University of Ottawa
Department of Mathematics and Statistics

An Introduction to Generative Adversarial Networks

by Bryan Paget

Thesis submitted to the Faculty of Graduate and Postdoctoral Studies in partial fulfillment of the requirements for the degree of Masters in Science in Statistics.[1]

September 11, 2019

[1]The master's program is a joint program with Carleton University, administered by the Ottawa-Carleton Institute of Mathematics and Statistics.

Table of Contents

1 Notation **iii**

2 Abstract **iv**

3 Preface **v**

4 Introduction **1**

5 Game Theory **6**

5.1 Nash Equilibrium and the Prisoner's Dilemma 7

5.2 Derivation of the Value Function 10

5.3 Generative Adversarial Networks 15

5.4 Discussion 15

6 Information Theory **16**

6.1 Information-Theoretic View of GANs 24

6.2 Optimization Dynamics 26

6.3 Discussion 29

7 Optimal Transport **30**

7.1 Limitations of the Kullback-Leibler Divergence 32

7.2 The Monge-Kantorovich Transportation Problem 36

7.3 The Wasserstein GAN 38

7.4 Discussion 40

8 Conclusion **41**

9 Appendix **43**

9.1 Optimization Dynamics 43

References **44**

Section 1

Notation

Generator	G
Discriminator	D
Space of parameter values for the Generator	Φ
Space of parameter values for the Discriminator	Θ
Generator Parameterized by $\phi \in \Phi$	G_ϕ
Discriminator Parameterized by $\theta \in \Theta$	D_θ
Target Space	$\mathcal{X}$
Prior set	$\mathcal{Z}$
Prior distribution	$p_{\mathcal{Z}}$
Generated Data Point	$\tilde{x} = G_\phi(z)$
Probability Distribution Induced by G_ϕ	p_ϕ
Target Probability Distribution	p^*
Value Function	$V(\phi, \theta)$

Section 2

Abstract

This thesis is a survey of the mathematical theory of Generative Adversarial Networks (GANs). The relevant theories discussed are game theory, information theory and optimal transport theory.

Section 3

Preface

One of the goals I set out for myself when I started this thesis was to provide a document rich in theory and intuition useful for anyone wanting to get up to speed on the theory of generative adversarial networks (GANs).

I have broken the theory into three different sections, *game theory*, *information theory*, and *optimal transport theory*. I chose this particular order because game theory provides the best introduction to the algorithm. An adversarial game is an intuitive and catchy way to conceptualize the alternating optimization. Information theory is required to deepen the understanding of the value function. Optimal transport was included at the end because it provides the theoretical foundation of a more recent implementation of the GAN algorithm which was inspired by the optimal transport problem of Monge, Kantorovich, and Rubinstein. The content of this thesis includes the following highlights:

(i) Section **??** contains definitions and examples related to the Nash equilibrium.

(ii) I have included a motivational derivation of the value function in sections **??**, **??**, and **??**.

(iii) In Section **??**, I provide an illustrative example demonstrating the difficulty in finding a Nash equilibrium.

(iv) Section **??** contains definitions of information theoretic quantities used in machine learning. I then show how those quantities are found in the value function in Section **??**.

(v) Section **??** contains rigorous proofs that show the discriminator is optimized to force the value function into an approximation of the Jensen-Shannon divergence and the generator is optimized to minimize this divergence.

(vi) Section **??** contains the history of optimal transport and introduction to the Wasserstein GAN.

(vii) Proposition **??** offers an information-theoretic perspective on what happens step-by-step during GAN training rather than the limiting behavior as discussed in Section **??**.

Section 4

Introduction

Figure 1: Crítica by Julio Ruelas, ca. 1907

This thesis is a survey on the theory behind the machine learning algorithm called *generative adversarial networks* (GANs).

The purpose of the GAN algorithm is to train two neural networks through a two-player, zero-sum game, hence the first chapter is on game theory.

The two neural networks model completely different functions. One is trained to become a generative model and it is called the generator. The other is trained to become an astute classifier between two classes (real data and generated data) and it is called the discriminator. The value function $V(\phi, \theta)$ used by the GAN algorithm can be understood through information theory, hence the second section of this thesis is on information theory.

The original formulation of the GAN algorithm had some limitations in practice. Some authors have made improvements to the original formulation by importing theory from optimal transport. This is explored in the third section of this thesis.

The GAN algorithm is best described as a framework for training two neural networks, one to generate some type of data and another to judge how closely the generated data

resemble the real data. A generative model g is a function that takes data from one probability space $(\mathcal{Z}, p_{\mathcal{Z}})$, called the prior space, and transforms it into something else entirely in some other probability space $(\mathcal{X}, p^*)$, called the target space.

Definition 4.1. *Let $\Phi \subset \mathbb{R}^n$ be a space of parameter values. A* generator *G is a function $G : \Phi \times \mathcal{Z} \mapsto \mathcal{X}$, which maps z drawn from $(\mathcal{Z}, p_{\mathcal{Z}})$ to $(\mathcal{X}, p^*)$. Let G_ϕ denote the generator parameterized by $\phi \in \Phi$.*

In the definition of the generator, we fixed a subset $\Phi \subset \mathbb{R}^n$ as the space of parameter values which are optimized during training. The parameter space for the discriminator is a different subset of $\mathbb{R}^n$, since the discriminator is a very different type of neural network than the generator, requiring a qualitatively different set of parameters. The space $\mathcal{X}$ is the output of the generator and is the input to the discriminator.

Definition 4.2. *Let $\Theta \subset \mathbb{R}^n$ be a space of parameter values. A* discriminator *D, is a function $D : \Theta \times \mathcal{X} \mapsto [0, 1]$, which computes the probability that x was drawn from $(\mathcal{X}, p^*)$. Let D_θ denote the discriminator parameterized by $\theta \in \Theta$.*

Definition 4.3. *The* generative adversarial networks *algorithm trains two neural networks, one called the generator G, which produces samples from the target space $(\mathcal{X}, p^*)$ and the other, called the discriminator D, which computes the probability that an observed sample came from $(\mathcal{X}, p^*)$. We can put it all together in this diagram.*

$$\begin{array}{ccccccc} \mathcal{Z} & \xrightarrow{G} & \tilde{\mathcal{X}} & \longrightarrow & (\mathcal{X}, \tilde{\mathcal{X}}) & \xrightarrow{D} & ([0,1],[0,1]) \xrightarrow{V(\phi,\theta)} \mathbb{R} \\ & & & \nearrow & & & \\ & & \mathcal{X} & & & & \end{array}$$

D maximizes and G minimizes the value function,

$$V(\phi, \theta) = \mathbb{E}_{x \sim p^*}[\log D_\theta(x)] + \mathbb{E}_{z \sim p_{\mathcal{Z}}}[\log (1 - D_\theta(G_\phi(z)))], \tag{4.1}$$

hence the GAN algorithm can be stated as

$$\min_\phi \max_\theta \mathbb{E}_{x \sim p^*}[\log D_\theta(x)] + \mathbb{E}_{z \sim p_{\mathcal{Z}}}[\log (1 - D_\theta(G_\phi(z)))]. \tag{4.2}$$

Many use cases for the GAN algorithm are in the generation of photo-realistic images. Other use cases include medical imaging, where data sets often suffer from class imbalances since it is easier to find data on healthy subjects. GANs have been used to

augment data sets by producing synthetic images of diseased plant and animal tissues as in [?], [?], and [?]. Synthetic data may also help anonymize sources of data for medical research, as explored in [?].

While most research focuses on the generative side of the algorithm, it is worth keeping in mind that the GAN algorithm also produces a discriminator, which is a classifier between two classes and has been used as a classifier in [?] to classify different plant diseases, which was inspired by [?].

As stated above, the GAN algorithm comprises two artificial, feedforward neural networks, a class of function made by composing many primitive functions.

Neural Networks

Definition 4.4. *A* Feedforward Neural Network *is a directed graph. At each node is a composition of linear (or affine) functions with a nonlinear activation function. The input and output of the nodes are determined by the connectivity of the graph.*

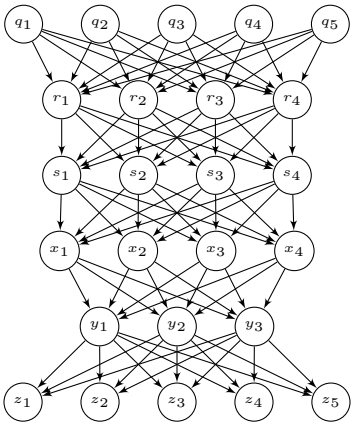

Figure 2: A Deep Neural Network

If we let f_θ denote the neural network parameterized by $\theta \in \Theta \subset \mathbb{R}^n$, we can think of training f_θ as searching through the space of all possible parameters Θ, in search of θ^* that minimizes some differentiable measure of loss.

An information theoretic explanation for what deep learning is and how the training works is given in [?] and [?]. Tishby, Zaslavsky, and Shwartz-Ziv touch on some important topics in deep learning from an information theory perspective. One such assertion of theirs is the arbitrary nature of the structure of the graph, i.e. there are many equivalent

graphs that minimize the loss function, therefore we are not really in search of some perfect $\theta^* \in \Theta$, as there are many equivalent θ.

There are many optimization algorithms for searching through Θ, most of which are variations of gradient descent. Gradients are taken with respect to the loss function $\mathcal{L}$ and the backpropagation algorithm assigns contribution of error. *Backpropagation* was introduced in [**?**], and is essentially an application of the chain rule of calculus to compute the derivatives of the loss function with respect to each parameter in the graph. At each node of the graph is a linear function, composed inside an activation function, as in

$$\frac{1}{1 + \exp\left(-\sum_{i=1}^{n} x_i \theta_i - b\right)}. \tag{4.3}$$

Like a neural network, a linear regression model is a function parameterized by a set of learnable parameters θ. The formula for linear regression is

$$\hat{y} = \sum_{i=1}^{n} x_i \theta_i - b. \tag{4.4}$$

The parameter vector θ of a linear regression model can be thought of as an expression for how important each entry of x is for a specific output y. A linear regression model may also include a bias term b, which in the case of a neural network can be interpreted as the activation threshold for the neuron. In a sense, a neural network provides a way to compose many linear regression models into a much larger model. [**?**] even go so far as to say that feed-forward neural networks are equivalent to polynomial regression.

Generative Adversarial Networks

The generator of the GAN algorithm trains a generative model, which we define as any mathematical or statistical model that mimics the process of creating the observed data [**?**]. Starting with randomly initialized parameters, which means we randomly choose a $\theta \in \Theta$, we optimize the model until it maps random input into the desired output with frequencies similar enough to the desired probability distribution.

The earliest reference to two-player games that result in the production of a generative model can be found in [**?**], a short article containing the following game. *"We are told the constraints, we pick a distribution, God gets to pick the 'real' distribution, satisfying the constraints of course. Some disinterested party picks an outcome according to the 'real' distribution that God has just picked, and we have to pay according to how surprised we*

are to see that outcome." This is not quite the GAN algorithm, but it has a similar moral, that is we seek a probability distribution and we learn what qualities define it through feedback. If we iterate the above procedure and formalize a way of learning from our surprise, we arrive at something resembling the GAN algorithm.

As for the history of adversarial games (where the players are trying to undermine each other), Jürgen Schmidhuber wrote more than one paper, the first of which appeared in 1992, on what he called *predictability minimization*, which is very similar to the GAN algorithm (see [**?**] and [**?**]). The idea behind predictability minimization was for one player to try to predict outcomes in some event space and the adversary tries to minimize the ability of that player to make those predictions.

A blog post from 2010 [**?**] essentially describes the GAN algorithm. In 2012, a paper on adversarial support vector machines was written by [**?**]. Finally, in 2014 an algorithm for training models to simulate the behavior of animals based on two populations, replicas and classifiers, was written by [**?**].

In 2014, while a student of Yoshua Bengio at the Université de Montréal, Ian Goodfellow and colleagues published *Generative Adversarial Nets*. This paper has since inspired many publications and has triggered a surge in the interest in generative models.

Now with this thesis we present a survey of the mathematical theory behind Generative Adversarial Networks. The relevant theories discussed are game theory, information theory and optimal transport theory.

Section 5

Game Theory

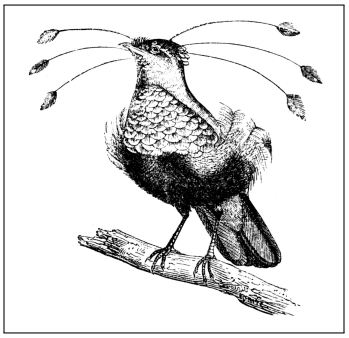

Figure 3: Louis Figuier, "Reptiles and Birds", 1869.

