GANs [Generative Adversarial Networks]

@Author : **Saurabh Kumar**
@date : **12-Jan**

Deep learning based geneative adversarial network

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 formalizedby 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

Architecture :

1. *Generator*
2. *Descriminators*

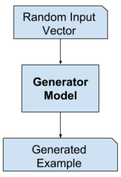
Both are implemented as MLP (Multilayer Perceptron)

- **Generator** : Model that is used to generate new plausible examplefrom 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

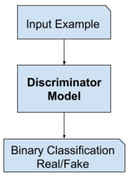
The Generator Model:

- 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.
- 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.



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.



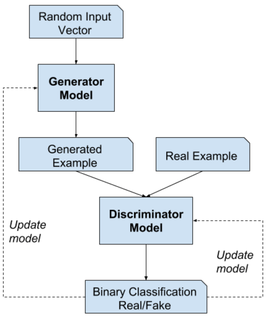
GANs as a Two Player Game

- Generative modeling is an unsupervised learning problem, 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.

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.

- 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.



> 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

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.

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.

Gans Applications

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

!!! Thanks For Reading ....

In [ ]: