GANs

Generative Adversarial Networks, or GANs for short, are an approach to generative modeling using deep learning methods, such as convolutional neural networks.

Generative modeling is an unsupervised learning task in machine learning that involves automatically discovering and learning the regularities or patterns in input data in such a way that the model can be used to generate or output new examples that plausibly could have been drawn from the original dataset.

GANs are a clever way of training a generative model by framing the problem as a supervised learning problem with two sub-models: the generator model that we train to generate new examples, and the discriminator model that tries to classify examples as either real (from the domain) or fake (generated). The two models are trained together in a zero-sum game, adversarial, until the discriminator model is fooled about half the time, meaning the generator model is generating plausible examples.

GANs are an exciting and rapidly changing field, delivering on the promise of generative models in their ability to generate realistic examples across a range of problem domains, most notably in image-to-image translation tasks such as translating photos of summer to winter or day to night, and in generating photorealistic photos of objects, scenes, and people that even humans cannot tell are fake.

In this post, you will discover a gentle introduction to Generative Adversarial Networks, or GANs.

After reading this post, you will know:

- Context for GANs, including supervised vs. unsupervised learning and discriminative vs. generative modeling.
- GANs are an architecture for automatically training a generative model by treating the unsupervised problem as supervised and using both a generative and a discriminative model.
- GANs provide a path to sophisticated domain-specific data augmentation and a solution to problems that require a generative solution, such as image-to-image translation.

Kick-start your project with my new book Generative Adversarial Networks with Python, including *step-by-step tutorials* and the *Python source code* files for all examples.

Let’s get started.

A Gentle Introduction to Generative Adversarial Networks (GANs)

Photo by Barney Moss, some rights reserved.

Overview

This tutorial is divided into three parts; they are:

1. What Are Generative Models?
2. What Are Generative Adversarial Networks?
3. Why Generative Adversarial Networks?

What Are Generative Models?

In this section, we will review the idea of generative models, stepping over the supervised vs. unsupervised learning paradigms and discriminative vs. generative modeling.

Supervised vs. Unsupervised Learning

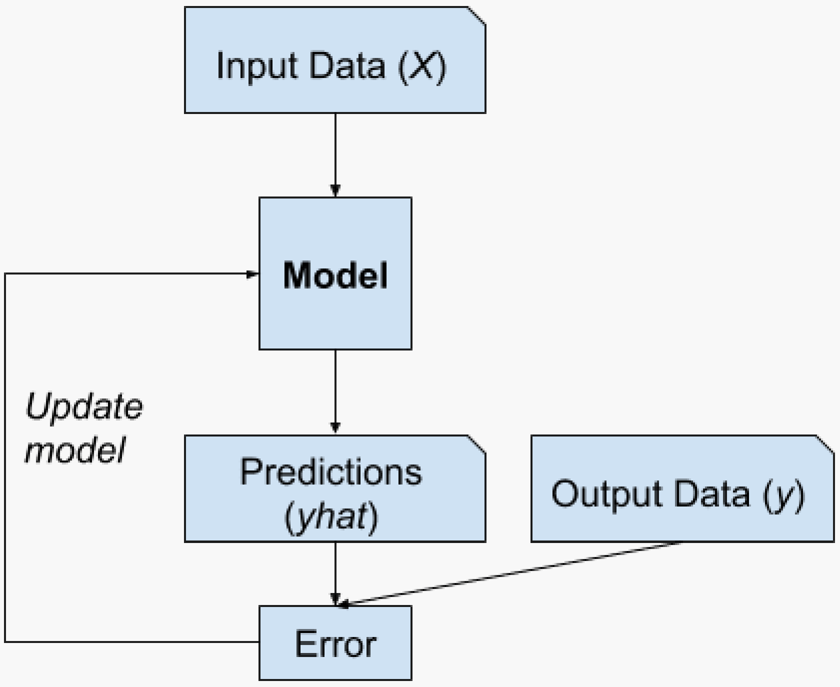
A typical machine learning problem involves using a model to make a prediction, e.g. predictive modeling.
This requires a training dataset that is used to train a model, comprised of multiple examples, called samples, each with input variables (*X*) and output class labels (*y*). A model

is trained by showing examples of inputs, having it predict outputs, and correcting the model to make the outputs more like the expected outputs.