The original theoretical formulation of the GAN algorithm, as in [**?**], is through *game theory*, which is the study of conflict and cooperation between rational decision makers through the use of mathematical and statistical modelling [**?**]. Throughout this thesis we only consider two-player games, and we will use D to represent the player and G to represent her opponent. We use the symbol V_i to represent the value function of player i for $i \in \{D, G\}$.

Definition 5.1. *A* value function V_i *is the reward (or loss if it is negative) for player i in a game as a function of that player's action(s).*

Remark. *Sometimes the loss is simply the difference between the target value y and the estimate $\hat{y}$. In the case of GANs, the value function is more complex and is essentially a measure on uncertainty.*

There are actually two value functions (see Section **??**) used in [**?**], the first allows the GAN algorithm to be interpreted as a *zero-sum game*, while the second value function does not [**?**].

Definition 5.2. *A* zero-sum game *is any game where the loss of one player is the gain of the other. In a zero-sum game, the players' rewards sum to zero, i.e.* $V_D + V_G = 0$.

Remark. *For this section we will stick with the first value function, therefore we will consider the GAN algorithm to be a zero-sum game between the generator and the discriminator, see (??) and (??) for the relevant definitions.*

What is the optimal strategy in a zero-sum-game? At any given turn, what action should a player take to maximize his or her chances of winning or at least minimize the expected loss? The *minimax decision rule* is a strategy to get the best outcome given that you know your opponent is trying to minimize your reward.

Definition 5.3. *In a two-player game, with players* D *and* G*, a* minimax decision rule *for* D *is a strategy that maximizes* $D's$ *expected reward, after* G *has minimized the maximum reward attainable by* D *that is. The* minimax value $\overline{V_D}$ *for* D*, is the largest reward* D *can win after* G *makes his move to minimize* D*'s reward, in symbols we write*

$$\overline{V_D} = \min_G \max_D V_D(D, G). \tag{5.1}$$

In a game between D and G, if G moves first to minimize the reward D can attain on her move, the minimax rule for D will allow D to attain the maximum of the reduced reward.

5.1 Nash Equilibrium and the Prisoner's Dilemma

The GAN algorithm searches for a *Nash equilibrium* in the space of parameters for the discriminator and the generator, as covered in [**?**] and [**?**].

Definition 5.4. *A* Nash equilibrium *in an* n*-player game is a set of strategies, one for each player, with the property that no player can benefit from unilaterally changing their strategy.*

Let S_i *denote the set of strategies for player* i*. Let* $S = S_1 \times S_2 \times \cdots \times S_n$ *denote the set of strategy profiles. Each strategy profile* $s \in S$ *is a combination of strategies, one for each player, which dictates what each player will do on their next turn. Let* $f_i(s)$ *be the payoff to player* i *of strategy profile* s*.*

A Nash equilibrium is a strategy profile $s^* = (s_1, s_2, \ldots, s_n)$ *such that*

$$f_i(s^*) \geq f_i((s_1, s_2, \ldots, s_{-i}, \ldots, s_n)) \tag{5.2}$$

for all i *and* s_{-i} *denotes any strategy of player* i *other than* s_i *from* s^*.

Remark. *Each player's strategy is an optimal response based on the anticipated behavior of the other player(s) in the game. A minimax solution to a zero-sum game is the same as a Nash Equilibrium. The Nash equilibrium is not necessarily globally optimal and a game may have more than one Nash equilibrium.*

We look to the *Prisoner's Dilemma* for an illustrative example of a Nash equilibrium. The Prisoner's Dilemma was originally formulated in a paper by Merrill Flood and Melvin Dresher in the 1950s, and Albert W. Tucker reformulated the game in terms of prison sentences and gave it the name *Prisoner's Dilemma* [**?**]. The point of the Prisoner's Dilemma is to show that the globally optimal state may not be realistically attainable in a non-cooperative game. Each player is trying to minimize their expected jail-time and they are assuming the other player is doing the same and, given that, they will not achieve the global minimum.

5.1.1 Prisoner's Dilemma

This game has two players, D and G, who are held separately in police custody with no means of communication. The players have each been arrested for some small crime of which the prosecution has enough evidence to convict them. However, the prosecution suspects one of D or G of having committed a more serious crime in the not-so-distant past. The problem is they lack sufficient evidence to convict either D or G of the more serious crime. So the prosecutors cut each player a deal. The deal is

1. if D defects and accuses G of having committed the more serious crime (and G denies involvement) D gets 1 year (as a reward for co-operation) while G gets 5 years of jail-time (and vice-versa);

2. if both D and G deny committing the larger crime, then they both get 2 years, since the prosecution does have enough evidence to convict them of the less serious crime;

3. if both D and G accuse each other of committing the larger crime, then they both get 3 years.

D \ G	Deny	Defect
Deny	(2, 2)	(5, 1)
Defect	(1, 5)	(3, 3)

Figure 4: Prisoner's Dilemma Payoff Matrix

Figure **??** has the game's payoff matrix, scores are for (D, G), i.e. D controls the rows and G controls the columns. It is obvious that (Deny, Deny) is the global optimum, but it turns out that it is an unstable state. The state (Defect, Defect) is the Nash equilibrium. We will go over all four states and show why this is true.

1. *(Deny, Defect).* If D is allowed to change her stance, knowing that G is going to defect, she will end up in a better situation, which means there is incentive for D to change her strategy, given that G will not change his. Therefore, this is not a Nash equilibrium, and vice-versa for (Defect, Deny) by symmetry.

2. *(Deny, Deny).* If D and G both deny involvement, and if D knows that G is not going to change his stance, D does gain from changing her strategy to defecting, since that will change the situation to (Defect, Deny), where D does 1 year and G does 5.

3. *(Defect, Defect).* If they both defect, neither D nor G will gain anything by changing their strategy, for if D changed to denying involvement, D would serve 5 years. A similar argument can be made about G changing his strategy.

Thus (Defect, Defect) is the Nash equilibrium for this game. It is a stable game state, i.e. there is no incentive for either player to change strategy (unless the other does as well). Remember, they can not communicate with each other. As D sits in the holding cell, she is thinking that G is going to either defect or deny involvement. If D wants to minimize the maximum amount of jail-time she could possibly do, she will want to defect, since that will eliminate the worst case scenario of having to do 5 years. When she defects, the worst that can happen to her is 3 years.

Since (in this situation) the minimax has not achieve the global minimum, what can we say about the minimax strategy and the GAN algorithm?

5.2 Derivation of the Value Function

Now that we have introduced the relevant concepts from game theory we can introduce the algorithm and its constituent neural networks. Let $(\mathcal{X}, p^*)$ be a probability space where $\mathcal{X}$ is any finite space. Let p^* be the probability distribution over this space. For instance, $\mathcal{X}$ may be the idealized space of all 8-bit RGB images of length H and width W, (i.e. $\mathcal{X} = [0, 255]^{3\times H\times W}$), and p^* is the probability distribution over this space which assigns mass to regions of $\mathcal{X}$ which correspond to RGB values that look like meaningful images, e.g. images of human faces. Each point in this space represents an image and most points in this space will look like meaningless noise.[2]

The goal of the GAN algorithm is to train a generator G_ϕ, parameterized by $\phi \in \Phi \subset \mathbb{R}^n$, to map random samples z drawn from $(\mathcal{Z}, p_{\mathcal{Z}})$, where $\mathcal{Z} \subset \mathbb{R}^n$ such that $G_\phi(z)$ is located in $\mathcal{X}$ in a region where p^* assigns a relatively large amount of mass. In practice $\mathcal{Z} \neq \mathcal{X}$ and n is usually smaller than $|\mathcal{X}|$ [**?**].

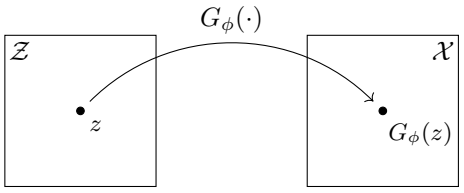

Figure 5: G_ϕ maps z to $G_\phi(z) \in \mathcal{X}$

The reason we use a generative model is to obtain samples from $\mathcal{X}$ characterized by a specific distribution of values. For instance, if $\mathcal{X}$ is the space of all RGB images and we desire images of human faces, then we want G_ϕ to map samples from $\mathcal{Z}$ to the region of $\mathcal{X}$ with RGB values within a very narrow range.

In the beginning, G_ϕ maps $z \in \mathcal{Z}$ to $\mathcal{X}$ haphazardly and it is from D_θ that G_ϕ obtains gradients from which to learn. We optimize the discriminator D_θ until it classifies samples sufficiently well, i.e. D_θ must learn to assign the correct probability that the sample came from the target distribution. Thus D_θ and G_ϕ both learn p^*, but from different perspectives.

[2] Meaningless to a human.

Let D_θ be a discriminator and G_ϕ be a generator. Let $V(\phi, \theta)$ be the value function, which is a function of the *actions* (which are parameterized by ϕ and θ) taken by D_θ and G_ϕ, given by

$$V(\phi, \theta) = \mathbb{E}_{x \sim p^*}[\log D_\theta(x)] + \mathbb{E}_{z \sim p_{\mathcal{Z}}}[\log (1 - D_\theta(G_\phi(z)))]. \tag{5.3}$$

The value function used by the GAN algorithm is related to the value function used by the *noise-contrastive estimator*, see [**?**] for more details. In the game between G_ϕ and D_θ, each will want to maximize the minimum reward it can earn and equivalently, minimize the maximum loss. In general, a player can minimize the maximum loss by playing its minimax decision rule (see Definition **??**).

5.2.1 Derivation of the Value Function from the Perspective of D_θ

We begin the derivation of $V(\phi, \theta)$ from the perspective of D_θ. We want $D_\theta(x)$ to assign mass to $x \in \mathcal{X}$ that are close to the mass assigned by p^*. We can do this by finding $\theta \in \Theta$ which maximize the likelihood function

$$L_D^{(1)}(\theta) = \prod_{i=1}^{n} D_\theta(x_i), \quad x_i \sim p^*. \tag{5.4}$$

Since this maximization is done numerically, we maximize the logarithm of the above equation,

$$\ell_D^{(1)}(\theta) = \log \prod_{i=1}^{n} D_\theta(x_i) = \sum_{i=1}^{n} \log D_\theta(x_i). \tag{5.5}$$

One benefit of using log is it is monotone increasing (any argument that maximizes (**??**) will also maximize (**??**)) and the curvature of log may make optimization more efficient by providing steeper gradients.

At the same time, since we also want the discriminator D_θ to distinguish generated samples from real samples, we want D_θ to assign low probability (preferably zero) to any generated data point $G_\phi(z) = \tilde{x}$. Therefore, for some fixed G_ϕ, θ must also minimize the following likelihood function,

$$^*L_D^{(2)}(\theta) = \prod_{i=1}^{n} D_\theta(G_\phi(z_i)), \quad z_i \sim p_{\mathcal{Z}}, \tag{5.6}$$

and as before, minimization of ${}^*L_D^{(2)}(\theta)$ can be done via minimization of its logarithm,

$$ {}^*\ell_D^{(2)}(\theta) = \log \prod_{i=1}^{n} D_\theta(G_\phi(z_i)) = \sum_{i=1}^{n} \log D_\theta(G_\phi(z_i)). \tag{5.7} $$

Alternatively, since we are already maximizing (**??**), we can turn (**??**) into a function to be maximized. Specifically, we ask D_θ to maximize

$$ \ell_D^{(2)}(\theta) = \sum_{i=1}^{n} \log\left(1 - D_\theta(G_\phi(z_i))\right), \tag{5.8} $$

since minimizing $D_\theta(G_\phi(z))$ is the same as maximizing $1 - D_\theta(G_\phi(z))$. When we combine $\ell_D^{(1)}(\theta)$ and $\ell_D^{(2)}(\theta)$ we arrive at the following objective function for D_θ to maximize,

$$ \sum_{i=1}^{n} \left(\log D_\theta(x_i) + \log\left(1 - D_\theta(G_\phi(z_i))\right)\right). \tag{5.9} $$

This is equivalent to maximizing the mean, which, by the law of large numbers, converges to

$$ \mathbb{E}_{x \sim p^*}[\log D_\theta(x)] + \mathbb{E}_{z \sim p_{\mathcal{Z}}}[\log\left(1 - D_\theta(G_\phi(z))\right)] \tag{5.10} $$

for n sufficiently large. Thus, we can say the objective of the GAN algorithm from the perspective of the discriminator is to find a set of parameters θ which maximize (**??**).

5.2.2 Derivation of the Value Function from the Perspective of G_ϕ

From the perspective of the generator, we search for ϕ that maximizes the likelihood (as defined by a fixed D_θ) that a generated sample comes from the target distribution p^*, i.e. we want G_ϕ to maximize

$$ {}^*\ell_G(\phi) = \log \prod_{i=1}^{n} D_\theta(G_\phi(z_i)) = \sum_{i=1}^{n} \log D_\theta(G_\phi(z_i)), \quad z_i \sim p_{\mathcal{Z}}. \tag{5.11} $$

Alternatively, we can minimize

$$ \ell_G(\phi) = \sum_{i=1}^{n} \log\left(1 - D_\theta(G_\phi(z_i))\right), \tag{5.12} $$

which is equivalent to minimizing $\mathbb{E}_{z\sim p_Z}\left[\log\left(1 - D_\theta(G_\phi(z))\right)\right]$.

5.2.3 Minimax value for D_θ

Then the training objectives for D_θ and G_ϕ lead us to the following formula for the minimax value of D_θ:

$$\overline{V_{D_\theta}} = \min_\phi \max_\theta \left(V_{D_\theta}(\theta, \phi)\right) \tag{5.13}$$

$$= \min_\phi \max_\theta \left(\mathbb{E}\left[\log D_\theta(x)\right] + \mathbb{E}\left[\log(1 - D_\theta(G_\phi(z)))\right]\right) \tag{5.14}$$

which we will solve in the next section by finding the minimax decision rule and minimax value for D_θ (see Theorem **??**), which means we are finding the Nash equilibrium. But this is not always easy to do in practice.

5.2.4 The Difficulty of Finding a Nash Equilibrium

It is not always easy to find a Nash equilibrium. In fact, if we look at the following example, we will see there is a tendency to oscillate around an equilibrium. This example was inspired by [**?**]. Let G control x and D control y.

Example. *Let $V(x, y) = xy$ be a value function and let the following be our game,*

$$\min_x \max_y V(x, y) = xy. \tag{5.15}$$

Then since $\frac{\partial V}{\partial x} = y$ and $\frac{\partial V}{\partial y} = x$ we obtain the following updates by following the gradient in the appropriate direction,

$$x^{(t+1)} \leftarrow x^{(t)} - \eta \cdot y; \tag{5.16}$$

$$y^{(t+1)} \leftarrow y^{(t)} + \eta \cdot x; \tag{5.17}$$

where $\eta > 0$ is the learning rate. The Nash equilibrium for this game is a pair of strategies $s^* = (s_G, s_D)$. If G's strategy is $s_G^* = (0, 0, 0, \dots)$, then $V(s_G^*, s_D) = 0\ \forall s_D$, so in particular D cannot improve the value function. And if D takes the same strategy $s_D^* = (0, 0, 0, \dots)$, then there is nothing G can do to improve the value function from its perspective. Hence $s^* = (s_D^*, s_G^*)$ and $V(s_G^*, s_D^*) = 0$.

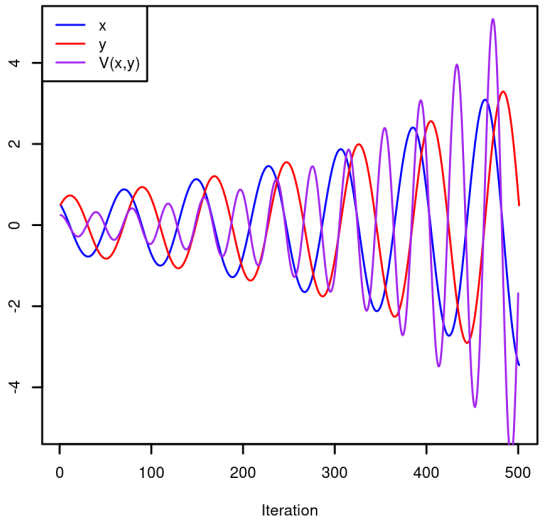

Figure 6: Oscillating around the optimal value

As we alternate updates, we observe oscillating behavior as depicted in Figure **??**. This oscillation has been observed in GAN training.

5.2.5 Two Value Functions

In [**?**], the paper introduced one value function as part of the theoretical discussion, but when it came to training the algorithm, they used a different value function which avoided the above issues. The first value function (equation **??**) is hard to train and often leads to vanishing gradients, which means the generator has nothing from which to learn. The decoupled objective functions

$$\max_{\theta} \left(\mathbb{E}\left[\log D_\theta(x)\right] + \mathbb{E}\left[\log(1 - D_\theta(G_\phi(z)))\right] \right) \tag{5.18}$$

$$\max_{\phi} \mathbb{E}\left[\log(D_\theta(G_\phi(z)))\right] \tag{5.19}$$

have the same fixed points as the previous one, but the gradients of the loss with respect to the generator are stronger, therefore providing better learning material for the generator. With equation **??** as the objective functions, the GAN algorithm is no longer a zero-sum game.

5.3 Generative Adversarial Networks

The full algorithm is presented below.

Algorithm 1: Generative Adversarial Networks

Let $\eta > 0$, *be the learning rate.*
Let $T > 0$, *be the number of training iterations.*
Let $D_\theta : \Theta \times \mathcal{X} \mapsto [0, 1]$
Let $G_\phi : \Phi \times \mathcal{Z} \mapsto \mathcal{X}$
while $t < T$ **do**
    Let $z = \{z_1, \ldots, z_m\}$, *where each* z_i *is sampled from* $(\mathcal{Z}, p_\mathcal{Z})$ *This batch will be used in the optimization of* D_θ.
    Let $x = \{x_1, \ldots, x_m\}$, *where each* x_i *is from the training data.*
    Let $\theta \leftarrow \theta + \eta \cdot \nabla_\theta \frac{1}{m} \sum_{i=1}^{m} \left[\log D_\theta\left(x_i\right) + \log\left(1 - D_\theta\left(G_\phi\left(z_i\right)\right)\right)\right]$, *i.e. we update the discriminator's parameters* θ *by ascending the gradient of* $V(\phi, \theta)$ *with respect to* θ.
    Let $z = \{z_1, \ldots, z_m\}$, *we sample a new batch from* $(\mathcal{Z}, p_\mathcal{Z})$. *This batch will be used in the optimization of* G_ϕ.
    $\phi \leftarrow \phi - \eta \cdot \nabla_\phi \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D_\theta\left(G_\phi\left(z_i\right)\right)\right)$, *i.e. we update the generator's parameters* ϕ *by descending the gradient of* $V(\phi, \theta)$ *with respect to* ϕ.
end

In the original paper, the discriminator is said to be optimized k times before the generator is optimized, however $k = 1$, so we chose to leave out that inner training loop in favour of readability and conceptual clarity.

5.4 Discussion

The minimax strategy for D can be thought of as follows: if D assumes G has done its worst, then D should assume that G is able to produce data points that are indistinguishable from the real thing. If that is the case, to minimize loss, D should return $\frac{1}{2}$ for all data points, which happens to be the maximum entropy distribution over the two states (real or synthetic). If $D(x) = \frac{1}{2}\ \forall x$, there is nothing G can do to improve its situation. The next section will show $D(x) = \frac{1}{2}\ \forall x$ is indeed the optimal strategy for D.

Section 6

Information Theory

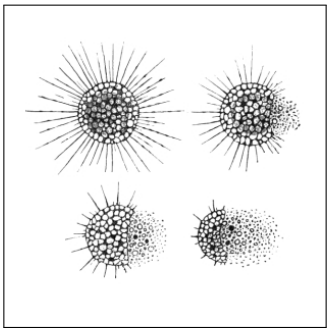

Figure 7: Conscious Entities, Peter Hankins

Information theory was born in 1948 with the publication of *A Mathematical Theory of Communication* by Claude Shannon (1916 – 2001). Shannon was inspired in part by earlier work by Boltzmann and Gibbs in thermodynamics and by Hartley and Nyquist at Bell, see [**?**]. Most of the theory and applications of information theory (compression, coding schemes, data transfer over noisy channels) are outside the scope of this thesis, but there are certain information theoretic quantities used regularly in machine learning, so it is useful to discuss them now.

Remark. *The information we talk about is restricted to the information about the probability distribution over the elementary outcomes, not information about the content of the outcomes. The significance of probability is that it tells us how certain we can be when making inference. The most important information in this regard is found in the probability distribution over the possible outcomes.*

Information theory is quite useful for deep learning. If we think of neural nets as noisy channels, the need for this theory becomes even more obvious. In [**?**], David Mackay said "brains are the ultimate compression and communication systems. And the state-of-the-art algorithms for both data compression and error-correcting codes use the same tools

as machine learning". Furthermore, "we might anticipate that the best data compression algorithms will result from the development of artificial intelligence methods".

The most fundamental quantity in information theory is entropy. Before we state the formal definition of entropy, we will motivate it as a measure of uncertainty by walking through its derivation. We will define a function η as a measure of uncertainty and we will derive entropy as a function based on the requirements it must satisfy using η as a starting point.

Definition 6.1. *Let $(\mathcal{X}, p)$ be a discrete probability space. We define* uncertainty *to be a real-valued function $\eta(\cdot) : \mathcal{X} \mapsto \mathbb{R}^+$ which depends only on the probabilities of the elementary outcomes and satisfies the following:*

(i) If an outcome x is guaranteed to occur, then there is no uncertainty about it and $\eta(x) = 0$;

(ii) For any two outcomes x, x', we have $p(x) < p(x') \iff \eta(x) > \eta(x')$;

(iii) For any two independent outcomes, x, x', the uncertainty of their joint occurrence, is the sum of their uncertainties, i.e. $\eta(x \cdot x') = \eta(x) + \eta(x')$.

Remark. *This definition is a modification of one given by [?].*

It should not be a surprise that it is new information we are interested in, since that is what reduces uncertainty. Common outcomes provide less information than rare outcomes, which means η should be inversely proportional to the probability of the outcome.

$$\eta(x) \propto \frac{1}{p(x)} \tag{6.1}$$

Since η must satisfy $\eta(x \cdot x') = \eta(x) + \eta(x')$, we must define η in terms of the logarithm. This is because the probability of two independent outcomes is the product of their probabilities whereas we want information to be additive. Thus,

$$\eta(x) \approx \log \frac{1}{p(x)}. \tag{6.2}$$

For probability distributions, we need a measure of uncertainty that says, on average, how much uncertainty is contained in $(\mathcal{X}, p)$. We need to weight the calculation by the probability of observing each outcome. This means what we are really seeking is

a measure on the probability distribution over $\mathcal{X}$. We adjust the notation, using the capital eta, which resembles the Latin H. Thus,

$$\mathcal{H}(p) = \sum_{x \in \mathcal{X}} p(x) \log \frac{1}{p(x)}. \tag{6.3}$$

This is what we will call entropy, a measure on the average amount of surprise associated to outcomes from $(\mathcal{X}, p)$. Entropy is maximized when we cannot say with any confidence if an outcome will occur. This upper bound occurs when the probabilities over the set of possible outcomes are uniformly distributed.

$$\mathcal{H}(p) \leq \log |\mathcal{X}| \tag{6.4}$$

We can also think of entropy as how much information, measured in binary information units (bits), is required to describe outcomes drawn from $(\mathcal{X}, p)$. The way to understand this last part is the logarithm tells us how many bits we need to describe this uncertainty, since

$$\log_2 \frac{1}{p(x)} = n \iff 2^n = \frac{1}{p(x)}. \tag{6.5}$$

However, any logarithm can be used. Base e and base 10 are also commonly used.

Definition 6.2. *Let $(\mathcal{X}, p)$ be any discrete probability space. The* entropy *of a probability distribution p with mass function p, denoted by $\mathcal{H}(p)$, is the average amount of uncertainty found in elementary outcomes from $(\mathcal{X}, p)$. We write*

$$\mathcal{H}(p) = -\mathbb{E}_{x \sim p} \left[\log p(x) \right]. \tag{6.6}$$

The entropy of a probability distribution tells us how much variation we should expect to see in samples drawn from $(\mathcal{X}, p)$. The probability distribution with maximum entropy is the uniform distribution since all outcomes are equally surprising.

Figure **??** depicts the entropy of a probability distribution over two states as a function of the symmetry of the distribution. As the probability of heads $p(H)$ approaches 0 or 1, we see the uncertainty vanishes, and uncertainty is maximized when probability is equally distributed over heads and tails.

