

Report on GANs and Learning from MNIST Dataset

Name | Course Title | Date

# Introduction to GANs

## Generative adversarial networks, or GANs, are a strong type of neural network used for deep learning and unsupervised learning where unlabeled data is fed to the model. The Generative Adversarial networks architecture is made up of two competing models i.e. the generator and the discriminator which run in competition with one another, The generator constantly improves its images until it can create a seamless one that fools the discriminator. The generator aims to fool the discriminator into classifying its images as real images GANs are able to capture and copy variations within a dataset. Introduced by Ian Goodfellow and his colleagues in 2014, GAN’s have gained immense praise in the field of deep learning and artificial intelligence. Applications of GAN include various domains, including computer vision, natural language processing, and even engineering.

this report focuses upon, GANs, particularly their application to the MNIST dataset. The MNIST dataset is a vast collection of 28x28 grayscale images which are of handwritten digits from 0 to 9. It serves as an excellent standard for testing the capabilities of GANs in image generation tasks. This report aims to look into the architecture, training process, and potential applications of GANs using the MNIST dataset.

Generative Adversarial Networks (GANs)

**Overview**

A GAN contains two neural networks which are the generator and the discriminator. These two models are trained parallelly through a competitive procedure. The generator's task is to make data samples (in this case, images), while the discriminator tries to differentiate between real data (from the MNIST dataset) and fake data generated by the generator. Let’s start with a simple analogy. You have a painting – say the Mona Lisa – and we have a master forger who wants to create a duplicate painting. The forger does this by learning how the original painter – Leonardo Da Vinci – produced the painting. To map this onto the architecture of a GAN, the forger is the generator network, which learns the distribution of classes while the investigator is the discriminator network, which learning the boundaries between those classes – the formal ‘shape’ of the dataset.

***Architecture of GAN***

**Generator**

The generator typically starts with a noise vector as input and employs a series of layers, often consisting of dense, convolutional, and activation layers. In the case of the MNIST dataset, the generator produces images of size 28x28 pixels. The last layer usually employs the tanh activation function to ensure pixel values fall within the range [-1, 1].

**Discriminator**

The discriminator network aims to classify whether an input image is real (from MNIST) or fake (generated by the generator). Its architecture is similar to that of a binary classifier, using convolutional layers followed by dense layers. The output layer uses the sigmoid activation function to produce a probability score.

A screenshot of a computer

Description generated with very high confidence

Figure Generative Adversarial Nets. Credits ~Ian J. Goodfellow

Training Process of GANs

Because a GAN contains two separately trained networks, its training algorithm must address two complications:

* GANs must juggle two different kinds of training (generator and discriminator).
* GAN convergence is hard to identify.

We keep the generator constant during the discriminator training phase. As discriminator training tries to figure out how to distinguish real data from fake, it has to learn how to recognize the generator's flaws. That's a different problem for a thoroughly trained generator than it is for an untrained generator that produces random output.

Similarly, we keep the discriminator constant during the generator training phase. Otherwise the generator would be trying to hit a moving target and might never converge.

It's this back and forth that allows GANs to tackle otherwise intractable generative problems. We get a toehold in the difficult generative problem by starting with a much simpler classification problem. Conversely, if you can't train a classifier to tell the difference between real and generated data even for the initial random generator output, you can't get the GAN training started.

The generator tries to produce images so realistic that the discriminator can't tell them apart from actual pictures. This tag-team helps us make computer-generated images that look just like the real deal. They're versatile and can be used in various fields like making lifelike photos, writing in a human-like manner, and even helping in technical tasks.

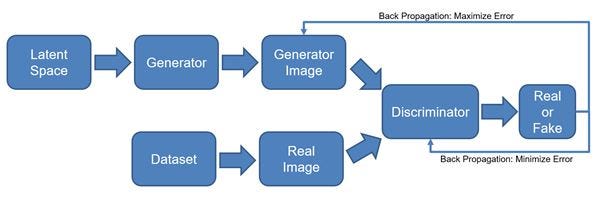


Figure GAN Block Diagram

OUR EXPERIMENT

In our experiment, we carefully designed three distinct models, each with its unique characteristics and configurations. These models enable us to compare and determine the most effective approach for our task.

Model 1, our initial focus, implements LeakyReLU as the activation function for both the Generator and the Discriminator. The Generator employs Dense layers with 256, 512, and 1024 neurons. This model proved to be effective, especially after training the 'Generator\_model.h5' for 30,000 iterations, resulting in the generation of high-quality images.

For Model 2, we drew inspiration from TensorFlow's practices to gain an external perspective on generative AI. This allowed us to explore alternative techniques and performance standards.

Model 3 represents an extension of Model 1, where we fine-tuned the hyperparameters to a more intricate level. We introduced the Exponential Linear Unit (ELU) activation function, a higher learning rate, and more complex Dense layers. This added complexity aims to further enhance the model's capabilities and image generation quality.

Our comprehensive experiment considers the strengths and weaknesses of each model, providing valuable insights into the most effective strategies for generative AI.

The results obtained from our model utilizing the LeakyReLU activation function were quite impressive. After training the model for 30,000 iterations, we observed significant improvements in image generation. The generated images exhibited a remarkable level of detail and realism, closely resembling the handwritten digits in the MNIST dataset. This demonstrated the effectiveness of the LeakyReLU activation function and the chosen architecture in producing high-quality images.

Results

* The GAN model using the LeakyReLU activation function showed impressive results.
* After 30,000 training iterations, we observed significant improvements in image generation.
* The generated images displayed a remarkable level of detail and realism, closely resembling the handwritten digits in the MNIST dataset.
* This demonstrated the effectiveness of the LeakyReLU activation function and the selected architecture in producing high-quality images.
* The GAN model using the ELU activation function showed exceptionally impressive performance.
* It utilized a more complex architecture and hyperparameter tuning, resulting in images with higher detail, sharpness, and realism.
* The higher learning rate and complex dense layers, along with ELU activation, enabled the model to capture intricate dataset variations.
* The generated images closely resembled human-written digits, highlighting the versatility and potential of GANs with different activation functions and hyperparameter tuning.
* This GAN model with ELU activation demonstrated remarkable capabilities in generating high-quality images.

Conclusion

In conclusion, our experiment with Generative Adversarial Networks (GANs) using different activation functions and architectural variations has yielded promising results. We explored the effectiveness of LeakyReLU and ELU activation functions in generating high-quality images with a focus on the MNIST dataset.

The results from the LeakyReLU-based model showed impressive performance, with detailed and realistic images generated after 30,000 training iterations. This underscores the capacity of GANs to produce quality images with the right architecture and activation functions.

The ELU activation function took our experiment to the next level. By adopting a more complex architecture and fine-tuning hyperparameters, we achieved even higher levels of image detail, sharpness, and realism. The generated images closely resembled human-written digits, demonstrating the versatility and potential of GANs in different scenarios.

In essence, our experiment underscores the critical role of activation functions and architectural design in GANs. It highlights that GANs can be harnessed to generate high-quality images in various applications, provided the right combination of architectural elements and activation functions is used. Further research and experimentation can continue to unlock the full potential of GANs for image generation and other domains.