Applied Machine Learning

Face Generation Using GANs

Group Presentation:
(GROUP 3)

--- **By Team Members:**

- Vihari Eyunni
- Shubham Kumar Singh
- Yaswanth Karri
- Prateesh Reddy Patlolla

LUDDY
SCHOOL OF INFORMATICS,
COMPUTING, AND ENGINEERING

What?

Proposed Project

Generative Adversarial Networks (GANs) are one of the most innovative ideas proposed in this decade. At its core, GANs are an unsupervised model for generating new elements from a set of similar elements.

- Today, we would like to provide a brief overview on our implementation of GANs for the purpose of generating human faces.
- In this proposed project, we chose a static dataset. Before we dive into the technical aspects for all those who are listening to the term GANs for the first time, let us provide a short introduction to the topic and provide relevance to its applications.

What?

Proposed Project

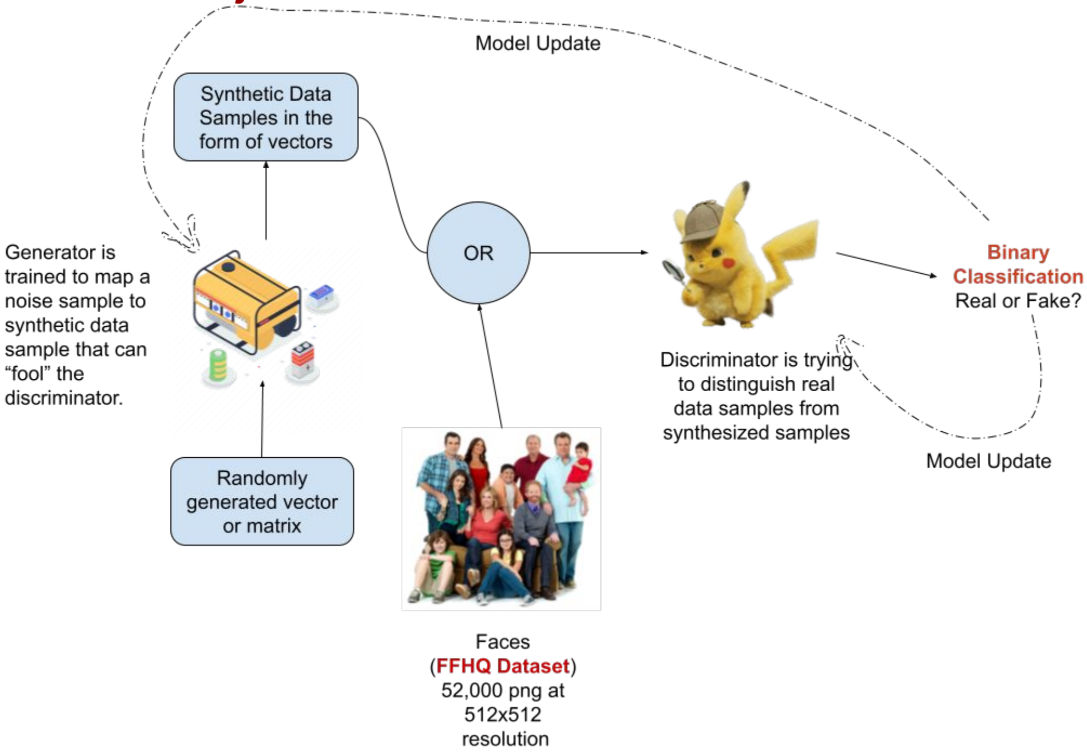
Why?

The purpose of this project:

- Adding only a little amount of noise to the original data can lead to wrongly classify the items. This has happened to many of the conventional neural nets. Sometimes, even the most sophisticated networks were fooled/tricked into misclassifying things.
- The funny thing is that the model's confidence in wrongly classifying the item is more than when it calculates the correct predictions. All this happens due to a small noise that we introduce into the data.
- This only leads to one conclusion that most ML models learn from a very small amount of data. This is bad because this leads to several problems including overfitting.
- Although the boundaries of separation between the different groups appear to be linear, they are actually made up of linearities, and even a slight shift in a point in the feature space will cause data to be misclassified.
- So, one of the most important reasons why we, as a team were interested in implementing GANs, is to address the above-mentioned shortage of data. Although this technique is now well implemented across multiple domains, there are still a lot of used cases to the most basic of GANs such as creating huge chunks of realistic-looking faces as data for face detection, object detection, and style transfer.

How?

- Architecture of this Project



Past Research Work

- And our future scope

- Goodfellow, Ian, et al. "Generative adversarial nets." Advances in neural information processing systems. 2014.
- Isola, Phillip, et al. "Image-to-image translation with conditional adversarial networks." Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.
- Salimans, Tim, et al. "Improved techniques for training GANs." Advances in neural information processing systems. 2016.

Although GANs evolved quickly, many results are stuck in academia. It is nice when models are made public as interactive tools. Many things can only be learned live, such as forcing algorithms to the extreme or playing with unusual choices.

We wish to deploy our much-refined model as an online platform to generate data for users. The main goal of this platform will be to generate new data for the user based on their requirements. We believe that this will have a major impact in accessing and acquiring more images as data which will help many projects and prospective research work.