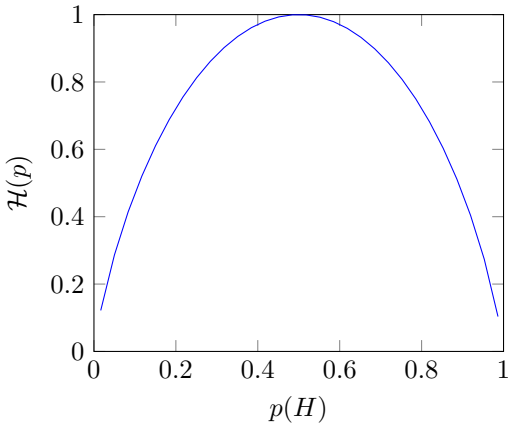

Figure 8: Entropy of a coin toss as a function of the symmetry of $p(H)$.

Example. *The entropy of the probability distribution corresponding to a fair coin toss is 1 bit, and the entropy of m tosses is m bits. If there are two states of equal probability, then we need 1 bit and if we have 3 states of equal probability, we need 1.584963 bits.*

Entropy-based Quantities

We include a definition of a metric below in order to make clear the distinction between it and a divergence, which will be defined afterwards.

Definition 6.3. *A* metric *on a set $\mathcal{X}$ is a function $d(\cdot,\cdot) : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}^+$ such that, $\forall x, y, z \in \mathcal{X}$:*

1. $d(x, y) \geq 0$, *and* $d(x, y) = 0 \iff x = y$
2. $d(x, y) = d(y, x)$
3. $d(x, z) \leq d(x, y) + d(y, z)$

Remark. *A divergence is a weaker notion than that of distance. A divergence need not be symmetric nor satisfy the triangle inequality.*

Definition 6.4. *Let $\mathcal{P}$ be any space of probability distributions over any finite set $\mathcal{X}$ such that all $P \in \mathcal{P}$ have the same support. A* divergence *on $\mathcal{P}$ is a function, $\mathcal{D}(\cdot||\cdot) : \mathcal{P} \times \mathcal{P} \mapsto \mathbb{R}^+$, such that $\forall p, q \in \mathcal{P}$ the following conditions are satisfied*

(i) $\mathcal{D}(p||q) \geq 0$

(ii) $\mathcal{D}(p||q) = 0 \iff p = q$.

Definition 6.5. *The* Kullback-Leibler divergence *is a measure of how different a probability distribution is from a second, reference probability distribution. It is also known by the following names:* relative entropy, directed divergence, information gain *and* discrimination information. *It is defined by*

$$\mathcal{D}_{KL}(p||q) = \sum_{x \in \mathcal{X}} p(x) \log \frac{p(x)}{q(x)}. \tag{6.7}$$

If p *and* q *have the same support, then* $\mathcal{D}_{KL}(p||q) = 0$ *if and only if* $p = q$.

Remark. *The Kullback-Leibler divergence is defined only if* p *is absolutely continuous with respect to* q*, i.e.* $\forall x \; q(x) = 0 \implies p(x) = 0$*. When* $p(x) = 0$*,* $\mathcal{D}_{KL}(p||q) = 0$ *since* $\lim_{x \to 0} x \log x = 0$.

Theorem 6.6. *For a closed convex set* $E \subset \mathcal{P}$*, where* $\mathcal{P}$ *is the space of all probability distributions over a finite set* $\mathcal{X}$*, and for a distribution* $Q \notin E$*, let* $P^* \in E$ *be defined by* $p^* = \arg\min_{P \in E} \mathcal{D}_{KL}(p||q)$*, then*

$$\mathcal{D}_{KL}(p||q) \geq \mathcal{D}_{KL}(p||p^*) + \mathcal{D}_{KL}(p^*||q)\,. \tag{6.8}$$

The interested reader can consult Theorem 11.6.1 in [?].

Remark. *The log-likelihood ratio test is used in comparing the goodness-of-fit of one statistical model over another. The Kullback-Leibler divergence of* p *and* q *is the average of the log-likelihood ratio test with respect to probabilities defined by* p*. For two models* $p(x) = f(x|\theta)$ *and* $q(x) = f(x|\phi)$*, the log-likelihood ratio test is*

$$\lambda(x) = \log \frac{\prod_{x \in \mathcal{X}} p(x)}{\prod_{x \in \mathcal{X}} q(x)} \tag{6.9}$$

$$= \log \prod_{x \in \mathcal{X}} \frac{p(x)}{q(x)} \tag{6.10}$$

$$= \sum_{x \in \mathcal{X}} \log \frac{p(x)}{q(x)} \tag{6.11}$$

and the average with respect to p *is*

$$\mathbb{E}_{x \sim p}\left[\lambda(x)\right] = \sum_{x \in \mathcal{X}} p(x) \log \frac{p(x)}{q(x)}. \tag{6.12}$$

Another way to think of GAN training is as fitting D and G to the data via optimizing a goodness-of-fit test since

$$\min_{\phi} \max_{\theta} \frac{1}{n} \sum_{i=1}^{n} \log D_\theta(x_i) + \frac{1}{n} \sum_{i=1}^{n} \log \left(1 - D_\theta(G_\phi(z_i))\right) \tag{6.13}$$

has the same fixed point as

$$\min_{\phi} \max_{\theta} \frac{1}{n} \sum_{i=1}^{n} \left[\log D_\theta(x_i) - \log \left(D_\theta(G_\phi(z_i))\right)\right] \tag{6.14}$$

$$= \min_{\phi} \max_{\theta} \frac{1}{n} \sum_{i=1}^{n} \log \left(\frac{D_\theta(x_i)}{D_\theta(G_\phi(z_i))} \right) \tag{6.15}$$

which is the Kullback-Leibler divergence or the average log-likelihood ratio test. Since $\forall x \ D_\theta(x) \in [0, 1]$, we can infer that when D_θ is optimized, it will place a larger amount of mass on x than on $G_\phi(z)$.

The term *information gain* refers to one interpretation of the Kullback-Leibler divergence. Specifically $\mathcal{D}_{\text{KL}}(p||q)$ is the amount of information gained about the data when q is used to model the data, rather than p. Equivalently, the amount of information lost when q is used to approximate p.

Definition 6.7. *The* reverse Kullback-Leibler divergence *is the asymmetrical counterpart.*

$$\mathcal{D}_{KL}(q||p) = \sum_{x \in \mathcal{X}} q(x) \log \frac{q(x)}{p(x)}. \tag{6.16}$$

The reverse Kullback-Leibler divergence is the average of the log-likelihood ratio test taken with respect to the model $q(x)$,

$$\mathbb{E}_{x \sim q} \left[\lambda(x)\right] = \sum_{x \in \mathcal{X}} q(x) \log \frac{q(x)}{p(x)}. \tag{6.17}$$

Minimizing the reverse Kullback-Leibler divergence is not equivalent to maximum likelihood methods.

The Kullback-Leibler divergence is related to another quantity used quite often in machine learning: cross entropy.

Definition 6.8. *The* cross entropy *of p and q (for a given data set) is the total amount*

of uncertainty incurred by modelling the data with q rather than p.

$$\mathcal{H}(p,q) = -\sum_{x\in\mathcal{X}} p(x)\log q(x) = -\mathbb{E}_{x\sim p}\left[\log q(x)\right]. \tag{6.18}$$

Lemma 6.9. *The cross entropy of p and q is the sum of the entropy of p and the Kullback-Leibler divergence of p and q.*

Proof.

$$\begin{aligned}
\mathcal{H}(p,q) &= -\sum_{x\in\mathcal{X}} p(x)\log q(x) && (6.19)\\
&= -\sum_{x\in\mathcal{X}} p(x)\log p(x) + \sum_{x\in\mathcal{X}} p(x)\log p(x) - \sum_{x\in\mathcal{X}} p(x)\log q(x) && (6.20)\\
&= -\sum_{x\in\mathcal{X}} p(x)\log p(x) + \sum_{x\in\mathcal{X}} p(x)\log\frac{p(x)}{q(x)} && (6.21)\\
&= \mathcal{H}(p) + \mathcal{D}_{\mathrm{KL}}(p||q) && (6.22)
\end{aligned}$$

□

This tells us the lower bound for cross entropy must be the entropy of the probability distribution p over $\mathcal{X}$. Thus, cross entropy is the uncertainty induced by assuming the wrong probability distribution over the data. The additional uncertainty is captured by the Kullback-Leibler divergence. Cross entropy is not symmetric since $\mathcal{H}(q,p) = \mathcal{H}(q) + \mathcal{D}_{\mathrm{KL}}(q||p)$.

As shown in [**?**], the generator minimizes an approximation of the Jensen-Shannon divergence.

Definition 6.10. *Let $p(x)$ and $q(x)$ be any two probability distributions over any space $\mathcal{X}$. The* Jenson-Shannon divergence *of $p(x)$ and $q(x)$ is a symmetrization of the Kullback-Leibler divergence of $p(x)$ and $q(x)$ over $\mathcal{X}$.*

$$\mathcal{D}_{JS}(p||q) = \frac{1}{2}\mathcal{D}_{KL}\left(p\middle\|\frac{p+q}{2}\right) + \frac{1}{2}\mathcal{D}_{KL}\left(q\middle\|\frac{p+q}{2}\right) \tag{6.23}$$

Remark. *The Jensen-Shannon divergence is the average of the Kullback-Leibler divergence and the reverse Kullback-Leibler divergence.*

Theorem 6.11. *The square root of the Jensen–Shannon divergence is a metric.*

Proof. See [**?**]. □

Mutual Information and other Measures

Information theory provides us with a measure of dependency, or at least how much information about one probability distribution is contained in another distribution. The following measure are defined in terms of random variables denoted by upper case letters such as X and Y.

Definition 6.12. *Let $(\mathcal{X}, p)$ and $(\mathcal{Y}, q)$ be any two finite probability spaces ($\mathcal{X}$ and $\mathcal{Y}$ need not be distinct) and consider two random variables $X \sim p$ and $Y \sim q$ with joint probability mass function γ and marginal probability mass functions $\pi_p \circ \gamma = p$ and $\pi_q \circ \gamma = q$. The* mutual information *$I(X;Y)$ is the Kullback-Leibler divergence of γ and the product of p and q, in other words*

$$\mathcal{I}(X;Y) = \sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}} \gamma(x,y) \log \frac{\gamma(x,y)}{p(x)q(y)} \tag{6.24}$$

Theorem 6.13. *If the random variables X and Y are independent, then $\gamma(x,y) = p(x)q(y)$ and $\mathcal{I}(X;Y) = 0$.*

Remark. *Mutual information is a measure of the amount of information contained in one probability distribution about another and makes for a useful measure of statistical dependence.*

Remark. *Mutual information can also be defined in terms of conditional entropy, defined in terms of random variables X and Y,*

$$\mathcal{I}(X;Y) = \mathcal{H}(X) - \mathcal{H}(X|Y) = \mathcal{H}(Y) - \mathcal{H}(Y|X) \tag{6.25}$$

where $\mathcal{H}(X|Y)$ is the conditional entropy of X given that Y has occurred. In this form the mutual information can be interpreted as the information contained in one probability distribution minus the information contained in the distribution when the other distribution is known.

The relationship different information theoretic quantities is depicted in the Venn diagram in Figure (**??**).

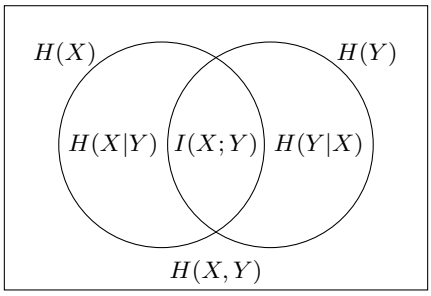

Figure 9: Venn Diagram of Information-Theoretic Quantities

6.1 Information-Theoretic View of GANs

We can now write a bit more about $V(\phi, \theta) = \mathbb{E}_{x\sim p^*}[\log D_\theta(x)] + \mathbb{E}_{z\sim p_{\mathcal{Z}}}[\log(1 - D_\theta(G_\phi(z)))]$ from the perspective of information theory. The proposition presented below does not consider the limiting behaviour of D and G. Rather, we consider the behaviour at each step of the algorithm. Section **??** considers the limiting behaviour of the dynamics between D and G.

We claim that the goal of GAN training is to minimize the Kullback-Leibler divergence of the target probability distribution p^* over the data and the distribution of mass the discriminator D_θ assigns to these data points. Additionally, GAN training also maximizes the Kullback-Leibler divergence of the generator's prior probability distribution $p_{\mathcal{Z}}$ and the distribution of mass the discriminator assigns to synthetic data points $D_\theta(G_\phi(z))$. Lastly, GAN training minimizes the Kullback-Leibler divergence of the generator's prior distribution $p_{\mathcal{Z}}$ and the amount of mass the discriminator $D_\theta(G_\phi(z))$ assigns to synthetic data points.