In the predictive or supervised learning approach, the goal is to learn a mapping from inputs x to outputs y, given a labeled set of input-output pairs …

— Page 2, Machine Learning: A Probabilistic Perspective, 2012.

This correction of the model is generally referred to as a supervised form of learning, or supervised learning.



Example of Supervised Learning

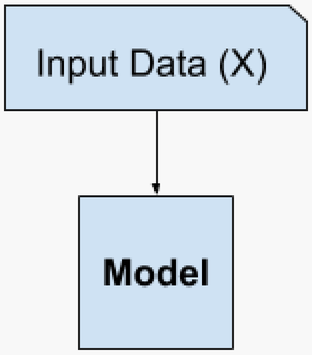
Examples of supervised learning problems include classification and regression, and examples of supervised learning algorithms include logistic regression and random forest.

There is another paradigm of learning where the model is only given the input variables (*X*) and the problem does not have any output variables (*y*).

A model is constructed by extracting or summarizing the patterns in the input data. There is no correction of the model, as the model is not predicting anything.

The second main type of machine learning is the descriptive or unsupervised learning approach. Here we are only given inputs, and the goal is to find "interesting patterns" in the *data. […] This is a much less well-defined problem, since we are not told what kinds of patterns to look for, and there is no obvious error metric to use (unlike supervised learning, where we can compare our prediction of y for a given x to the observed value).*

— Page 2, Machine Learning: A Probabilistic Perspective, 2012.
This lack of correction is generally referred to as an unsupervised form of learning, or unsupervised learning.



Example of Unsupervised Learning

Examples of unsupervised learning problems include clustering and generative modeling, and examples of unsupervised learning algorithms are K-means and Generative Adversarial Networks.

Want to Develop GANs from Scratch?

Take my free 7-day email crash course now (with sample code).

Click to sign-up and also get a free PDF Ebook version of the course.

Download Your FREE Mini-Course

Discriminative vs. Generative Modeling

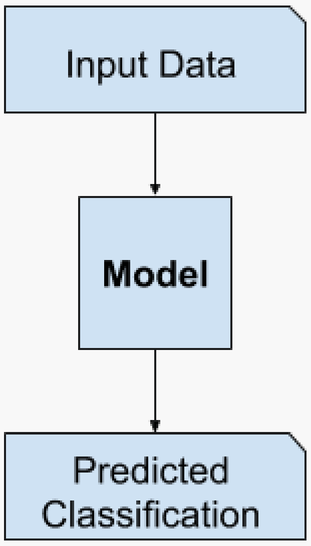
In supervised learning, we may be interested in developing a model to predict a class label given an example of input variables.

This predictive modeling task is called classification.

Classification is also traditionally referred to as discriminative modeling.

… we use the training data to find a discriminant function f(x) that maps each x directly onto a class label, thereby combining the inference and decision stages into a single learning problem.

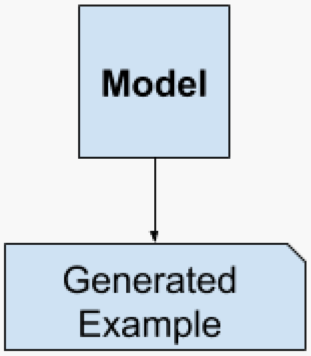
— Page 44, Pattern Recognition and Machine Learning, 2006.
This is because a model must discriminate examples of input variables across classes; it must choose or make a decision as to what class a given example belongs.



Example of Discriminative Modeling

Alternately, unsupervised models that summarize the distribution of input variables may be able to be used to create or generate new examples in the input distribution.

As such, these types of models are referred to as generative models.



Example of Generative Modeling

For example, a single variable may have a known data distribution, such as a Gaussian distribution, or bell shape. A generative model may be able to sufficiently summarize this data distribution, and then be used to generate new variables that plausibly fit into the distribution of the input variable.
Approaches that explicitly or implicitly model the distribution of inputs as well as outputs are known as generative models, because by sampling from them it is possible to generate synthetic data points in the input space.

