Deep Learning with generative adversarial networks

GANs IN ACTION

Jakub Langr
and Vladimir Bok

MEAP

MANNING

MEAP Edition
Manning Early Access Program
GANs in Action
Deep learning with generative adversarial networks
Version 4

For more information on this and other Manning titles go to
www.manning.com

welcome

Welcome to MEAP for *GANs in Action: Deep Learning with Generative Adversarial Networks.* GANs are an exciting new class of machine learning models whose ability to generate synthetic yet realistic-looking data has led to mind-bending applications. As perhaps the first successful general-purpose way of generating new data, GANs have shown a great potential for a wide range of practical applications (including ones in art, fashion, medicine, and finance) and the list of GAN-related academic papers keeps growing exponentially.

This book is intended for readers who already have some experience with machine learning and neural networks. You will need intermediate Python knowledge as well as basic knowledge of probability, statistics, and calculus. You will also need some elementary familiarity with or willingness to independently learn the Python-based machine learning libraries Keras and TensorFlow.

As one of the very first publications dedicated to this topic, *GANs in Action* will guide you through generative adversarial learning from the ground up. Starting from the simplest examples, we will advance to some of the most innovative GAN implementations and techniques, including those that have only been presented this year, such as Progressive Growing of GANs.

We will make cutting-edge research accessible by providing the intuition behind these advances while sparing you all but the most essential math and theory. Ultimately, our goal is to give you the knowledge and tools necessary to not only understand what has been accomplished in GANs to-date but also to empower you to find new applications of your own choosing. The generative adversarial paradigm is full of potential to be unravelled by enterprising individuals like you who can make an impact through academic and real-world applications alike.

Your feedback is important to us; we hope you will leave us comments in the Author Online forum so we can make this book the best it can be—much like GAN's Generator continues to improve through the feedback it receives. (After reading Chapter 1, you will understand this pun. We promise!)

We—as many other machine learning professionals—see GANs as a revolutionary new paradigm and we are glad to have you join us on this exciting journey.

—Jakub Langr & Vladimir Bok

brief contents

1 *Introduction to GANs*

2 *Autoencoders as a Path to GANs*

3 *Your First GAN: Generating Handwritten Digits*

4 *Deep Convolutional GAN (DCGAN)*

5 *Training & Common Challenges: GANing for Success*

6 *Progressing with GANs*

7 *Semi-Supervised GAN*

8 *Conditional GAN*

9 *Cycle-Consistent Adversarial Networks*

10 *Adversarial Examples*

11 *Practical Applications of GANs*

12 *Conclusion*

Appendix A: Technical/ deployments

1 Introduction to GANs

In this chapter, we will:

- Introduce Generative Adversarial Networks (GANs) and how they work.
- Preview some of the exciting GAN applications that will be covered later in this book.

1.1 Introduction

The notion whether machines can think is older than the computer itself. In 1950, the famed mathematician, logician, and computer scientist Alan Turing—perhaps best known for the role he played in decoding the Nazi wartime enciphering machine, Enigma—penned a paper that would immortalize his name for generations to come, “Computing Machinery and Intelligence.”

In the paper, Turing proposed a test he called the "imitation game", better known today as the “Turing Test”—a hypothetical scenario in which an unknowing observer talks with two counterparts behind closed door: one, a fellow human; the other, a computer. If, Turing reasons, the observer is unable to tell which is the person and which is the machine, the computer passed the test must be deemed intelligent.

Anyone who engaged in a dialogue with an automated chatbot or a voice-powered intelligent assistant, such as Amazon's Alexa or Google Home, knows that computers have a long way to go to pass this deceptively simple test. However, in other tasks, computers have not only matched human performance but also surpassed it—even in areas that were, until

recently, thought to be out of reach of even the smartest algorithms, such as face recognition[1] or mastering the game of Go[2].

One of the key reasons for this gap is that while artificial intelligence and machine learning have historically been excellent at teaching computers to discern ever more intricate patterns in data and master ever more complex gameplays, they have been poor at teaching them to generate new data—something that we, humans, do every day as we converse with one another, come up with innovative solutions to problems, and express our creativity through art.

This all changed in 2014 when Ian Goodfellow invented Generative Adversarial Networks (GANs). This technique has enabled computers to generate realistic data by using not one but two separate neural networks. GANs were not the first computer program used to generate data, but their results and versatility set them apart from all the rest.

