BrandGAN: Unsupervised Structural Image Correction

Mostafa El Katerji

Thesis submitted to the University of Ottawa
in partial Fulfillment of the requirements for the
Masters in Computer Science:

School of Electrical Engineering and Computer Science (EECS)
Faculty of Engineering
University of Ottawa

Abstract

Recently, machine learning models such as Generative Adversarial Networks and Autoencoders have received significant attention from the research community. In fact, researchers have produced novel ways for using this technology in the space of image manipulation for cross-domain image-to-image transformations, upsampling, style imprinting, human facial editing, and computed tomography correction. Previous work primarily focuses on transformations where the output inherits the same skeletal outline as the input image.

This work proposes a novel framework, called BrandGAN, that tackles image correction for hand-drawn images. One of this problem's novelties is that it requires the skeletal outline of the input image to be manipulated and adjusted to look more like a target reference while retaining key visual features that were included intentionally by its creator.

GANs, when trained on a dataset, are capable of producing a large variety of novel images derived from a combination of visual features from the original dataset. StyleGAN is a model that iterated on the concept of GANs and was able to produce high-fidelity images such as human faces and cars. StyleGAN includes a process called projection that finds an encoding of an input image capable of producing a visually similar image. Projection in StyleGAN demonstrated the model's ability to represent real images that were not a part of its training dataset. StyleGAN encodings are vectors that represent features of an image. Encodings can be combined to merge or manipulate features of distinct images.

In BrandGAN, we tackle image correction by leveraging StyleGAN's projection and encoding vector feature manipulation. We present a modified version of projection to find an encoding representation of hand-drawn images. We propose a novel GAN indexing technique, called GANdex, capable of finding encodings of novel images derived from the original dataset that share visual similarities with the input image. Finally, with vector feature manipulation, we combine the GANdex vector's features with the input image's projection to produce the final image-corrected output. Combining the vectors results in adjusting the input imperfections to resemble the original dataset's structure while retaining novel features from the raw input image. We evaluate seventy-five hand-drawn images collected through a study with fifteen participants using objective and subjective measures. BrandGAN reduced the Fréchet inception distance from 193 to 161 and the Kernel-Inception distance from 0.048 to 0.026 when comparing the hand-drawn and BrandGAN output images to the reference design dataset. A blinded experiment showed that the average participant could identify 4.33 out of 5 images as their own when presented with a visually similar control image. We included a survey that collected opinion scores ranging from one or "strongly disagree" to five or "strongly agree." The average participant answered 4.32 for the retention of detail, 4.25 for the output's professionalism, and 4.57 for their preference of using the BrandGAN output over their own.

Acknowledgment and Dedication

My pursuit of this degree would have never been possible without the support that I had from my supervisor, my family, and my friends.

To my supervisor Dr. Hussein Al Osman, I will be forever grateful for providing me with this opportunity and going above and beyond with your guidance and support.

To my parents and siblings, I am genuinely privileged to have your unconditional support. You are my main source of motivation.

To my grandma Adleh, my favorite person and my biggest fan.

To Iman, who was the first to encourage me to pursue a career path that I am extremely passionate about.

To my best friends Reem, Firas, Omar, Mostafa, Joseph, Daniella, Stephanie, and Jack.

Table of Contents

List of Tables

List of Figures

Glossary of Terms

GAN: Generative Adversarial Network

StyleGAN: Style-Based Generator Architecture for Generative Adversarial Networks

CycleGAN: Cycle-Consistent Adversarial Networks

GANdex: Generative Adversarial Network Index

ADA: Adaptive Discriminator Augmentation

LPIPS: Learned Perceptual Image Patch Similarity

VGG-16: Visual Geometry Group 16 model

UD: User Drawings Image Set

IC: Image Corrected Image Set

RDD: Reference Design Dataset

KID: Kernel Inception Distance

FID: Fréchet Inception Distance

MSE: Mean Squared Error

FFHQ: Flickr-Faces-HQ Dataset

CT: Computed Tomography

CBCT: Cone-beam Computed Tomography Systems

MLFcGAN: Multi-level Feature Fusion based Conditional GAN for Underwater Image Color Correction

ST-GAN: Spatial Transformer Generative Adversarial Networks for Image Compositing

PULSE: Photo Upsampling via Latent Space Exploration

Chapter 1 - Introduction

When a group of individuals has been tasked with hand-drawing the same image, they would each produce a different version of the image based on their drawing skills and hand-eye coordination. Even skilled designers, when given the same task, could produce different versions of the same image.

Design software allowed for the creation of more consistent illustrations capable of being edited and reproduced digitally. As a result, software tools reduced undesired visual artifacts due to the user's motor movement. Designers can use free-hand drawing tools such as drawing tablets [1][2] for advanced use-cases such as customized illustrations; artifacts are still often produced and need to be corrected through software or by the designers themselves. SAI [3][4] and similar tools offer stabilizer features to smoothen digital tablet drawings.

1.1. The Problem of Image Correction

For an individual with no background in graphic design, using design software applications imposes a barrier to entry associated with the steep learning curve and the financial cost of purchasing these tools. Suppose that the individual has a short-term design need, such as a logo for their business, it is more time and cost-effective to hire a designer to create their needed artifact.

Before the designer can serve their client's requests, they need to collect information on the design requirements [5]. A typical process involves the client describing what they need. The designer develops a set of sample mock-ups [6], followed by the client selecting one they choose to proceed with, which the designer from that point forward will use. The process of producing the mock-ups requires the designer to invest time in creating each one to achieve

“professional-looking” results that the client can review. Some designers’ approach is to hand-draw the mock-ups on a piece of paper and then use it as a reference to produce the “professional-looking” final version of the mock-up. Depending on the number of mock-ups being made, this process can be tedious and involves work that the designer will discard for the mock-ups that the client does not favor.

We define a “professional-looking” image as an image that seems to be designed by a professional graphic designer to the typical viewer. We recognize that this is a subjective property that cannot be easily quantified with objective metrics. Hence, in this thesis, we use subjective measures when we assess this visual quality.

If a designer can automate the process of transforming their hand-drawn images into “professional-looking” images, it would save them time and allow them to provide clients with more mock-ups if needed. Automation will decrease the turnaround time and provide clients with more options that they could use to make an informed decision. Such a process could be used by any individual that aims to accomplish a similar goal. They could provide their image to such a process and expect it to adjust the image into a more esthetically pleasing form while retaining their original intended visual features.

We define the process of producing a “professional-looking” asset from a hand-drawn image as Image Correction. A common practice for designers is to brand [7][8] their designs on inspirations from other creators that are usually available as image bundles [9][10][11]; we define such branding inspirations as Reference Design Datasets (RDD). This thesis aims to research a solution for automating the Image Correction process that adjusts hand-drawn image structure to resemble a target RDD.

We propose the BrandGAN framework, an automated branding process that transforms a hand-drawn image to resemble a target reference design using a Generative Adversarial Network (GAN) [12]. It leverages a novel GAN [12][13] indexing technique called GANdex, and a modified GAN exploration process to solve Image Correction. We chose to use UI icons developed by professional graphic designers as our RDD throughout our study because they play a role in brand identity [7][8].

1.2. Cross-domain Image-to-image Translation

During our study of previous work, we found that research work is scarce in Image Correction. Therefore, we decided to widen our investigation of existing work by evaluating the cross-domain image-to-image translation literature.

Cross-domain image-to-image translation is the process of translating an input image from one domain to another. With this definition, Image Correction would be an image-to-image translation task that maps images from the hand-drawn to the Image Corrected domain.

This section describes existing works that tackle various Image Translation tasks across a variety of domains. We explore these works in this Chapter to highlight their limitations and provide a motivation for the proposed work.

1.2.1. Unsupervised image-to-image Translation Networks

Previous work [14] tackled this problem for paired images from different domains, such as satellite imagery and map data. Satellite imagery is more widely available than maps [15]; an image-to-image model would be valuable for predicting map information directly from satellite images rather than being collected through more costly means such as through third-party data

partners [16]. The authors use a Coupled GAN [17] approach to learning the mapping between two image domains. This approach requires the paired images' structure in the training dataset to overlap and is not compatible with arbitrary unpaired datasets.

1.2.2. Unpaired image-to-image Translation

Some works iterated on image-to-image translation [18][19] by leveraging a constrained named cycle-consistency loss. This technique enabled image-to-image translations between domains without requiring paired datasets. The advantage of this technique is that getting dataset pairs could be expensive or complex in some situations.

Some examples provided by the papers [20] include making a Horse resemble a Zebra or transforming a painting into a photograph. The translation task does not require the Zebra to Horse datasets to be correlated with each other. In previous techniques [14], accomplishing the same goal would require the expensive process of collecting paired images of Horses and Zebras from the same angle that have similar proportions.

1.2.3. GANs and StyleGAN

The Generative Adversarial Network (GAN) [12] is a model capable of generating fake examples of a real dataset. It is composed of two competing AI models named Generator and Discriminator. The generator model takes a noise vector as input and generates an image vector; the vector is sampled from a uniform distribution commonly referred to as the latent space [21][22]. The discriminator model takes real images from a dataset and fakes from the Generator as input and classifies them as real or fake. During training, Generator optimizes on generating fakes while the Discriminator optimizes on classifying fakes.

StyleGAN [23] builds on the GAN model by introducing the concept of disentanglement. Instead of feeding a noise vector directly to the Generator, StyleGAN introduces input latent (or **z**) and intermediate latent (or **w**) spaces. A translation network is used to convert **z** vectors to **w** vectors. The trainable translation network is used alongside a Progressive GAN [24] during training, resulting in disentangling features in the **w** space. We discuss disentanglement further in Section 2.1.2.

StyleGAN2 [13] is the second iteration of StyleGAN that introduces enhancements to the Generator's training process leading to improved quality of the generated images. StyleGAN2-ADA [25] is the third iteration of StyleGAN that leverages a novel data augmentation technique called adaptive discriminator augmentation (ADA) that enables StyleGAN to produce higher-quality fake images when trained on smaller datasets.

Projection is a process supported by GAN and StyleGAN capable of finding a noise vector that can approximate the representation of an RGB image in the model's latent space.

GAN's ability to generate novel images and StyleGAN's disentanglement of features in the **w** latent space enabled image-to-image techniques (discussed in Sections 1.2.3.1 and 1.2.3.2).

1.2.3.1. StyleGAN2 Distillation for Feed-forward Image Manipulation

This work [26] focuses on manipulating StyleGAN2 generated images by learning mappings within the StyleGAN2's latent space between different domains. They demonstrate their work's ability to learn how to control perceived gender and age on human faces.

1.2.3.2. Correction By Projection: Denoising Images With Generative Adversarial Networks

Correction By Projection: Denoising Images With Generative Adversarial Networks [27] tackles the problem of denoising images. They use projection, a process for GANs that searches for a vector capable of generating an approximation of an input image through a GAN's generator. Given that the Generator only produces images from the training dataset that did not include noisy images, the result is a denoised but visually similar image to the input.

1.3. Current Limitation of In-Place Image Translation

Most of the previous work that we explored is limited to in-place image translation. In-place translation modifies image textures or colors, but the initial structure of the input image is retained in the output.

Image Correction is inherently dependent on adjusting the input image structure and not just its textures or colorization. To solve the problem, we considered supervised and unsupervised approaches. A supervised approach would require collecting hand-drawn samples from a diverse demographic of individuals for every domain that requires image correction. Therefore, collecting data for a supervised approach would be costly, restricted to the data collection domains, and does not guarantee an inclusive representation of all individuals' drawing styles. With that said, an unsupervised approach that adapts to an RDD and is capable of tackling image correction in a generalized way regardless of an individual's drawing style is a better fit for this task.

Our Image Correction solution is inspired by how previous work on StyleGAN2 image translation with face editing [26][18] showed the possibility of using the StyleGAN model to adjust image structure predictably. Facial editing is different from other solutions as the

individual's facial features in the portrait are retained while structural elements such as their orientation and head shape are changed. Image Correction aims to achieve a similar outcome by retraining detail from the hand-drawn input image while adjusting its structure to align with the RDD.

We used this inspiration to prototype a similar StyleGAN2-based translation capable of editing images with a flawed structure to resemble a corrective structure inspired by a reference design dataset. In the prototype, we explored manipulating facial structure using the StyleGAN2 FFHQ model [29], where we aimed to adjust jaw width predictably. We discuss the prototype in greater detail in Section 2.3.1.

1.4. Problem Statement and Motivation

The process of creating "professional-looking" mockups is time-consuming, costly, and requires design expertise; this imposes a burden on all the parties involved and makes it less accessible for individuals that are unable to bear its costs. While platforms such as Fiverr [30] and Upwork [31] helped make professional designers more accessible to customers by connecting customers directly with freelancers, the traditional design process remains to be exhaustive [32] and expensive [33].

Graphic designers charge customers differently depending on their expertise and the proposed project's scope [34]. Two typical pricing models include compensation based on an hourly rate or a lump sum paid on a project basis [34].

At the start of a project, designers meet with their clients to collect requirements to understand their needs [35][36]. Even when requirements are communicated clearly, the vision of the clients and designers do not necessarily align. Designers present mockups of varying designs to the

customer to avoid misalignment and ask them to choose their preferred direction [35][36]. A common practice while creating mockups is for designers to sketch their ideas before using vectorization software to digitize their drawings [32][37][9]. The sketching step is where the designer uses their creative process to materialize the client's requirements, while vectorization is a mundane task where they make their sketch presentable and professional. Some designers tend to present two to four mockups and then iterate on the mockup their client selected before finalizing their design [38][39]. The more mockups and iterations are presented, the more time would be required for the designer to sketch and vectorize; this would lead to higher costs imposed on the client and delay the project's timeline.

We identified the mundane process of digitizing hand-drawn images to be a scaling problem. The motivation of finding a solution to this problem is the impact on reducing cost on clients and increasing designer throughput. As an added benefit, we were motivated to explore such a solution as a tool that can be used by individuals with any level of design expertise to produce high-quality content from their sketches.

Therefore, in this thesis, we will address the following problems:

1. Choosing an approach for transforming structures in an unsupervised manner
2. Automating the process of transforming hand-drawn sketches into "professional-looking" mockups

1.5. BrandGAN

The result of our research is BrandGAN, a framework that leverages StyleGAN2-ADA [25], a novel GAN indexing called GANdex, and latent space interpolation to accomplish structural image correction.

The Generator within StyleGAN2-ADA accepts noise vectors, sampled from the intermediate latent space **w**, as input and produces an image as output. Given two latent vectors, if the latent space were to be traversed from the first to the second vector, the traversal's produced image would show the first vector's image slowly warping into the image produced from the second vector. Taking both vectors' average generates an image in the midpoint between the two (equally mixes elements from both images).

After GANs are trained on an image dataset, they can generate an infinite number of fake images based on the training dataset. Projection is a process where a latent vector is searched for in the input latent space **z** capable of generating a close image approximation to an RGB image.

The ability to combine features from multiple latent vectors applies to projected vectors. A projected vector could be based on a structurally flawed image. If there were to be another vector that produces a similar but refined structure, blending the two vectors would combine the feature representations of the two structures. The resulting vector would generate an image that skews both structures into each other, resulting in a correction of the flawed projection. Applying this process to a projected hand-drawn image with a vector representing a similar but refined structure would produce the image-corrected version of the hand-drawn image.

BrandGAN aims to do the following tasks:

- Parse an input hand-drawn image
- Find a corrective latent space vector that strongly resembles the RDD and the input image using a novel index called GANdex.
- Project the input image starting from the corrective latent-space vector and find a vector that closely resembles the input image

- Interpolate a latent space vector between the corrective and projected vectors to find an output vector that maintains structures learned from the RDD while inheriting features and details from the input image

1.6. Contributions

We list our thesis contributions below:

1. We introduce a novel GAN indexing technique, called GANdex, capable of efficiently finding a StyleGAN2-ADA encoding in latent space closely resembling an input raw image.
2. We propose BrandGAN, a framework that uses GANdex, a modified version of StyleGAN2-ADA's projection process, and a vector interpolation technique. It tackless image correction of hand-drawn images in an unsupervised manner using an RDD.
3. We evaluate BrandGAN using objective and subjective measures.
    a. The objective measures use programmatic quantitative measures.
    b. We collected the subjective measures through a study with 15 participants. The participants were asked to draw images and do a blind test. We evaluated their ability to discern their image from a control.

1.7. Thesis Outline

The rest of the thesis is organized as follows:

Chapter 2: We present background and related work pertaining to image translation from one domain to another that leverages machine learning techniques.