— Page 43, Pattern Recognition and Machine Learning, 2006.
In fact, a really good generative model may be able to generate new examples that are not just plausible, but indistinguishable from real examples from the problem domain.

Examples of Generative Models

Naive Bayes is an example of a generative model that is more often used as a discriminative model.
For example, Naive Bayes works by summarizing the probability distribution of each input variable and the output class. When a prediction is made, the probability for each possible outcome is calculated for each variable, the independent probabilities are combined, and the most likely outcome is predicted. Used in reverse, the probability distributions for each variable can be sampled to generate new plausible (independent) feature values.

Other examples of generative models include Latent Dirichlet Allocation, or LDA, and the Gaussian Mixture Model, or GMM.

Deep learning methods can be used as generative models. Two popular examples include the Restricted Boltzmann Machine, or RBM, and the Deep Belief Network, or DBN.

Two modern examples of deep learning generative modeling algorithms include the Variational Autoencoder, or VAE, and the Generative Adversarial Network, or GAN.

What Are Generative Adversarial Networks?

Generative Adversarial Networks, or GANs, are a deep-learning-based generative model.

More generally, GANs are a model architecture for training a generative model, and it is most common to use deep learning models in this architecture.

The GAN architecture was first described in the 2014 paper by Ian Goodfellow, et al. titled "Generative Adversarial Networks."

A standardized approach called Deep Convolutional Generative Adversarial Networks, or DCGAN, that led to more stable models was later formalized by Alec Radford, et al. in the 2015 paper titled "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks".

Most GANs today are at least loosely based on the DCGAN architecture …

— NIPS 2016 Tutorial: Generative Adversarial Networks, 2016.

The GAN model architecture involves two sub-models: a *generator model* for generating new examples and a *discriminator model* for classifying whether generated examples are real, from the domain, or fake, generated by the generator model.

- **Generator**. Model that is used to generate new plausible examples from the problem domain.
- **Discriminator**. Model that is used to classify examples as real (*from the domain*) or fake (*generated*).

Generative adversarial networks are based on a game theoretic scenario in which the generator network must compete against an adversary. The generator network directly produces samples. Its adversary, the discriminator network, attempts to distinguish between samples drawn from the training data and samples drawn from the generator.

— Page 699, Deep Learning, 2016.

The Generator Model

The generator model takes a fixed-length random vector as input and generates a sample in the domain.

The vector is drawn from randomly from a Gaussian distribution, and the vector is used to seed the generative process. After training, points in this multidimensional vector space will correspond to points in the problem domain, forming a compressed representation of the data distribution.

This vector space is referred to as a latent space, or a vector space comprised of latent variables. Latent variables, or hidden variables, are those variables that are important for a domain but are not directly observable.

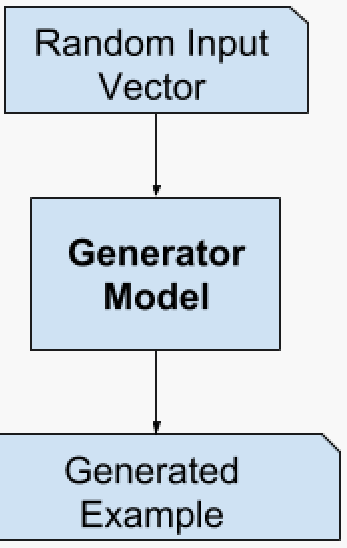
A latent variable is a random variable that we cannot observe directly.

— Page 67, Deep Learning, 2016.

We often refer to latent variables, or a latent space, as a projection or compression of a data distribution. That is, a latent space provides a compression or high-level concepts of the observed raw data such as the input data distribution. In the case of GANs, the generator model applies meaning to points in a chosen latent space, such that new points drawn from the latent space can be provided to the generator model as input and used to generate new and different output examples.

Machine-learning models can learn the statistical latent space of images, music, and stories, and they can then sample from this space, creating new artworks with characteristics similar to those the model has seen in its training data.

— Page 270, Deep Learning with Python, 2017.
After training, the generator model is kept and used to generate new samples.



Example of the GAN Generator Model

The Discriminator Model

The discriminator model takes an example from the domain as input (real or generated) and predicts a binary class label of real or fake (generated).