Remark. *The Kullback-Leibler divergence in this context can be thought of as how much more surprised we will be by outcomes drawn from $(\mathcal{X}, p)$ since we are using an approximation q of the true probability distribution p over the set $\mathcal{X}$.*

Proposition 6.14. *If we consider*

$$\min_\phi \max_\theta \mathbb{E}_{x\sim p^*}[\log D_\theta(x)] + \mathbb{E}_{z\sim p_{\mathcal{Z}}}[\log(1 - D_\theta(G_\phi(z)))] \tag{6.26}$$

from an information theoretic perspective, we can infer the training objectives of the GAN algorithm are to find

(i) $\theta \in \Theta$ to minimize $\mathcal{D}_{KL}(p^*(x)||D_\theta(x))$ and maximize $\mathcal{D}_{KL}(p_{\mathcal{Z}}(z)||D_\theta(G_\phi(z)))$;

(ii) $\phi \in \Phi$ to minimize $\mathcal{D}_{KL}(p_{\mathcal{Z}}(z) \,||\, D_\theta(G_\phi(z)))$.

Proof. The first term of $V(\phi, \theta)$ is the negative cross entropy of p^* and the distribution induced by $D_\theta(x)$,

$$-\mathcal{H}(p^*, D_\theta) = \sum_{x \in \mathcal{X}} p^*(x) \log(D_\theta(x)) = \mathbb{E}_{x \sim p^*} \left[\log(D_\theta(x))\right]. \tag{6.27}$$

Since we train the discriminator to maximize (**??**) we can think of this as minimizing its negative, which is equivalent to minimizing the cross entropy of p^* and the distribution induced by $D_\theta(x)$, which occurs when the Kullback-Leibler divergence of p^* and $D_\theta(x)$ is 0, since

$$\begin{aligned}
\mathcal{H}(p^*, D_\theta) &= -\sum_{x \in \mathcal{X}} p^*(x) \log D_\theta(x) && (6.28)\\
&= -\sum_{x \in \mathcal{X}} p^*(x) \log p^*(x) + \sum_{x \in \mathcal{X}} p^*(x) \log p^*(x) - \sum_{x \in \mathcal{X}} p^*(x) \log D_\theta(x) && (6.29)\\
&= -\sum_{x \in \mathcal{X}} p^*(x) \log p^*(x) + \sum_{x \in \mathcal{X}} p^*(x) \log \frac{p^*(x)}{D_\theta(x)} && (6.30)\\
&= H(p^*) + \mathcal{D}_{\mathrm{KL}}(p^* \,||\, D_\theta). && (6.31)
\end{aligned}$$

Since we do not touch p^* during training, the only way to minimize $\mathcal{H}(p^*, D_\theta)$ is to train D_θ to minimize $\mathcal{D}_{\mathrm{KL}}(p^* \,||\, D_\theta)$, which occurs when $p^* = D_\theta$. Therefore, when D_θ has been optimized, $D_\theta(x)$ will return the probability that x was sampled from p^*, which equals p^* when D_θ is optimal. The second term of (**??**),

$$\mathbb{E}_{z \sim p_{\mathcal{Z}}} \left[\log(1 - D_\theta(G_\phi(z)))\right] = \sum_{z \sim p_{\mathcal{Z}}} p_{\mathcal{Z}}(z) \log(1 - D_\theta(G_\phi(z))) \tag{6.32}$$

was maximized by D_θ, which is equivalent to D_θ maximizing

$$-\sum_{z \sim p_{\mathcal{Z}}} p_{\mathcal{Z}}(z) \log(D_\theta(G_\phi(z))), \tag{6.33}$$

which is the cross entropy of $p_{\mathcal{Z}}$ and $D_\theta(G_\phi(z))$. This means D_θ is trying to make $p_{\mathcal{Z}}$ and $D_\theta(G_\phi(z))$ to be as different as possible. And given a fixed discriminator D_θ, we train the G_ϕ to minimize the same equation, which can be expressed equivalently as

minimizing

$$-\sum_{z\sim p_{\mathcal{Z}}} p_{\mathcal{Z}}(z)\log(D_\theta(G_\phi(z))) \tag{6.34}$$

the cross entropy of $p_{\mathcal{Z}}$ and $D_\theta(G_\phi(z))$. □

The interpretation of the above theorem is that the generator wants the discriminator's decision on the generated data to be as uninformative as random noise. The discriminator wants the distribution of its decision over the training data to match the empirical probability distribution, while at the same time, the discriminator wants its decision on the generated data to be more informative than noise.

Next, we look at the optimization steps as the dynamics between approximating a divergence and minimizing the same divergence.

6.2 Optimization Dynamics

We discussed training D_θ and G_ϕ from the perspective of game theory in Section **??**. Now let us look at these results more rigorously and with some actual calculations. Throughout this section, we will use the following notation. Let $\mathcal{X}$ be our data and p^* the true probability distribution over $\mathcal{X}$. Let $\tilde{\mathcal{X}}$ be the generated data and p_ϕ be the distribution over $\tilde{\mathcal{X}}$ induced by G_ϕ. Let X be a random sample from $(\mathcal{X}, p^*)$, $\tilde{X}$ be a random sample from $(\tilde{\mathcal{X}}, p_\phi)$, and let Z be a random sample from the prior probability space $(\mathcal{Z}, p_{\mathcal{Z}})$.

Throughout this section we will make use of not only

$$V(\phi,\theta)(X,Z) = \sum_{x\in X} p^*\log D_\theta(x) + \sum_{z\in Z} p_{\mathcal{Z}}\log\left(1 - D_\theta(G_\phi(z))\right), \tag{6.35}$$

but of two equivalent variations as well. The first variation, $\tilde{V}(\phi,\theta)(X,\tilde{X})$, is obtained by changing the second argument of $V(\phi,\theta)$ from a sample of noise Z to a sample from the generator $G_\phi(Z) = \tilde{X}$. We compute the second expectation with respect to p_ϕ, i.e. we have

$$\tilde{V}(\phi,\theta)(X,\tilde{X}) = \sum_{x\in X} p^*(x)\log D_\theta(x) + \sum_{\tilde{x}\in\tilde{X}} p_\phi(\tilde{x})\log\left(1 - D_\theta(\tilde{x})\right). \tag{6.36}$$

For the second variation, $\mathcal{V}$, we are going to embed our data into $U \subset \mathbb{R}^2$, where each u_i is associated with a pair $(x_i, \tilde{x}_i)$. Then we can define the following function

$$\mathcal{V}(\phi, \theta)(U) = \sum_{u \in U} p^*(u) \log D_\theta(u) + p_\phi(u) \log(1 - D_\theta(u)), \tag{6.37}$$

which we will use as short-hand for

$$\sum_{u \in U} (p^* \circ \pi)(u) \log(D_\theta \circ \pi)(u) + (p_\phi \circ \tilde{\pi})(u) \log(1 - (D_\theta \circ \tilde{\pi})(u)), \tag{6.38}$$

where π and $\tilde{\pi}$ are projections, i.e. $(f \circ \pi)(u) = (f \circ \pi)((x, \tilde{x})) = f(x)$ and $(f \circ \tilde{\pi})(u) = (f \circ \tilde{\pi})((x, \tilde{x})) = f(\tilde{x})$.

Theorem 6.15. *For any fixed G_ϕ, D_θ maximizes $V(\phi, \theta)$ by playing its minimax decision rule and in doing so returns the following function*

$$\arg\max_{D_\theta} V(\phi, \theta)(\cdot) = \frac{p^*(\cdot)}{p^*(\cdot) + p_\phi(\cdot)}. \tag{6.39}$$

Proof. We compute the derivative of $\mathcal{V}(\phi, \theta)$ with respect to θ,

$$\frac{\partial \mathcal{V}(\phi, \theta)(U)}{\partial \theta} = \sum_{u \in U} \left[p^*(u) \frac{\frac{\partial D_\theta(u)}{\partial \theta}}{D_\theta(u)} - p_\phi(u) \frac{\frac{\partial D_\theta(u)}{\partial \theta}}{(1 - D_\theta(u))} \right], \tag{6.40}$$

and when we let the derivative equal zero (looking only at the summand for clarity of notation) we can uncover $D^*_\theta(u)$, by solving for $D_\theta(u)$

$$\frac{p^*(u)}{D_\theta(u)} = \frac{p_\phi(u)}{(1 - D_\theta(u))} \iff D^*_\theta(u) = \frac{p^*(u)}{p_\phi(u) + p^*(u)}. \tag{6.41}$$

Hence, for any fixed generator G_ϕ, $\mathcal{V}(\phi, \theta)(U)$ is maximized when D_θ takes the following action

$$D^*_\theta(u) = \frac{p^*(u)}{p^*(u) + p_\phi(u)}. \tag{6.42}$$

Hence $\arg\max_{D_\theta} V(\phi, \theta)(\cdot, \cdot) = \frac{p^*(\cdot)}{p^*(\cdot) + p_\phi(\cdot)}$. □

Now that we have the minimax decision rule, we can place it in the value function

$\tilde{V}$, to uncover $\max_{D_\theta} \tilde{V}(\phi, \theta)(X, \tilde{X})$, equal to

$$\mathbb{E}_{x \sim p^*} \left[\log \frac{p^*(x)}{p_\phi(x) + p^*(x)} \right] + \mathbb{E}_{\tilde{x} \sim p_\phi} \left[\log \frac{p_\phi(\tilde{x})}{p_\phi(\tilde{x}) + p^*(\tilde{x})} \right], \tag{6.43}$$

which is the sum of two Kullback-Leibler divergences (which is related to the Jenson-Shannon divergence of p^* and p_ϕ) i.e. $\max_{D_\theta} V(\phi, \theta)(X, Z)$ is equal to

$$\mathcal{D}_{\text{KL}}(p^* \,||\, p_\phi + p^*) + \mathcal{D}_{\text{KL}}(p_\phi \,||\, p_\phi + p^*)\,. \tag{6.44}$$

Lemma 6.16. *The minimax value for D_θ, i.e. $\min_\phi \max_\theta V(\phi, \theta)(X, Z)$, is $-\log 4$.*

Proof. The training goal for G_ϕ is to learn p^*, so the optimal G_ϕ is $G_\phi^* = p^*$. If this optimal G_ϕ^* is place inside (**??**), we obtain

$$\begin{aligned} V(\phi^*, \theta)(X, \tilde{X}) &= \sum_{x \in X} p^*(x) \log \frac{p^*(x)}{2p^*(x)} + \sum_{\tilde{x} \in \tilde{X}} p^*(\tilde{x}) \log \frac{p^*(\tilde{x})}{2p^*(\tilde{x})} && (6.45) \\ &= \sum_{x \in X} p^*(x) \log \frac{1}{2} + \sum_{\tilde{x} \in \tilde{X}} p^*(\tilde{x}) \log \frac{1}{2} && (6.46) \\ &= -\log 4 && (6.47) \end{aligned}$$

Hence, the minimax value is $-\log 4$. □

Theorem 6.17. *When D_θ is optimal, minimizing $V(\phi, \theta)$ as a function of ϕ is equivalent to optimizing the Jensen-Shannon divergence.*

Proof. We can make the relationship between (**??**) and the Jensen-Shannon divergence by adding and subtracting $\log 4$ from (**??**), which yields

$$V(\phi, \theta^*) = 2 \cdot \mathcal{D}_{\text{JS}}(p^* \,||\, p_\phi) - \log 4, \tag{6.48}$$

which is what we wanted. See Appendix **??** for complete details. □

6.3 Discussion

This section included two information theoretic perspectives on GAN training. Theorem **??** considered the theoretical, limiting behaviour of GANs and **??** considered what happens at each step of the optimization.

As for Theorem **??**, the measure that G minimizes changes at each training step since D forces $V(\phi, \theta)$ into a better approximation of the Jensen-Shannon divergence one step at a time by performing the actions enumerated in Theorem **??**. It is useful to consider what happens at each step of training. When we do that, we observe the generator and discriminator minimizing and maximizing the Kullback-Leibler divergence to optimize the fit of D and G to $(\mathcal{X}, p^*)$.

Section 7

Optimal Transport

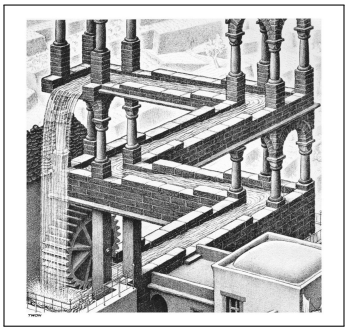

Figure 10: M.C. Escher, "Waterfall", 1961.