Datasets

- Dataset used in this project

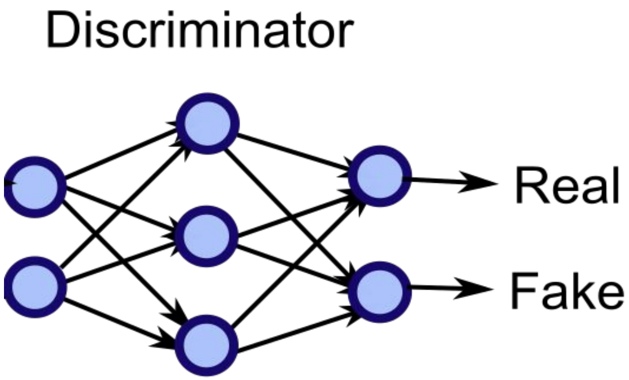
We checked on the following Datasets initially :

- CelebA Dataset
- Faces94 Dataset
- Flickr-Faces-HQ Dataset (FFHQ) Dataset

After checking the datasets and the implementations , we have decided to work on the Flickr-Faces-HQ Dataset (FFHQ) Dataset.
Flickr-Faces-HQ Dataset (FFHQ) is a dataset which contains 52k high-quality PNG images relating to human faces at 512x512 resolution .This dataset includes more variation than CELEBA-HQ dataset in terms of age, ethnicity and image background, and also has much better coverage of accessories such as eyeglasses, sunglasses, hats, etc. The images were crawled from Flickr and then automatically aligned and cropped.

Approach

- Discriminator



- ➔ Discriminator is a 4 layer strided with batch normalisation (except input layer) and LeakyRelu activations with sigmoid on the final layer thus technically making it a binary classifier.

- ➔ Discriminator Loss :

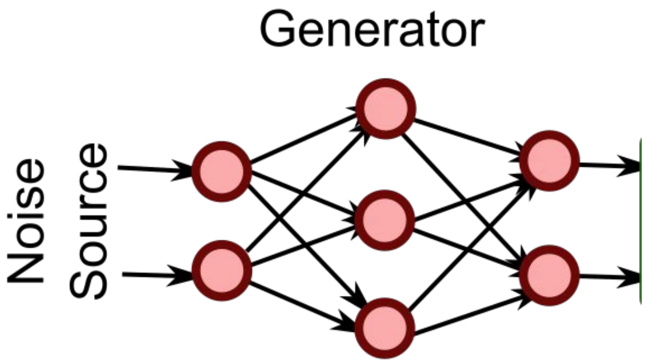
$$BCE(D(x), 1) + BCE(D(G(z)), 0)$$

- ➔ The approach followed here is min-max algorithm. As Discriminator wants to maximize log probability of predicting one(real) for D(x) and minimize log prob of predicting zero(fake) for G(z)

- ➔ Nash Equilibrium of the model is attained when model can't reduce loss without changing G(z) parameters.

Approach

- Generator



➔ Generator starts with input latent (random noise) and the loss is used to update the weights in such a way that it generates realistic images that fools discriminator.

➔ It is 4 layered upsampled strided with batch normalization and Relu activations with Tanh on the output layer (normalized images).

➔ Generator Loss:

`BCE(D(G(z)) , 1)`

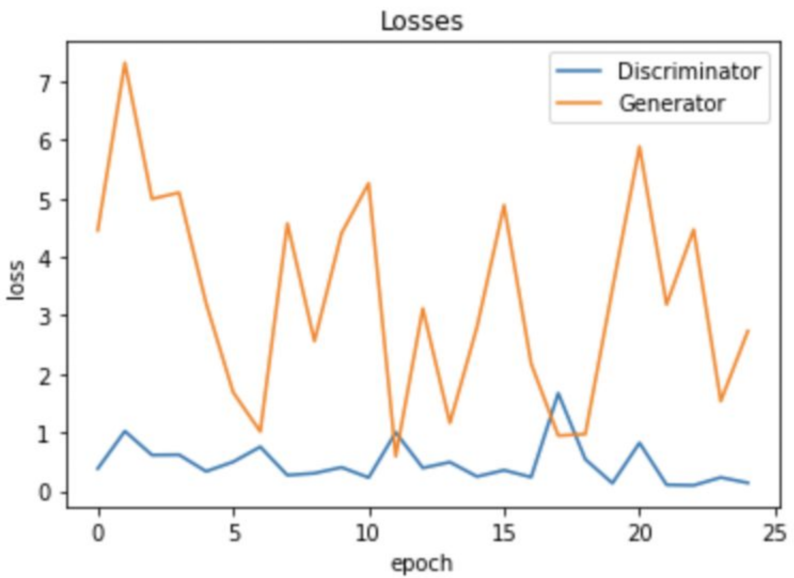
➔ The approach followed here is images output by generator should be assigned a high probability (perfect being 1) by discriminator.