The real example comes from the training dataset. The generated examples are output by the generator model.

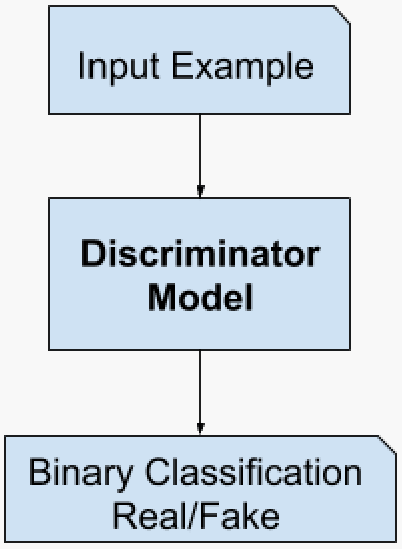
The discriminator is a normal (and well understood) classification model.

After the training process, the discriminator model is discarded as we are interested in the generator.

Sometimes, the generator can be repurposed as it has learned to effectively extract features from examples in the problem domain. Some or all of the feature extraction layers can be used in transfer learning applications using the same or similar input data.

We propose that one way to build good image representations is by training Generative Adversarial Networks (GANs), and later reusing parts of the generator and discriminator networks as feature extractors for supervised tasks

— Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks, 2015.



Example of the GAN Discriminator Model

GANs as a Two Player Game

Generative modeling is an unsupervised learning problem, as we discussed in the previous section, although a clever property of the GAN architecture is that the training of the generative model is framed as a supervised learning problem.

The two models, the generator and discriminator, are trained together. The generator generates a batch of samples, and these, along with real examples from the domain, are provided to the discriminator and classified as real or fake.

The discriminator is then updated to get better at discriminating real and fake samples in the next round, and importantly, the generator is updated based on how well, or not, the generated samples fooled the discriminator.

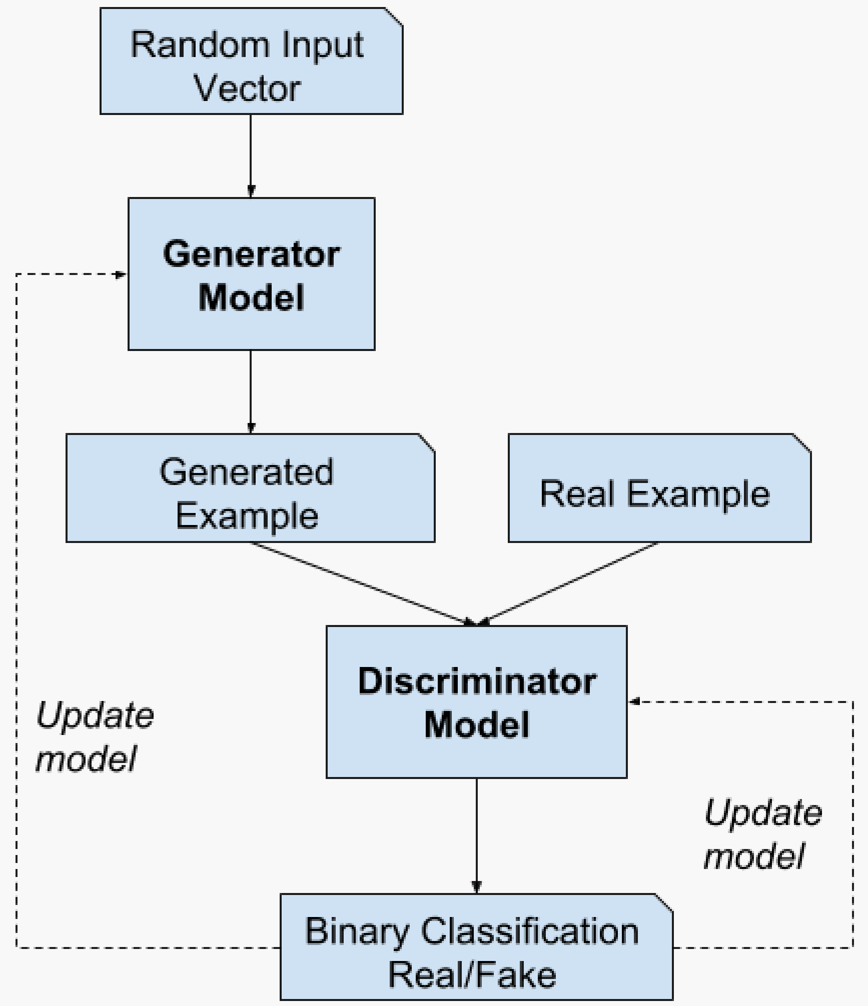
We can think of the generator as being like a counterfeiter, trying to make fake money, and the discriminator as being like police, trying to allow legitimate money and catch counterfeit money. To succeed in this game, the counterfeiter must learn to make money that is indistinguishable from genuine money, and the generator network must learn to create samples that are drawn from the same distribution as the training data.