Section **??** uncovered how the generator minimizes an approximation of the Jensen-Shannon divergence of p^* and p_ϕ which the discriminator approximated at each training step. In this section we discuss the limitations of $V(\phi, \theta)$ as the GAN objective function which come from the metric topology induced by the Jensen-Shannon divergence. To this end, we provide an introduction to some relevant concepts from topology in order to understand the aforementioned limitations.

We also provide an introduction to optimal transport and what is commonly referred to as the earth mover or Wasserstein distance in order to properly cover a variant of the GAN algorithm called the Wasserstein GAN (WGAN). The WGAN was introduced by [**?**] from the Courant Institute of Mathematical Sciences as an attempt to overcome certain obstacles in GAN training.

The moral of this section is that the choice of distance function with which to furnish a space $\mathcal{X}$ has profound consequences on properties of continuity and therefore on the ability of sequences of probability distributions to converge within $\mathcal{X}$.

A topological space is the most general notion of a mathematical space that allows for the definition of concepts like continuity and convergence. Other spaces such as

manifolds and metric spaces are specializations of topological spaces with extra structure or constraints. A space $\mathcal{X}$ can be furnished with various different distance functions. The open balls form the base for a topology on $\mathcal{X}$, which makes $\mathcal{X}$ into a topological space.

Definition 7.1. *Let $\mathcal{X}$ be any set. A* topology *$\mathcal{T}$ on $\mathcal{X}$ is a collection of subsets of $\mathcal{X}$, each called an open set, such that*

(i) the empty set and the set containing all of $\mathcal{X}$ are open;

(ii) the intersection of finitely many open sets is an open set;

(iii) the union of any collection of open sets is an open set.

The set $\mathcal{X}$ along with a topology $\mathcal{T}$ on $\mathcal{X}$ is called a topological space*.*

The notion of the open ball is fundamental to the topology of a metric space. Useful topological definitions (useful from the perspective of the practising statistician) can be formed from the notion of the open ball.

Definition 7.2. *Let $(\mathcal{X}, d)$ be a metric space. An* open ball *of radius $r \in \mathbb{R}^+$ around the point $x_0 \in \mathcal{X}$ is the set*

$$\mathcal{B}_d(x_0, r) = \{x \in \mathcal{X} : d(x, x_0) < r\}. \tag{7.1}$$

That is to say $\mathcal{B}_d(x_0, r)$ is the set of all points in $\mathcal{X}$ that are within r distance from x_0.

Definition 7.3. *Let $(\mathcal{X}, d)$ be a metric space. The topology generated by the basis of open balls $\mathcal{B} = \{\mathcal{B}_d(x, r) \mid x \in \mathcal{X}, r > 0\}$ is called the* topology induced by *d and is referred to as a* metric topology*.*

Different metrics induce different topologies which are characterized by the quality of granularity.

Theorem 7.4. *Let d and d' be metrics on a set $\mathcal{X}$, and let $\mathcal{T}$ and $\mathcal{T}'$ be the respective topologies they induce. $\mathcal{T}'$ is* finer *than $\mathcal{T}$ if and only if for each $x \in \mathcal{X}$ and $\epsilon > 0$, there exists a $\delta > 0$ such that $\mathcal{B}_{d'}(x, \delta) \subset \mathcal{B}_d(x, \epsilon)$.*

Definition 7.5. *Let $\mathcal{X}$ be any finite set of elementary outcomes and $\mathcal{P}$ be the space of all probability distributions over $\mathcal{X}$ with equal support. Let $d : \mathcal{P} \times \mathcal{P} \mapsto \mathbb{R}$ be a metric on this space. A sequence of probability distributions $(P_n)_{n \in \mathbb{N}}$* converges *to a probability distribution P if $d(P_n, P) \to 0$ as $n \to \infty$.*

The Kullback-Leibler and Jensen-Shannon divergences induce a coarser topology than the topology induced by the earth mover distance. The interested reader can see the proof of this in [**?**]. To ease convergence, we want to place a finer topology on $\mathcal{P} \times \mathcal{P}$. A finer topology means we can pack more open sets over $\mathcal{P} \times \mathcal{P}$, which makes it easier to define a continuous map from $\mathcal{P} \times \mathcal{P}$ to $\mathbb{R}^+$. If a metric d induces a finer topology on a space than another metric d', then we say d is a weaker notion of distance than d'.

We can think of continuity and convergence in more than one way. Below we include the relevant definitions from both the topological and metric space points of view.

Definition 7.6. *A function $f : \mathbb{R} \mapsto \mathbb{R}$ is* continuous *if for every $x_0 \in \mathbb{R}$, $\epsilon > 0$, there exists $\delta > 0$ such that if $|x - x_0| < \delta$, then $|f(x) - f(x_0)| < \epsilon$.*

Continuous functions between topological spaces preserve proximity, i.e. a continuous function maps points that are close together in one space to points that are close together in the other space.

Definition 7.7. *In a topological space $(\mathcal{X}, \mathcal{T})$, a sequence of points* converges *to $x \in \mathcal{X}$ if for every neighbourhood U of x, there is an $N \in \mathbb{N}$ such that $x_n \in U$ for all $n \geq N$.*

The following is the topological definition of continuity. Briefly, f is continuous if the preimage of every open set is open.

Definition 7.8. *Given a function $f : \mathcal{X} \mapsto \mathcal{Y}$ and a point $y \in \mathcal{Y}$, define $f^{-1}(y)$, the* preimage *of y, to be the set $\{x \in \mathcal{X} \mid f(x) = y\}$. For any set $A \subset \mathcal{Y}$, the preimage of A is $f^{-1}(A) = \{x \in \mathcal{X} \mid f(x) \in A\}$.*

Definition 7.9. *Let $\mathcal{X}$ and $\mathcal{X}'$ be topological spaces. A function $f : \mathcal{X} \mapsto \mathcal{X}'$ is* continuous *if $f^{-1}(V)$ is open in $\mathcal{X}$ for every open set V in $\mathcal{X}'$.*

The problem with the Kullback-Leibler and Jensen-Shannon divergences is that they are strong notions of distance. That means continuity of the loss function may be lost under certain commonly encountered circumstances in GAN training.

7.1 Limitations of the Kullback-Leibler Divergence

In [**?**], the authors use the example of learning parallel lines and here we present the same example. This is an example of what happens with the Kullback-Leibler and Jensen-Shannon divergences when we compare distributions over the same space but

with disjoint supports (see Definition **??** for more information on the Kullback-Leibler divergence).

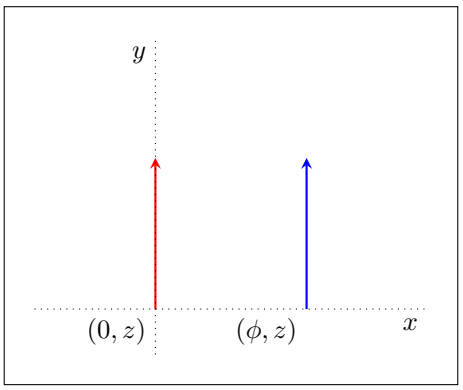

Figure 11: Parallel Lines

Example (Learning Parallel Lines)**.** *Let $\mathcal{X} = \mathbb{R}^2$ and let $p_0(z)$ be the distribution of pairs $(0, z) \subset \mathbb{R}^2, z \in [0, 1]$, i.e. we have a uniform distribution over the y-axis of $\mathbb{R}^2$ of length 1 starting at the origin. Let $G_\phi(z)$ be the distribution of pairs $(\phi, z) \subset \mathbb{R}^2$ (a generative model), where ϕ is the parameter that specifies the location of the distribution with respect to the x-axis. We want to train $G_\phi(z)$ to approximate $p_0(z)$, i.e. we want $\phi \to 0$.*

(i) If we use the Kullback-Leibler divergence to measure the distance between $p_0(z)$ and $G_\phi(z)$, we observe the following discontinuity between changes in the parameter ϕ and the divergence

$$\mathcal{D}_{KL}(p_0(z)||G_\phi(z)) = \mathbb{E}_{z\sim p_0(z)}\left[\log \frac{p_0(z)}{G_\phi(z)}\right]. \tag{7.2}$$

Since the expectation above is taken with respect to the distribution over the line $(0, z)$, the line will have measure zero with respect to the distribution $G_\phi(z)$. Thus $\mathcal{D}_{KL}(p_0(z)||G_\phi(z)) = \infty$, unless $\phi = 0$, then $\mathcal{D}_{KL}(p_0(z)||G_\phi(z)) = 0$. The same occurs with $\mathcal{D}_{KL}(G_\phi(z)||p_0(z))$.

(ii) The Jensen-Shannon divergence ends up being equally useless since

$$\mathcal{D}_{JS}(p_0(z)||G_\phi(z)) = \frac{1}{2}\mathbb{E}_{z\sim p_0(z)}\left[\log\frac{p_0(z)}{p_m(z)}\right] + \frac{1}{2}\mathbb{E}_{z\sim G_\phi(z)}\left[\log\frac{G_\phi(z)}{p_m(z)}\right] \tag{7.3}$$

$$= \frac{1}{2}\mathbb{E}_{z\sim p_0(z)}\left[\log 2\right] + \frac{1}{2}\mathbb{E}_{z\sim G_\phi(z)}\left[\log 2\right] \tag{7.4}$$

$$= \log 2, \tag{7.5}$$

where $p_m(z) = \frac{p_0(z)+G_\phi(z)}{2}$. *In the first term* $G_\phi(z) = 0$ *because* $z \sim p_0(z)$ *and in the second term* $p_0(z) = 0$ *because* $z \sim G_\phi(z)$ *i.e.* $\mathcal{D}_{JS}(p_0(z)||G_\phi(z)) = \log 2$, *unless* $\phi = 0$, *in which case* $\mathcal{D}_{JS}(p_0(z)||G_\phi(z)) = 0$.

The reason why this happens is a central topic in [**?**] and is due to the combination of the strong notion of distance of the Jensen-Shannon divergence and the artificially high dimensionality of most data sets. For example, a data set may live in $\mathbb{R}^n$ for some large n but there may only be variation of interest in a small number of dimensions. This is one of the ideas motivating dimensionality reduction algorithms like PCA and manifold learning algorithms in particular like Isomap [**?**].

GANs are often used in the generation of realistic looking images. If we consider an image to be a point in $\mathcal{X} = [0, 255]^{3\times H\times W}$, the space of 8-bit RGB images, and if we were to sample a point from this space, it would most likely look like noise. Thus, any data set of images would correspond to a very small subset, or data manifold, of $\mathcal{X}$. An informal definition of a manifold is more useful to our discussion than a formal one. When we say manifold, we mean a continuous geometrical structure with finite dimension (e.g. a line, a curve, a plane, a surface etc...) embedded inside a space of higher dimension than the manifold itself. Locally, manifolds resemble $\mathbb{R}^n$ for some n, i.e. they are locally flat.

Richard E. Bellman coined the term *the curse of dimensionality* in [**?**] to refer to the fact that when the dimensionality of the data increases, the volume of the space scales at an exponential rate and consequently data become effectively sparse. For instance, 10 points can be evenly arrange along the unit interval with 0.1 units of distance between them. If we want to cover the unit square with points 0.1 units of distance apart, we would need 100 points; for the unit cube? 1000 points. Each time we add a dimension, we need (in our case) 10 times as many points. Since the space of all $L \times W$ 8-bit RGB images is very large, we get the idea that even "large" image data sets are effectively small.

In the case of the GAN algorithm, it is unlikely that the generator will generate points

that are from the same data manifold in which the true data lie. This means p_ϕ and p^* are likely to be supported by disjoint lower dimensional manifolds and $\mathcal{D}_{\text{KL}}(p_\phi || p^*) = 0$, when the distributions are not equal, or $\mathcal{D}_{\text{KL}}(p_\phi || p^*) = \infty$.

Perfect Discriminator

The goal of GAN training is to optimize D_θ until it converges to D^*_θ, forcing (**??**) into a function related to the Jensen-Shannon divergence. G_ϕ is then optimized to minimize this divergence but in practice D_θ very quickly converges to be what [**?**] call a perfect discriminator.

