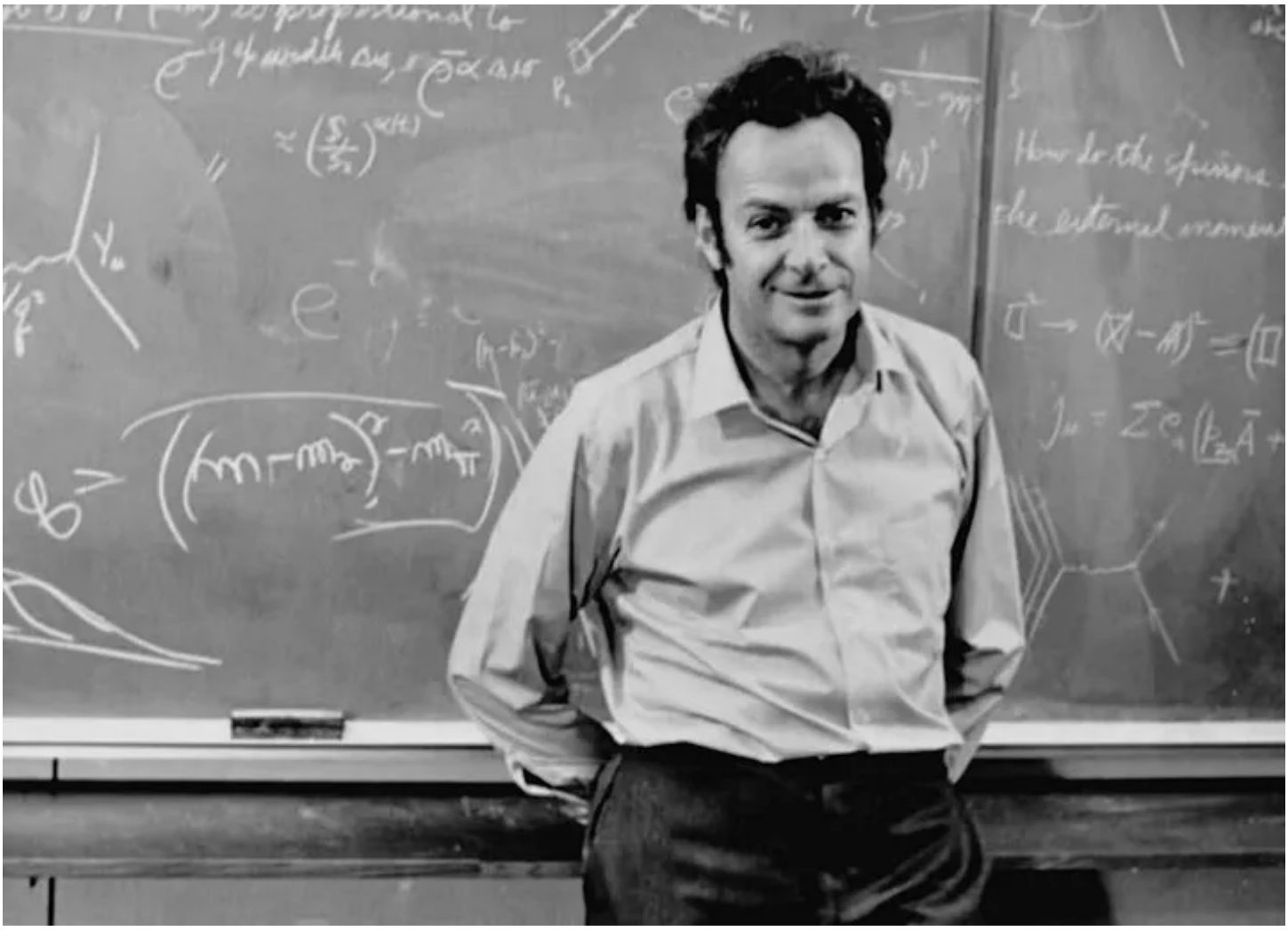
Get unlimited access to the best of Medium for less than $1/week. Become a member