Chapter 3: We describe the BrandGAN architecture and framework. We present our novel indexing technique GANdex, our modified StyleGAN2-ADA projection approach, and the image correction process.

Chapter 4: We evaluate BrandGAN in a study that included fifteen participants. Each participant provided drawings and answered a survey. We present and analyze objective and subjective measures based on the data collected during the study.

Chapter 5: We provide insights on future work and summarize our thesis findings.

Chapter 2 - Related Works

In this section, we discuss image-to-image translation techniques proposed in the literature. We review different GAN-based models such as StyleGAN and CycleGAN and various image-to-image translations that leverage GANs.

2.1. Generative adversarial network

This section reviews previous work on the Generative adversarial network framework (GAN) [12]. In our image correction task, we aim to translate hand-drawn images to resemble the structural styles present in the RDD in an unsupervised manner. Therefore, we focused on GANs due to their ability to generate images with structures that resemble the RDD when trained.

2.1.1. GAN

A Generative Adversarial Network [12] is a framework where two AI models compete to achieve opposing tasks. A Generator model is tasked with generating fake images that resemble a dataset, and the Discriminator is tasked with differentiating real from fake images.

The Generator's input is a random vector sampled from a distribution, called the latent space, while its output is an image. The Discriminator takes an image as input and outputs a binary real or fake classification. The models train independently on a turn-by-turn basis.

During the Discremenator's turn, the weights of the Generator are frozen. Fake images are generated using the Generator and combined with Real images from the training dataset—the Discriminator trains on the combination of images and updates the weights based on the labels it produces.

During the Generator's turn, the weights of the Discriminator are frozen. Vectors sampled from

the latent space are provided as input to the Generator before it outputs an image. The image is then passed to the Discriminator for classification. Backpropagation is used for the entire pipeline where only the Generator's weights are updated. The Generator only trains using the Discriminator and can eventually generate images similar to the training dataset without being provided any examples throughout the training process. GANs have been used in domains outside of images, such as generating novel molecules [40].

2.1.2. StyleGAN

While powerful, the base GAN model suffers from several limitations. The use of vectors sampled from a latent space as direct input to the Generator inhibits the vectors' ability to have a predictable influence on the generated images. The arbitrary nature of how the latent space maps to the Generator's output features leads to feature entanglement [23]. Feature entanglement is when distinct features share their representation in the latent space. An example of feature entanglement is having the shape of a generated face and the color of the eye sharing a feature representation where the manipulation of the shape of the face would not be possible without interfering with the eye color.

StyleGAN [23][41] is a model that iterates over the GAN by solving feature entanglement using what the authors refer to as Styles. They define three styles which are Coarse, Medium, and Fine.

- **Coarse** styles represent features such as the shape of a face, orientation, and other abstract structures.
- **Middle** styles represent lower-level structural features such as eye size, skin texture, hairstyle.

- **Fine** styles represent the lowest-level features such as hair and skin color.

To solve the problem of entanglement for the latent space and have the ability to represent styles independently of each other, the authors propose an intermediate latent representation and use a Progressive GAN [24].

The input latent space is referred to as **z**, while the intermediate latent space is referred to as **w**. During training, **z** vectors are mapped to **w** vectors through a novel Mapping Network before **w** vectors are provided as input to the Generator. The Mapping Network is responsible for learning mappings for styles from **z** to **w** independently of one another; by learning the styles independently, this process results in producing disentangled vectors. StyleGAN is trained similarly to a Progressive GAN, where the generator initially produces low-resolution images and slowly scales the generator's output size until reaching a threshold. Progressive GAN [24] training enables the model to learn the three distinct styles independently from Coarse to Fine.

StyleGAN produced high-fidelity images of objects, such as human faces, which decisively appear realistic. The improvement in detail can be attributed to the separation between the different abstractions that define an image's appearance and the GAN's entangled features.

Latent space manipulation improved as a result of having disentangled features. Vector manipulation can be executed in a predictable way to produce an intended outcome. For example, consider three vectors from **z**; vector AN represents person A, vector AG represents the face of person A with glasses, and vector BN represents person B without glasses. The result of subtracting the vector AN from AG is a vector G. Adding vectors BN and G result in a vector BG where BG represents person B with glasses.

Images generated from a weighted sum of vectors result in having a combination of features from all vectors with varying intensities depending on weights.

An intermediate latent vector **w** can represent a seemingly infinite combination of features, giving StyleGAN the ability to generate a significant variety of novel images. Through a process called Projection, RGB images can be used to find a latent vector in the intermediate latent space that generates an image that closely resembles the input using a loss function. A projected image can benefit from latent space manipulation, such as augmenting a real human image with glasses. Projection is discussed in further detail in Section 3.5.

As of the writing of this thesis, the authors of StyleGAN published two subsequent works [13][25] that build on the original StyleGAN paper [23]. The second paper proposed StyleGAN2 [13][42], which incorporated improvements in the training process that generated higher-quality images from the Generator than StyleGAN. The third paper proposed StyleGAN2-ADA [25][43], which introduced data augmentation techniques that enabled the network to train in smaller datasets than it was capable of before.

2.1.3. StyleGAN2

StyleGAN2 [13] focused on improving the quality of the images produced by the original StyleGAN work. They focused on introducing improvements such as regularization and redesigning generator normalization, and revisiting the model architecture. As a result, they could resolve visual artifacts present in the base model and produce higher-fidelity fakes.

2.1.4. StyleGAN2-ADA

StyleGAN2 with adaptive discriminator augmentation (ADA) [25][43] is the third publication for StyleGAN that tackles training GAN models with less training data. GANs, when trained on

small datasets, lead to discriminator overfitting and the divergence of the training. They demonstrate discriminator overfitting by training StyleGAN2 on randomized subsets of the FFHQ [44] and observing how the training progresses using the Fréchet inception distance (FID) [25] score. They observe that the FID score decreases similarly across all subsets but ultimately starts to rise again. A rising FID score indicates that the Generator's output is increasingly generating lower-quality fakes, suggesting that the GAN is diverging and is no longer training effectively.

To solve GAN training divergence when a small dataset is used, they propose using data augmentation techniques for the discriminator. This approach allowed the discriminator to generalize more on smaller datasets, leading to less overfitting and a lower chance of divergence from the GAN. They use eighteen data augmentation transforming techniques categorized under pixel blitting, geometric transformations, color transformations, and more.

While augmentation slows down overfitting for the discriminator, it comes at the cost of slowing down the GAN's convergence. The authors introduce a parameter called augmentation probability represented as **p**, which is the probability that an augmentation transformation is applied at any given time. Therefore, increasing **p** lowers overfitting but slows down convergence. The authors propose a heuristic method to adjust **p** every number of epochs based on the degree of overfitting. By doing so, **p** is increased to counteract overfitting and is then decreased when the model is evaluated to generalize better. The approach is called the adaptive discriminator augmentation (ADA) technique and is meant to overcome the drawback of slow convergence with a **p** fixed at a high value.

2.2. Image Translation using GANs

In this section, we cover previous works that use GANs for the task of image translation. We surveyed this work to learn more about various techniques in that literature that can be adapted for image correction.

2.2.1. Effects and Color Transfer

Icons are typically produced and shipped as packages of varying structures that share a common artistic design pattern. This pattern exists to maintain visual consistency between such artifacts when they are used in contexts such as user interfaces. Google's Material Design Icons [45][46] is a popular icon pack commonly found on Google products and Android applications.

Icon packs can have commonly used shapes such as UI elements or novel shapes unique to the pack. The number of icons in a pack can vary depending on how and where they will be used, but they all typically share a consistent style.

The style needs to be defined only once and then replicated for every icon type produced. Since designing the style is done at the beginning, the process of designing the majority of the icons becomes redundant.

Image translation work [47][48] has tackled automated effects and color transfer. Their models accept structural and style-reference images as input and output an image with the reference's style applied to the input structure. With such a tool, an icon pack designer can create a style for one icon then automatically replicate it on other structures they choose to include in their pack.

2.2.2. Color Correction

Color correction is the process of adjusting an image to fix artifacts from the environment in which it was taken or to alter the image's presentation.

MLFcGAN [49] is a GAN-based model that aims to correct underwater images. Underwater imaging often exhibits artifacts due to the specialized waterproofed cameras and the lighting conditions that differ considerably from standard imaging. The authors demonstrated MLFcGAN's ability to reconstruct realistic underwater colors while retaining more detail than the previous work.

2.2.3. Image Compositing

ST-GAN [50] is a GAN-based model that uses a geometric transformation to create an image composition of a scene and a foreground object. During the training process, the model learns the mapping between an object and where it would typically be located in a scene. When the model is provided background and foreground images, it follows an iterative process that narrows down a possible combination of the two images.

They demonstrate the model's capabilities with examples such as placing furniture in a room and glasses on human portraits.

2.2.4. Upscaling Pixelated Images using GANs

Upscaling is the process of transforming a low-resolution image into a high-resolution image. Previous works, such as PULSE [51], explored the upscaling of human-facial images using GANs.

Their approach explores StyleGAN2's latent **z** space, searching for a vector capable of generating the upscaled version of the pixelated input. While the standard StyleGAN2 exploration approach uses a loss function between the input and the generated images, PULSE computes the loss between the input and the pixelated version of the generated image. PULSE's modified loss function acts as a heuristic to find the best vector that would generate the pixelated input's upsampled transformation.

2.2.5. Paired cycle-GAN-based image correction for quantitative cone-beam computed tomography

Computed tomography scan, or CT Scan, is a medical imaging technique that uses X-ray imaging to construct cross-sectional images of bones, blood vessels, and soft tissue inside a body [52]. Cone-beam computed tomography [53], or CBCT, serves a similar purpose to CT but uses rotating cone-shaped X-ray beams to capture instead of image slices. CBCT is faster than CT imaging at the cost of reduced accuracy in the detail of the images.

A previous work [54] pursued bridging the gap between CBCT and CT by developing an AI image-correction model claimed to correct CBCT artifacts that are not present in CT images. The proposed solution uses a cycle-GAN [19] model alongside paired CBCT and CT images of the same subjects. The cycle-GAN learns the mapping between CBCT to CT images through cycle-consistency loss; these mappings include CBCT-specific artifacts or loss of detail. As a result, the image correction model was shown to produce CT from CBCT images comparable to the ground truth.

2.3. Gap Analysis

To our knowledge, none of the previous works focus on the structural translation of images; they

mainly augment the input domain with stylistic changes or pixel-level color adjustments.

Image correction of hand-drawn images requires structural translation where noisy outlines are adjusted to resemble refined structures using corrective structures derived from the RDD. Therefore, we see that our work's novelty focuses on the structural rather than stylistic adjustment of our input domain.

We broke down the problem scope into three essential requirements:

1. Transform structures predictably.
2. Learn corrective structures in an unsupervised manner.
3. Translate noisy structures into refined structures with the least amount of changes

Satisfying the three requirements is the basis of our solution to the image correction problem. The first requirement is for the tool that is capable of adjusting the structure of hand-drawn images. The second requirement provides the ability to capture a wide variety of structures from the reference design dataset without prior knowledge of its content. The third requirement is a process that accepts a hand-drawn image, finds a corrective structure that resembles it the most, then uses the ability to manipulate structures to adjust the input's noisy structure to resemble the corrective structure closely.

2.3.1. Transform Structures Predictably

To satisfy this requirement, we wanted a base model capable of representing structure and support making adjustments using learnable approaches.

We were inspired by StyleGAN2's [13] ability to manipulate coarse features. In the model's paper, the authors demonstrated how manipulating coarse features on a generated face could drastically alter head structure and orientation while keeping all other facial features constant.

To experiment with controllable transformation of structure with GAN, we attempted to manipulate a generated image's jaw without adjusting any other feature. The experiment went as follows:

- Using the author's pre-trained FFHQ StyleGAN2 model [29], we generated twenty faces using twenty random vectors sampled from StyleGAN2's **z** space.
- We selected two vectors where one generated a face with a narrow jaw (NJ) and another with a wide jaw (WJ). The head structure and orientation of both faces were equivalent.
    - Our goal was to attempt to widen NJ's jaw without changing any of their other features.
- Using trial-and-error, we found the latent vectors' components that directly influence jaw width by swapping random permutations of components between the vectors until we could generate a new vector WNJ that has the facial features of NG but with the jaw-width of WJ.
- We averaged vectors WNJ and NJ and used the result to produce an image. The image generated from the average vector contained the same subject as NJ but with a jaw width that roughly falls halfway between those of NJ and WJ.
- We attempted the previous step but used weighted averages. It confirmed that the more weight we allocate to WJ, the wider the jaw becomes, and the opposite is true if more weight is given to NJ.

This experiment showcased StyleGAN2's ability to manipulate image structure. However, such image translation necessitates a source and a target. In our example, the source was NJ, and the target was WJ. Hence, to widen the jaw for NJ, we simply need to generate a large number of images using random vectors from the input latent space **z** and employ a classifier that identifies WJ candidate images to select our target. Then, by calculating a weighted average of NJ and WJ, we achieve the intended effect. When it comes to a hand-drawn image, there is no image of the drawing that exhibits the corrective structure that we can drive the input image towards. Therefore, we cannot use this widely applied approach for our Image Correction. The fulfillment of the other two requirements presented below allows us to resolve this gap in existing methods.

2.3.2. Learn Corrective Structures In An Unsupervised Manner

StyleGAN2 is capable of generating "fake" images that resemble a given dataset. While the "fake" images are novel, their structure/coarse features are derived from the dataset's perceived structures. Vectors generated from the **z** space have structure derived from a combination of the training dataset's coarse features.

We define a corrective structure as a structure generated from a vector sampled from the input latent space **z** and closely resembles an input image. By sampling it from the **z** space, it guarantees that it is derived from coarse features present in the original dataset learned during StyleGAN2's training. An input's corrective structure needs to be retrievable without prior knowledge of the dataset; we accomplish corrective structure resolution using our proposed GANdex indexing approach.

2.3.3. Translate Noisy Structures Into Refined Structures With The Least Amount Of Changes

Image correction is expected to make fine adjustments to the input image without drastically altering the abstract structure intended by the user. Overcorrecting can lead to a drastically different image, while under-correcting can lead to a marginally adjusted version of the input image.

We were able to satisfy this requirement by combining the first two requirements. The result is as follows:

1. GANdex is used to find the corrective structure for an input image
    a. This step is satisfied through the fulfillment of the second requirement
2. Using the corrective structure's vector from GANdex as a start, the input image is projected using StyleGAN2-ADA. Since the corrective structure is used as the starting point, the distance between the corrective and projected vector would be minimized due to its inherent similarity to the input image.
3. The weighted average of the corrective and projected vectors can produce an image that blends coarse features from the corrective structure and project image influenced by the weights.
4. Since the distance between the vectors is minimized within the latent space, the weighted average would exhibit minimal visual changes between vectors as their feature vectors are close. Therefore, this approach would reduce the number of changes imposed on the input image and only affect minor noisy structural details.

Chapter 3 - Proposed Method

Inspired by the previous work on image-to-image translation, we sought to find a solution for image correction capable of translating images from the hard-drawn to the image-corrected domains in an unsupervised manner.

In the following subsections, we will present our detailed work and implementation of the BrandGAN framework. We start by describing the dataset we used for our study. Next, we discuss the BrandGAN initialization steps, including training the underlying StyleGAN2-ADA model and building the GANdex cache. Finally, we provide a thorough description of the BrandGAN end-to-end image correction process.

3.1. Reference Design Dataset

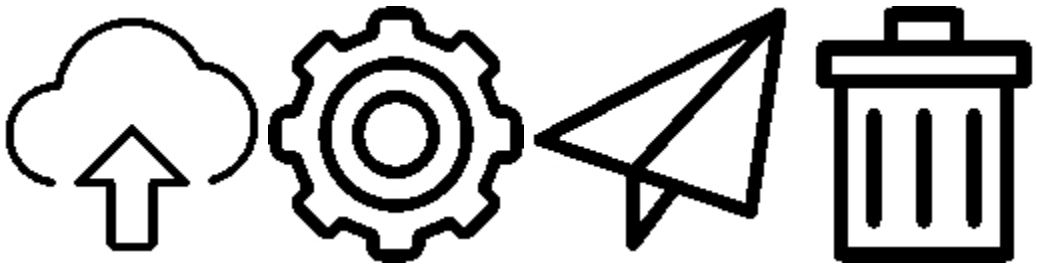

Figure 3.1 - Examples of UI Icons from the Reference Design Dataset

Figure 3.2: Examples of UI Icons excluded from our study.