GANs have achieved remarkable results that have long been considered virtually impossible for artificial systems, such as the ability to generate fake images in real-world-like quality, turn a scribble into a photograph-like image, or turn a video footage of a horse into a running zebra—all without the need for vast troves of painstakingly-labeled training data. As such, they have have been hailed by industry experts as one of the most important innovations in deep learning. Yann LeCun, the Director of AI Research at Facebook, went as far as to say that GANs and their variations are "the coolest idea in deep learning in the last 20 years."[3]

The goal of this book is to provide the definitive guide for anyone interested in learning about this emerging deep learning technique from the ground up. Starting from the simplest examples, we will advance to some of the most innovative GAN implementations and techniques, including those that have only been presented this year, such as Progressive Growing of GANs. We will make cutting-edge research accessible by providing the intuition behind these advances while sparing you all but the most essential math and theory. Ultimately, our goal is to give you the knowledge and tools necessary to not only understand what has been accomplished in GANs to-date but also to empower you to find new applications of your own choosing. The generative adversarial paradigm is full of potential to be unravelled by enterprising individuals like you who can make an impact through academic and real-world applications alike.

1.2 Prerequisites

Before we begin, here is what you should ideally know. While we try to do our best to explain most things as we go, you should be confident about at least 70% of this list:

[1] Chaochao Lu: "Surpassing Human-Level Face Verification Performance on LFW with GaussianFace", 2014; arXiv:1404.3840.
[2] https://www.nytimes.com/2017/05/23/business/google-deepmind-alphago-go-champion-defeat.html
[3] https://www.wired.com/2017/04/googles-dueling-neural-networks-spar-get-smarter-no-humans-required/

- We expect you to be able to run intermediate Python programs. You do not need to be a Python master, but you should have at least two years of Python experience (ideally as a full-time data scientist or software engineer). Example tasks include:
    - Being able to start your own Conda environment
    - Installing and using packages
    - Using context and decorators
- You should understand Object Oriented Programming, how to work with objects and now to figure out their attributes and methods. You need to be able to understand fairly typical (e.g., Pandas DataFrames) as well as atypical Python objects (e.g., Tensors, Keras Layer object).
- You should understand basics of delayed computation in TensorFlow / Python settings and the basics of distributed computation.
- You should understand basics of Machine Learning theory such as train / test split, overfitting, weights, hyper-parameters, as well as the basics of supervised, unsupervised and reinforcement learning. Also metrics such as accuracy and Mean Squared Error.
- You should understand basic statistics and calculus, such as probability, density functions, probability distributions, differentiation, and convex optimization.
- You should understand elementary linear algebra, such as matrices, high dimensional spaces and ideally also Principal Component Analysis.
- You should understand basics of Deep Learning—things such as feedforward networks, weights & biases, activation functions, regularization, stochastic gradient descent. and backpropagation.
- You should also have some elementary familiarity with or willingness to independently learn the Python-based machine learning libraries, Keras and TensorFlow.

We are not trying to scare you, but rather ensure that you will be getting the most out of this book. You may try to take a stab at it anyway, but the less you know, the more you should expect to search online on your own. But if this list does not seem scary to you, we should be good to go.

1.3 What Are Generative Adversarial Networks?

A Generative Adversarial Network (GAN) is a machine learning technique which uses a game-like competitive dynamic between two neural networks to learn to generate fake examples indistinguishable from real data from a given training dataset (e.g., images of handwritten digits).

The two networks are called "Generator" and "Discriminator". The Generator's goal is to produce data that is indistinguishable from the training dataset. The Discriminator's goal is to correctly determine whether a particular example is real (i.e., coming from the training dataset) or fake (i.e., created by the Generator).

Starting from what is usually no more than a vector of random numbers, the Generator learns to produce realistic-looking examples. It does so indirectly, through the feedback it receives from the Discriminator's decisions. Each time the Discriminator is fooled into classifying a fake image as real, the Generator knows it did something well. And each time the Discriminator correctly rejects a Generator-produced image as fake, the Generator knows it needs to improve.

The Discriminator continues to improve as well: For each classification it makes, it is given feedback whether its guess was correct or not. So, as the Generator gets better at producing realistic-looking data, the Discriminator gets better at telling fake data from the real. Both networks continue to simultaneously improve through this cat-and-mouse game.