Understanding GANs — Deriving the Adversarial loss from scratch

Hrithick Sen · Follow

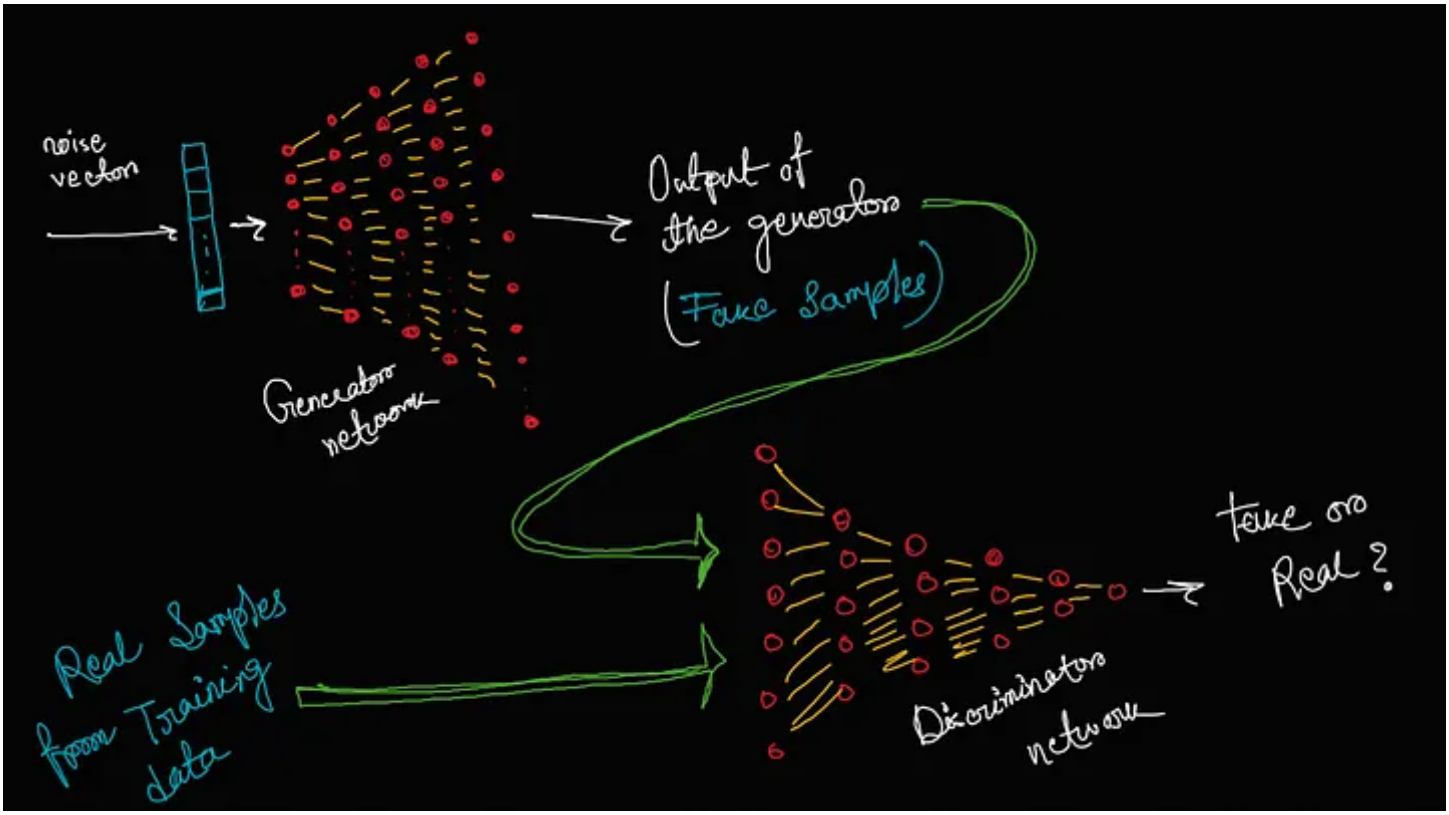
Published in Analytics Vidhya · 5 min read · Mar 3, 2021

120 2



Generative adversarial networks or GANs for short are an unsupervised learning task where the generator model learns to discover patterns in the input data in such a way that the model can be used to generate new samples of the training data.

The main idea of GAN is **adversarial training**, where two neural networks fight against each other and improve themselves to fight better.