This research aims to explore image correction that is primarily structurally driven. We chose to

use a black and white dataset that has recognizable but simplistic images. As a result, we used Flaticon [55] and downloaded free flat UI icons (see Figure 3.1). We initially downloaded 8233 images that contained 57 unique icon labels based on crawling metadata. To allow the StyleGAN2-ADA model to generalize better, we aimed to remove outlier images that may affect the quality of the fakes generated. Images with text, 50% or more black pixels, or fewer than three examples in the dataset were removed (see Figure 3.2). The final dataset contained 2667 images.

The dataset is referred to as the Reference Design Dataset (RDD) as it would act as the reference for design structures that BrandGAN would attempt to replicate.

3.2. Architecture

In this subsection, we will present the architecture of the BrandGAN framework. We will first describe the prerequisites of using BrandGAN: the StyleGAN2-ADA model training and GANdex initialization. Afterward, we describe BrandGANs end-to-end process for image correction.

3.2.1. Training

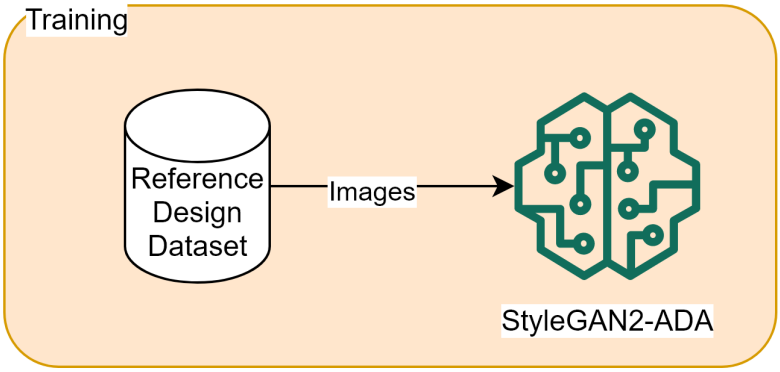

Figure 3.3 - StyleGAN2-ADA Model Training on the Reference Design Dataset

The core GAN component of BrandGAN is StyleGAN2-ADA [25]. We explored other models such as Wasserstein GAN (WGAN) [56] and the base StyleGAN2 model [13].

We started with WGAN, and we trained for one week before it plateaued. It produced novel shapes, but they did not share any structural similarity to the dataset. WGAN's inability to represent coarse features did not allow us to manipulate its latent encodings in predictable ways to control structure and detail.

After WGAN, we experimented with StyleGAN2. We trained StyleGAN2 on the dataset for four weeks on an NVIDIA RTX 2070 super [57]. It supports the generation of images from its initial latent space z or its intermediate latent space w. We generated images using StyleGAN2 with vectors sampled from the **z** space; the generated images were fakes that did not belong to the training dataset, but it was evident that their structure was derived from the dataset. When we manipulated the z vectors' mapping in the **w** space, we could adjust the generated image's structure.

We experimented with StyleGAN2's projection module using images from the dataset and generated latent vectors from the **w** space. We added noise to the vector to produce structural anomalies, then took the latent vector's average with and without the noise. The image produced from the average vector adjusted the anomalies to resemble an icon's reference structure closely. This result was promising as the averaging image corrected the noisy image anomalies while retaining minor visible details. One drawback to the base StyleGAN2 model is that it failed to capture most of the dataset in its generated outputs due to our dataset's small size.

StyleGAN2 with adaptive discriminator augmentation (ADA) [25][43] was released while we experimented with the base StyleGAN2 model. StyleGAN2-ADA iterated over the base model

by introducing data-augmentation techniques that allowed it to train a smaller dataset. We trained the StyleGAN2-ADA model (see Figure 3.3) on our dataset using an NVIDIA Tesla V100 GPU [58] for three weeks.

StyleGAN2-ADA's trained model could generate images representing the entire dataset alongside enhanced image details compared to the base StyleGAN2 model. As a result, we chose it as the primary GAN model for BrandGAN.

3.2.2. GANdex Initialization

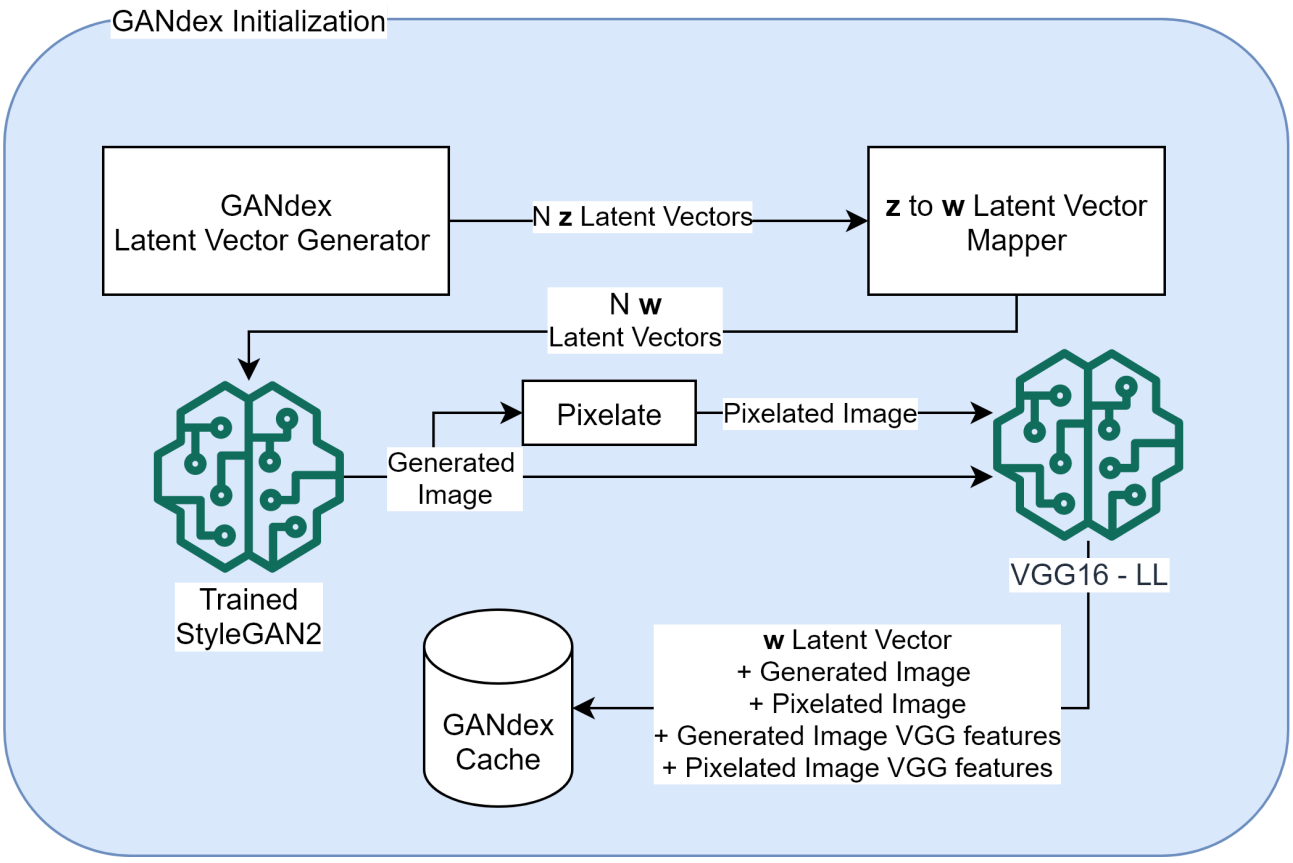

Figure 3.4 - GANdex Initialization Process

GANdex Initialization (see Figure 3.4), is a process where a cache of vectors with corrective structures is generated using StyleGAN2-ADA. The corrective structures are used during the

image correction process and are meant to be quickly accessible. The cache is created by sampling vectors from StyleGAN2-ADA's **z** space, transforming them into the **w** space, and generating images by vectors from the **w** space to StyleGAN2-ADA's Generator.

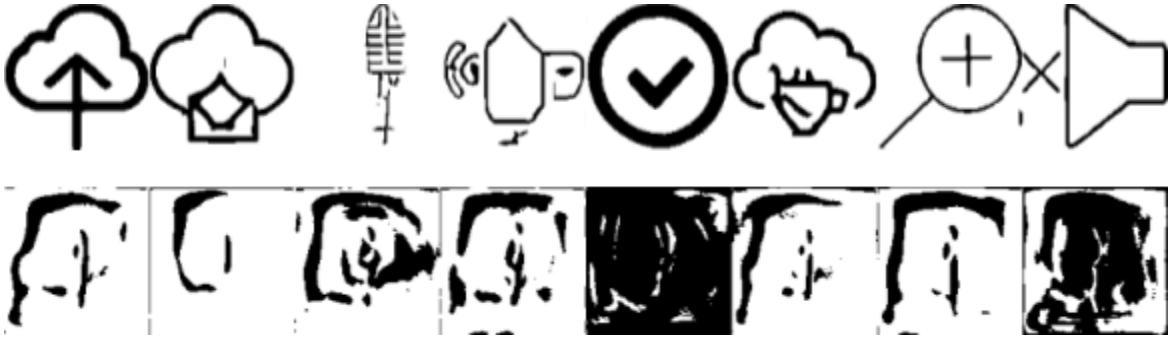

*Figure 3.5: The top images were generated using eight random vectors sampled from the **z** latent space. The bottom images were generated using eight random vectors from the **w** space.*

The GANdex Latent Vector Generator samples N random vectors from a normal distribution that belongs to the **z** space. The number of vectors N is a hyperparameter for GANdex that controls the number of samples available during the lookup process and is discussed further in Section 3.4. Sampling is done from the **z** space to ensure that the generated images belong to the same distribution as the fake images used during StyleGAN2-ADA's training. If the **w** space were sampled directly, it would result in random noise images (see Figure 3.5).

The **z** to **w** Latent Vector Mapper passes the N **z** vectors to StyleGAN2-ADA's mapping network and generates N **w** vectors as a result. Since the **w** vectors were derived through the mapping network, they would generate high-quality fakes, unlike if the **w** vectors were directly sampled from a random distribution. This step is required since the Generator only accepts **w** vectors as input.

The N **w** vectors are passed as input to StyleGAN2-ADA's Generator, which produces N images as a result. The generated images are used during the GANdex Lookup process to match hand-drawn inputs with cached fakes. The lookup process, discussed with greater detail in Section 3.4, attempts to match images in their raw and pixelated form to increase the chances of finding a suitable corrective structure for the input. Finally, N pixelated images are generated.

Another step in the GANdex Lookup process, which will be discussed in greater detail in Section 3.4, matches the raw and pixelated sets of images using features generated vectors from the VGG-16 [59] model pre-trained on the ImageNet dataset. Generating the N images' features is costly, and they need to be readily queryable for later use. For these reasons, the VGG-16 features are generated for the raw and pixelated images during the initialization of GANdex.

The result of the previous computations is N tuples. Each tuple contains an intermediate latent vector **w**, a raw image, a pixelated version of the raw image, a VGG feature vector for the raw image, and a VGG feature vector for the pixelated image. All tuples are saved for later use.

BrandGAN checks for the cached data on every run; if the GANdex cache has been previously generated, BrandGAN directly loads the results into memory, making it ready for image correction executions. It is used in the BrandGAN Image Correction Process described in Section 3.2.3.

3.2.3. BrandGAN Image Correction Process

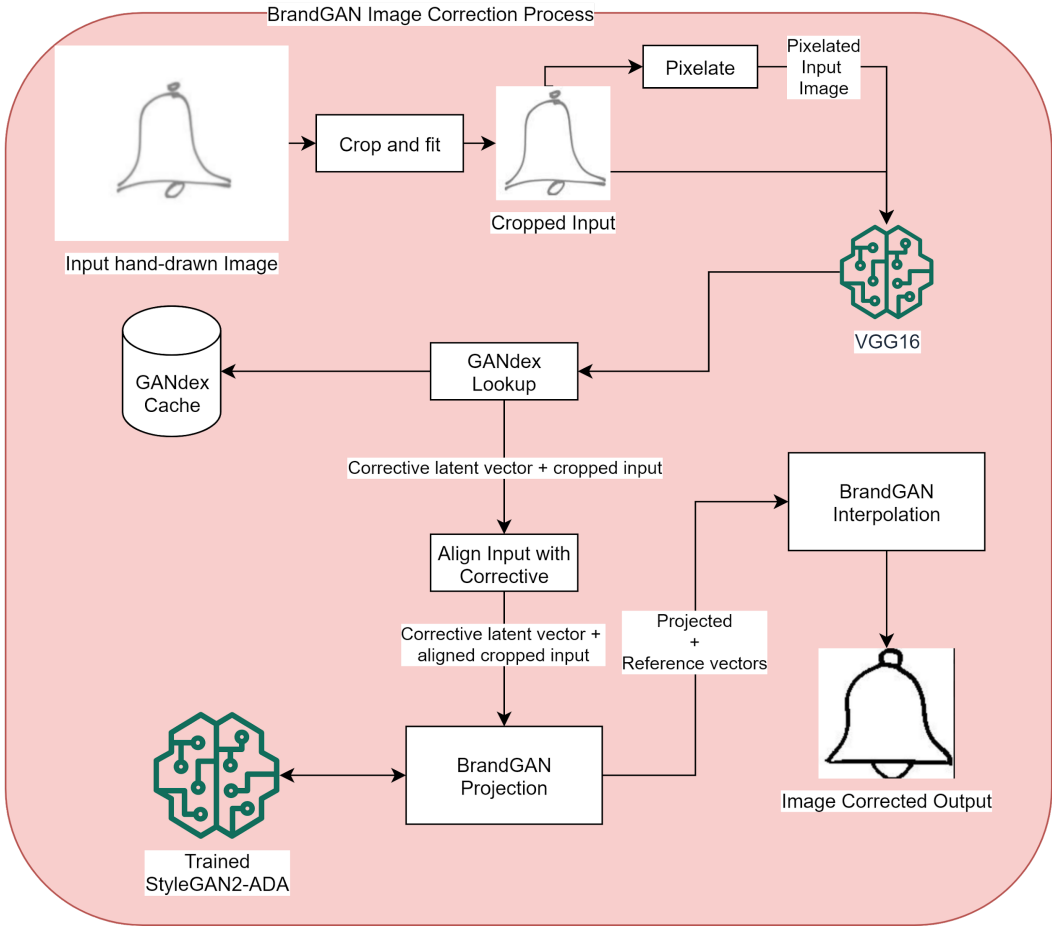

Figure 3.6 - BrandGAN Image Correction Process

The BrandGAN Image Correction Process (see Figure 3.6) is the end-to-end process responsible for transforming a hand-drawn image into its image-corrected counterpart.

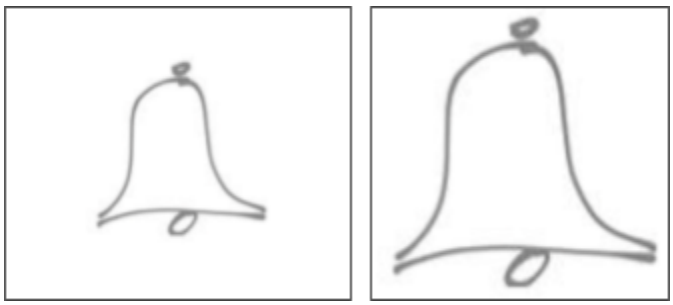

Figure 3.7: The image on the left is an example of hand-drawn image input. The image on the right is the input image with its excess padding removed and scaled to fit a square image.

The input image can have an arbitrary size and aspect ratio. Therefore, the first step in the process is to normalize input to produce consistent behavior. StyleGAN2-ADA, when trained on our dataset, produces 128x128 square images. For simplicity, we chose to use the exact resolution universally across all of BrandGAN. The Crop and fit step removes excessive padding in the input image and then produces a 128x128 image that maintains the drawing's aspect ratio (see Figure 3.7).