Table 1.1 summarizes the key takeaways about the two GAN subnetworks.

Table 1.1: Generator and Discriminator Networks

	Generator	Discriminator
Input	A vector of random numbers	The Discriminator receives input from two sources: • Real examples coming from the training dataset • Fake examples coming from the Generator
Output	Fake examples that strive to be as convincing as possible	Likelihood that the input example is real
Goal	Generate fake data that are indistinguishable from members of the training dataset	Distinguish between fake examples coming from the Generator and real examples coming from the training dataset

GENERATOR AND DISCRIMINATOR ANALOGY: COUNTERFEITING MONEY

If all this sounds too abstract, consider the following analogy. Hopefully, it will make it easier to understand the two networks and how they interact. In a small town, the local mafia (the "Generator") sets out to counterfeit money. Each time they produce a new batch of bills, they send one of their associates to a local bank (the "Discriminator") and attempt to deposit the money as real.

If the associate gets arrested, the mafia knows they have to get better at counterfeiting. Just like a Generator whose example was rejected as fake, the mafia does not know where exactly they went wrong; all they know is they need to improve *something*.

Similarly, the bank needs to continue to improve. Accepting fake money could prove costly so if the bank uncovers that they have recently accepted fake bills as a real deposit, they may invest in better money scanning technology and employee training to help prevent such mishaps from happening again.

As a result, the bank may now be able to recognize high-quality counterfeit bills that previously slipped through the cracks. This, in turn, prompts the mafia to improve its counterfeiting techniques—and the feedback loop continues.

1.3.1 GAN in Action

Now that we have a high-level understanding of GAN and its constituent networks, it is time to take a closer look at the system in action. Imagine our goal is to teach a GAN to produce realistic-looking handwritten digits. (In fact, we will learn to implement such model in Chapter 3 and expand on it in Chapter 4.) The diagram in Figure 1.1 shows the core GAN architecture:

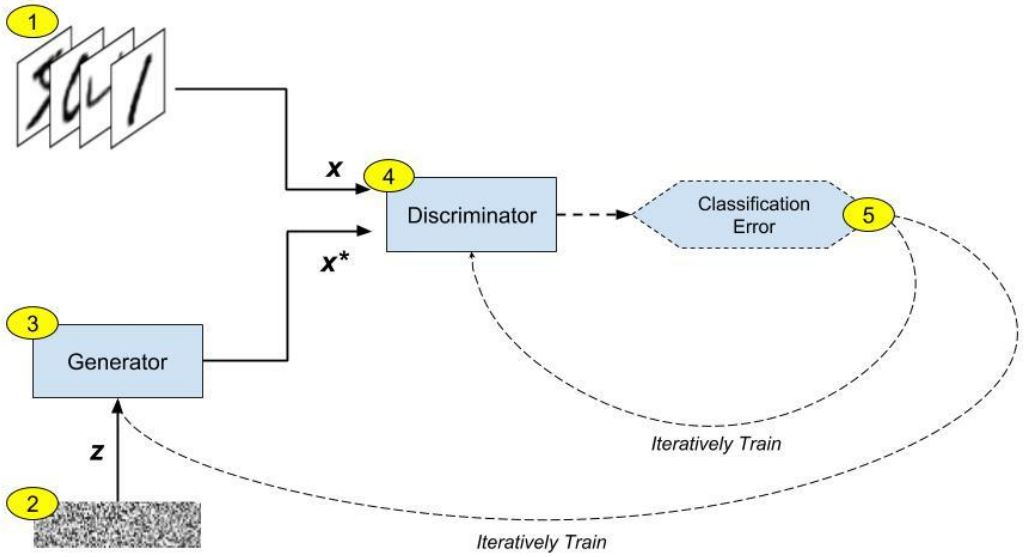


Figure 1.1: GAN Architecture Diagram. This diagram shows the two GAN subnetworks, their inputs and outputs, and how they interact.

① **Training Dataset:** The dataset of real examples which we want the Generator to learn to emulate with near-perfect quality. In this case, the dataset consists of images of handwritten digits. This dataset serves as input (***x***) to the Discriminator network.
② **Random Noise Vector (*z*):** The raw input to the Generator network. ***z*** is a vector of random numbers which the Generator uses as a starting point for synthesizing fake examples.
③ **The Generator Network:** The Generator takes in a vector of random numbers ***z*** as input and outputs fake example (***x****). Its goal is to make the fake examples it produces indistinguishable from the real examples in the training dataset.
④ **The Discriminator Network:** The Discriminator takes as input either a real example ***x*** coming from the training set or the fake example ***x**** produced by the Generator. For each example, it determines and outputs the likelihood (on a scale from 0 to 1) whether the example is real.