The Generator takes a noise vector as input and then transforms the noise vector into a fake training sample, which is then passed through the discriminator. The discriminator takes both real samples (from the training data) and fake samples (generated by the generator), and then it tries to discriminate between fake and real samples. In other words, **the generator tries to fool the discriminator by showing it fake training data samples** and the discriminator tries to be as clever as possible.

The main idea is if the generator fools the discriminator then it means the discriminator should improve itself. On the other hand, if the discriminator

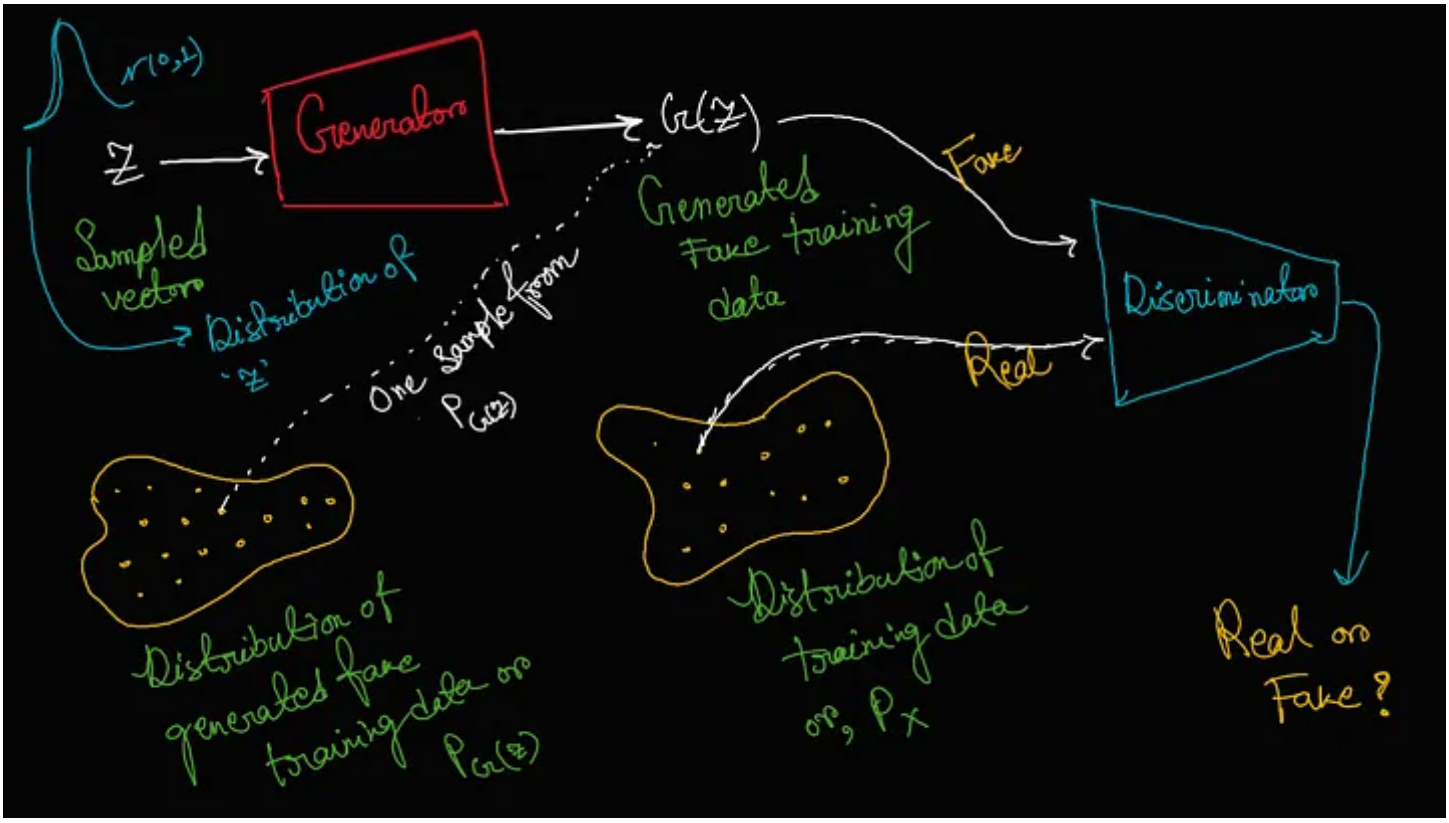
classifies fake and real samples perfectly then it means that the generator should improve itself so *that it can fool the discriminator!*