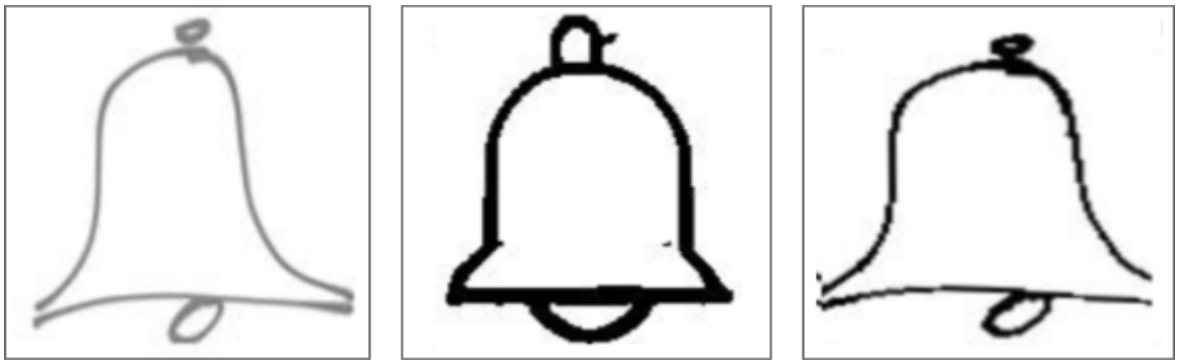

Figure 3.8: A cropped input image, its GANdex corrective vector, and the aligned version of the input image

Before the image correction is executed, a corrective vector that resembles the input's structure must be retrieved from GANdex. GANdex Lookup, which is discussed in further detail in Section 3.4, is responsible for retrieving an input image's corrective vector. GANdex Lookup's inputs are the raw image, its pixelated transformation, the raw image's VGG-16, and the pixelated image's VGG-16 features extracted. All four parameters are produced from the input image and sent to the GANdex Lookup process as parameters; it returns a corrective vector as a result.

The next step is BrandGAN Projection, which is discussed with further detail in Section 3.5.5 and is responsible for finding a representation of the input image within the StyleGAN2-ADA model's space. The input image needs to be aligned with the corrective vector (see Figure 3.8) to

maximize the projector's effectiveness. Alignment, discussed in Section 3.5.4, uses OpenCV's Motion Homography transform [60] to maximize the overlapping between the input image and the corrective vector. Motion Homography transform is typically used for aligning similar images that differ in rotation or translation. The projection process starts from the corrective vector and projects the aligned input image on StyleGAN2-ADA, which results in a final vector in the **w** space that produces an image that resembles the input image.

The final step is the BrandGAN interpolation, discussed in Section 3.6, which combines the corrective and projected vectors into the final image-corrected output. The combination is done through averaging, where the details of the projected image are combined with the collective's coarse features to produce an image corrected result.

3.3. GANdex

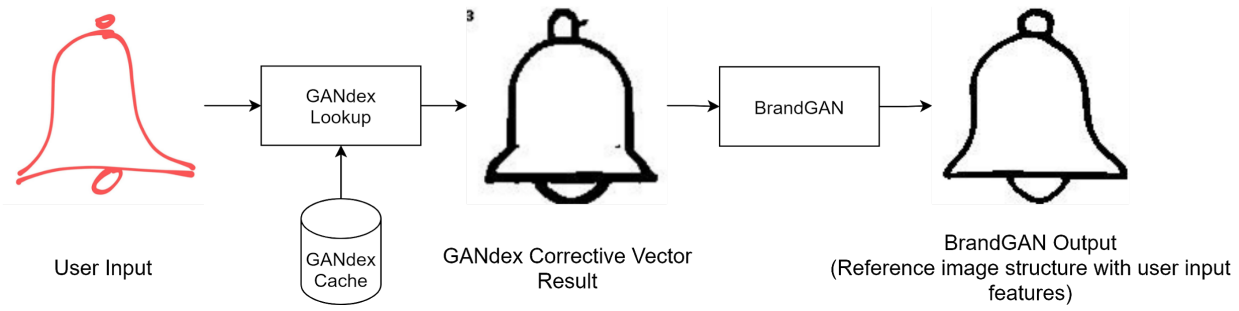

Figure 3.9 - GANdex lookup process before BrandGAN translation

Structural image correction requires the ability to have a corrective structure derived from the RDD, which resembles the structural elements of the input image. BrandGAN blends the corrective structural features with details from the user's image to produce the image-corrected result (see Figure 3.9).

The GANdex cache has N images produced from N latent vectors sampled from a normal distribution during the GANdex initialization process discussed in Section 3.2.2. The novel images inherit structure from the original dataset even though they have unique visual features. The larger the number of samples, the larger the variety of structures available for BrandGAN to use during the image correction process. The added variety increases the chance of finding a closer structural match in GANdex to the hand-drawn input's structure. We experimented with four values for N (1000, 10000, 50000, and 250000) during our research.

50000 showed the best improvements for BrandGAN's output as compared to 1000 and 10000. 250000 only showed marginal improvements compared to 50000. The cache file generated for 50000 was around 2GB in size, while 250000 was around 9GB. We chose to use 50000 as it provided the best balance between cache size and output improvement.

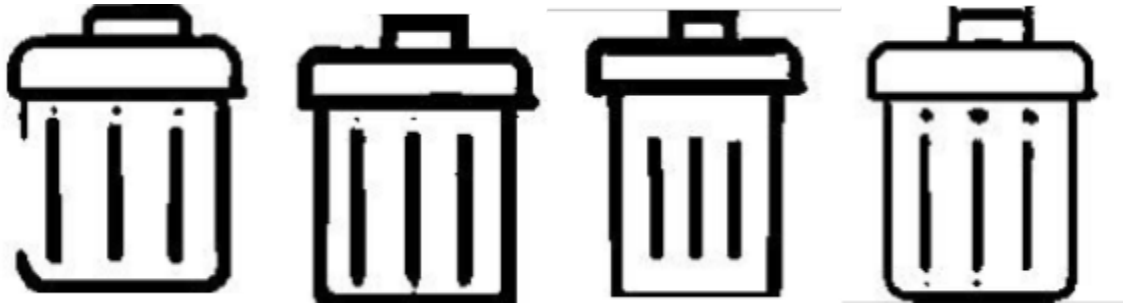

Figure 3.10 - Four trash icons generated using StyleGAN2-ADA

The four trash sample icons (see Figure 3.10) are generated from vectors sampled from the **z** space. They are novel to the training dataset and are composed of learned features from real trash icons. GANdex is constructed with random vectors that produce a large variety of novel images composed of a combination of features from one or more training samples. The variance in their structure increases the likelihood of finding a corrective vector from GANdex comparable to the user's input image. The closer the corrective vector and the input image, the

more likely StyleGAN2-ADA will retain the user's hand-drawn features during the image-correction process.

3.4. GANdex Lookup

GANdex Lookup is the process of finding a corrective vector from the GANdex cache that structurally resembles an input image. To find the corrective structure of a hand-drawn image, GANdex uses an ensemble of feature-matching algorithms to find a cached vector that resembles the input image.

Figure 3.11: Variations of the same icon as drawn by three separate individuals from our study.

Even when drawing the same intended target image, hand-drawn images from different individuals may differ in features such as pen stroke, hand-steadiness, and color (see Figure 3.11). For the GANdex Lookup task, we wanted to find an algorithmic approach capable of matching drawings with corrective vectors from the GANdex cache while considering the drawing styles' variation.

We explored several scores for algorithmic image-matching, such as using VGG-16 [59] features and the Learned Perceptual Image Patch Similarity (LPIPS) metric [61]. VGG-16 is a neural network built for the ImageNet Challenge 2014 [62] named after its creators, Visual Geometry Group [63], and the 16 layers it contains. Even though VGG-16 was initially built for

ImageNet's classification task, it has been used in recent years for image searching [64][65]. Image searching is executed using the weights produced by VGG-16's second-to-last layer as feature vectors followed by clustering algorithms to group images that look similar to one another. The Learned Perceptual Image Patch Similarity (LPIPS) metric uses perceptual loss [66] and is a metric that quantifies the similarity between two images.

To assess each metric, we aimed to visually evaluate how often they can match a real image from the RDD with an image from GANdex that is a fake sharing close structural features to the real image. We arbitrarily selected fifty unique images from the original dataset and used the N corrective vectors generated for GANdex discussed in Section 3.3. We implemented a script that uses each of the three metrics to match every real image with the entire set of GANdex reference images and output a file with an image tuple with a real image and the metric's best guess the most visually similar fake.

VGG-16 was the fastest to execute at 3ms per image to output all fifty outputs and match 35 out of the 50 images properly. Our custom classifier struggled and took 4ms per image and was only able to match 15 images. LPIPS was much slower than the other two at over 4 seconds per image but matched 47 images.

VGG-16's speed was preferred, but we also wanted to benefit from the higher performance of LPIPS. We looked into VGG-16's top five candidates for each of the 50 images we used in our testing and found that most of them included the image output from LPIPS, which was not VGG's top candidate. Therefore, we chose to combine the two metrics by first using VGG to look up K candidates and then use LPIPS to select the best candidate match. After testing multiple values for K, we found the value 20 to give the best results. The resulting hybrid approach matched 43 out of 50 images at a total cost of 5ms per image.

Figure 3.12: A hand-drawn image with mismatched candidates from VGG-16

While the hybrid approach was a notable improvement, we still noticed that VGG-16 struggled with small features such as the image stroke and noisy strokes that often led it to yield candidates that do not belong to the same class. This variance was even more evident when hand-drawn images are used instead of the real images (see Figure 3.12).

Figure 3.13 - Left to right: Original Input, Pixelated Input, and Skeleton of Input

Upon evaluating more examples of hand-drawn images, minor imperfections in the outlines of the images and image strokes that are large or small were evident in most mismatched images. As a result, we chose to use normalized versions of the input images during the matching process. We use image pixelation to remove minor imperfections and skeletonization [67] (see Figure 3.13). The pixelated input would be compared with the pixelated versions of all the lookup images; this resulted in improved generalizability for the lookup process, which no longer depended on the minor detail. The skeleton of the input consistently produced a version of the input image with the same stroke width, and this resulted in lookups that are invariant of the stroke of the input's outline.

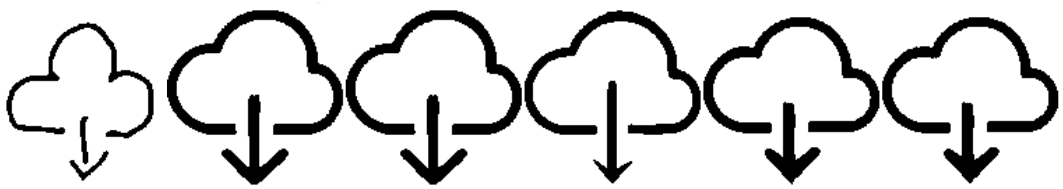

Figure 3.14: Matched cloud icon that makes use of pixelization and skeletonization

After incorporating both transformations into our scoring function, we noticed that the algorithm yielded results more in line with the input (see Figure 3.14). As a result, we incorporated these improvements into the GANdex lookup process we discuss in this section.

Figure 3.15 - Input image, a pixelated input image, a skeletonized input image, and GANdex corrective vector of the input image

Figure 3.16 - GANdex lookup candidates where M is set to 20. The first row is for the SMIL collection, the second row is for the SMPL collection, the third is for the SMCL collection, and the fourth is the SBCL collection.

The GANdex lookup process is responsible for retrieving the best candidate used in the image correction process. Finding a close structural approximation of the input image in the cache maximizes the retention of intentional user features in the image corrected output.

$$distance\left(v_1, v_2\right) = 1 - \frac{v_1 \cdot v_2}{||v_1||\,||v_2||} \quad (3.1)$$

The lookup process begins with generating the pixelated and skeletonized [67] versions of the input image (see Figure 3.15). VGG feature vectors of the input and pixelated images are then generated. We calculate the least cosine distance [68] (see Equation 3.1) between the input features and each of the vectors in the GANdex cache, producing the input lookup collection **IL**. Similarly, the pixelated lookup collection **PL** is produced using the pixelated features and the pixelated vectors in the GANdex cache. We yield the **M** vectors with the smallest distance values for each of the two sets, where **M** is the number of cache candidates we would consider for each collection. As a result, we produce the smallest distance M input lookup collection **SMIL** and the smallest distance M pixelated lookup collection **SMPL**.

While the collections **SMIL** and **SMPL** yielded close candidates that resemble the input, the superiority of one over the other relied on the input's nature. We observed that the pixelated lookup was superior for inputs with detailed inputs, while the raw lookup was superior with inputs that are mainly composed of abstract features. With this observation, we pursued an objective solution that can elect a corrective structure from either set. We would not expect the user to do the selection themselves.

Based on the assessment we discussed earlier in this section, we chose to use the LPIPS score with skeletonization to find the best candidates from collections **SMIL** and **SMPL**. We calculated the LPIPS distances between the skeletonized versions of the input and the images from collections **SMIL** and **SMPL**; this produced the combined lookup collection **CL**. Finally, we yield the M smallest distances for the combined lookup, producing the smallest M combined lookup collection **SMCL**.

While **SMCL** produced more consistent and reliable candidates than **SMIL** and **SMPL**, we observed that the best candidate vector would occasionally be located at the second or third

element of the collection. Since **SMCL** performed well, we could ask for user input to select from the top three candidates before producing. However, we wanted to explore one final step that may be capable of sorting the top candidates correctly, enabling the confident selection of the top candidate more confidently. As a result, we chose to calculate the LPIPS distances between the input image and the unmodified images from the **SMCL**, producing the best candidate lookup collection **BCL**. Finally, we yield the sorted best candidate lookup collection **SBCL**. We show the **SMIL**, **SMPL**, **SMCL**, and **SBCL** collections in Figure 3.16 for the input shown in Figure 3.15.

The vector with the least distance from the **SBCL** collection is selected as the corrective vector of the input image (see Figure 3.15), which is returned by GANdex and used in the Projection set discussed in Section 3.5.5.

While selecting the vector with the least distance worked for most of our study, it exhibited false positives with drawings that resemble more than one icon type from the original dataset. For example, a hand-drawing on Pointer icons was associated with a corrective structure of a Volume icon. We manually selected between the top five candidates with the least distances to circumvent this shortcoming for the study. In future work, a classifier can be used to automate this process.

3.5. Projection

3.5.1. Conventional StyleGAN Projection

StyleGAN2-ADA's implementation [13] includes a module named projector. Given an image and the trained StyleGAN2-ADA model, the projector does the following:

1. Sample N random vectors from a normal distribution in the intermediate latent space
2. Calculate an intermediate latent vector from the component-wise average of the N random vectors
3. The projector then traverses the latent space, starting from the average vector. It minimizes the LPIPS loss between the input image and the image generated from the vector retrieved at every projection step. At the end of every step, the neighboring vector with the lower loss is chosen.
4. The lower the loss value, the closer the generated image is to the input image.

The projection would lead to a local minimum; the local minimum could potentially be an image that resembles the input image. Examples are provided from StyleGAN2-ADA. They project real faces on-to StyleGAN2-ADA models trained on the FFHQ dataset [44] to find an intermediate latent space vector that generates an image that resembles them. Previous works [28] enabled manipulating the image by changing isolated features based on pre-trained vectors. For a StyleGAN2-ADA model trained on faces, the authors were able to apply transformations such as manipulating the face's age while keeping all other features constant.

3.5.2. Projection Improvements using GANdex

Although the conventional projection process can resolve a latent vector that corresponds to an image similar to the input image, it begins its search from a random starting point. The average vector generated from the N random intermediate vectors (as described in Section 3.3) is arbitrary, and the projection process explores nearby vectors until it reaches a local minimum. The main drawback of this approach is that the projection could halt at a local minimum where it would not be able to optimize further; this may lead to a vector that does not resemble the input image. Another drawback is that even if a neighboring local minimum exists that produces a

close approximation to the input image, the number of projection iterations could, in some cases, be significant depending on how far the average vector is.

3.5.3. Projection Improvements using GANdex

BrandGAN relies heavily on the final projected vector to retain the input features for its image-corrected output. We leveraged our novel GANdex indexing technique comparable to a literature book's index to solve this problem.

When a reader is looking for a specific piece of information, they look through the book's index to find the closest subject related to what they are looking for, go to the referred page and then read the neighboring pages to locate their desired content [69]. Without the index, the reader would need to explore the book in arbitrary ways. They could read the book from the beginning until the desired information is found or jump to a random page and follow the content to find what is closest to what they desire. With either technique, they may find content related to what they are looking for, but it is impossible to tell if there is more desired content without reading the entire book. Moreover, there are no guarantees that they would find their topic of interest quickly. As a result, book traversal without an index is comparable to traversing the GAN latent space from an arbitrary location that is not influenced by the search query.

