**Department of Information Engineering**

**Digital Forensics Course**

**Project 1:** **Generative Adversarial Networks**

**Supervisor: prof. Simone Milani**

**Name**

**Esraa Alnajjar 2009462**

**Marian Ashaq 2041388**

**1. Introduction**

Generative Adversarial Networks are unsupervised learning task in machine learning that are able to cluster and analyze unlabeled datasets, discover and learn the patterns in the input data in order to be able to generate or output new examples that could have been drawn from the original dataset.

GANs is a Machine learning model that is composed of two neural networks where each network tries to become more accurate than the other network with respect to predictions.

The generator and the discriminator are the two neural networks that make up a GAN. The generator represents convolutional neural network and the discriminator represents deconvolutional neural network. GANs begin to generate outputs with high quality to deceive discriminator and discriminator becomes better at finding out the data that has been created artificially by the generator.

Generator and discriminator have two different goals. The first (generator) aims to generate fake data to deceive the discriminator and maximize the overall loss function. The discriminator goal is to discriminate between real and fake data and to minimize the overall loss function.

**2. Method of working of GANs**

Firstly, decide on the desired output and gathering an initial dataset based on these parameters. Secondly, the data is randomized and input into the generator until it acquires basic accuracy in producing outputs. Thirdly, the images from the generator are fed into the discriminator. The discriminator provides a probability between 0 and 1 where (0 refers to fake data and 1 refers to real data).Finally, the values are manually checked for success and repeated until reaching the desired outcome.

**3. GAN Training Procedure**

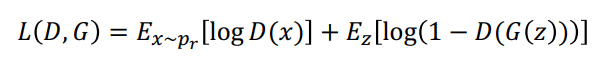
1. Train discriminator D for one or more epochs -while training D only discriminator loss is used -weights of D are updated through backpropagation 2. Generator trains for one or more epochs -sample random noise -generate fake samples -get discriminator output -compute loss -backpropagate through D and G to get gradients -use gradients to change generator weigh

3. Repeat steps 1 and 2

**4. GAN loss functions**

The loss functions in GAN is to reflect the distance between the distribution of the data generated by the GAN and the distribution of the real data. There are two common GAN loss functions which are Minmax loss and the Wasserstein loss.

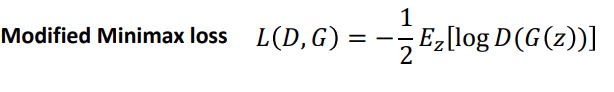
**Firstly the Minmax loss**



In this equation:

-D(x) represents the discriminator’s estimate of the probability that real data instance x is real -E(x) is the expected value over all real data instances -G(z) is the generator’s output when given noise z -D(G(z)) represents the discriminator’s estimate of the probability that a fake instance is real -Ez represents the expected value over all random inputs to the generator

The modified Minmax loss The need for of Modified Minmax loss over the Minmax loss appears because the above Minmax loss cause the GAN to get stuck in first stages of GAN training while the discriminator task is easy. Thus, modifying the generator loss in order to maximize log D(G(x))

****

**Secondly Wasserstein loss**

This loss function depends on a modification of the GAN scheme in which the discriminator does not actually classify instances, but for each instance it outputs a number. This number does not have to be less than one or greater than 0.In this case discriminator training just tries to make the output b for real instances bigger than that for fake instances.

Critic Loss**:** D(x) - D(G(z)) The discriminator tries to maximize the difference between its output on real instances and its output on fake instances.

Generator Loss**:** D(G(z)) The generator tries to maximize the discriminator's output for its fake instances.

In these functions:

-D(x) is the critic's output for a real instance -G(z) is the generator's output when given noise z -D(G(z)) is the critic's output for a fake instance

**5. GAN Problems**

1. Vanishing gradient problem Discriminator always guess; no possibility to learn directions for better parameters Solutions: modified Minmax, Wasserstein loss

2. Mode collapse Generator produces only a specially-plausible output Solutions: Wasserstein loss, unrolled GAN

3. Failure to converge Solutions: adding noise, penalizing discriminator weights

**6. GAN Evaluation**

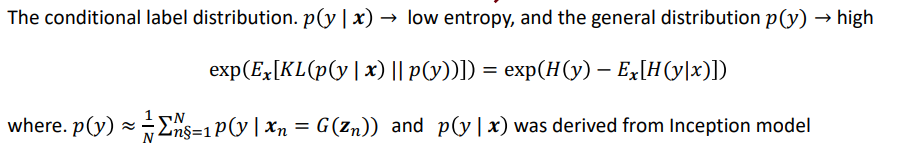
Measuring the performance of GAN could be done by both quantitative and qualitative methods. Qualitative measures are the types of measurement that are not numerical and sometimes involve human subjective evaluation using comparison while quantitative measures refer to the calculation of specific numerical scores that are used to evaluate the quality of generated images.