The possible causes are,

1. **The generator fools the discriminator** means the discriminator failed to classify a fake image sample. In this case, the discrimination should improve hence the loss will backpropagate through the discriminator only!
2. **The discriminator does a good job at classifying fake and real images**, which means the fake images are not good enough to confuse the discriminator. That means the generator should improve itself hence the loss will backpropagate through the generator network only!

BUT, the question is how can the generator fool the discriminator?

Intuitively, the generator learns the probability distribution of our training data. The below picture describes the idea intuitively.



In one sentence, The generator learns to approximate the distribution of the actual training data, and then it samples from the learned distribution. While training, there will be a fight between the generator and the discriminator, and both are trained alternatively while keeping the other one fixed!

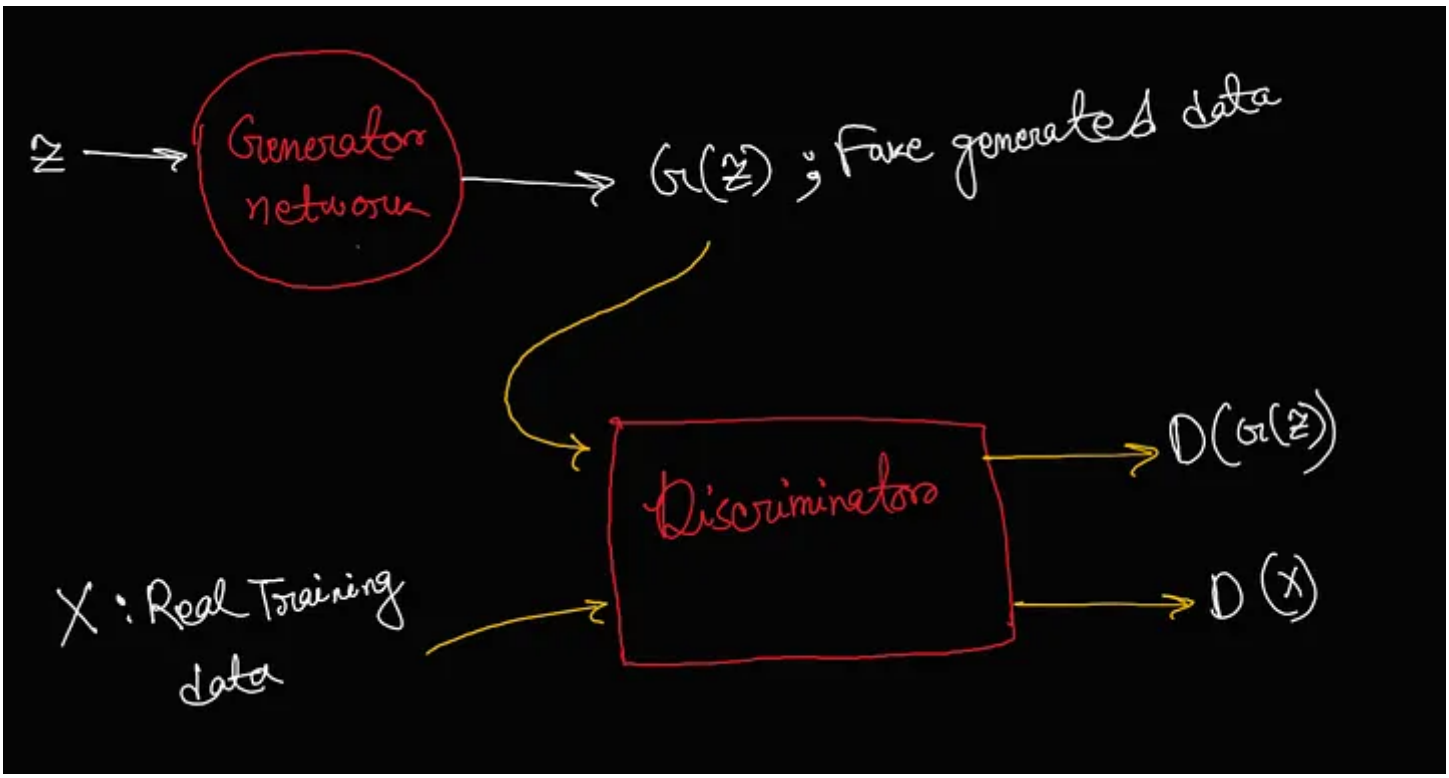
Deriving the adversarial loss:

