Generating logo images with Generative Adversarial Networks

Hristo Todorov   Hristo Kanev
Scientific advisors: Dr. Stoyan Vellev

August 14, 2020

Introduction

- What are we trying to achieve?
- How are we trying to achieve it?
- What is the main question we want to answer?

Dataset

- Large Logo Dataset (LLD)
- Composed of over 600 000 high-quality diverse images
- Can be used freely for academic purposes

Figure: Sample images from the dataset

Generative Adversarial Networks

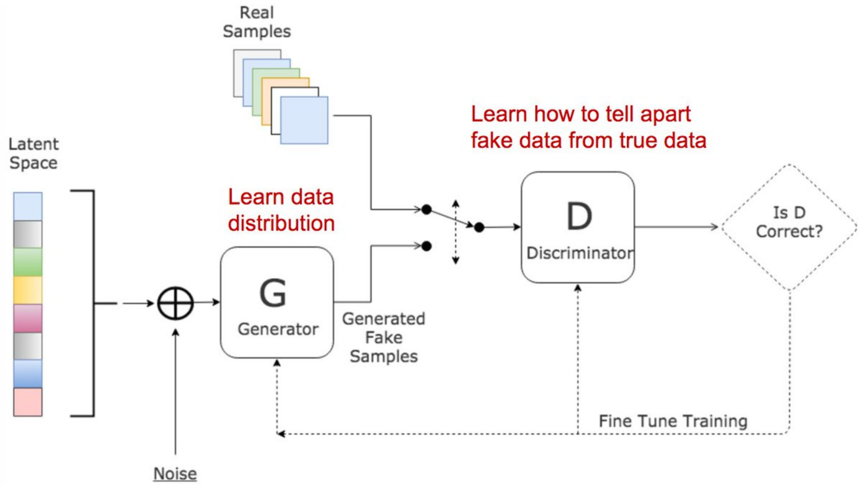

Figure: Architecture of a GAN

Training a Generative Adversarial Network

- Trained via backpropagation - no need for any Markov chains
- Really unstable to train
- The loss (cost) function heavily affects the training process

Binary cross-entropy

$$\min_G \max_D V(D, G) = \mathbb{E}_{x \sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))] \quad (1)$$

Least Squares Loss

$$\min_D V_{LSGAN} = \frac{1}{2}\mathbb{E}_{x \sim p_{data}(x)}[(D(x) - 1)^2] + \frac{1}{2}\mathbb{E}_{z \sim p_z(z)}[(D(G(z))^2] \quad (2)$$

$$\min_G V_{LSGAN} = \frac{1}{2}\mathbb{E}_{z \sim p_z(z)}[(D(G(z))^2] \quad (3)$$

Wasserstein Generative Adversarial Network

- Minimal changes to the architecture of the discriminator
- Harder to train
- No sign of vanishing gradients and mode collapse

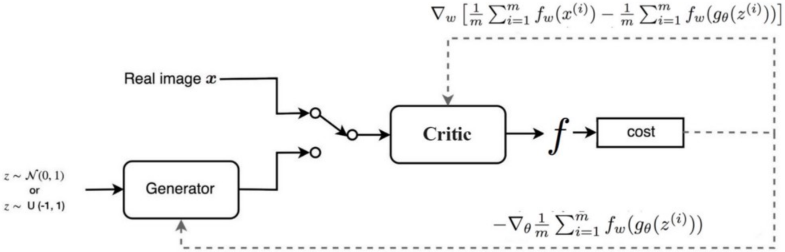

Figure: Architecture of a WGAN

Conditional Generative Adversarial Network

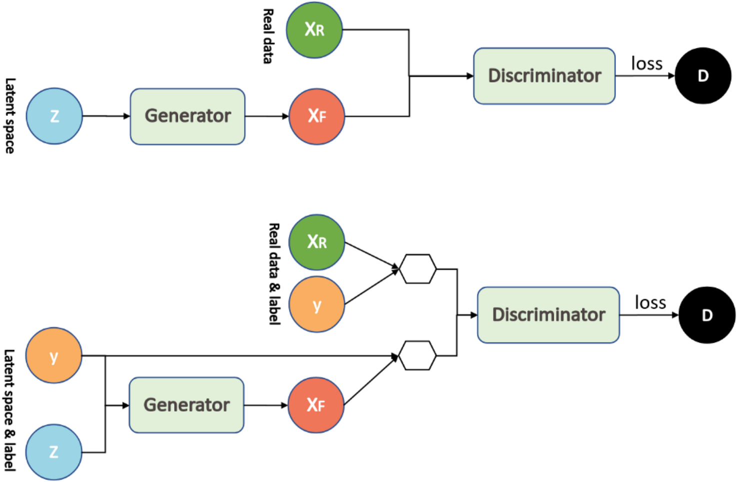

Figure: Comparison between a regular GAN and a CGAN

Feature extraction via Transfer learning

- Transfer learning - using a neural network trained at solving one problem to solve different, but similar problem
- VGG16 - convolutional neural network trained at classifying objects from the famous dataset ImageNet

