Generative Adversarial Network

Ashutosh Kakadiya, Harsh Patel, Nisarg Tike, Prerak Raja, Riddhesh Sanghvi

School of Engineering and Applied Science
Ahmedabad University
Group-6

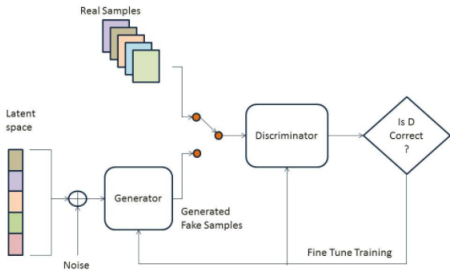
***Abstract*—Generative adversarial networks are a type of artificial intelligence algorithms used in unsupervised machine learning to generate new similar looking data. They were first introduced by Ian Goodfellow et al. in 2014. In this report, two approaches are discussed for generating new data. In the first approach, DCGAN (Deep Convolutional Generative Adversarial Network) is used and is tested on two datasets namely MNIST and celebA. In the second approach, two different neural networks are used, one for generating principal components and another for generating weight vectors of independent images.**

Keywords: Computer Vision, Convolutional Neural Network (CNN), Generative Adversarial Network (GAN)

I. Introduction

Generative Adversarial Network is a relatively new framework for generating new and similar data using existing data. GAN can be considered to be composed of two competing neural network models, generator and discriminator. The generator takes noise as input and generates samples. The discriminator receives samples from both the generator and the training data, and tries to distinguish between the two sources. The generator learns to produce more and more realistic samples while the discriminator learns to get better at distinguishing generated data from real data.

This report describes two approaches and their respective results. Section II describes the generative adversarial network and approaches. Section III is about the results. Section IV describes various applications of GAN. Finally Section V concludes the report.



II. Generative Adversarial Network

The generative network is a deconvolutional neural network which is taught to map from a latent space to a particular data distribution of interest, while the discriminative network is taught to distinguish between samples from the actual data distribution and generated samples. The discriminator is trained by feeding samples from the actual dataset and generated samples. In order to generate a sample, the generator is fed with a randomized input noise that is sampled from predefined latent space. Upon receiving the information that the discriminator was successful in distinguishing the generated image, the generator adjusts its parameters so that the training data and generated data becomes more difficult to be distinguished by the discriminator. The aim is to find parameter values that make generated data indistinguishable from the training data to the discriminator.

A. GAN using Deep Convolutional Neural Network

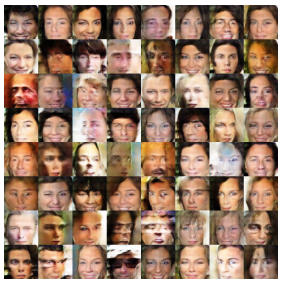
We have trained discriminator using MNIST and celebA database. The mnist dataset contains 60,000 input images of size ($28\times28\times1$) and in the celebA dataset we have taken first 1,00,000 images with crop size of ($108\times108\times3$) which are then fed to the neural network with a batch size of 64 images at a time and learning rate of 0.0002 for training. Then the trained DCGAN is initialized with random weights, so a random code plugged into the network would generate a completely random image. However, the network has been configured with the learned parameters of the input images, hence after each iteration (by trying to fool discriminator network) DCGAN learns to generate images similar to training images.

B. GAN using PPCA

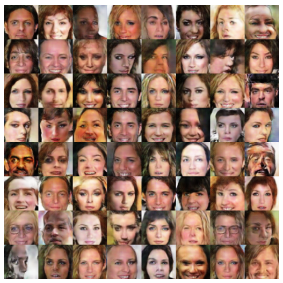
To create generative model using PPCA we have made some changes in the DCGAN model. Instead of giving input as image matrix, first the image matrix is given to PPCA module which generates the Principal Components of the image set, it is then given to the GAN model, where randomly selected 256 Principal Components are used to train the discriminator thus the dimensions of the training set at a time to discriminator would be($256\times48\times20$). We have selected generation batch size of 16 samples, hence an initial noise of dimension(16×100) is the input to generator. Generator generates new Principal Components of dimension($16\times48\times20$) which is fed to the discriminator. Discriminator then tells whether the generated Principal Components are fake or real and the feedback is given to generator.

III. Results

DCGAN Results
celebA Dataset

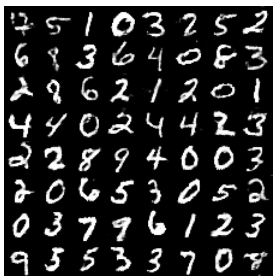


5 Epochs



10 Epochs

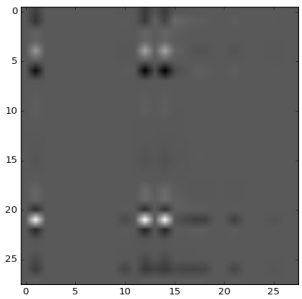
MNIST Dataset



5 Epochs

25 Epochs

PPCA Approach Results



450 Iterations MNIST Dataset

IV. Applications

Following are the applications of GANs.

- Generative Design (current research topic in AutoDesk)
- Image and matrix Completion
- A real time rendered, one of a kind virtual reality
- Image Generation
- High resolution image generation
- Image Inpainting
- Semantic Segmentation
- Video Generation
- Impressive text to image generation

V. Conclusion

Using Deep convolutional neural network we were able to create great results. The generated MNIST output was very near to an actual handwritten digit and the generated celebrity faces were very much real looking. Although in the PPCA approach we could not create satisfactory results. The method to verify the output of PPCA is also difficult.

Also, to train the neural network with features and diverse dataset is not possible using PPCA. In DCGAN the network learned the features of the images using convolutional kernels, stridings and other pixel based processing possible on the image which is not possible while using PPCA.

VI. Future Work

Generative Design: How people and computers can cocreate things that humans alone could never imagine.

Step by step process:

- Designer/Engineer inputs design goal and constraints, using generative design system (DCGAN) and humans enter specifics such as strength, weight, cost.

- Computer uses algorithms and generate thousands of design, running performance analysis each.

- Designer/Engineer studies options and modify constraints. Computer regenerates. Humans and computational AI together identify most relevant solution.

- Designer/Engineer fabricate prototype using 3D printing and returns to step 3 if needed.

References

[1] Goodfellow, I., Pouget, J., Mirza, M., Xu, B., Warde, D., Ojair, S., Courville, A. and Bengio, Y. (2014). Generative Adversarial Networks.

[2] Deshpande, A. (2016). Deep Learning Research Review Week [online] Adeshpande3.github.io. https://adeshpande3.github.io/adeshpande3.github.io/ Deep-Learning-Research-Review-Week-1-Generative-Adversarial-Nets

[3] GitHub. (2017). carpedm20/DCGAN-tensorflow [online] https://github.com/carpedm20/DCGAN-tensorflow
[4] Brownlee, J. (2017). Deep Learning with Python. 1st ed. Melbourne.

[5] TensorFlow. (2017). TensorFlow. [online] https://www.tensorflow.org/