**Generative AI and Applications: CSE-848 (Assignment-2)**

**Name: Shahana Ali**

1. **Generative Adversarial Network (GAN)?**

Generative Adversarial Network (GAN) [1] entails training two neural networks to compete against each other to create new data from a given training set. For instance, we can generate new data from original data. Data could be in any form, such as images, text, audio, or signal data. As the network suggests, it has two neural networks, and one network is responsible for generating new data from the given input data (and modifying it as much as possible). The other predicting network determines whether the generated data is fake or real. The training of these two networks persists until the prediction network can no longer distinguish fake from original. The basic network architecture of the GAN is provided in Figure 1.

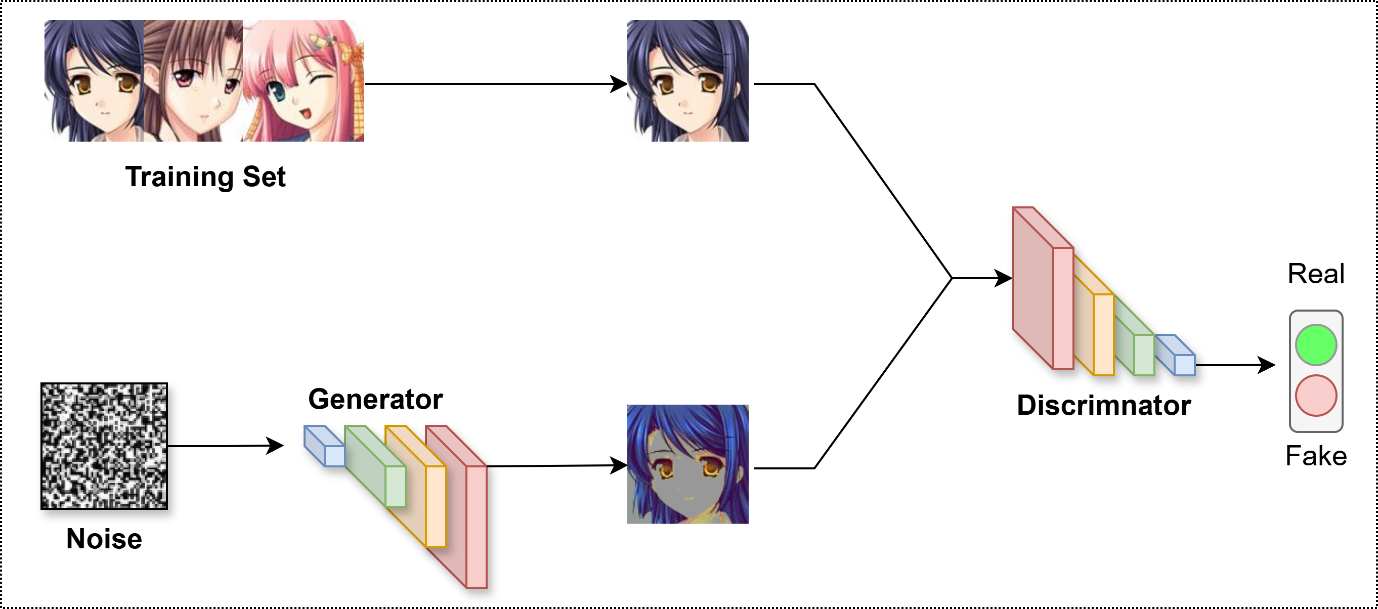


Fig.1. GAN Network.

1. **Choosing Dataset and Preprocessing**

I chose the Cartoon (Face Anime) Dataset [2] to train the GAN network. The dataset contains around 63000 images. However, due to computational and close assignment deadlines, images were removed from it, which made it around 12000 for training, which falls in the requirement range (5000-50000). In terms of preprocessing, the adopted dataset is resized to 64x64 and further normalized to range [-1, 1] for stable GAN training. The same preprocessing techniques as suggested in the assignment description. Finally, create a DataLoader, which prepares the dataset for efficient training.

1. **Implementation of Deep Convolutional GAN (DCGAN)**

DCGAN was implemented successfully according to the requirements mentioned in the assignment description. The DCGAN code was adopted and modified (according to our custom dataset) from the given Github repository [3].

* Used transposed convolutions at the Generator side
* Used ReLU activation and Tanh for output at the Discriminator side
* Used convolutions, batch normalization, and Leaky ReLU in the Training phase
* Uses Binary Cross-Entropy Loss
* Optimized with Adam optimizer
* Trains for 50 epochs
* Saved Training process logs into a text file
* Saves generated images at different epochs



Fig. 2. Generated images from different epochs

**Results and Discussions**

All the training logs depict the generator and discriminator loss, as the model is not that efficient. However, we need further training and validations with different hyperparameters and tuning. Due to the close assignment deadline and computational resources, I could not play with many parameters and training strategies. However, the GAN implementation enabled me to understand the overall intuition of the GAN and its applications while training and generating different synthetic images.

*Likewise, due to the close assignment deadline and limited computational resources, I was also not able to deploy it as a web interface and faced an issue. The potential problem was not saving the complete model directly. Thus, I need to modify and retrain my model to save it directly and as a complete deployable model. If I get more time and chance, I can re-modify and retrain my model after necessary changes, and then I could deploy it as a web interface.*

**GitHub**:

**References**

1. Goodfellow, Ian, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. "Generative adversarial nets." Advances in neural information processing systems 27 (2014).
2. https://www.kaggle.com/datasets/splcher/animefacedataset
3. https://github.com/eriklindernoren/PyTorch-GAN/tree/master