Definition 7.10. *A* perfect discriminator *is a function $D : \mathcal{X} \mapsto [0, 1]$ that has accuracy 1 for all x in the supports of p^* and p_ϕ. In other words*

$$\mathbb{P}_{\mathcal{X}}(\{\{x \in \mathcal{X} : p^* > 0\} : D(x) = 1\}) = 1, \tag{7.6}$$

$$\mathbb{P}_{\tilde{\mathcal{X}}}(\{\{\tilde{x} \in \tilde{\mathcal{X}} : p_\phi > 0\} : D(\tilde{x}) = 0\}) = 1. \tag{7.7}$$

Theorem 7.11. *If D_θ is a perfect discriminator, then D_θ is constant on both supports of p_ϕ and p^* and $\nabla V_\phi = 0$, which means the gradient updates provide G_ϕ no amount of movement in the ϕ-parameter space.*

Proof. Let $\alpha > 0$ denote the learning-rate parameter and when we write $G_\phi^{(t+1)} \leftarrow G_\phi^{(t)} \cdots$ we mean the parameters ϕ are replaced with the output of the calculation on the right-hand side of the assignment operator. Then

$$G_\phi^{(t+1)} \leftarrow G_\phi^{(t)} - \alpha \nabla_\phi \frac{1}{m} \sum_{i=1}^{m} \log \left(1 - D_\theta^{(t+1)}(G_\phi^{(t)}(z_i))\right) \tag{7.8}$$

$$\implies G_\phi^{(t+1)} \leftarrow G_\phi^{(t)} - \alpha \nabla_\phi \frac{1}{m} \sum_{i=1}^{m} \log \left(1 - D_\theta^{(t+1)}(\tilde{x}_i)\right) \tag{7.9}$$

$$\implies G_\phi^{(t+1)} \leftarrow G_\phi^{(t)} - \alpha \nabla_\phi \frac{1}{m} \sum_{i=1}^{m} \log 1 \tag{7.10}$$

$$\implies G_\phi^{(t+1)} \leftarrow G_\phi^{(t)} \tag{7.11}$$

which is to say, G_ϕ does not get updated. □

Theorem 7.12. *If D_θ converges to a perfect discriminator too early in training and G_ϕ has stopped learning, then D_θ will not learn anything from the gradient updates.*

Proof.

$$D_\theta^{(t+1)} \leftarrow D_\theta^{(t)} - \alpha \nabla_\theta \frac{1}{m} \sum_{i=1}^{n} \Big(\log D_\theta^{(t)}(x_i) + \log \left(1 - D_\theta^{(t)}(G_\phi^{(t)}(z_i))\right) \Big) \tag{7.12}$$

$$\implies D_\theta^{(t+1)} \leftarrow D_\theta^{(t)} - \alpha \nabla_\theta \frac{1}{m} \sum_{i=1}^{n} \left(\log 1 + \log 1\right) \tag{7.13}$$

$$\implies D_\theta^{(t+1)} \leftarrow D_\theta^{(t)} \tag{7.14}$$

which is to say, D_θ does not get updated. □

Even though D_θ is a perfect discriminator, it is only good at telling apart obviously different distributions. This begets the need for a "gentler" discriminator. The WGAN paper tackles the issues mentioned above by using a different objective function and training routine. Looking into the history of the Wasserstein distance we will see that the WGAN is a modern implementation to an old transportation problem.

7.2 The Monge-Kantorovich Transportation Problem

Convergence issues that arise from the original GAN formulation have inspired novel takes on GAN training and implementation. One influential variation, inspired by *optimal transport theory*, is the Wasserstein GAN [**?**]. Before we introduce the Wasserstein GAN, it will be informative to introduce optimal transport theory.

The Wasserstein GAN is named after the Wasserstein-1 distance, otherwise known as the earth mover distance. The distance in question however, was not discovered by Leonid Wasserstein, rather it was discovered by the work of Gaspard Monge and Leonid Kantorovich. Wasserstein did publish a paper with a definition of the distance in 1969, but he did not discover it.

Gaspard Monge (1746 – 1818) was a mathematician, physicist, and founder and head of the École Polytechnique, located just outside of Paris. In 1781 he formulated the transportation problem *Excavation and Embankments*, which was about how to transport soil during the construction of forts and roads with minimal transportation cost.

Leonid Kantorovich (1912 – 1986) is regarded as one of the founders of mathematical economics and received a Nobel prize in 1975 for his contributions. He also established the theory of linear programming in 1938. In 1939 Kantorovich published a booklet *Mathematical Methods of Organizing and Planning of Production* and later he wrote a

brief paper called *On Translocation of Masses* in 1942.

In 1947 Kantorovich read the proceedings to a public session dedicated to Monge. The proceedings contained the transcript of a talk about Monge's transportation problem. When Kantorovich read the problem, he saw how it related to his own work and it was then that the transportation problem became known as the Monge-Kantorovich transportation problem.

The following is the definition of a transport plan as defined by Kantorovich in 1942 [**?**] and can be found in [**?**].

Definition 7.13. *Let $\mathcal{X}$ be any space. A* transport plan *is a probability measure γ on $\mathcal{X} \times \mathcal{X}$, whose projection $\gamma \circ \pi_x = p$ and whose projection $\gamma \circ \pi_y = q$ (i.e. the marginal distributions of γ with respect to x and y) are the measures p and q.*

Remark. *Each joint distribution γ represents the amount of mass needed to be move from each x to each corresponding y in order to transform p into q.*

Definition 7.14. *The* transport cost *for a given transport plan is given by*

$$\left(\int_{\mathcal{X}}\int_{\mathcal{Y}} ||x-y||^p \gamma(x,y)dxdy\right)^{\frac{1}{p}} \tag{7.15}$$

The integral of the distance the mass needs to travel, $||x-y||^p$, multiplied by the amount of mass γ gives the expected amount of work required to transform p into q.

The goal of optimal transport is to find the transport plan with minimal cost, thus we wish to find

$$\mathcal{W}_p(p,q) = \inf_{\gamma\in\Gamma}\left(\int_{\mathcal{X}}\int_{\mathcal{Y}} ||x-y||^p \gamma(x,y)dxdy\right)^{\frac{1}{p}} \tag{7.16}$$

where $p \geq 1$. For $p = 1$ we call $\mathcal{W}_1(p,q)$ the Kantorovich distance. This is what is used in the Wasserstein GAN. The algorithm in (**??**) can be stated more succinctly as

$$\mathcal{W}_1(p,q) = \inf_{\gamma\in\Gamma} \mathbb{E}_{(x,y)\sim\gamma}\left[d(x,y)\right] \tag{7.17}$$

Remark. *The definition given above in (**??**) involves an intractable infimum (it is not computationally feasible to find the infimum over all possible γ).*

7.3 The Wasserstein GAN

The distance given in (**??**) does not yet improve anything given the intractability of the infimum. However, the following theorem, due to Kantorovich and his student Rubinstein, saves the day. See [**?**].

Definition 7.15. *Let $(\mathcal{X}, \mathcal{D}_\mathcal{X})$ and $(\mathcal{Y}, \mathcal{D}_\mathcal{Y})$ be two metric spaces and let $f : \mathcal{X} \mapsto \mathcal{Y}$ be a function mapping elements from $\mathcal{X}$ to $\mathcal{Y}$. A* K-Lipshitz *function is defined as*

$$\mathcal{D}_\mathcal{Y}(f(x_1), f(x_2)) \leq K \cdot \mathcal{D}_\mathcal{X}(x_1, x_2) \tag{7.18}$$

*for all x_1 and $x_2 \in \mathcal{X}$. (**??**) can be stated as*

$$\frac{\mathcal{D}_\mathcal{Y}(f(x_1), f(x_2))}{\mathcal{D}_\mathcal{X}(x_1, x_2)} \leq K, \tag{7.19}$$

which is to say the rate of change of f is bounded above by the Lipshitz constant K.

Theorem 7.16. *The* Kantorovich-Rubinstein duality *says*

$$\inf_{\gamma \in \Gamma} \mathbb{E}_{(x,y)\sim\gamma} \left[d(x, y) \right] \tag{7.20}$$

is equivalent to

$$\sup_{f \in \mathcal{F}} \mathbb{E}_{x\sim p} \left[f(x) \right] - \mathbb{E}_{x\sim q} \left[f(x) \right] \tag{7.21}$$

where $\mathcal{F}$ is the set of all 1-Lipshitz functions.

The topology induced by the Kullback-Leibler divergence is much coarser than the topology induced by the Kantorovich-Rubinstein distance, which is a weak enough notion of distance to relax convergence requirements. See [**?**] for more information on the Kantorovich-Rubinstein distance.

Example (Learning Parallel Lines (Revisited))**.** *When we use the Kantorovich-Rubinstein*

distance in the parallel lines problem.

$$\mathcal{W}_1(G_\phi(z), p_0(z)) = \inf_{\gamma \in \Gamma(G_\phi(z), p_0(z))} \mathbb{E}_{(x,y)\sim\gamma} \left[||x - y|| \right] \tag{7.22}$$

$$= \inf_{\gamma \in \Gamma(G_\phi(z), p_0(z))} \mathbb{E}_{(x,y)\sim\gamma} \left[\sqrt{(\phi - 0)^2 + (z - z)^2} \right] \tag{7.23}$$

$$= |\phi| \tag{7.24}$$

The Kantorovich-Rubinstein distance provides a continuous measure of distance even for probability distributions with disjoint supports.

The Wasserstein GAN Algorithm

The following is the WGAN algorithm as presented in [**?**].

Algorithm 2: WGAN

Let $\eta > 0$ *be the learning rate.*
Let $c > 0$ *be the clipping parameter.*
Let $T > 0$ *be the number of training iterations.*
Let $K > 0$ *be the number of training iterations for the critic.*
Let $f_\theta : \Theta \times \mathcal{X} \mapsto [0, 1]$
Let $G_\phi : \Phi \times \mathcal{Z} \mapsto \mathcal{X}$
for $t \in \{0, \ldots, T\}$ **do**
    for $k \in \{0, \ldots, K\}$ **do**
        Let $z = \{z_1, \ldots, z_m\}$, *where each* z_i *is sampled from* $(\mathcal{Z}, p_\mathcal{Z})$*, This batch will be used in the optimization of* D_θ*.*
        Let $x = \{x_1, \ldots, x_m\}$, *where each* x_i *is from the training data.*
        Let $\frac{\partial V}{\partial \theta} = \nabla_\theta \left[\frac{1}{m} \sum_{i=1}^{m} f_\theta(x_i) - \frac{1}{m} \sum_{i=1}^{m} f_\theta(G_\phi(z_i)) \right]$.
        Let $\theta \leftarrow \theta + \eta \cdot \text{RMSProp}(\theta, \frac{\partial V}{\partial \theta})$.
        Let $\theta \leftarrow \text{clip}(\theta, -c, c)$.
    end
    Let $z = \{z_1, \ldots, z_m\}$, *we sample a new batch from* $(\mathcal{Z}, p_\mathcal{Z})$*. This batch will be used in the optimization of* G_ϕ*.*
    Let $\frac{\partial V}{\partial \phi} = -\nabla_\phi \frac{1}{m} \sum_{i=1}^{m} f_\theta(G_\phi(z_i))$.
    Let $\phi \leftarrow \phi + \eta \cdot \text{RMSProp}(\phi, \frac{\partial V}{\partial \phi})$.
end

7.4 Discussion

The original GAN objective function inherits the strong topology from the Jensen-Shannon and Kullback-Leibler divergences, which means the original GANs formulation was not as well suited for working with real data as it could have been. The WGAN is better suited than the original GAN since the earth mover distance is always defined, unlike the Jensen-Shannon and Kullback-Leibler divergences.

One issue with the WGAN is found in the way **[?]** achieve the 1-Lipshitz constraint. They clip the parameters after each update (see Algorithm **??** above), which effectively reduces the amount the parameters can change with each update. They acknowledge this is not a very good way to maintain 1-Lipshitz continuity.

Section 8

Conclusion

In Section **??** we looked at game theory and how the GAN algorithm can be cast as a game between two competing neural networks. The minimax strategy for D was $D(x) = \frac{1}{2}$ for all x, which, if attained, would make it impossible for the generator to minimize the value function, since G's actions would no longer affect the actions of D. This strategy makes sense for D, since it is the best anyone can do when confronted with maximum uncertainty.

Section **??** included two information theoretic perspectives on GAN training. Theorem **??** considered the theoretical, limiting behaviour of GANs and **??** considered what happens at each step of the optimization. This is similar to a macroscopic and microscopic view of GAN training.

Section **??** introduced optimal transport and showed how the GAN algorithm has benefited greatly from the application of the Kantorovich-Rubinstein distance from optimal transport. It takes a great deal of theoretical understanding to build a well-functioning learning system. By proposing a variant of the GAN algorithm based on optimal transport theory, [**?**] have opened up an additional theoretical avenue of research. Future research for GANs can come from game theory, information theory and optimal transport.