— NIPS 2016 Tutorial: Generative Adversarial Networks, 2016.
In this way, the two models are competing against each other, they are adversarial in the game theory sense, and are playing a zero-sum game.
Because the GAN framework can naturally be analyzed with the tools of game theory, we call GANs "adversarial".

— NIPS 2016 Tutorial: Generative Adversarial Networks, 2016.
In this case, zero-sum means that when the discriminator successfully identifies real and fake samples, it is rewarded or no change is needed to the model parameters, whereas the generator is penalized with large updates to model parameters.

Alternately, when the generator fools the discriminator, it is rewarded, or no change is needed to the model parameters, but the discriminator is penalized and its model parameters are updated.

At a limit, the generator generates perfect replicas from the input domain every time, and the discriminator cannot tell the difference and predicts "unsure" (e.g. 50% for real and fake) in every case. This is just an example of an idealized case; we do not need to get to this point to arrive at a useful generator model.



Example of the Generative Adversarial Network Model Architecture

[training] drives the discriminator to attempt to learn to correctly classify samples as real or fake. Simultaneously, the generator attempts to fool the classifier into believing its samples are real. At convergence, the generator's samples are indistinguishable from real data, and the discriminator outputs 1/2 everywhere. The discriminator may then be discarded.

— Page 700, Deep Learning, 2016.

GANs and Convolutional Neural Networks

GANs typically work with image data and use Convolutional Neural Networks, or CNNs, as the generator and discriminator models.

The reason for this may be both because the first description of the technique was in the field of computer vision and used CNNs and image data, and because of the remarkable progress that has been seen in recent years using CNNs more generally to achieve state-of-the-art results on a suite of computer vision tasks such as object detection and face recognition.

Modeling image data means that the latent space, the input to the generator, provides a compressed representation of the set of images or photographs used to train the model. It also means that the generator generates new images or photographs, providing an output that can be easily viewed and assessed by developers or users of the model.

It may be this fact above others, the ability to visually assess the quality of the generated output, that has both led to the focus of computer vision applications with CNNs and on the massive leaps in the capability of GANs as compared to other generative models, deep learning based or otherwise.

Conditional GANs

An important extension to the GAN is in their use for conditionally generating an output.

The generative model can be trained to generate new examples from the input domain, where the input, the random vector from the latent space, is provided with (conditioned by) some additional input.

The additional input could be a class value, such as male or female in the generation of photographs of people, or a digit, in the case of generating images of handwritten digits.