GANdex is initialized with index vectors that produce diverse images representing corrective structures learned from the original dataset. To produce such an index, we aimed to generate a large set of random input latent vectors **z** sampled from a normal distribution (we discussed this process in Section 3.3). The rationale is that the random vectors would represent fake images where each of them has a unique combination of features from the images in the dataset. The

more significant the variation, the bigger the index, the more likely it is to have an image resembling an input image.

BrandGAN uses GANdex to find the vector that resembles the input image; it is used as the initial vector for the projection process. Since the selected vector is optimized for the input image, the initial similarity loss calculated during the projection process is lower than expected with the conventional projection process that relies on an initial random average vector.

3.5.4. Alignment

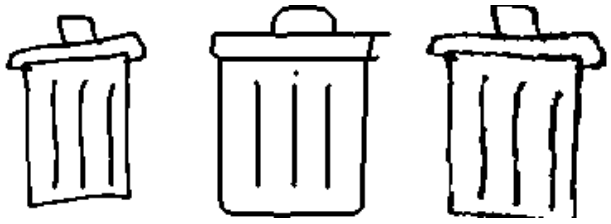

Figure 3.17 - Left to right: Drawing, GANdex corrective image, Drawing aligned with the corrective image.

While the image generated from the corrective vector obtained from GANdex is visually similar to the input image, its dimension and orientation can differ (see Figure 3.17). Dimension and orientation may force the projection process to halt on a local-minima that is incapable of visually representing the input image.

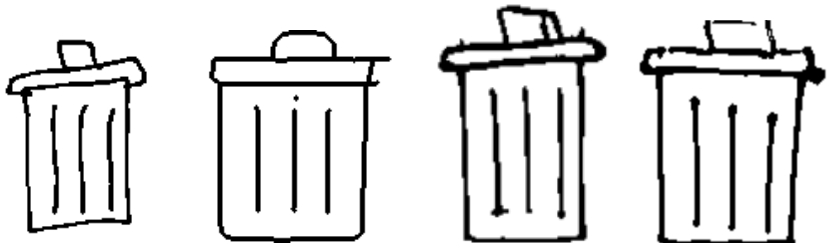

Figure 3.18 - Left to right: Drawing, GANdex corrective image, Projected Image w/o alignment, Projected Image with alignment.

To solve this problem, we leveraged image alignment before projection (see Figure 3.17). To align the input and the corrective images, we transform them into outlines using scikit-learn [67]

then compute a warping matrix using CV2 [70]. The warping matrix is applied using CV2's Warp Perspective [60] to adjust the input to align with the corrective image. As a result, the aligned image would reduce the number of iterations for the projection and minimizes the distance between the projected and corrective vectors, reducing the amount of lost detail from the original drawing (see Figure 3.18).

3.5.5. BrandGAN Projection

This section explains BrandGAN's projection process, which projects the input image and yields a vector that generates a close approximation of it. Projection is required for the Interpolation step that we discuss in Section 3.6.

Figure 3.19: A hand-drawn image of a miniature version of the trash icon. The image on the right is input to BrandGAN, while the image on the right is the cropped version of the input.

Projection starts with an input image where the drawing may be located in an arbitrary location and surrounded by blank space. The input is cropped and centered, ensuring consistent behavior for any input image BrandGAN may receive (see Figure 3.19).

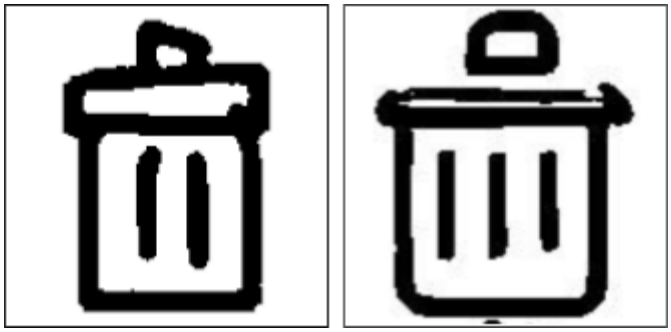

Figure 3.20: The image on the left is the cropped input, and the image on the right is the result of GANdex lookup using the cropped image.

The cropped image input is then used in the GANdex lookup process discussed in Section 3.4. As a result, GANdex would provide a corrective vector that visually simulates the input (see Figure 3.20). The corrective image may have some visual artifacts; they do not affect the BrandGAN output as they disappear during the averaging process. For example, the disconnected lid of the corrective vector in Figure 3.20 would be reconstructed when combined with the projected vector when the features representing the lid of both vectors are combined.

Figure 3.21: The images are the cropped input, corrective image from GANdex, and the result of aligning the input image with the corrective image. The alignment is calculated using the skeletonized versions of the input and corrective images (represented on each image's top-left area).

Even an optimal corrective image may have a different scale than the input (see Figure 3.20. We found that misalignment between the initial projection vector and the input image could result in

a suboptimal projection that does not replicate the input's features during our research.

We experimented with two alignment methods: resizing the width and height and using OpenCV's Warp Perspective [60] algorithm. Resizing the width and height produced inconsistent results. We analyzed the results and found that this was due to all of the image's features being scaled proportionally relative to each other; this is problematic because not all of the input's features need to be scaled similarly to overlap the corrective image. CV2's Warp Perspective algorithm uses Perspective Transformation [71], an algorithm used to align two 2D images with one another in the 3D space when taken at different angles. Perspective Transformation overcomes the limitations of the first approach by allowing for the input image to be scaled in 3D; this higher-dimensional alignment allows for portions of the image to be scaled independently of one another, leading to improved preservation of the input image's relative proportions. As a result, we decided to incorporate Warp Perspective into our pipeline.

Warp perspective had failure cases where the two images' outline stroke was different; this led to exaggerated transformations leading to unusable results in the transformation. To overcome this issue, we evaluated the Perspective Transformation on the skeletonized versions of the input and output to normalize their outline strokes before applying the resulting transformation on the input. This approach resolved outline-related inconsistencies. The final alignment steps are shown in Figure 3.21

Figure 3.22: The three projection candidates. They optimize on LPIPS loss, VGG-16 MSE, and pixel-level distance.

The aligned input image is projected on StyleGAN2-ADA starting from the corrective vector from GANdex. When we evaluated the conventional projection approach discussed in Section 3.5.1, we occasionally found flawed projections. Upon further analysis, we found that the conventional approach occasionally resulted in flawed projections due to its reliance on LPIPS distance [62] only. While the LPIPS metric often produced good results, it has the drawback of being an unsupervised objective approach. Therefore, LPIPS alone is a single point of failure where the result of projection becomes unusable if the score fails to converge on the given input.

BrandGAN relies on the result of the projection process as it is where the features from the original input image are captured. We aimed to overcome the issue of having one heuristic as a single point of failure by leveraging an ensemble of heuristics instead where each of them produces their best guess of a final projected image.

$$v_p = \frac{1}{n}\sum_{i=1}^{n}\left(v_{ri} - v_{ti}\right)^2 \qquad (3.2)$$

VGG-16, as discussed in Section 3.3, produced favorable results for image search showing its effectiveness as a similarity measure. As a result, we chose to use it as our second metric during

projection (denoted as v_p, as shown in Equation 3.2). The score is defined as the mean square error between the VGG features of the input image (denoted as v_r) and the VGG features of the projected image at the current projection step t(denoted as v_t). All $v$vectors have a size n equals to the total number of elements in the VGG-16 feature vector.

$$i_p = \frac{1}{m}\sum_{j=1}^{m}\left(i_{rj} - i_{tj}\right)^2 \qquad (3.3)$$

Since we needed to capture as many of the input image's visual features, we introduced a third score that measures the pixel-level distance between the input and projected images (denoted as i_p, as shown in Equation 3.3). It is defined as the mean square error between the pixel matrices of the corrective i_r and the projected image i_t at step t. Both images have the exact dimensions where m denotes the total number of pixels in each image.

BrandGAN projection is executed over k steps, where k is a hyperparameter that can control the projection process's intensity. In every step t where $0 \leq t \leq k$, all three heuristics are computed for the projected image. Every heuristic at every step saves the projected vector if it has a value smaller than previous steps. As a result, at projection step k, BrandGAN would have a projection vector candidate for every heuristic used (see Figure 3.22).

$$p_a = \frac{(p_1 + p_2 + p_3)}{3} \qquad (3.4)$$

$$\hat{w} = 1 - ||\{l_1, l_2, l_3\}|| \qquad (3.5)$$

BrandGAN requires a final vector that is combined with the corrective vector to produce the image corrected result. We initially averaged all three vectors resulting in p_a(as shown in Equation 3.4) to calculate the final vector. If any of the three metrics produced a suboptimal image compared to the others, it would carry its features to the final vector with averaging. This behavior produced artifacts even if the other metrics produced good projection results. We decided to use a weighted average instead to avoid this issue.

The weighted average's purpose is to give larger weights to the candidates that look more similar to the input image. We used the LPIPS distance between every candidate and the input image to calculate the LPIPS candidate distance-vector $\{l_1, l_2, l_3\}$. We then normalize the vector to get relative weights that favor the heights distance measure. Finally, since we need to favor the weights of the lowest distances, we subtract every element in the vector from 1 to produce the final weights vector $\hat{w}$(as shown in Equation 3.5).

$$p_f = \frac{\left(\hat{w}_1 p_1 + \hat{w}_2 p_2 + \hat{w}_3 p_3\right)}{3} \qquad (3.6)$$

We calculator the final projection vector p_f(as shown in Equation 3.6) using the weights vector $\hat{w}$. Compared to the average vector p_a, where all projections are treated the same, p_f penalized heuristics that misrepresented the input image. As a result, p_f was able to produce a more optimal representation of the input image even when one of the heuristics failed to produce an optimal projection vector candidate.

The final projection vector p_f is BrandGAN's most optimal guess of a vector that can reproduce the original input. The p_f vector is passed to the BrandGAN Interpolation step discussed in

Section 3.6 to generate the final image-corrected output.

3.6. Interpolation

Figure 3.23 - Left to right: Vector A's Image, Vector B's Image

*Figure 3.24 - Left to right: Vector (A*0.9 + B*0.1), Vector(0.5*(A + B)), Vector (A*0.1 + B*0.9)*

For any vector in the intermediate latent space **w**, it is possible to use vector arithmetics to manipulate images differently. For example, averaging two distinct vectors produces a vector that can generate an image with features that interpolate each of the two vectors' features. If the two vectors have a weighted average, the vector with the more significant weight will influence the final image's appearance (see Figure 3.24).

After the projection step is complete, BrandGAN would have produced the following:

1. Input Image
2. Corrective Vector
3. Aligned Input Image

4. Final Projected Vector

BrandGAN interpolation combines the corrective and projected vectors (see Figure 3.23) to produce an output vector that inherits features from both. The intended behavior is to generate a vector that uses the corrective vector's features to adjust the input image's imperfections to look more similar to the corrective structure based on the RDD. The interpolated vector would therefore be the image-corrected version of the input image. The weights used to combine the corrective and projected vectors produce various images depending on the weights used; selecting weights is a subjective process depending on the degree of image correction needed on the input image (see Figure 3.24). In our study, we used the average between the corrective and projected vectors as our output. Alternatively, BrandGAN can include a slider control that a user can use to adjust the image correction intensity with a live preview before selecting the combination that best fits their case.

3.7. BrandGAN Sample Outputs

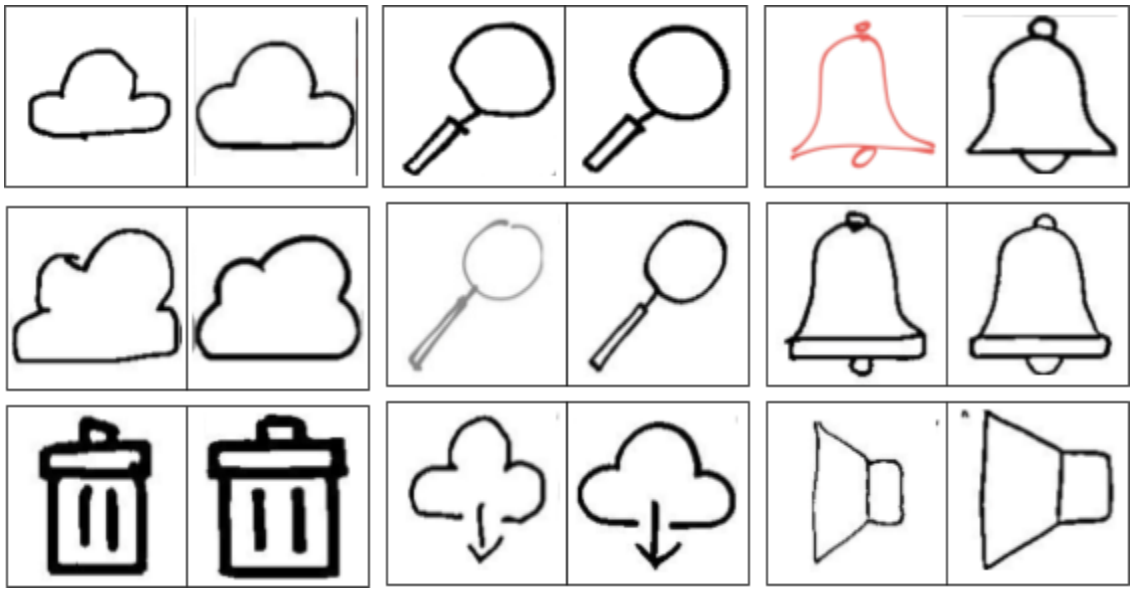

Figure 3.25: Sample input/output pairs from the data collected during our study (discussed in Chapter 4)

The results of BrandGAN, discussed in further detail in Chapter 4, were able to generalize on different icon types. Figure 3.25 shows input/output pairs. The pairs show how detail was preserved, including the stroke of the outlines in most cases.

Chapter 4 - Evaluation

In this section, we present the evaluation of BrandGAN using objective and subjective measures based on data collected from a study with fifteen participants.

In the following subsections, we describe our study and how data was collected from the participants. Afterward, we present the subjective and objective evaluations derived from the study's data.

4.1. The BrandGAN Study

We ran a study with 15 participants. The study's main goal was to evaluate the participants' ability to identify their images in a Blinded Experiment [72] between a BrandGAN output of their image and a control image drawn by the study's investigator. If they could identify their image from the control, this would indicate that they could recognize their features retained in the image-corrected version of their drawings.

The study's recruitment followed a snowball strategy [73] where participants received invites and contacted the investigator when they chose to participate. After the participants signed the consent forms, we adopted a two-phase approach for every participant. In the first phase, the participant provided their drawings. In the second phase, they are asked to evaluate the image-corrected versions of the drawings they provided in the first phase in a survey.

4.1.1. Phase 1 - Drawing Images

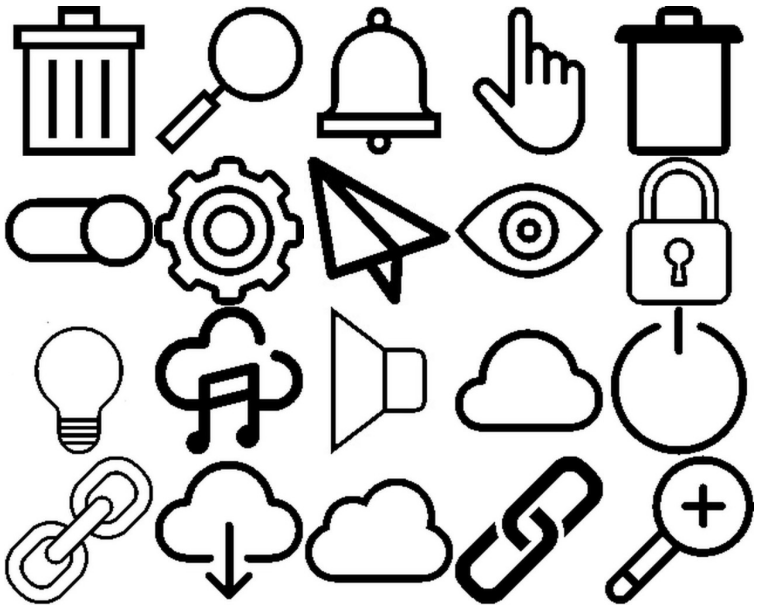

Figure 4.1 - 20 images presented during the study.

BrandGAN is limited to the reference design dataset it was trained on; therefore, the study needs to be restricted to the RDD's domain. Since the participants do not have prior knowledge of the domain, we provided them with a 5x4 grid of 20 images (see Figure 4.1) from the RDD. We then asked them to draw any five images of their choosing using their phone's touch-screen displays.