The discriminator is nothing but a classifier that performs a binary classification(either Real or Fake). So, what loss function do we use for binary classification? Is not it **binary cross-entropy**?

The equation of the Binary cross-entropy loss function is given below.

$$L(\hat{y}, y) = y.\log \hat{y} + (1 - y).\log (1 - \hat{y})$$

Where, $\hat{y}$ is predicted and y is the ground truth label.



Z: Noise vector (Dimension of the noise vector is a hyperparameter).

G(Z): Output of the generator when given the noise vector Z.

X: Real Training data.

D(G(Z)): Output of the discriminator when given fake generated data or G(Z).

D(X): Output of the discriminator when given real training data from X.

The discriminator takes either X or G(Z). Note that the discriminator is nothing but a binary classifier so, **we label D(X) as 1 and D(G(Z)) as 0.**

We want our discriminator to label all D(X) as 1 and all D(G(Z)) as 0. Right?

So,

Open in app ↗

Search Write R

Or, $L(D(X), 1) = \log D(X)$

The discriminator should maximize Log(D(X)), and as Log is a monotonic function so Log(D(X)) will automatically get maximized if **the discriminator maximized D(X).**

On the other hand,

Given, $G(Z)$ as input, the predicted value is $D(G(Z))$ and the ground truth label is 0. Then,

$\therefore L(D(G(Z)), 0) = 0. \log D(G(Z)) + (1 - 0). \log (1 - D(G(Z))$

Or, $L(D(G(Z)), 0) = \log (1 - D(G(Z))$

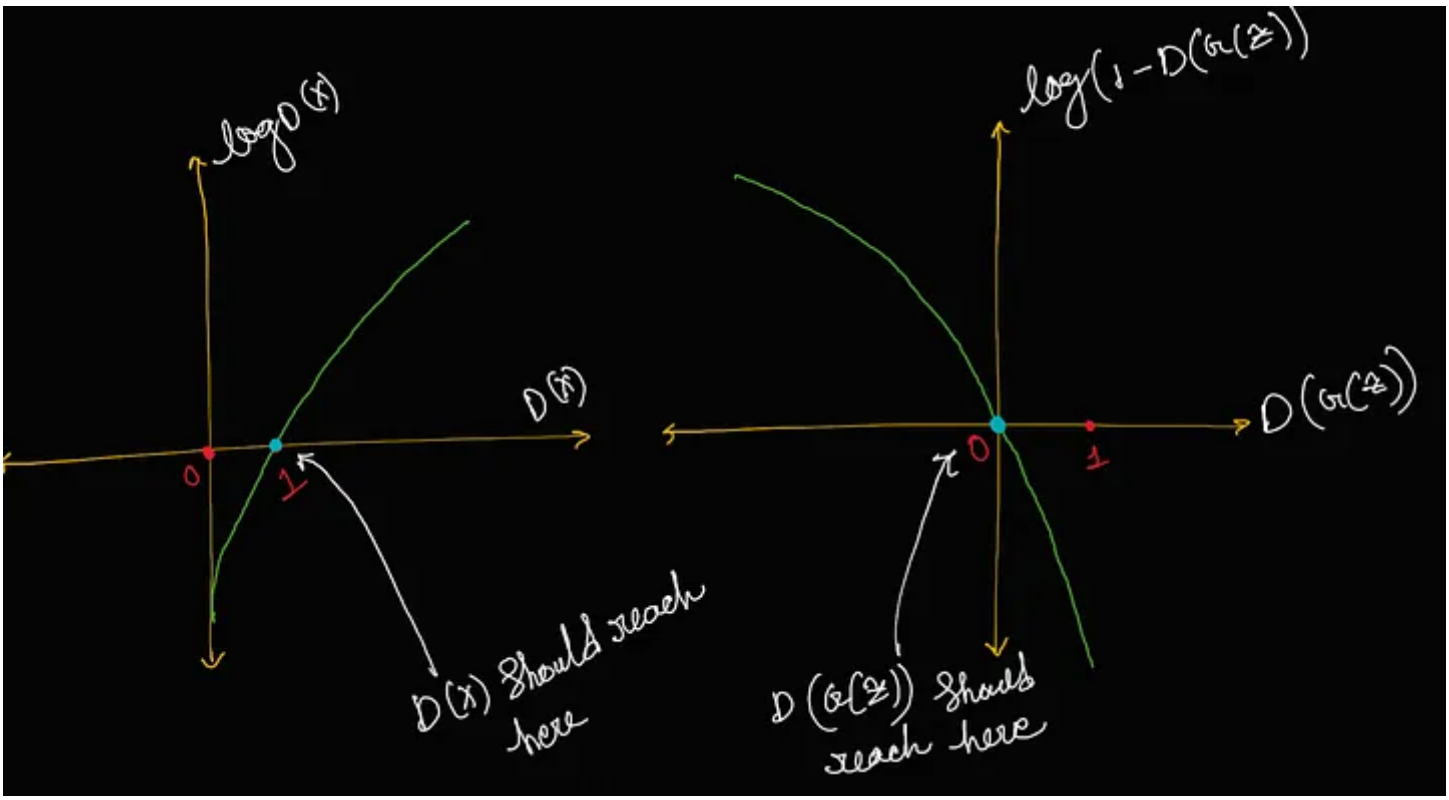
The discriminator needs to maximize log(1 — D(G(Z))), which means it must have to **minimize D(G(Z))**.