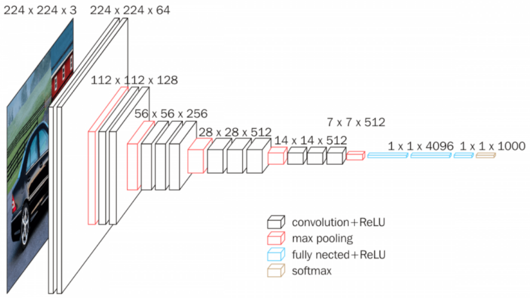

Figure: Architecture of VGG16 - deep convolutional neural network with 16 layers

Feature extraction via autoencoder

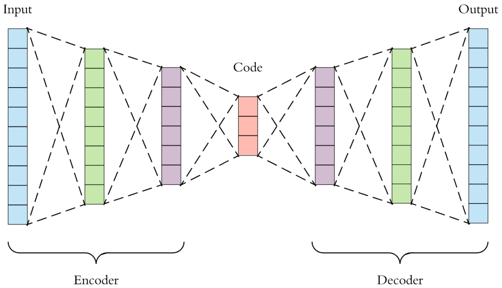

Figure: Architecture of an autoencoder

$$\phi : X \to F \tag{4}$$

$$\psi : F \to X \tag{5}$$

$$\phi, \psi = \underset{\phi,\psi}{\arg\min} \left\| X - (\psi \circ \phi) X \right\|^2 \tag{6}$$

K-means Clustering

By using K-means clustering, we can group the images and create synthetic labels. Images that have more in common that they do with the other images will have the same label.

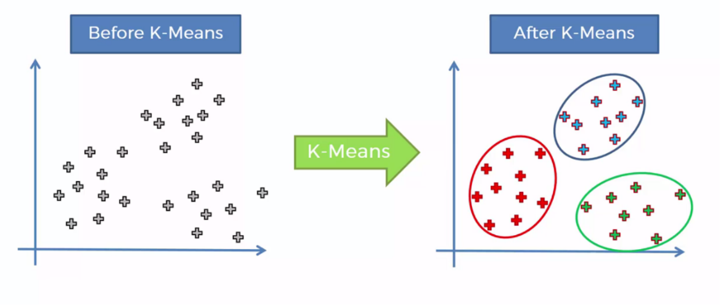

Figure: K-means clustering example

Image Processing Algorithms

- Blurring
- Complementing
- Color changing
- Denoising Autoencoder

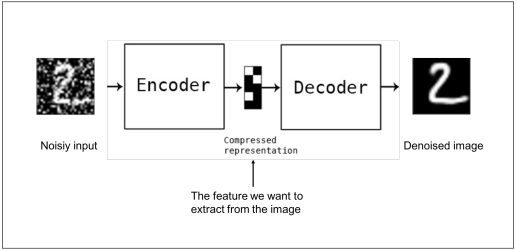

Figure: Architecture of a Denoising Autoencoder

Implementation

- Python 3
- Tensorflow 2.0
- NumPy
- OpenCV
- Matplotlib
- Scikit-learn
- PIL
- Django
- HTML
- CSS
- JavaScript
- Nvidia CUDA

Results

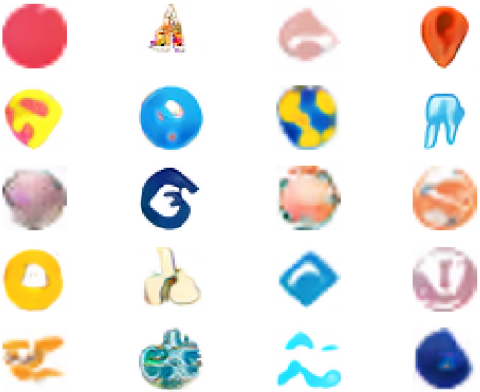

Figure: Sample of generated images

Conclusion & Future Work

Conclusion:

- Pleasurable results, although graphic designers still cannot be completely replaced
- Room for further improvements

Future plans:

- Train the neural networks for more iterations
- Implement new image processing algorithms
- Expand the research into other UI elements such as pictograms

Personal contribution

- Architecture + hyperparameter tuning
- Development of a system, capable of producing great results
- Clarification and addressing of the main challenges in UI elements synthesis

Demo

https://calliope.pythonanywhere.com

Thank you for your attention!
Any questions?