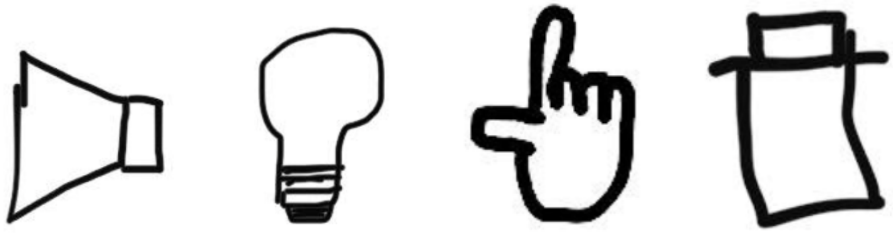

Figure 4.2 - Four drawings collected from participants during the study

After providing their drawings (see Figure 4.2), the participants were informed of a follow-up date when the second phase of the study would occur.

4.1.2. Phase 2 - Evaluating BrandGAN

The images from the participants are transformed through BrandGAN. In this phase, we aimed to carry out Blinded Experiment [72], where the participants are asked to classify which BrandGAN output image is theirs when presented with a control. The control images were the image-corrected versions of images drawn by the investigator based on the 20 images used during the study (see Figure 4.1).

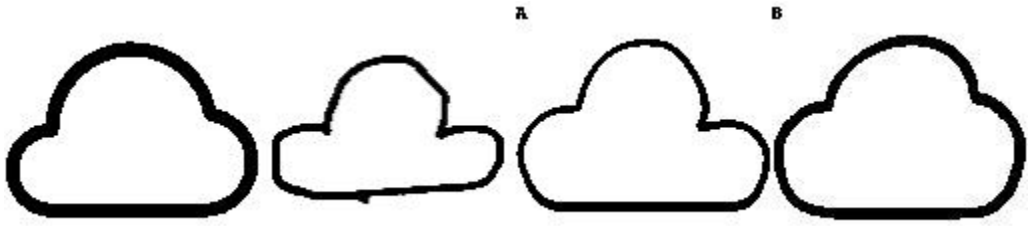

Figure 4.3 - Blinded Experiment presented to the participants during the study. The first image is an icon from the grid used as a reference for the participant's drawing. The second image is the participant's drawing. The third and fourth images are BrandGAN outputs where one is a control, and the other image is based on the participant's drawing. The third and fourth images are labeled with A and B and are randomly shuffled.

For each of the five images, they received a row of images (see Figure 4.3) with the following:

1. The reference image they used to base their drawing on
2. Their provided hand-drawn image
3. Image A
4. Image B

Images A and B are the user's BrandGAN output and the control image from the investigator. The order of A and B are shuffled at random and are only known to the investigator.

The participants are then asked the following survey questions:

1. Which picture, "A" or "B", is based on the image you drew? They answer with A or B.

2. “The image that I selected retains detail from my original drawing.” They answer with how much they agree with the statement.
3. “The image I selected looks more "professional-looking" compared to my original drawing.” They answer with how much they agree with the statement.
4. “I would more likely use the image I selected over my original drawing.” They answer with how much they agree with the statement.

Questions 2 to 4 were answered with a numerical value between 1 and 5. 1 indicates that they strongly disagree, while five indicates that they strongly agree. 3 is the midpoint and indicates that they are neutral. We refer to the scores provided as Opinion Scores.

4.2. Subjective Evaluation

In this section, we evaluate the data gathered during the study. We first investigate the results from the blinded experiment, then we analyze the survey results.

4.2.1. Blinded Test Results

Table 4.1. Correctly Classified Images.

Measure	Mean (0-5)	Mode (0-5)	Median (0-5)	95 Percentiles (0-5)	Variance	Standard Deviation
# of Participants	4.33	4	4	5	0.67	0.82

The numeric value ranges from 0 to 5 and indicates how many images were classified correctly.

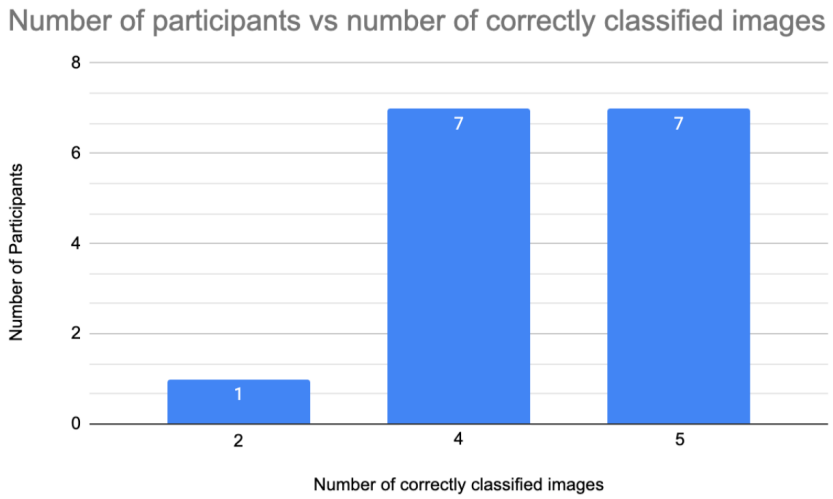

Figure 4.4: One participant classified only 2 of their drawings correctly, seven classified 4, and seven classified 5.

In Table 4.1, we report the results of the blinded experiment from all fifteen participants. The scale of the classification result is zero to five. Zero indicates that none of the images were correctly classified, while five indicates that all images were correctly classified. The average participant was able to correctly identify 4.33 out of 5 images as their own. The median value and mode value of 4 indicate that a significant amount of the participants could not guess one of their images. Figure 4.4 further confirms this observation, where it shows that the number of participants who could not guess one of their images is the same as the ones that got all of them correctly. One participant guessed only two images correctly; their images are included in Figure 4.5 and will be discussed later in this section.

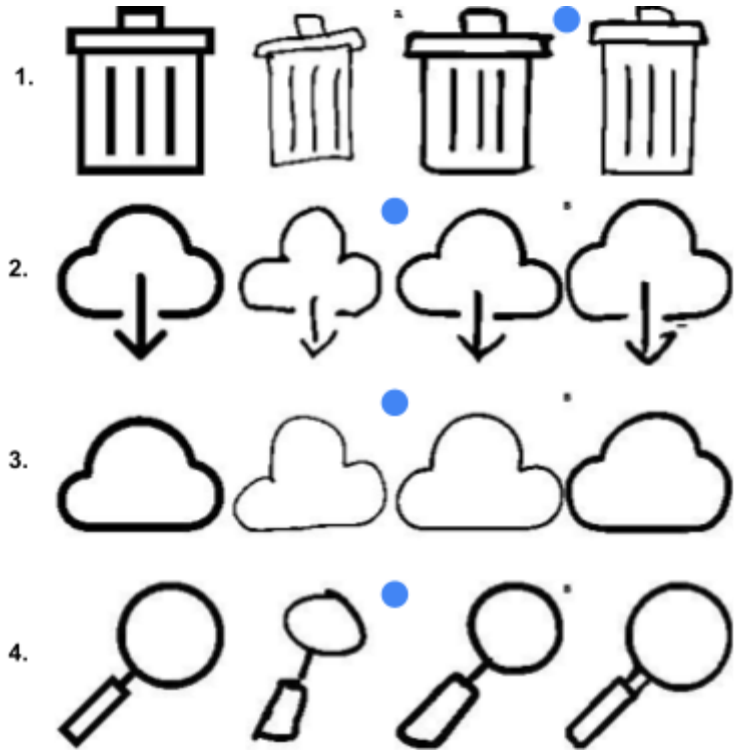

Figure 4.5: Samples of correctly classified rows. The blue circles represent the BrandGAN output based on the participant's drawing.

Figure 4.5 shows samples of images that were correctly classified. Row 1's BrandGAN output was able to maintain the orientation of the bin cover and the height and stroke style of the bin. Row 2's BrandGAN output preserved the dimensions of the cloud icon and the location of the arrow's pointer. Row 3's BrandGAN output replicated the cloud dimensions and the relative parts of the input image's angular features. Finally, Row 4's BrandGAN output shows how the magnifying glass handle was replicated from the input.

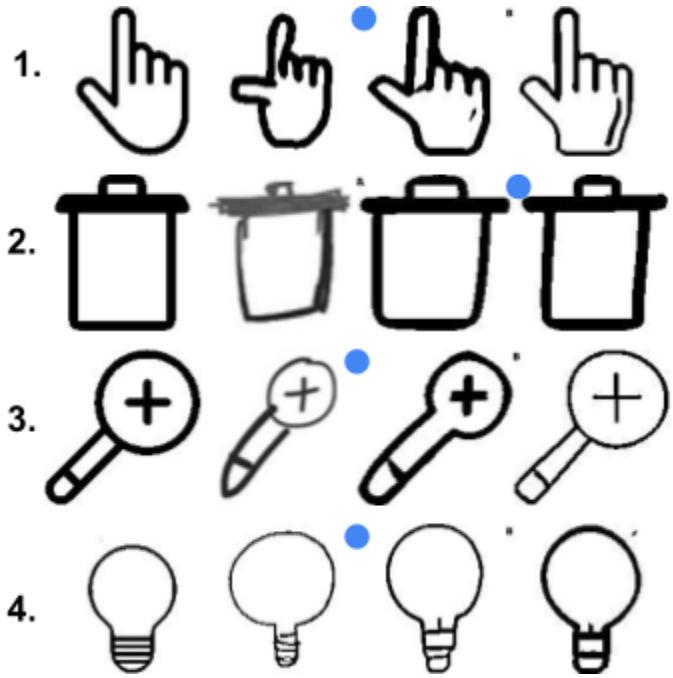

Figure 4.6: Four blinded tests from our study where participants could not identify their image correctly. The blue circle was added during the writing of the thesis, and it overlays the image based on the participant's drawing.

We observed that some of the misclassified images had a strong resemblance to the control image, making them hard to distinguish. Other misclassifications were a result of image detail that BrandGAN was not able to replicate.

Samples of misclassified images are shown in Figure 4.6. Row 1 shows a failure in representing the thumb of the training which only manifested in a small line on the index finger, other than the outline's stroke. There are no other distinctive features between the two images. Row 2's outputs only differ with their dimensions, making it difficult to differentiate between the two. Row 3's output bears a solid resemblance to the drawing, but control is a better fit than the image from the original dataset; we hypothesize that the participant selected B for that reason. While producing the results for Row 4, we noticed that the bulb drawing's wide circular portion misled BrandGAN into interpreting a corrective image that generates a vertical magnifying glass

instead of a bulb. The misclassified corrective image resulted in an output image that is a combination of magnifying glass and bulb features, leading to an unrecognizable logo.

4.2.2. Survey Results

Table 4.2 summarizes the opinion scores for the correctly classified images in terms of Detail, Professionalism, and Preference. These scores were collected from questions 2, 3, and 4, respectively. We introduced these questions in Section 4.1.2.

Table 4.2. Opinion Score of Correctly Classified Images.

Measure	Mean Opinion Score (1-5)	Mode (1-5)	Median (1-5)	95 Percentiles (1-5)
Detail	4.32	5	4	5
Professionalism	4.25	5	4	5
Preference	4.57	5	5	5

The opinion scores of correctly classified images from the survey. Scores range from 1 to 5.

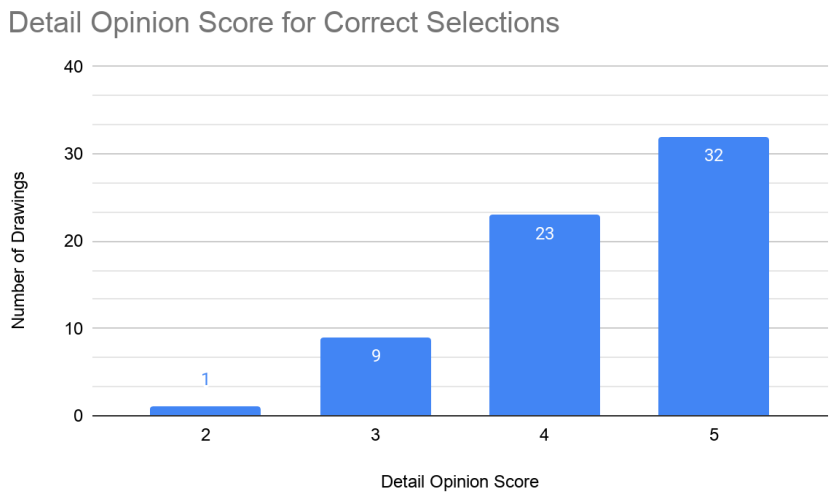

Figure 4.7: A histogram that reflects the number of participants for every opinion score recorded for the Detail survey question.

The mean for the Detail score is 4.32, while the Mode and Median are five and four, respectively. Figure 4.7 further explains the Mode and Median. The majority score was five,

with 32 images (42.6% of the study's images) being given that score. The second most frequent score was four, with 23 images (30.6% of the study's images). Despite the prominence of scores five and four, we noticed 9 images scored poorly at three while 1 image was scored at two. We explore the two scoring extremes below.

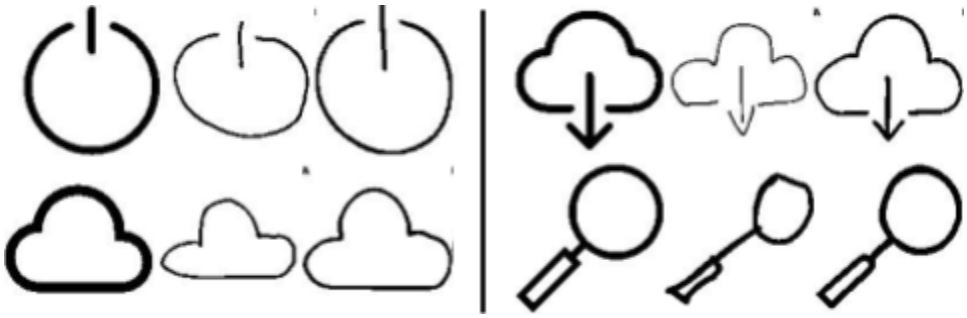

Figure 4.8: The two examples on the left scored 5 for detail, while the two examples on the right scored 3 and 2, respectively.

We present examples of both extremes in Figure 4.8. We noticed that the two examples that scored five retained a significant amount of detail. The power icon retained the relative length of the line while maintaining its skewness. Furthermore, the circular portion of the icon inherited a similar bend in its underside. The cloud icon replicated every one of the three oval areas from the original drawing. It made them more circular and less pointy while maintaining their relative positions and proportions. As for the failure cases, we notice that the BrandGAN output is a close approximation but failed to represent key features of the drawings. The cloud, which scored three on detail, has its right side oval pointing downwards while the original drawing pointed upwards. As for the magnifying glass, that scored a two, superimposed the handle to the middle, and did not reflect the length of the magnifier's neck in the BrandGAN output.

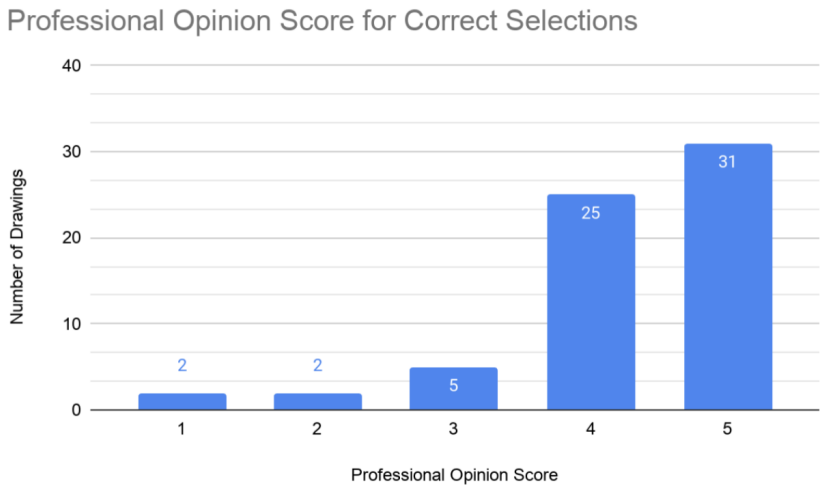

Figure 4.9: A histogram that reflects the number of participants for every opinion score recorded for the Professional survey question.