⑤ **Iterative Training/Tuning:** For each of the Discrimiantor's guesses, we determine how good it was—much like we would for a regular classifier—and use the results to iteratively tune the Discriminator and the Generator networks through backpropagation:

- The Discriminator's weights and biases get updated to maximize its classification accuracy; i.e., maximizing the likelihood of correct prediction: x as real and x^* as fake.
- The Generator's weights and biases get updated to maximize the likelihood that the Discriminator incorrectly classifies x^* as real.

1.3.2 GAN Training

Now that we covered the GAN architecture and explained its components and how they interact, let's see it all in action by examining the GAN training process:

GAN TRAINING ALGORITHM

For each training iteration ***do***:

1. Train the Discriminator:
    a) Take a random real example ***x*** from the training dataset
    b) Get a new random noise vector ***z*** and, using the Generator network, use it to synthesize a fake example ***x****
    c) Use the Discriminator network to classify the ***x*** and the ***x****.
    d) Compute the classification errors and backpropagate the total error to update the Discriminator weights and biases, so as to minimize the classification errors.
2. Train the Generator:
    a) Get a new random noise vector ***z*** and, using the Generator network, use it to synthesize a fake example ***x****
    b) Use the Discriminator network to classify the ***x****
    c) Compute the classification error and backpropagate the error to update the Generator weights and biases, so as to maximize the Discriminator's error.