So, the loss function for the discriminator (for a single sample) becomes,

$$\text{Maximise}\Big[\log D(X) + \log\left(1 - D(G(Z))\right)\Big]$$

> *The discriminator will **maximise D(X)** and **minimise D(G(Z))** to overall maximize the above loss function.*

Note that, D(X) and D(G(Z)) both are probability values and **both of them lie in between 0 and 1.**



Now, the loss function of the discriminator over a batch is,

$$\text{Max}\left[\mathbb{E}_{(X\sim P(X))}\left[\log D(X)\right]+\mathbb{E}_{(Z\sim P(Z))}\left[\log\left(1-D(G(Z))\right)\right]\right]$$

> *Where, P(X) is the probability distribution of real training data and P(Z) is the probability distribution of the noise vector Z. Typically, P(Z) is gaussian or Uniform.*

The **Generator** needs to fool the discriminator by generating images as real as possible. This means **the generator should generate such G(Z) which if we pass through the discriminator will label is as 1.**

> *So, discriminator wants to make D(G(Z)) equal to 1 and generator wants to make D(G(Z)) equal to 0.*

So, from binary cross-entropy,

$$L(D(G(Z)),0)=\log\left(1-D(G(Z))\right)$$

The above one is for just one sample. Over a batch, it will be,

$$\therefore L(D(G(Z)),0)=\mathbb{E}_{Z\sim P(Z)}\left[\log\left(1-D(G(Z))\right)\right]$$

The generator will minimize the above loss function and to minimize the above the generator must maximize D(G(Z)). Now, it is very clear that **the**

discriminator wants to minimize D(G(Z)) and the generator wants to maximize D(G(Z)).

Understand that, the generator is never going to see any real data but for completeness, the generator loss function can be written as follows!

$$\text{Min}\left[\mathbb{E}_{(X\sim P(X))}\left[\log D(X)\right]+\mathbb{E}_{(Z\sim P(Z))}\left[\log\left(1-D(G(Z))\right)\right]\right]$$

Note that, the generator has no control over the first term so the **generator will only minimize the second term.**

Assume that, **D** is the parameters of the Discriminator and **G** is the parameters of the generator. So, we can write the loss function as,

$$\underset{G}{Min}\ \underset{D}{Max}\left[\mathbb{E}_{(X\sim P(X))}\left[\log D(X)\right]+\mathbb{E}_{(Z\sim P(Z))}\left[\log\left(1-D(G(Z))\right)\right]\right]$$

This means the discriminator parameters (defined by D) will maximize the loss function and the generator parameters (defined by G) will minimize the loss function.

