

# PyTorch-GAN

---

Collection of PyTorch implementations of Generative Adversarial Network varieties presented in research papers. Model architectures will not always mirror the ones proposed in the papers, but I have chosen to focus on getting the core ideas covered instead of getting every layer configuration right. Contributions and suggestions of GANs to implement are very welcomed.

See also: [Keras-GAN](#)

## Table of Contents

---

- [Installation](#)
- [Implementations](#)
  - [Auxiliary Classifier GAN](#)
  - [Adversarial Autoencoder](#)
  - [BEGAN](#)
  - [BicycleGAN](#)
  - [Boundary-Seeking GAN](#)
  - [Conditional GAN](#)
  - [Context-Conditional GAN](#)
  - [Context Encoder](#)
  - [Coupled GAN](#)
  - [CycleGAN](#)
  - [Deep Convolutional GAN](#)
  - [DiscoGAN](#)
  - [DRAGAN](#)
  - [DualGAN](#)
  - [Energy-Based GAN](#)
  - [GAN](#)
  - [InfoGAN](#)
  - [Least Squares GAN](#)
  - [MUNIT](#)
  - [Pix2Pix](#)
  - [PixelDA](#)
  - [Semi-Supervised GAN](#)
  - [Softmax GAN](#)
  - [StarGAN](#)
  - [Super-Resolution GAN](#)
  - [UNIT](#)
  - [Wasserstein GAN](#)
  - [Wasserstein GAN GP](#)
  - [Wasserstein GAN DIV](#)