end for

1.3.3 GAN Training Visualized

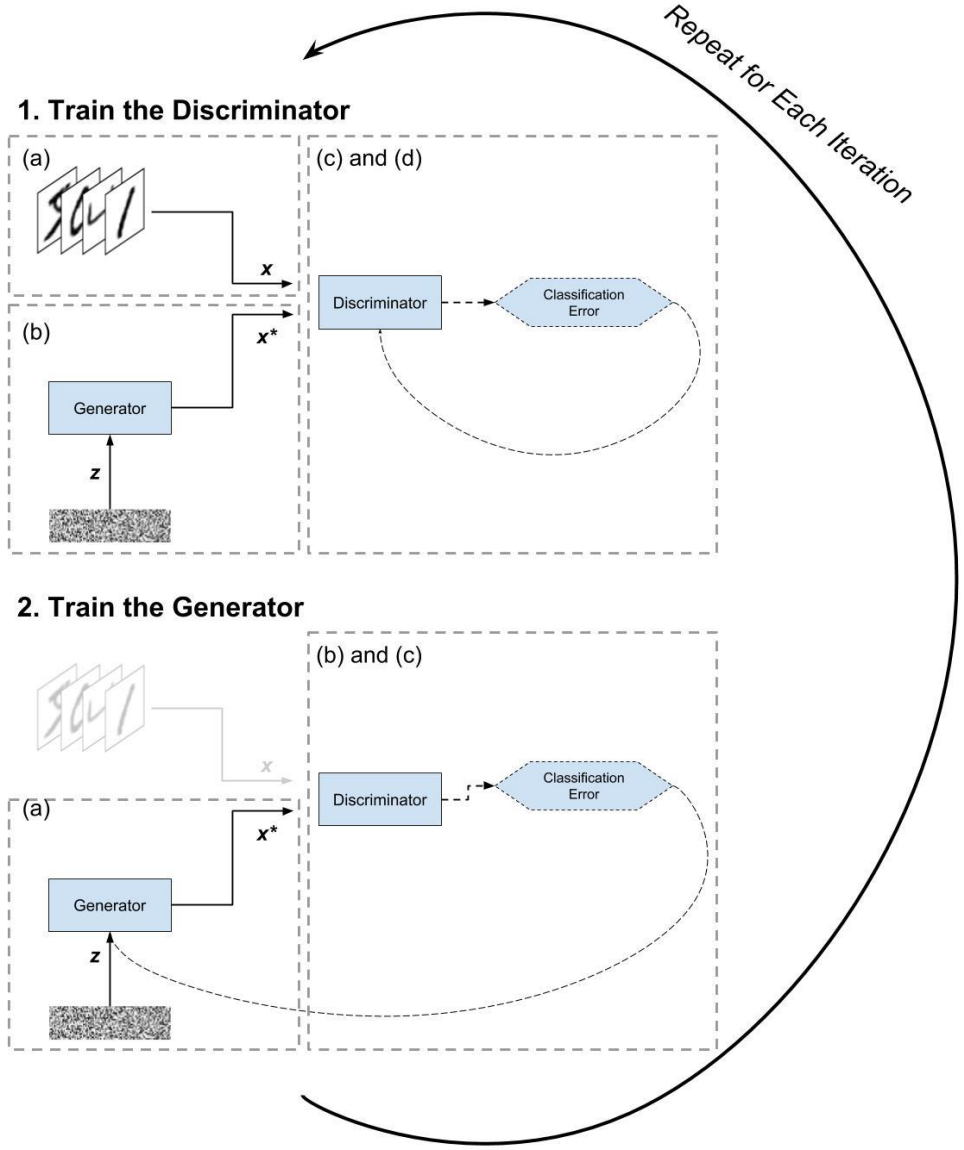
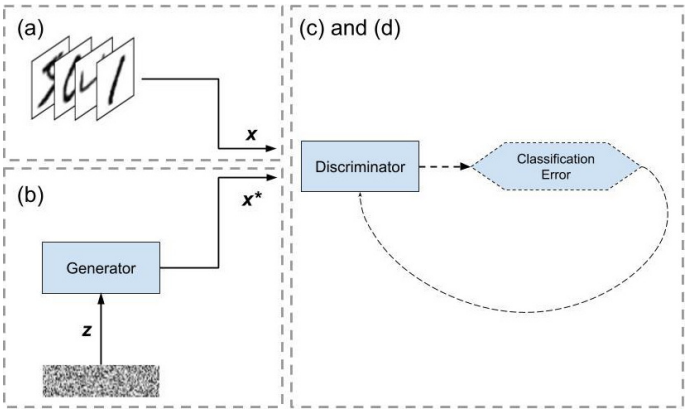
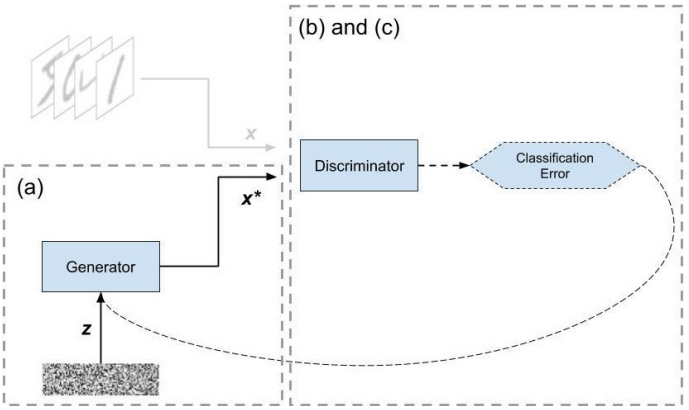


Figure 1.2: GAN Training Process. This diagram shows the GAN Training Algorithm described above. Please note that Step 1 (Discriminator Training) and Step 2 (Generator Training) depict the same GAN Network at different time snapshots in the corresponding stages in of the training process.

Subdiagram	Legend
1) Train the Discriminator 	(a) Take a random real example ***x*** from the training dataset (b) Get a new random noise vector ***z*** and, using the Generator network, use it to synthesize a fake example ***x**** (c) Use the Discriminator network to classify the ***x*** and the ***x**** . (d) Compute the classification errors and backpropagate the total error to update the Discriminator weights and biases, so as to minimize the classification errors.
2) Train the Generator 	(a) Get a new random noise vector ***z*** and, using the Generator network, use it to synthesize a fake example ***x**** (b) Use the Discriminator network to classify the ***x**** (c) Compute the classification error and backpropagate the error to update the Generator weights and biases, so as to maximize the Discriminator's error.

1.3.4 Reaching Equilibrium

You may wonder when this training loop ever ends. Or, to be precise, how do we know when a GAN is fully trained so that we can set the right number of training iterations. When training a regular neural network, we usually have a clear objective to achieve and measure. For example, if we have a classifier network seeking to distinguish pictures of dogs from pictures of cats, we measure the classification accuracy (proportion of correctly classified images) and stop the training when we achieve a satisfactory level (e.g., 90%). In a GAN, there are two networks with two competing objectives; when one network gets slightly better, the other gets a bit worse. How do we know when to stop?