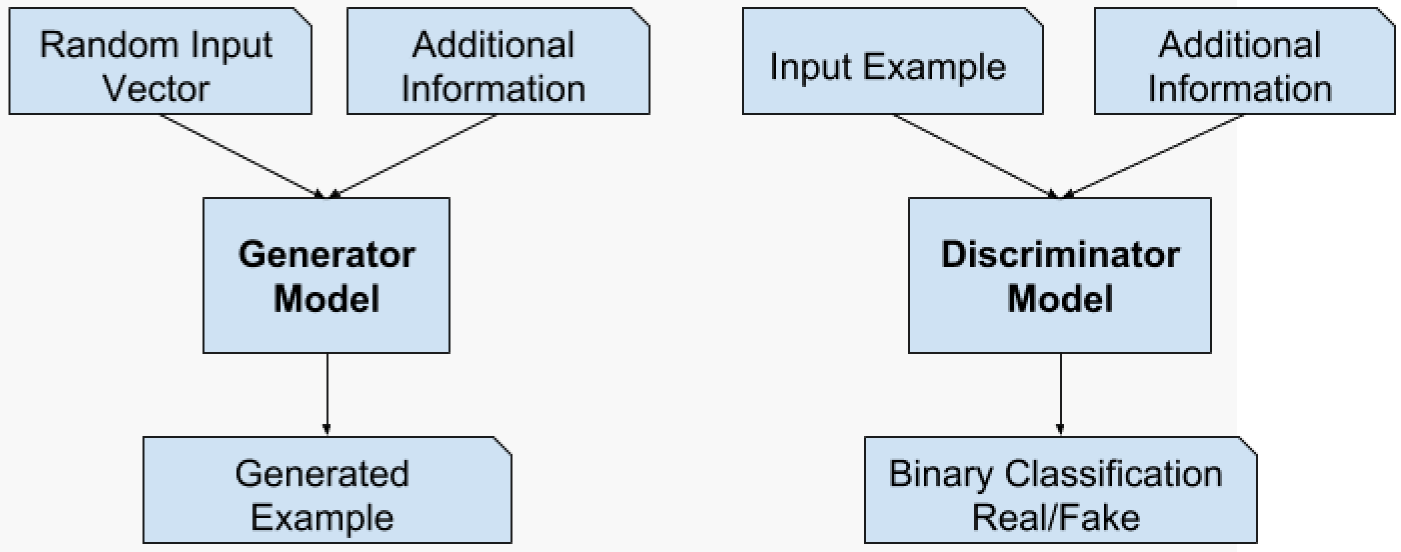
Generative adversarial nets can be extended to a conditional model if both the generator and discriminator are conditioned on some extra information y. y could be any kind of auxiliary information, such as class labels or data from other modalities. We can perform the conditioning by feeding y into the both the discriminator and generator as [an] additional input layer.

— Conditional Generative Adversarial Nets, 2014.
The discriminator is also conditioned, meaning that it is provided both with an input image that is either real or fake and the additional input. In the case of a classification label type conditional input, the discriminator would then expect that the input would be of that class, in turn teaching the generator to generate examples of that class in order to fool the discriminator.

In this way, a conditional GAN can be used to generate examples from a domain of a given type.

Taken one step further, the GAN models can be conditioned on an example from the domain, such as an image. This allows for applications of GANs such as text-to-image translation, or image-to-image translation. This allows for some of the more impressive applications of GANs, such as style transfer, photo colorization, transforming photos from summer to winter or day to night, and so on.

In the case of conditional GANs for image-to-image translation, such as transforming day to night, the discriminator is provided examples of real and generated nighttime photos as well as (conditioned on) real daytime photos as input. The generator is provided with a random vector from the latent space as well as (conditioned on) real daytime photos as input.



Example of a Conditional Generative Adversarial Network Model Architecture

Why Generative Adversarial Networks?

One of the many major advancements in the use of deep learning methods in domains such as computer vision is a technique called data augmentation.
Data augmentation results in better performing models, both increasing model skill and providing a regularizing effect, reducing generalization error. It works by creating new, artificial but plausible examples from the input problem domain on which the model is trained.

The techniques are primitive in the case of image data, involving crops, flips, zooms, and other simple transforms of existing images in the training dataset.

Successful generative modeling provides an alternative and potentially more domain-specific approach for data augmentation. In fact, data augmentation is a simplified version of generative modeling, although it is rarely described this way.

… enlarging the sample with latent (unobserved) data. This is called data augmentation. […] In other problems, the latent data are actual data that should have been observed but are missing.

— Page 276, The Elements of Statistical Learning, 2016.
In complex domains or domains with a limited amount of data, generative modeling provides a path towards more training for modeling. GANs have seen much success in this use case in domains such as deep reinforcement learning.