> *The adversarial loss can be optimized by gradient descent. But while training a GAN **we do not train the generator and discriminator simultaneously**, while training the generator we freeze the discriminator and vice-versa!*

The original GAN paper provides a pseudo-code that shows how a GAN is trained.

for number of training iterations **do**

for k steps **do**

- Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
- Sample minibatch of m examples $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\boldsymbol{x})$.
- Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right) \right].$$

end for

- Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
- Update the generator by descending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right).$$

I think this was not too hard to follow! :)

References:

Generative adversarial network

A generative adversarial network (GAN) is a class of machine learning frameworks designed by Ian Goodfellow and his...

en.wikipedia.org

Generative Adversarial Networks

We propose a new framework for estimating generative models via an adversarial process, in which we simultaneously...

arxiv.org

Deep Learning | Machine Learning | AI | Math | Programmin

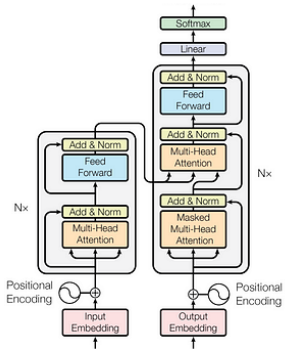
Written by Hrithick Sen

29 Followers · Writer for Analytics Vidhya

Engineer | Data Scientist | Thinker LinkedIn: https://www.linkedin.com/in/hrithick-sen-58ab1619b

Follow

More from Hrithick Sen and Analytics Vidhya



Hrithick Sen

The Math Behind the Machine: A Deep Dive into the Transformer...

The transformer architecture was introduced in the paper “Attention is All You Need” by...

12 min read · 5 days ago

57 2

Shashank Singhal in Analytics Vidhya

Hadoop : How to install in 5 Steps in Windows 10

An easy to go guide for installing the Hadoop in Windows

7 min read · Mar 27, 2021

115 14

Ampofo Amoh - Gyebi in Analytics Vidhya

How to build your first Desktop Application in Python

Search the entire internet for uses of the Python programming language and they list...

12 min read · Dec 12, 2020

1.3K 19

Hrithick Sen

Understanding principle component analysis (PCA)—Fro...

Principle component analysis is the most basic and simple dimensionality reduction...

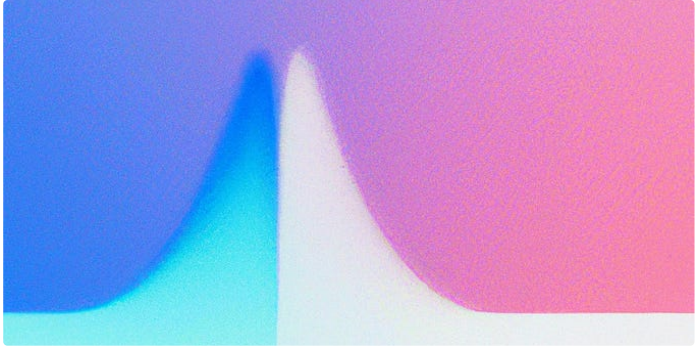
5 min read · Dec 4, 2021

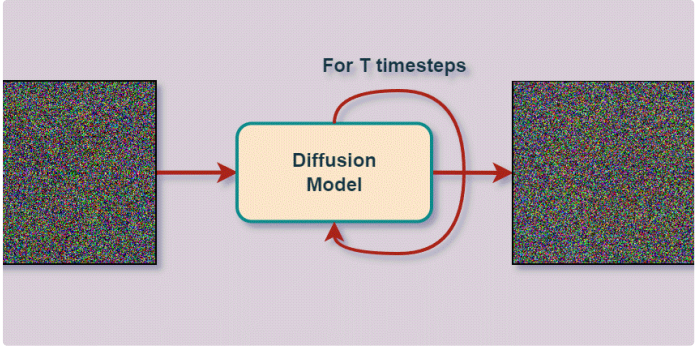
73 1

See all from Hrithick Sen

See all from Analytics Vidhya

Recommended from Medium





Aparna Dhinakaran in Towards Data Science

How to Understand and Use the Jensen-Shannon Divergence

A primer on the math, logic, and pragmatic application of JS Divergence—including ho...

