Using DCGAN to generate images of brains

|  |  |  |
| --- | --- | --- |
| Dan Mohler | Jordan Winkler | Prad Ejner |
| dmohler@iu.edu | jomawink@iu.edu | [prad.ejner@gmail.com](mailto:prad.ejner@gmail.com) |
| Computer and Information Sciences | Computer and Information Sciences | Computer and Information Sciences |
| Indiana University South Bend | Indiana University South Bend | Indiana University South Bend |
| South Bend, IN, USA | South Bend, IN, USA | South Bend, IN, USA |

*Abstract*— *Using a DCGAN(Deep Convolutional Generative Adversarial Network) on a dataset of brain MRIs we were able to generate fake images of brain MRIs. Then comparing generated images with the real images using Frechet Inception Distance(FID) to measure how well the DCGAN model performed.*

Keywords—GAN, DCGAN, FID, medical, brain

# Introduction

Generative adversarial networks (GANs) are a recent innovation (2014) in machine learning invented by Ian Goodfellow and his colleagues. Given a training set, a neural network will learn to generate new data with the same statistics as the training set. For example, GANs can use a training set of photographs of human faces to create images of human faces. These generated images don't belong to any real person.

GANs consist of two neural networks that contest with each other. The generative network generates plausible candidates while the discriminative network evaluates candidates by distinguishing the generator's fake data from real data. At the beginning of training, the generator will produce data that is obviously fake and the discriminator quickly learns to tell that it is fake data. As training continues, the generator will continue to get closer to producing output that can fool the discriminator. Eventually, if the generator training goes well, the generator will produce data which is difficult to distinguish as fake or real. The discriminator gets worse at telling the differences and accuracy decreases.

# Discriminator

The discriminator in a GAN is a classifier. The discriminator tries to distinguish real data from the data created by the generator.

## Discriminator Training

During discriminator training, the generator does not train. The weights for the generator remain constant while it produces images for the discriminator to train on.

The data used for training the discriminator comes from two sources: real data are considered positive instances during training, and fake data are considered negative instances during training.

If the discriminator misclassifies real instance as fake or fake data as real, the discriminator loss penalizes the discriminator. Discriminator will use the discriminator loss to update its weights through backpropagation.

# Generator

The goals of the generator is to fool the discriminator. Using feedback from the discriminator, the generator will create fake data and learn to make the discriminator classify generator output as real data.

Typically with with deep learning models, input data is an instance that we want to classify or make a prediction about. However, with GANs we want output that is entirely new data instances. Therefore, the generator will start by taking random noise as its input. The generator will then transform this noise into meaningful output.

Typically, we alter a neural network's weights to reduce the error or loss of its output. However, with GANs, the generator is not directly connected to the loss we are trying to affect(the discriminator is). The generator loss penalizes the generator whenever it produces data that the discriminator classifies as fake.

Backpropagation adjusts each weight by calculating the weight's impact on the output. Backpropagation start with the output from the discriminator and flows back through the disciminator into the generator to obtain gradients. The gradients are used to change the generator weights.

# Measuring Results

In GANs, the generator and the discriminator measure how well they are doing relative to each other. For example, we measure how the generator is fooling the discriminator and how well the discriminator identifies data instances as real or fake. These metrics do not measure image quality or its diversity.

## Fréchet Inception Distance (FID)

The Frechet Inception Distance score, or FID for short, is a metric that calculates the distance between feature vectors calculated for real and generated images using a statistical representation of both. The probability variable for the real image can be represented as a multivariate Normal distribution Xr ~ N(μr,  Σr) and the generated one as Xr ~ N(μr,  Σr). Giving

FID=||*μr*−*μ**g*||2+Tr(Σ*r*+Σg−2(ΣrΣg)1/2)

Tr is the trace of a matrix. And ||.|| is the euclidean distance between vectors. Lower FID score indicates the two groups are more similar or have more similar statistics. The perfect score would be 0. 0 indicating that the two groups of images are identical. FID scores are used to evaluate the quality of images generated by GANs and lower scores have been shown to correlate well with higher quality images.

## History of FID

FID scores were proposed in 2017 by Martin Heusel, et al. in "GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium".

FID scores was proposed as an improvement over the existing Inception Score (IS) since the Inception Score did not capture how synthetic images compared to real images. Heusel et al. showed how lower FID scores correlate with better-quality images when systematic distortions were applied such as blur and random noise.

**Original Values:**

| Dropout | bn\_momentum | adam\_lr | adam\_beta | FID Score |
| --- | --- | --- | --- | --- |
| 0.25 | 0.80 | 0.0002 | 0.50 | 192.77 |

**Modified Values:**

| Dropout | bn\_momentum | adam\_lr | adam\_beta | FID Score |
| --- | --- | --- | --- | --- |
| 0.27 | 0.80 | 0.0002 | 0.50 | 205.93 |
| 0.30 | 0.80 | 0.0002 | 0.50 | 144.54 |
| 0.33 | 0.80 | 0.0002 | 0.50 | 164.24 |
| 0.36 | 0.80 | 0.0002 | 0.50 | 161.72 |
| 0.39 | 0.80 | 0.0002 | 0.50 | 161.21 |
| 0.42 | 0.80 | 0.0002 | 0.50 | 117.77 |
| 0.45 | 0.80 | 0.0002 | 0.50 | 110.27 |
| 0.25 | 0.70 | 0.0002 | 0.50 | 187.83 |
| 0.25 | 0.75 | 0.0002 | 0.50 | 173.23 |
| 0.25 | 0.82 | 0.0002 | 0.50 | 156.22 |
| 0.25 | 0.80 | 0.00015 | 0.50 | 189.31 |
| 0.25 | 0.80 | 0.00025 | 0.50 | 201.41 |
| 0.25 | 0.80 | 0.0003 | 0.50 | 222.07 |
| 0.25 | 0.80 | 0.0001 | 0.50 | 184.46 |
| 0.25 | 0.80 | 0.0002 | 0.40 | 186.82 |
| 0.25 | 0.80 | 0.0002 | 0.45 | 176.28 |
| 0.25 | 0.80 | 0.0002 | 0.55 | 190.34 |
| 0.25 | 0.80 | 0.0002 | 0.60 | 217.35 |

##### References

1. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., & Hochreiter, S. (2017). Gans trained by a two time-scale update rule converge to a local nash equilibrium. In Advances in Neural Information Processing Systems (pp. 6626–6637).
2. Radford, L. Metz, and S. Chintala, (2016) “Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks,” ArXiv Preprint, ArXiv:1511.06434