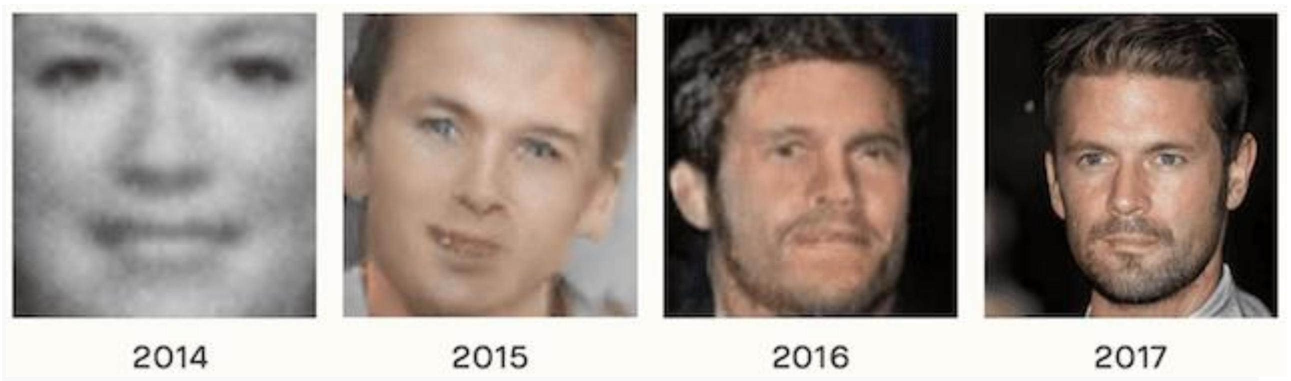
There are many research reasons why GANs are interesting, important, and require further study. Ian Goodfellow outlines a number of these in his 2016 conference keynote and associated technical report titled "NIPS 2016 Tutorial: Generative Adversarial Networks." Among these reasons, he highlights GANs' successful ability to model high-dimensional data, handle missing data, and the capacity of GANs to provide multi-modal outputs or multiple plausible answers.

Perhaps the most compelling application of GANs is in conditional GANs for tasks that require the generation of new examples. Here, Goodfellow indicates three main examples:

- **Image Super-Resolution**. The ability to generate high-resolution versions of input images.
- **Creating Art**. The ability to great new and artistic images, sketches, painting, and more.
- **Image-to-Image Translation**. The ability to translate photographs across domains, such as day to night, summer to winter, and more.

Perhaps the most compelling reason that GANs are widely studied, developed, and used is because of their success. GANs have been able to generate photos so realistic that humans are unable to tell that they are of objects, scenes, and people that do not exist in real life.

Astonishing is not a sufficient adjective for their capability and success.



Example of the Progression in the Capabilities of GANs From 2014 to 2017. Taken from The Malicious Use of Artificial Intelligence: Forecasting, Prevention, and Mitigation, 2018.

Further Reading

This section provides more resources on the topic if you are looking to go deeper.

Posts

- Best Resources for Getting Started With Generative Adversarial Networks (GANs)
- 18 Impressive Applications of Generative Adversarial Networks (GANs)

Books

- Chapter 20. Deep Generative Models, Deep Learning, 2016.
- Chapter 8. Generative Deep Learning, Deep Learning with Python, 2017.
- Machine Learning: A Probabilistic Perspective, 2012.
- Pattern Recognition and Machine Learning, 2006.
- The Elements of Statistical Learning, 2016.

Papers

- Generative Adversarial Networks, 2014.
- Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks, 2015.
- NIPS 2016 Tutorial: Generative Adversarial Networks, 2016.
- Conditional Generative Adversarial Nets, 2014.
- The Malicious Use of Artificial Intelligence: Forecasting, Prevention, and Mitigation, 2018.

Articles

- Generative model, Wikipedia.
- Latent Variable, Wikipedia.
- Generative Adversarial Network, Wikipedia.

Summary

In this post, you discovered a gentle introduction to Generative Adversarial Networks, or GANs.

Specifically, you learned:

- Context for GANs, including supervised vs. unsupervised learning and discriminative vs. generative modeling.
- GANs are an architecture for automatically training a generative model by treating the unsupervised problem as supervised and using both a generative and a discriminative model.
- GANs provide a path to sophisticated domain-specific data augmentation and a solution to problems that require a generative solution, such as image-to-image translation.