Those familiar with Game Theory may recognize this setup as a zero-sum game—a situation in which one player's gains are equal to another's losses; that is, when one player gets better off by some amount, the other player gets worse off by the exact same amount. All zero-sum games have a so-called **Nash Equilibrium**[4], a point at which neither player can improve his or her situation or payoff by changing his or her actions.

The Generator and the Discriminator reach their Nash Equilibrium when the fake examples produced by the Generator are indistinguishable from the real data, and the Discriminator can at best randomly guess whether a particular example is real or fake (i.e., make a 50/50 guess that an example is real or fake).

Let's convince ourselves why this is the case. When each of the fake examples $\boldsymbol{x}^*$ is truly indistinguishable from the real examples $\boldsymbol{x}$ coming from the training dataset, there is nothing the Discriminator can use to tell them apart from one another. Since half of the examples it receives are real and half fake, the best it can do is to flip a coin and classify each example as real or fake with 50% probability.

The Generator is likewise at a point where it has nothing to gain from further tuning. Because the examples it produces are already perfectly indistinguishable from the real ones, even a tiny change to the process it uses to turn the random noise vector $\boldsymbol{z}$ into a fake example $\boldsymbol{x}^*$ may give the Discriminator a cue about how to tell apart the fake example from the real data, making the Generator worse off.

With equilibrium achieved, a GAN is fully trained. In practice, such equilibrium is almost impossible to find so—as is often the case in deep learning—we stop the training as soon as we achieve our desired objective; e.g, when the fake examples produced by the Generator are convincing enough for our purposes.

1.3.5 The Pros and Cons of Studying GANs

THE PROS

GANs have become very popular due to their very promising early results. These subsequently fueled a set of practical applications that then led to theoretical advancements which fed back to practical applications. In this book, we will cover some of the most notable practical applications and touch on some of the theoretical advancements; however this is not an academic book and therefore our focus will be on those exciting results that probably drove you to buy this book in the first place (Don't lie!).

But there are many more reasons to be excited about GANs. Let us just hint at some of the reasons why we are so excited about them beyond just their amazing results:

[4] Nash Equilibrium is named after the American economist and mathematician John Forbes Nash Jr., whose life and career were captured in the biography titled *A Beautiful Mind* and inspired the eponymous film.

- GANs are pretty flexible technique meaning that their applications, especially in conjunction with other machine learning techniques, are usually viable and therefore, we dare to hope that they will stand the test of time.
- A number of authorities have expressed their belief in generative adversarial networks as a technique fundamentally changing AI: including heads of industry such as Jen-Hsun Huang[5], NVIDIA CEO, and machine learning researchers from across different research groups such as Yann LeCun[6], the head of Facebook AI Research. You may choose not to believe them, but there's some value of genuine experts giving their opinions on the future of AI, especially the ones that choose not to promote themselves too much on social media.
- Generative Adversarial Networks are also a very intuitive idea so there is plenty of scope to build on and further and it is relatively easy for newcomers to join in and contribute. Indeed, we have seen contribution from many adjacent disciplines, such as computer graphics and design. There have been a number of recent applications by NVIDIA, numerous researchers in 3D image generation as well as, for example, the work done by Guerin et al. (2017) in 3D terrain creation using GANs.
- The point above also has an interesting implication in terms of future research: a lot of obvious research areas have not been explored or have barely been touched on at the time of writing. For example, sampling from the real data or the seeding factors is typically implemented in the most naive way possible. We can see how more research into this area could bring about further advancements in the efficiency of this technique.
- Generative adversarial networks can also be applied in the variety of setups: unsupervised learning, semi-supervised learning and applications where we need to fill in only partially known data. Not to mention more experimental approaches such as reinforcement learning or the way they contribute to supervised learning.
- Last but not least, Generative Adversarial Networks seem to match the way that lot of intelligent systems behave: they had their own mental model of what is likely to happen and try to simulate different futures or realities and then adjust based on the outcomes of these realities.

This list is a brief overview of why you could be interested in GANs; however, there are also legitimate reasons to remain cautions about this emerging technique.

[5] Nusca, A. (2017). Nvidia CEO Jensen Huang Is Fortune's 2017 Businessperson of the Year | Fortune. Retrieved June 21, 2018, from http://fortune.com/2017/11/16/nvidia-ceo-jensen-huang/

[6] LeCun, Y. (2016). Session with Yann LeCun - Quora. Retrieved June 21, 2018, from https://www.quora.com/session/Yann-LeCun/1

THE CONS

The promise of AI, GANs included, tends to be misrepresented by the popular media. So let's take some time to understand why it may be a good idea to hold off before diving into this topic.

- GANs are still in very early days both in terms of tooling and best practices for stability (as we will cover later). To build a stable in-production model is still incredibly difficult and very few companies have managed to do this successfully so far, at least as far as is publicly known.
- As any young field, it is not yet completely clear whether GANs will stand the test of time. One could apply the same argument to deep learning as a whole, but the risk to GANs is even greater. But it seems increasingly clear that the lower bound for GANs' impact is fundamentally changing at least parts of computer vision, which already seems like a significant feat.

Hopefully this will give you as balanced view as possible of reasons why to study or not to study this field. Now, it can be the case that this young field will get surpassed by the next hottest thing; however, it seems highly likely that any such novel technique would at least be inspired by the advancements brought about by GANs, thereby making their study worthwhile.

1.4 Applications of GANs

> *"GANs. All the cool kids are doing it."*
>
> **—Literally everyone**

GANs have achieved imagination-capturing results. In this book, we will be able to only scratch the surface of what is possible with GANs; however, our hope is that this book will provide you with the necessary theoretical knowledge and practical skills to continue exploring any facet of this field that you find most interesting. Below are just a few of the exiting applications we will learn to implement in this book.

GENERATING PHOTOREALISTIC FAKE IMAGES

Figure 1.3 Image Source: Tero Karras, Timo Aila, Samuli Laine: "Progressive Growing of GANs for Improved Quality, Stability, and Variation", 2017; arXiv:1710.10196.

None of the faces shown above are of real humans; they showcase GAN's ability to produce fake images in photo-realistic quality. These images were produced using Progressive Growing of GAN, a technique we will cover in Chapter 6.

IMAGE-TO-IMAGE TRANSLATION

Figure 1.4 Image Source: Jun-Yan Zhu, Taesung Park, Phillip Isola: "Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks", 2017; arXiv:1703.10593.

Through an implementation called CycleGAN (covered in Chapter 10), GANs have been able to translate an image of a horse into an image of zebra (and back!) and a photo into a Monet-like painting—all this with essentially no supervision and no labels whatsoever.

1.5 Guide to this Book

- **Chapter 2** discusses autoencoders, some of the most important theoretical and practical precursors to GANs.
- **Chapter 3** dives deeper into the theory behind GANs and adversarial training. In this chapter, we will also implement our first GAN.
- **Chapter 4** guides us through a more advanced GAN implementation (the so-called "Deep Convolutional GAN" or "DCGAN") which uses convolutional neural networks for its Generator and Discriminator.
- **Chapter 5** discusses many of the theoretical and practical hurdles to training GANs and how to overcome them.
- **Chapter 6** looks at the groundbreaking training methodology called "Progressive Growing of GANs" which is the technique that produced the photorealistic faces we saw earlier.
- **Chapter 7** covers the use of GANs in semi-supervised learning (training models using only a small fraction of labeled examples compared to fully supervised models), an area of immense practical importance.

- **Chapter 8** teaches us about the Conditional GAN, a technique that allows for targeted data generation by using labels or other conditioning information while training its Generator and Discriminator.
- **Chapter 9** explores the CycleGAN, a general-purpose technique for image-to-image translation: turning one image (e.g., a photo of a horse) into another (e.g., a photo of a zebra).
- **Chapter 10** discusses adversarial examples (a means of intentionally deceiving neural networks into making mistakes), a topic of great practical and theoretical importance.
- **Chapter 11** looks at several promising areas of practical applications of GANs.

1.6 Summary

- In this chapter, we introduced GANs and described how they work. We provided the intuition and outlined the theory behind the adversarial learning dynamic between the two networks that comprise a GAN:
    - The Generator, whose goal is fool the Discriminator by producing data indistinguishable from the training dataset.
    - The Discriminator, whose goal is to correctly distinguish between real data coming from the training dataset and fake data produced by the Generator.
- We also gave pros and cons for studying GANs and listed some of their most promising practical applications, such as image-to-image translation.