The mean for the Professional score is 4.25, while the Mode and Median are 5 and 4, respectively. Figure 4.9 further explains the Mode and Median. The majority score was five, with 31 images (41.3% of the study's images) being given that score. The second most frequent score was four, with 25 images (33.3% of the study's images) being given that score. As for the rest of the images, 5 images scored three, 2 scored two, and 1 scored one. We explore the two scoring extremes below.

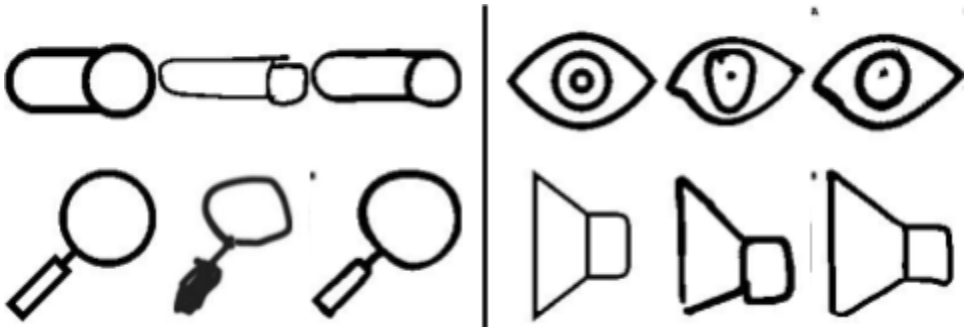

Figure 4.10: The two examples on the left scored 5 for Professionalism, while the two examples on the right scored 2 and 1, respectively.

We present examples of both extremes in Figure 4.10. This Professional score differs from Detail as it is more concerned with the subjective observation that the produced image is one the participant perceives as fitting within a professional context. The toggle button on the top right shows how BrandGAN reproduces the image with more circular edges and continuous lines. The magnifying glass reproduced the proportions of the handle and the glass but failed to include the filled handle color. We see that, in both cases, the output images maintained symmetry and similarity of their inputs. The view icon, which scored a two, shows how BrandGAN could resolve the proportions of the eye icon and the dot for the iris. However, we noticed that the dot might have been the reason behind the low score since a professional image would have more sophisticated details such as a circular iris instead. The Sound icon, which scored a one, demonstrates the retention of the proportions and angles of the two shapes but ultimately retained the asymmetrical shape found in the input image. We hypothesize that the participants may have made the reasonable assumption that BrandGAN would reproduce better versions of those finer details from their drawings for the failure cases.

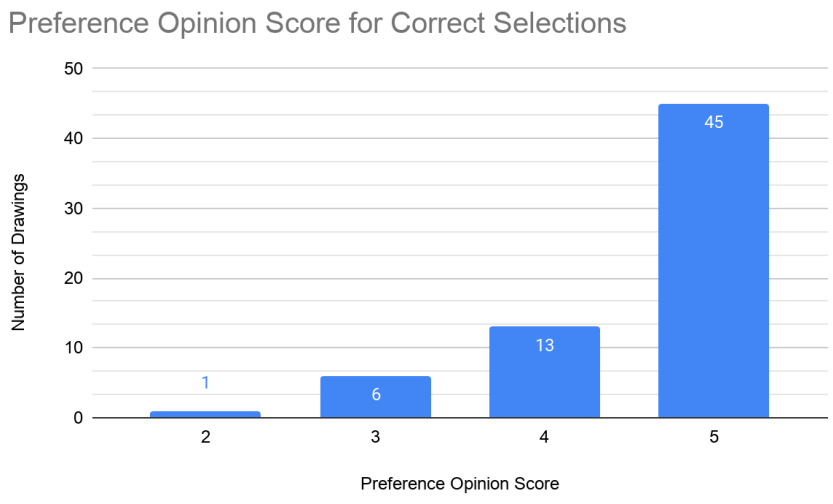

Figure 4.11: A histogram that reflects the number of participants for every opinion score recorded for the Preference survey question.

The mean for the Preference score is 4.57, while the Mode and Median are both five. Figure 4.11 reflected those metrics as 45 images scored a five (60% of the study's images). Despite the high metrics, we still saw that 6 images were scored a three while 1 was scored a two. We explore both extremes below.

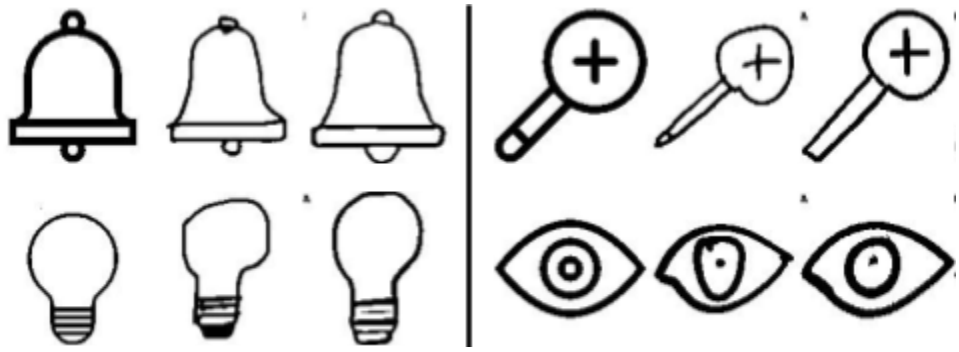

Figure 4.12: The two examples on the left scored 5 for Preference while the two examples on the right scored 3 and 2, respectively.

Preference is more difficult to explain without more context from the participants. We attempt to hypothesize on the results using the examples in Figure 4.12. For the bell and the bulb icons, their BrandGAN outputs retain many details compared to the magnifying glass and view icons on the right. We hypothesize that those examples scored a five because they could correct the finer errors in the images while retaining most of the broader details. As a result, since all else is equal, the participants had a more substantial bias towards selecting their image's BrandGAN output. The examples on the right had scores of three for the magnifying glass and two for the view icon. We hypothesize that even though they could identify their features, the BrandGAN output may not have provided them with enough value to prefer it over their drawing.

In the upcoming section, we present our objective evaluation of the study's data.

4.3. Objective Evaluation

We pursued an objective evaluation using quantitative measures to assess our algorithm's effectiveness on the image correction task. We aimed to find an algorithmic score capable of producing objective heuristics of our data that we can use during our evaluation.

BrandGAN handles three sets of data: the user drawings, BrandGAN's image corrected versions of the drawings, and the reference design dataset used to train the underlying StyleGAN2-ADA model. We used the images collected during the study for this evaluation.

The following are the definitions of the three image sets:

1. User Drawings (UD): A set containing 75 hand-drawn images from our study (5 from each of the 15 participants)
2. BrandGAN Image Corrected (IC): A set containing 75 images that are the result of running BrandGAN on the set Drawings
3. Reference Design Dataset (RDD): The reference design dataset used for training the core StyleGAN2-ADA model

BrandGAN's output **IC** is expected to resemble the reference design dataset **RDD** while retaining features from the original drawings **UD**. Therefore, a score measuring the similarity between image sets should reflect that the distance between **UD** and **RD** is greater than the distance between **IC** and **RD**. We used the Kernel Inception Distance (KID) [74] and Fréchet Inception Distance (FID) [75] as our objective scores. They are used in previous work [76][77][78] to evaluate the quality of images generated by GANs by computing the similarity metric between the real and fake images using their InceptionV3 [79] features.

Kernel Inception Distance (KID) is defined as:

$$k(x, y) = \left(\frac{1}{d} x^{\intercal} y + 1\right)^3 \qquad (4.1)$$

The KID score is computed using Equation 4.1, a polynomial kernel [80] where x and y are Inception representations of two sets of images. An Inception representation is a feature vector computed from an image using an InceptionV3 model [79] trained on the ImageNet dataset. The feature vector represents the activations of the second-to-last layer of the Inception model when an image is provided as input.

Fréchet Inception Distance (FID) is defined as:

$$FID = \left|\mu - \mu_w\right|^2 + tr\left(\Sigma + \Sigma_w - 2\left(\Sigma\Sigma_w\right)^{1/2}\right) \qquad (4.2)$$

The FID score is computed as the Wasserstein distance [81], as shown in Equation 4.2, between the Inception representations of both image sets. The Wasserstein distance would therefore represent the distance between the feature vectors of the two image sets.

We were inspired by the StyleGAN2-ADA paper [25] to use KID as it is more reliable for smaller datasets compared to FID. We chose to use FID to provide us with more metrics to validate our results, despite not being as sensitive as KID for small datasets.

We calculated the following distances for every metric:

1. **UD to IC**: The distance between the user drawings and their image corrected outputs
2. **IC to RDD**: The distance between the image corrected images and reference dataset
3. **UD to RDD**: The distance between the user drawings and the reference dataset

Table 4.3. Study Image Corrected Evaluation Scores.

Distance	FID	KID
UD to IC	169.25	0.036
UD to RDD	193.22	0.048
IC to RDD	161.31	0.026

Table 4.3 summarizes the FID and KID scores for all three distances.

The IC set's image features are derived from combining the UD and RDD sets' features, making IC an independent set. The non-zero values for **UD to IC** and **IC to RDD** for both metrics show that IC does not belong to either one, showing that BrandGAN's outputs belong to an independent image distribution.

The set IC is derived from UD and transformed by BrandGAN to look more similar to the RDD set. Therefore, the distance between **UD to RDD** is expected to be greater than the distance between **IC to RDD**. Both metrics confirmed the expectations. The FID score dropped from 193.22 to 161.31, while the KID score dropped from 0.048 to 0.026. The drop of both metrics indicates that the features of the IC set are closer to RDD's features than UD, showing that BrandGAN was able to translate the drawings into a domain that is closer to the reference design dataset.

Chapter 5 - Conclusion and Future Work

In this chapter, we summarize our work in a conclusion and propose future work.

5.1. Conclusion

The goal of the thesis was to explore the problem of structural image correction of hand-drawn images and find an approach that can tackle it in an unsupervised manner. This problem is complex due to the high variability of drawing styles between individuals alongside the vast concepts they would materialize into their drawings.

We make use of StyleGAN2-ADA's ability to generate fakes, a GAN indexing approach that captures corrective structures, a modified projection of images on the StyleGAN model, and an interpolation process that blends the projection and the corrective vectors producing an image corrected output. We summarize our contributions as follows:

1. We introduced GANdex, a novel GAN indexing technique used to find corrective structures efficiently from images generated using a StyleGAN2-ADA model.
2. We implement the BrandGAN framework, which aims to solve the problem of image correction. BrandGAN uses GANdex, a modified version of StyleGAN2-ADA's projection process, and a vector interpolation technique. We show that it could image correct structures of hand-drawn UI icons in an unsupervised manner.
3. We evaluate BrandGAN using objective and subjective measures on data collection through a fifteen-participant study.

Finally, to evaluate BrandGAN's image-correction ability, we ran a fifteen-participant study where they underwent a blinded experiment and were asked to fill a survey. The experiment

showed that the average participant correctly identified 4.33 out of 5 images as their own when presented with their drawing's BrandGAN output and a similar image as a control. The survey collected three opinion scores ranging from one or "strongly disagree" to five or "strongly agree" for each participant's images. The average scores were 4.32 for retention of detail, 4.25 for the output's professionalism, and 4.57 for their preference of BrandGAN's output image over their own. We also use objective scores on the study's data to show that BrandGAN's outputs have a smaller distance to the reference design dataset when compared with the study's images. The Fréchet inception distance was reduced from 193 to 161, and the Kernel-Inception distance was reduced from 0.048 to 0.026. Overall, the results demonstrate BrandGAN's effectiveness in the task of image correction.

5.2. Future Work

We summarize our future work as the following:

- Despite the efficiency of GANdex for corrective vector lookups and interpolation, projection is a costly operation as it has a constant number of iterations regardless of the image used. Possible improvements include using a more optimal loss function during projection or replacing the projector with an alternative to find the target image's latent vector.
- Exploring the use of more complex datasets that include text or other image features that only appear in a small number of instances in the Reference Design Dataset. By getting more insights on such datasets, we would learn more about how BrandGAN can fix finer image features during the image correction process.

- Exploring the projection process and replacing it with a mapping network capable of predicting a latent vector for an image relative to a reference vector. The mapping network would make the image correction process faster as it would require a single iteration rather than the projector's multiple steps.
- Improving GANdex by replacing the random latent vector sampling step with a context-aware process that uses objective scores that ensure a balanced representation of the image instances in the reference design dataset while minimizing the size of the cache for faster lookups.
- Using a classifier during the GANdex lookup to resolve issues that arise from ambiguous shapes that resemble distinct classes from the RDD

References

[1] Wacom, “Wacom pen tablets,” *Wacom*. [Online]. Available: https://www.wacom.com/en-ca/products/pen-tablets. [Accessed: 12-Apr-2021]

[2] Huion, “Huion Pen Tablets,” *Huion*. [Online]. Available: https://store.huion.com/collections/pen-tablet. [Accessed: 12-Apr-2021]

[3] SYSTEMAX, “SYSTEMAX Software Development - PaintTool SAI.” [Online]. Available: https://www.systemax.jp/en/sai/. [Accessed: 09-Apr-2021]

[4] Adobe, “Industry-leading vector graphics software | Adobe Illustrator.” [Online]. Available: https://www.adobe.com/ca/products/illustrator.html. [Accessed: 12-Apr-2021]

[5] concept9, “The Graphic Design Process in 5 Steps,” *concept9*, 05-Jun-2015. [Online]. Available: https://concept9.ca/the-graphic-design-process-in-5-steps/. [Accessed: 12-Apr-2021]

[6] Jamahl Johnson, “4 techniques for creating mockups to show off your designs,” *99designs*, 01-Apr-2021. [Online]. Available: https://99designs.ca/blog/tips/creating-professional-mock-ups/. [Accessed: 15-Apr-2021]

[7] Creative Bloq Staff and 2013, “Design a dynamic icon-based brand identity,” *Creative Bloq*. [Online]. Available: https://www.creativebloq.com/branding/design-dynamic-icon-based-identity-4132726. [Accessed: 12-Apr-2021]

[8] Jordan Prindle, “Why Your Brand Needs Custom Icons and How to Create Your Own,” *Jordan Prindle Designs | Brand and Squarespace Designer for Entrepreneurs*. [Online]. Available: https://www.jordanprindledesigns.com/blog/brand-icons. [Accessed: 12-Apr-2021]

[9] Billy Sweetman, “UI Designer’s Guide to App Icon Design (Free Icon Set),” *Headway.io*. [Online]. Available: https://www.headway.io/blog/designers-guide-to-icons. [Accessed: 09-Apr-2021]

[10] Dribbble, “Browse thousands of Icon Set images for design inspiration | Dribbble.” [Online]. Available: https://dribbble.com/search/icon%20set. [Accessed: 09-Apr-2021]

[11] The Noun Project, "Noun Project: Free Icons & Stock Photos for Everything." [Online]. Available: https://thenounproject.com/. [Accessed: 09-Apr-2021]

[12] I. J. Goodfellow *et al.*, "Generative Adversarial Networks," *ArXiv14062661 Cs Stat*, Jun. 2014 [Online]. Available: http://arxiv.org/abs/1406.2661. [Accessed: 09-Apr-2021]

[13] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, "Analyzing and Improving the Image Quality of StyleGAN," *ArXiv191204958 Cs Eess Stat*, Mar. 2020 [Online]. Available: http://arxiv.org/abs/1912.04958. [Accessed: 09-Apr-2021]

[14] M.-Y. Liu, T. Breuel, and J. Kautz, "Unsupervised Image-to-Image Translation Networks," *ArXiv170300848 Cs*, Jul. 2018 [Online]. Available: http://arxiv.org/abs/1703.00848. [Accessed: 09-Apr-2021]

[15] Matteo Luccio, "AI helps create street maps from satellite imagery," *GPS World*, 05-Mar-2020. [Online]. Available: https://www.gpsworld.com/ai-helps-create-street-maps-from-satellite-imagery/. [Accessed: 11-Apr-2021]

[16] Andrew Lookingbill, "Google Maps 101: how we map the world," *Google*, 22-Jul-2019. [Online]. Available: https://blog.google/products/maps/google-maps-101-how-we-map-world/. [Accessed: 15-Apr-2021]

[17] M.-Y. Liu and O. Tuzel, "Coupled Generative Adversarial Networks," *ArXiv160607536 Cs*, Sep. 2016 [Online]. Available: http://arxiv.org/abs/1606.07536. [Accessed: 10-Apr-2021]

[18] H. Tang, H. Liu, D. Xu, P. H. S. Torr, and N. Sebe, "AttentionGAN: Unpaired Image-to-Image Translation using Attention-Guided Generative Adversarial Networks," *ArXiv191111897 Cs Eess*, Feb. 2020 [Online]. Available: http://arxiv.org/abs/1911.11897. [Accessed: 09-Apr-2021]

[19] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks," *ArXiv170310593 Cs*, Aug. 2020 [Online]. Available: http://arxiv.org/abs/1703.10593. [Accessed: 09-Apr-2021]

[20] Jun-Yan Zhu, Phillip Isola, and Alexei A. Efros, "Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks." [Online]. Available:

https://junyanz.github.io/CycleGAN/. [Accessed: 09-Apr-2021]

[21] J. Wu, C. Zhang, T. Xue, B. Freeman, and J. Tenenbaum, "Learning a Probabilistic Latent Space of Object Shapes via 3D Generative-Adversarial Modeling."