This thesis provided much of the necessary background material for anyone wanting to get up to speed on the theory of generative adversarial networks (GANs). The most important thing going forward with GAN research may be the detection of fake news related media. GANs have made it incredibly easy to produce fake images and videos. There are many apps to produce deep fakes, just use your favorite search engine to find them.

The following quote from [**?**] hints at the social impact the GAN algorithm may have in the near future.

> *The generative model can be thought of as analogous to a team of counterfeiters, trying to produce fake currency and use it without detection, while the discriminative model is analogous to the police, trying to detect the counterfeit currency. Competition in this game drives both teams to improve their*

methods until the counterfeits are indistinguishable from the genuine articles.

The competition between counterfeiters and police may be soon played out by the producers of fake images and videos, to be used in fake news, and concerned researchers. See [?], [?], [?], [?], [?], [?], [?], [?], [?], [?], [?] and [?] for more information on fake image and video detection.

Section 9

Appendix

9.1 Optimization Dynamics

$$\mathbb{E}\left[\log \frac{p^*(x)}{p_\phi(\tilde{x}) + p^*(x)}\right] + \mathbb{E}\left[\log \frac{p_\phi(\tilde{x})}{p_\phi(\tilde{x}) + p^*(x)}\right] + \log 4 - \log 4 \tag{9.1}$$

$$= \sum_x p^*(x) \log \frac{p^*(x)}{p_\phi(\tilde{x}) + p^*(x)} + \sum_x p_\phi(\tilde{x}) \log \frac{p_\phi(\tilde{x})}{p_\phi(\tilde{x}) + p^*(x)} + \log 4 - \log 4 \tag{9.2}$$

$$= \sum_x p^*(x) \log \frac{p^*(x)}{p_\phi(\tilde{x}) + p^*(x)} + \sum_x p_\phi(\tilde{x}) \log \frac{p_\phi(\tilde{x})}{p_\phi(\tilde{x}) + p^*(x)} + \log 2 + \log 2 - \log 4 \tag{9.3}$$

$$= \sum_x p^*(x) \log \frac{p^*(x)}{p_\phi(\tilde{x}) + p^*(x)} + \sum_x p_\phi(\tilde{x}) \log \frac{p_\phi(\tilde{x})}{p_\phi(\tilde{x}) + p^*(x)} \tag{9.4}$$

$$+ \sum_x p^*(x) \log 2 + \sum_x p_\phi(\tilde{x}) \log 2 - \log 4 \tag{9.5}$$

$$= \sum_x p^*(x) \log \frac{2p^*(x)}{p_\phi(\tilde{x}) + p^*(x)} + \sum_x p_\phi(\tilde{x}) \log \frac{2p_\phi(\tilde{x})}{p_\phi(\tilde{x}) + p^*(x)} - \log 4 \tag{9.6}$$

$$= \sum_x p^*(x) \log \frac{p^*(x)}{\frac{p_\phi(\tilde{x}) + p^*(x)}{2}} + \sum_x p_\phi(\tilde{x}) \log \frac{p_\phi(\tilde{x})}{\frac{p_\phi(\tilde{x}) + p^*(x)}{2}} - \log 4 \tag{9.7}$$

$$= \mathcal{D}_{\mathrm{KL}}\left(p^*(x) \middle|\middle| \frac{p_\phi(\tilde{x}) + p^*(x)}{2}\right) + \mathcal{D}_{\mathrm{KL}}\left(p_\phi(\tilde{x}) \middle|\middle| \frac{p_\phi(\tilde{x}) + p^*(x)}{2}\right) - \log 4 \tag{9.8}$$

$$= 2 \cdot \mathcal{D}_{\mathrm{JS}}(p_\phi(\tilde{x}) \,||\, p^*(x)) - \log 4 \tag{9.9}$$

References

[Afchar et al., 2018a] Afchar, D., Nozick, V., Yamagishi, J., and Echizen, I. (2018a). Mesonet: a compact facial video forgery detection network. *2018 IEEE International Workshop on Information Forensics and Security (WIFS)*.

[Afchar et al., 2018b] Afchar, D., Nozick, V., Yamagishi, J., and Echizen, I. (2018b). Mesonet: a compact facial video forgery detection network. *CoRR*, abs/1809.00888.

[Agarwal and Varshney, 2019] Agarwal, S. and Varshney, L. R. (2019). Limits of deepfake detection: A robust estimation viewpoint.

[Arjovsky and Bottou, 2017] Arjovsky, M. and Bottou, L. (2017). Towards principled methods for training generative adversarial networks. *arXiv preprint arXiv:1701.04862*.

[Arjovsky et al., 2017] Arjovsky, M., Chintala, S., and Bottou, L. (2017). Wasserstein GAN. *arXiv:1701.07875 [cs, stat]*.

[Bellman, 1957] Bellman, R. (1957). E. 1957. dynamic programming. *Princeton UniversityPress. BellmanDynamic programming1957*, page 151.

[Bishop, 2006] Bishop, C. M. (2006). *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg.

[Cheng et al., 2018] Cheng, X., Khomtchouk, B., Matloff, N., and Mohanty, P. (2018). Polynomial regression as an alternative to neural nets. *arXiv preprint arXiv:1806.06850*.

[Cortes, 2017] Cortes, E. (2017). Plant disease classification using convolutional networks and generative adverserial networks.

[Cover and Thomas, 2012] Cover, T. M. and Thomas, J. A. (2012). *Elements of information theory*. John Wiley and Sons.

[Doyle, 1982] Doyle, P. (1982). Why maximize entropy?

[Endres and Schindelin, 2003] Endres, D. M. and Schindelin, J. E. (2003). A new metric for probability distributions. *IEEE Transactions on Information Theory*, 49(7):1858–1860.

[Frid-Adar et al., 2018] Frid-Adar, M., Klang, E., Amitai, M., Goldberger, J., and Greenspan, H. (2018). Synthetic data augmentation using gan for improved liver lesion classification. In *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)*, pages 289–293. IEEE.

[Gidel et al., 2018] Gidel, G., Berard, H., Vignoud, G., Vincent, P., and Lacoste-Julien, S. (2018). A Variational Inequality Perspective on Generative Adversarial Networks. *arXiv:1802.10551 [cs, math, stat]*. arXiv: 1802.10551.

[Goodfellow, 2017] Goodfellow, I. J. (2017). NIPS 2016 tutorial: Generative adversarial networks. *CoRR*, abs/1701.00160.

[Goodfellow et al., 2014] Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative Adversarial Networks. *arXiv:1406.2661 [cs, stat]*. arXiv: 1406.2661.

[Gutmann and Hyvärinen, 2010] Gutmann, M. and Hyvärinen, A. (2010). Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 297–304.

[Kantorovich, 1942] Kantorovich, L. V. (1942). On the translocation of masses. *Proceedings of the USSR Academy of Sciences*, page 2.

[Kantorovich and Rubinstein, 1958] Kantorovich, L. V. and Rubinstein, G. S. (1958). On a space of completely additive functions. *Vestnik Leningrad. Univ*, 13(7):52–59.

[Korshunov and Marcel, 2018] Korshunov, P. and Marcel, S. (2018). Deepfakes: a new threat to face recognition? assessment and detection.

[Li et al., 2014] Li, W., Gauci, M., and Groß, R. (2014). Coevolutionary learning of swarm behaviors without metrics. In *Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation*, pages 201–208. ACM.

[Li et al., 2018] Li, Y., Chang, M.-C., and Lyu, S. (2018). In ictu oculi: Exposing ai generated fake face videos by detecting eye blinking.

[Li and Lyu, 2018] Li, Y. and Lyu, S. (2018). Exposing deepfake videos by detecting face warping artifacts.

[Losee, 1997] Losee, R. M. (1997). A discipline independent definition of information. *Journal of the American Society for information Science*, 48(3):254–269.

[MacKay et al., 2003] MacKay, D., Kay, D., and Press, C. U. (2003). *Information Theory, Inference and Learning Algorithms*. Cambridge University Press.

[Martin and England, 2011] Martin, N. F. and England, J. W. (2011). *Mathematical theory of entropy*, volume 12. Cambridge university press.

[Matern et al., 2019] Matern, F., Riess, C., and Stamminger, M. (2019). Exploiting visual artifacts to expose deepfakes and face manipulations. pages 83–92.

[McCloskey and Albright, 2018] McCloskey, S. and Albright, M. (2018). Detecting gan-generated imagery using color cues.

[Myerson, 1997] Myerson, R. B. (1997). *Game Theory: Analysis of Conflict*. Harvard University Press, 1st paperback edition edition.

[Nataraj et al., 2019] Nataraj, L., Mohammed, T. M., Manjunath, B. S., Chandrasekaran, S., Flenner, A., Bappy, J. H., and Roy-Chowdhury, A. K. (2019). Detecting gan generated fake images using co-occurrence matrices.

[Nazki et al., 2018] Nazki, H., Lee, J., Yoon, S., and Park, D. S. (2018). Synthetic data augmentation for plant disease image generation using gan. *Future Convergence Contents Realization the 4th Industrial Revolution, At Mokpo National University, Mokpo, South Korea*, pages 459–460.

[Niemitalo, 2010] Niemitalo, O. (2010). A method for training artificial neural networks.

[Odena, 2016] Odena, A. (2016). Semi-supervised learning with generative adversarial networks. *arXiv preprint arXiv:1606.01583*.

[Poundstone, 1993] Poundstone, W. (1993). *Prisoner's Dilemma/John von Neumann, Game Theory and the Puzzle of the Bomb*. Anchor.

[Rumelhart et al., 1986] Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323:533–536.

[Rössler et al., 2019] Rössler, A., Cozzolino, D., Verdoliva, L., Riess, C., Thies, J., and Nießner, M. (2019). Faceforensics++: Learning to detect manipulated facial images.

[Sabir et al., 2019] Sabir, E., Cheng, J., Jaiswal, A., AbdAlmageed, W., Masi, I., and Natarajan, P. (2019). Recurrent convolutional strategies for face manipulation detection in videos.

[Salimans et al., 2016] Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., Chen, X., and Chen, X. (2016). Improved techniques for training gans. In Lee, D. D., Sugiyama, M., Luxburg, U. V., Guyon, I., and Garnett, R., editors, *Advances in Neural Information Processing Systems 29*, pages 2234–2242. Curran Associates, Inc.

[Schmidhuber, 1992] Schmidhuber, J. (1992). Learning factorial codes by predictability minimization. *Neural Computation*, 4(6):863–879.

[Schmidhuber, 2018] Schmidhuber, J. (2018). Unsupervised neural networks fight in a minimax game.

[Shin et al., 2018] Shin, H.-C., Tenenholtz, N. A., Rogers, J. K., Schwarz, C. G., Senjem, M. L., Gunter, J. L., Andriole, K. P., and Michalski, M. (2018). Medical image synthesis for data augmentation and anonymization using generative adversarial networks. In *International Workshop on Simulation and Synthesis in Medical Imaging*, pages 1–11. Springer.

[Shwartz-Ziv and Tishby, 2017] Shwartz-Ziv, R. and Tishby, N. (2017). Opening the Black Box of Deep Neural Networks via Information. *arXiv:1703.00810 [cs]*. arXiv: 1703.00810.

[Tenenbaum et al., 2000] Tenenbaum, J. B., De Silva, V., and Langford, J. C. (2000). A global geometric framework for nonlinear dimensionality reduction. *science*, 290(5500):2319–2323.

[Tishby and Zaslavsky, 2015] Tishby, N. and Zaslavsky, N. (2015). Deep Learning and the Information Bottleneck Principle. *arXiv:1503.02406 [cs]*. arXiv: 1503.02406.

[Valerio Giuffrida et al., 2017] Valerio Giuffrida, M., Scharr, H., and Tsaftaris, S. A. (2017). Arigan: synthetic arabidopsis plants using generative adversarial network. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2064–2071.

[Vershik, 2013] Vershik, A. M. (2013). Long history of the monge-kantorovich transportation problem: (marking the centennial of l.v. kantorovich's birth!). *The Mathematical Intelligencer*, 35(4):1–9.

[Villani, 2008] Villani, C. (2008). *Optimal transport: old and new*, volume 338. Springer Science and Business Media.

[Weng, 2017] Weng, L. (2017). From gan to wgan.

[Xuan et al., 2019] Xuan, X., Peng, B., Dong, J., and Wang, W. (2019). On the generalization of GAN image forensics. *CoRR*, abs/1902.11153.

[Zhou et al., 2012] Zhou, Y., Kantarcioglu, M., Thuraisingham, B., and Xi, B. (2012). Adversarial support vector machine learning. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1059–1067. ACM.