9 min read · Mar 2, 2023

346 2

Gabriel Mongaras in Better Programming

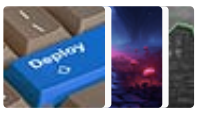
Diffusion Models—DDPMs, DDIMs, and Classifier Free Guidance

A guide to the evolution of diffusion models from DDPMs to Classifier Free guidance

28 min read · Mar 13, 2023

566 8

Lists



Predictive Modeling w/ Python

20 stories · 813 saves

Practical Guides to Machine Learning

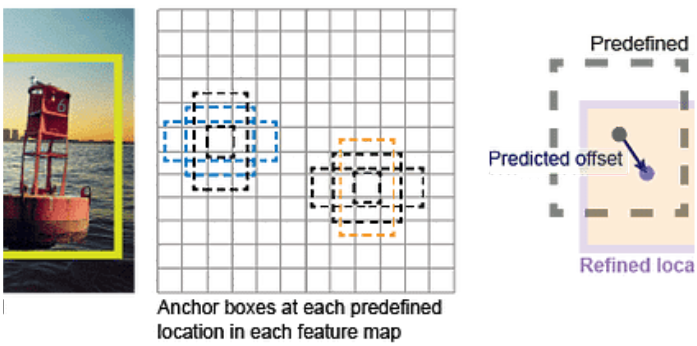
10 stories · 947 saves

Natural Language Processing

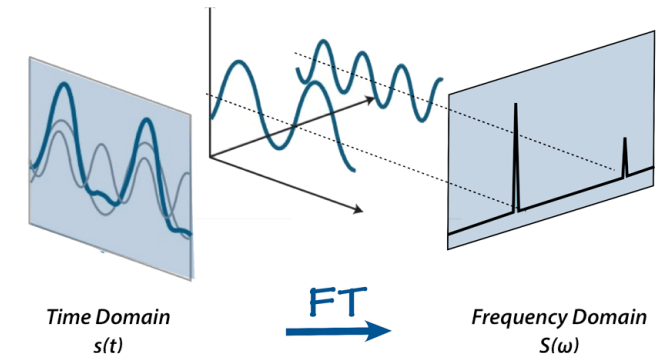
1112 stories · 586 saves

The New Chatbots: ChatGPT, Bard, and Beyond

12 stories · 276 saves



Nikita Malviya



Everton Gomede, PhD in The Modern Scientist

Object Detection — Anchor Box VS Bounding Box

In object detection algorithms like Faster R-CNN and YOLO, anchor boxes are used to...

2 min read · Aug 6, 2023

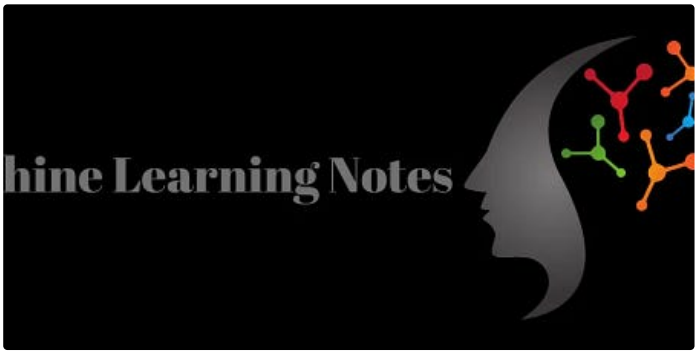
31

The Fourier Transform and its Application in Machine Learning

Introduction

5 min read · Nov 3, 2023

1.4K 10



Rahul S

Machine Learning: Cross Entropy and Cross-Entropy Loss

Cross Entropy and Cross-Entropy Loss are closely related concepts, but they serve...

· 2 min read · Oct 3, 2023

4

Wasurat Soontronchai

Building a simple GAN model

Generative AI is super hype nowadays, let learn to build simple GAN model from...

10 min read · Aug 1, 2023

72 1

See more recommendations