[22] P. Bojanowski, A. Joulin, D. Lopez-Paz, and A. Szlam, "Optimizing the Latent Space of Generative Networks," *ArXiv170705776 Cs Stat*, May 2019 [Online]. Available: http://arxiv.org/abs/1707.05776. [Accessed: 10-Apr-2021]

[23] T. Karras, S. Laine, and T. Aila, "A Style-Based Generator Architecture for Generative Adversarial Networks," Dec. 2018 [Online]. Available: https://arxiv.org/abs/1812.04948v3. [Accessed: 11-Apr-2021]

[24] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive Growing of GANs for Improved Quality, Stability, and Variation," *ArXiv171010196 Cs Stat*, Feb. 2018 [Online]. Available: http://arxiv.org/abs/1710.10196. [Accessed: 09-Apr-2021]

[25] T. Karras, M. Aittala, J. Hellsten, S. Laine, J. Lehtinen, and T. Aila, "Training Generative Adversarial Networks with Limited Data," *ArXiv200606676 Cs Stat*, Oct. 2020 [Online]. Available: http://arxiv.org/abs/2006.06676. [Accessed: 09-Apr-2021]

[26] Y. Viazovetskyi, V. Ivashkin, and E. Kashin, "StyleGAN2 Distillation for Feed-forward Image Manipulation," *ArXiv200303581 Cs*, Oct. 2020 [Online]. Available: http://arxiv.org/abs/2003.03581. [Accessed: 09-Apr-2021]

[27] S. Tripathi, Z. C. Lipton, and T. Q. Nguyen, "Correction by Projection: Denoising Images with Generative Adversarial Networks," *ArXiv180304477 Cs*, Mar. 2018 [Online]. Available: http://arxiv.org/abs/1803.04477. [Accessed: 09-Apr-2021]

[28] Y. Shen, J. Gu, X. Tang, and B. Zhou, "Interpreting the Latent Space of GANs for Semantic Face Editing," *ArXiv190710786 Cs*, Mar. 2020 [Online]. Available: http://arxiv.org/abs/1907.10786. [Accessed: 09-Apr-2021]

[29] NVIDIA Research Projects, "StyleGAN2 Pre-trained Models." [Online]. Available: https://nvlabs-fi-cdn.nvidia.com/stylegan2/networks/. [Accessed: 09-Apr-2021]

[30] Fiverr, "Get Everything You Need Starting at $5 - Fiverr," *Fiverr.com*. [Online]. Available: https://www.fiverr.com/. [Accessed: 09-Apr-2021]

[31] Upwork, “In-demand talent on demand.™ Upwork is how.™,” *Upwork*. [Online]. Available: https://www.upwork.com/. [Accessed: 09-Apr-2021]

[32] Maria De La Riva, “Here’s How To Design An Icon From Scratch: A Step-By-Step Guide.” [Online]. Available: https://careerfoundry.com/en/blog/ui-design/icon-design-process/. [Accessed: 09-Apr-2021]

[33] C. E. October 05 and 2020, “16 essential tools for graphic designers in 2021,” *Creative Bloq*. [Online]. Available: https://www.creativebloq.com/graphic-design/tools-every-graphic-designer-should-have-6133208. [Accessed: 09-Apr-2021]

[34] Delia Tugui, “The Cost of Hiring a Freelance Graphic Designer,” *Business 2 Community*. [Online]. Available: https://www.business2community.com/small-business/the-cost-of-hiring-a-freelance-graphic-designer-02143632. [Accessed: 09-Apr-2021]

[35] Arek Dvornechuck, “Logo Design Process From Start To Finish (A Step-by-Step Guide).” [Online]. Available: https://www.ebaqdesign.com/blog/logo-design-process. [Accessed: 09-Apr-2021]

[36] Matt Ellis, “Logo design process: how professionals do it -,” *99designs*, 06-Aug-2019. [Online]. Available: https://99designs.ca/blog/tips/logo-design-process-how-professionals-do-it/. [Accessed: 09-Apr-2021]

[37] Justas, “My Icon Design Process,” *Icon Utopia*, 14-Jul-2015. [Online]. Available: https://iconutopia.com/my-icon-design-process/. [Accessed: 09-Apr-2021]

[38] Logo Design Love, “How many logo options do you present to clients?,” *Logo Design Love*, 09-Jul-2015. [Online]. Available: https://www.logodesignlove.com/how-many-options. [Accessed: 09-Apr-2021]

[39] David Anderson, “How many mockups or design comps do you show clients?,” *E-Learning Heroes*. [Online]. Available: https://community.articulate.com/discussions/building-better-courses/how-many-mockups-or-design-comps-do-you-show-clients. [Accessed: 09-Apr-2021]

[40] GREGORY BARBER, “A Molecule Designed by AI Exhibits ‘Druglike’ Qualities | WIRED.” [Online]. Available: https://www.wired.com/story/molecule-designed-ai-exhibits-druglike-qualities/. [Accessed: 09-Apr-2021]

[41] NVIDIA Research Projects, *StyleGAN — Official TensorFlow Implementation*. NVIDIA [Online]. Available: https://github.com/NVlabs/stylegan. [Accessed: 12-Apr-2021]

[42] NVIDIA Research Projects, *StyleGAN2 — Official TensorFlow Implementation*. NVIDIA [Online]. Available: https://github.com/NVlabs/stylegan2. [Accessed: 12-Apr-2021]

[43] NVIDIA Research Projects, *StyleGAN2 with adaptive discriminator augmentation (ADA) — Official TensorFlow implementation*. NVIDIA [Online]. Available: https://github.com/NVlabs/stylegan2-ada. [Accessed: 09-Apr-2021]

[44] NVIDIA Research Projects, *Flickr-Faces-HQ Dataset (FFHQ)*. NVIDIA [Online]. Available: https://github.com/NVlabs/ffhq-dataset. [Accessed: 09-Apr-2021]

[45] Google, “Material Design Guidelines,” *Material Design*. [Online]. Available: https://material.io/design. [Accessed: 09-Apr-2021]

[46] Google, *Material design icons is the official icon set from Google*. Google [Online]. Available: https://github.com/google/material-design-icons. [Accessed: 09-Apr-2021]

[47] S. Yang, J. Liu, Z. Lian, and Z. Guo, “Awesome Typography: Statistics-Based Text Effects Transfer,” *ArXiv161109026 Cs*, Dec. 2016 [Online]. Available: http://arxiv.org/abs/1611.09026. [Accessed: 09-Apr-2021]

[48] T.-H. Sun, C.-H. Lai, S.-K. Wong, and Y.-S. Wang, “Adversarial Colorization Of Icons Based On Structure And Color Conditions,” *ArXiv191005253 Cs Eess*, Oct. 2019 [Online]. Available: http://arxiv.org/abs/1910.05253. [Accessed: 09-Apr-2021]

[49] X. Liu, Z. Gao, and B. M. Chen, “MLFcGAN: Multi-level Feature Fusion based Conditional GAN for Underwater Image Color Correction,” *IEEE Geosci. Remote Sens. Lett.*, vol. 17, no. 9, pp. 1488–1492, Sep. 2020, doi: 10.1109/LGRS.2019.2950056.

[50] C.-H. Lin, E. Yumer, O. Wang, E. Shechtman, and S. Lucey, “ST-GAN: Spatial Transformer Generative Adversarial Networks for Image Compositing,” in *2018 IEEE/CVF*

Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, 2018, pp. 9455–9464, doi: 10.1109/CVPR.2018.00985 [Online]. Available: https://ieeexplore.ieee.org/document/8579083/. [Accessed: 09-Apr-2021]

[51] S. Menon, A. Damian, S. Hu, N. Ravi, and C. Rudin, "PULSE: Self-Supervised Photo Upsampling via Latent Space Exploration of Generative Models," *ArXiv200303808 Cs Eess*, Jul. 2020 [Online]. Available: http://arxiv.org/abs/2003.03808. [Accessed: 09-Apr-2021]

[52] Mayo Clinic, "CT scan - Mayo Clinic." [Online]. Available: https://www.mayoclinic.org/tests-procedures/ct-scan/about/pac-20393675. [Accessed: 09-Apr-2021]

[53] Paul Crawford, "Cone Beam CT vs. Traditional CT." [Online]. Available: https://info.blockimaging.com/and-the-top-used-cone-beam-ct-is. [Accessed: 09-Apr-2021]

[54] J. Harms *et al.*, "Paired cycle‐GAN‐based image correction for quantitative cone‐beam computed tomography," *Med. Phys.*, vol. 46, no. 9, pp. 3998–4009, Sep. 2019, doi: 10.1002/mp.13656.

[55] Flaticon, "Flaticon, the largest database of free vector icons," *Flaticon*. [Online]. Available: https://www.flaticon.com/. [Accessed: 09-Apr-2021]

[56] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein GAN," *ArXiv170107875 Cs Stat*, Dec. 2017 [Online]. Available: http://arxiv.org/abs/1701.07875. [Accessed: 09-Apr-2021]

[57] NVIDIA, "GeForce RTX 2070 SUPER Graphics Cards," *NVIDIA*. [Online]. Available: https://www.nvidia.com/en-us/geforce/graphics-cards/rtx-2070-super/. [Accessed: 09-Apr-2021]

[58] NVIDIA, "NVIDIA TESLA V100," *NVIDIA*. [Online]. Available: https://www.nvidia.com/en-gb/data-center/tesla-v100/. [Accessed: 09-Apr-2021]

[59] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *ArXiv14091556 Cs*, Apr. 2015 [Online]. Available: http://arxiv.org/abs/1409.1556. [Accessed: 09-Apr-2021]

[60] Open Source Computer Vision, "CV2's Warp Perspective," *Geometric Image*

Transformations. [Online]. Available: https://www.docs.opencv.org/2.4/modules/imgproc/doc/geometric_transformations.html#void%20warpPerspective(InputArray%20src,%20OutputArray%20dst,%20InputArray%20M,%20Size%20dsize,%20int%20flags,%20int%20borderMode,%20const%20Scalar&%20borderValue). [Accessed: 09-Apr-2021]

[61] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The Unreasonable Effectiveness of Deep Features as a Perceptual Metric," *ArXiv180103924 Cs*, Apr. 2018 [Online]. Available: http://arxiv.org/abs/1801.03924. [Accessed: 09-Apr-2021]

[62] O. Russakovsky *et al.*, "ImageNet Large Scale Visual Recognition Challenge," *ArXiv14090575 Cs*, Jan. 2015 [Online]. Available: http://arxiv.org/abs/1409.0575. [Accessed: 13-Apr-2021]

[63] Visual Geometry Group, "Visual Geometry Group - University of Oxford," *University of Oxford*. [Online]. Available: https://www.robots.ox.ac.uk/~vgg/. [Accessed: 13-Apr-2021]

[64] P. A. Prihatmaja, "Reverse Image Search Using Pretrained Deep Learning Model," *Medium*, 12-Nov-2019. [Online]. Available: https://medium.com/swlh/reverse-image-search-using-pretrained-deep-learning-model-83f16ef4aec8. [Accessed: 13-Apr-2021]

[65] Jun Gu, "Building a Reverse Image Search System Based on Milvus and VGG - DZone AI," *dzone.com*. [Online]. Available: https://dzone.com/articles/building-a-reverse-image-search-system-based-on-mi. [Accessed: 13-Apr-2021]

[66] DeepAI, "Perceptual Loss Functions," *DeepAI*, 17-May-2019. [Online]. Available: https://deepai.org/machine-learning-glossary-and-terms/perceptual-loss-function. [Accessed: 13-Apr-2021]

[67] scikit-image, "Module: morphology — skimage v0.19.0.dev0 docs." [Online]. Available: https://scikit-image.org/docs/dev/api/skimage.morphology.html. [Accessed: 09-Apr-2021]

[68] National Institute of Standards and Technology, "COSINE DISTANCE, COSINE SIMILARITY, ANGULAR COSINE DISTANCE, ANGULAR COSINE SIMILARITY." [Online]. Available:

https://www.itl.nist.gov/div898/software/dataplot/refman2/auxillar/cosdist.htm. [Accessed: 09-Apr-2021]

[69] S. Hurwitz, "A History of Index Creation," *Elephant*. [Online]. Available: http://www.elephant.org.il/indexing/a-history-of-index-creation. [Accessed: 17-Apr-2021]

[70] Open Source Computer Vision, "OpenCV: Geometric Image Transformations." [Online]. Available: https://docs.opencv.org/master/da/d54/group__imgproc__transform.html. [Accessed: 09-Apr-2021]

[71] J. Mezirow, "Perspective Transformation," *Adult Educ.*, vol. 28, no. 2, pp. 100–110, Jan. 1978, doi: 10.1177/074171367802800202.

[72] K. F. Schulz, I. Chalmers, and D. G. Altman, "The Landscape and Lexicon of Blinding in Randomized Trials," *Ann. Intern. Med.*, vol. 136, no. 3, pp. 254–259, Feb. 2002, doi: 10.7326/0003-4819-136-3-200202050-00022.

[73] Stephanie Glen, "Snowball Sampling: Definition, Advantages and Disadvantages," *Statistics How To*. [Online]. Available: https://www.statisticshowto.com/probability-and-statistics/statistics-definitions/snowball-sampling/. [Accessed: 09-Apr-2021]

[74] M. Bińkowski, D. J. Sutherland, M. Arbel, and A. Gretton, "Demystifying MMD GANs," *ArXiv180101401 Cs Stat*, Jan. 2021 [Online]. Available: http://arxiv.org/abs/1801.01401. [Accessed: 09-Apr-2021]

[75] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium," *ArXiv170608500 Cs Stat*, Jan. 2018 [Online]. Available: http://arxiv.org/abs/1706.08500. [Accessed: 09-Apr-2021]

[76] E. R. Chan, M. Monteiro, P. Kellnhofer, J. Wu, and G. Wetzstein, "pi-GAN: Periodic Implicit Generative Adversarial Networks for 3D-Aware Image Synthesis," *ArXiv201200926 Cs* [Online]. Available: http://arxiv.org/abs/2012.00926. [Accessed: 11-Apr-2021]

[77] S. Tan, K. Wong, S. Wang, S. Manivasagam, M. Ren, and R. Urtasun, "SceneGen: Learning to Generate Realistic Traffic Scenes," *ArXiv210106541 Cs*, Jan. 2021 [Online]. Available:

http://arxiv.org/abs/2101.06541. [Accessed: 11-Apr-2021]

[78] T. n, C. Li, L. Theis, C. Richardt, and Y.-L. Yang, “HoloGAN: Unsupervised learning of 3D representations from natural images,” *ArXiv190401326 Cs*, Oct. 2019 [Online]. Available: http://arxiv.org/abs/1904.01326. [Accessed: 11-Apr-2021]

[79] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the Inception Architecture for Computer Vision,” *ArXiv151200567 Cs*, Dec. 2015 [Online]. Available: http://arxiv.org/abs/1512.00567. [Accessed: 11-Apr-2021]

[80] Y. Goldberg and M. Elhadad, “splitSVM: fast, space-efficient, non-heuristic, polynomial kernel computation for NLP applications,” in *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies Short Papers - HLT ’08*, Columbus, Ohio, 2008, p. 237, doi: 10.3115/1557690.1557758 [Online]. Available: http://portal.acm.org/citation.cfm?doid=1557690.1557758. [Accessed: 16-Apr-2021]

[81] I. Olkin and F. Pukelsheim, “The distance between two random vectors with given dispersion matrices,” *Linear Algebra Its Appl.*, vol. 48, pp. 257–263, Dec. 1982, doi: 10.1016/0024-3795(82)90112-4.