Hyperparameter Settings

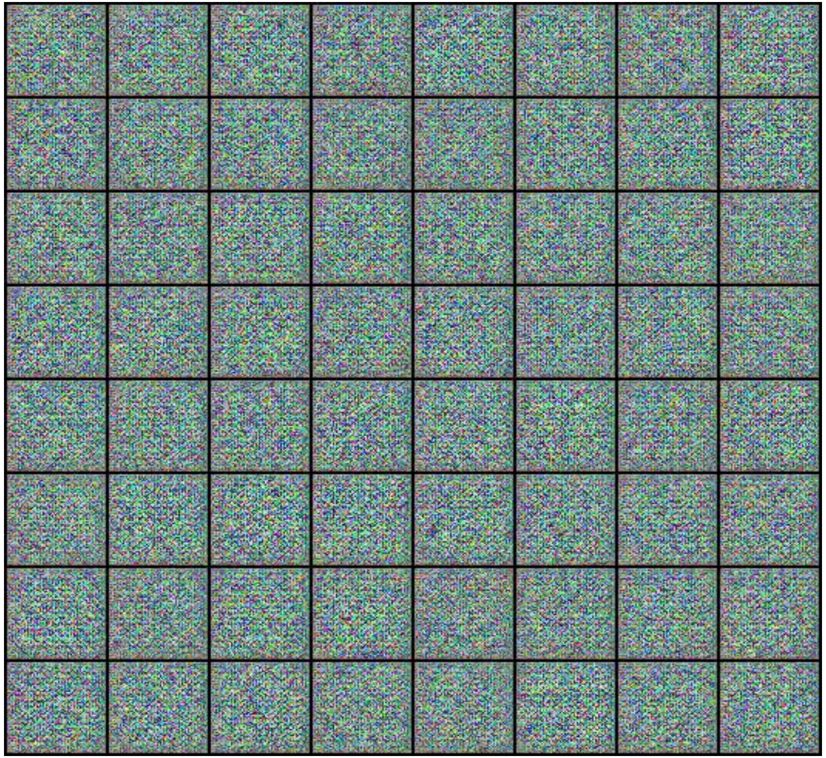
- → **Learning rate**: 0.0002
- → **Batch size**: 128
- → **Number of epochs**: 25
- → **Generator optimizer**: Adam optimizer
- → **Discriminator optimizer**: Adam optimizer
- → **Momentum**: (0.5, 0.999)
- → **# layers in Generator**: 4
- → **# layers in Discriminator**: 4
- → **Activation functions for Discriminator** : LeakyRelu function on all layers Sigmoid on output layer
- → **Activation functions for Generator** : Relu function on all layers Tanh on output layer
- → **Loss function**: Binary Cross-Entropy Loss

Facts & Figures

Epoch	loss_g	loss_d	real_score	fake_score
1/25	4.4531	0.3823	0.7548	0.4391
2/25	7.3169	1.0208	0.8865	0.5212
3/25	4.9899	0.6132	0.9353	0.3746
...	...	...	...	...
23/25	4.4676	0.0955	0.9483	0.0394
24/25	1.5339	0.2306	0.8536	0.0514
25/25	2.7309	0.1374	0.9131	0.0415



Our Generator in Action



Neural Style Transfer

- An interesting approach to increase the spectrum of variation in output

While working on the GANs we thought what if the user wanted a data that was in alignment with his requirements but wanted lot of variations in the dataset to improve the the training of the model. That's when NST's popped up. We took a single image of output from our GAN implementation and performed style transfer on it.

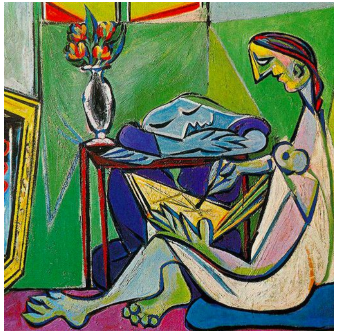
Implementation details:

We use a VGG network. It is an implementation of CNNs. Here, we use 0th,5th,10th,19th,28th,layers which correspond to to conv1_1, conv2_1, conv3_1, conv4_1, conv5_1 mentioned in NST paper i.e pretrained model vgg19.

We first, obtain convolution features in specifically chosen layers i.e Generated image features, Style image features and original image features.

Neural Style Transfer

An interesting approach to increase the spectrum of variation in output.



Final Thoughts

- Likes, Challenges, Learning Opportunities, Questions

Initial & Final Thoughts:

- For our final project we thought to work on a challenging problem, where we can implement our current skill set. Our goal was to gain even deeper knowledge of Machine Learning and Neural Networks.
- Seeing our results of completely new human's faces is quite extraordinary. Therefore, we are excited for the future aspects of this project.
- We believe that we can achieve even better results, if given more time and produce such realistic looking fake human images that are difficult to differentiate from real human images.

Final Thoughts

- Likes, Challenges, Learning Opportunities, Questions

Challenges :

- ❖ loss Stability
- ❖ Early Stopping
- ❖ Mode Collapse

Learning :

- ❖ Stick with your setting
- ❖ learning rate tuning
- ❖ Adding Noise to discriminator

In coming days, we aim to deploy our model such that any researcher in needing of that particular style of data could grab it. Our only question left unanswered is how will we reduce time of image generation after request.

Thank You!