Firstly, qualitative GAN evaluation There are five qualitative techniques used for evaluating GAN generator models: Nearest Neighbors, Rapid Scene Categorization, Rating and Preference Judgment, Evaluating Mode Drop and Mode Collapse, Investigating and Visualizing the Internals of Networks. The nearest neighbor approach is useful to give context for evaluating how realistic the generated images happen to be. The Rapid Scene Categorization method images are presented to human judges for a very limited amount of time, such as a fraction of a second, and classified as real or fake. These aforementioned images are often presented in pairs and the human judge is asked which image they prefer in terms of being more realistic. Thus, a score or rating is determined based on the number of times a specific model generated images on such tournaments and variance in the judging can be reduced by averaging the ratings across multiple different human judges.

## Secondly, quantitative GAN evaluation There are Twenty-four quantitative techniques that are used for evaluating GAN generator models: Average Log-likelihood, Coverage Metric, Inception Score(IS),Modified Inception Score (m-IS),Mode Score, AM Score, Fréchet Inception Distance (FID),Maximum Mean Discrepancy (MMD),The Wasserstein Critic, Birthday Paradox Test, Classifier Two-sample Tests (C2ST),Classification Performance, Boundary Distortion, Number of Statistically-Different Bins (NDB),Image Retrieval Performance, Generative Adversarial Metric (GAM),Tournament Win Rate and Skill Rating, Normalized Relative Discriminative Score (NRDS),Adversarial Accuracy and Adversarial Divergence, Geometry Score, Reconstruction Error, Image Quality Measures (SSIM, PSNR and Sharpness Difference),Low-level Image Statistics, Precision, Recall and F1 Score. Two widely adopted metrics for evaluating generated images are the Inception Score and the Fréchet Inception Distance.

The inception score was introduced by [Tim Salimans](https://ai.google/research/people/106222), et al. in 2016 in their paper with title “[Improved Techniques for Training GANs](https://arxiv.org/abs/1606.03498)”. **Calculation of the inception score is done by using a pre-trained deep learning neural network model for image classification to classify the generated images**. Using the model, a large number of the generated images are classified and the probability of the image belonging to each class is predicted. Then, the probabilities are provided in the score to clarify how much each image is similar to a known class and how diverse the set of images are across the known classes. **The higher the inception score the better the quality of the generated images.**

Inception score Calculation



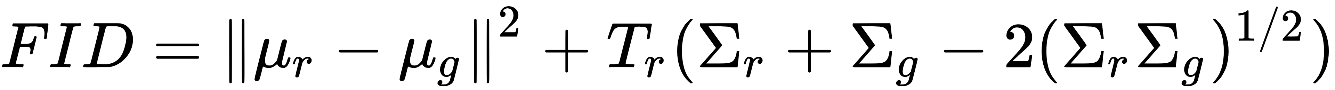
**fig(1) Inception score equation**

Inception score has some problems such as favors memory GAN, fails to detect some bad trapping model, favors detectable objects rather than realistic, favors generator, asymmetric, affected by image resolution.

The Fréchet Inception Distance, or FID, score was introduced by [Martin Heusel](https://www.linkedin.com/in/mheusel/), et al. in their paper with title “[GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium](https://arxiv.org/abs/1706.08500017)” .This score was proposed as an improvement and modification over the existing one(Inception Score).

FID is better in terms of discriminability, robustness and computational efficiency. In addition to this, FID is consistent with human judgments and is more robust to noise than Inception Score. Both the IS and the FID scores use the inception v3 model. The main goal of the coding layer of the model which is the last pooling layer prior to the output classification of images is used to capture computer vision specific features of an input image. **These activations are always calculated for a collection of real and generated images (fake)**.in addition to that, these activations are then summarized as a multivariate **Gaussian and the distance between these two distributions is calculated using FID**. **The lower FID score indicates more realistic images which is equivalent to the statistical properties of real images**.

FID Score Calculation



**fig(2) Inception score equation**

FID has some limits such as Limits: assumes Gaussian distribution, consistent with human judgment and more robust to noise than IS, FID detect mode dropping, FID is sensitive to some artifacts.

**7. GAN Applications**

GAN applications in different fields:

Generate Examples for Image Datasets

Generate Photographs of Human Faces

Generate Realistic Photographs

Generate Cartoon Characters

Image-to-Image Translation

Text-to-Image Translation

Semantic-Image-to-Photo Translation

Face Frontal View Generation

Generate New Human Poses

Photos to Emojis

Photograph Editing

Face Aging

Photo Blending

Super Resolution

Photo Inpainting

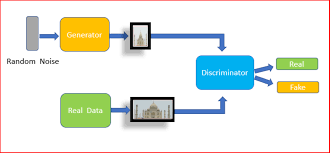
Clothing Translation

Video Prediction

3D Object Generation

**8. Differences between GAN Types**

**Simple GAN** (Generative Adversarial Network) is explained in the Introduction.



**fig(3) shows simple GAN model**

**The DCGAN** (Deep Convolutional Generative adversarial Network),it is composed mainly of convolutional layers without Max-pooling or fully connected layers.It uses transposed convolution and strides for down sampling and up sampling of 2D images.starting from Dense layer that takes this random noise as an input, and then upsample this seed several times until reaching the desired image with size  28x28 x1.

DCGAN steps few lines:

First replace all max pooling with convolutional stride. Then use transposed convolution for up sampling. After that eliminate the fully connected layers. Then use Batch normalization except the output layer for the generator and the input layer of the discriminator. Use ReLU in the generator model for activation except for the output which uses tanh.And finally use LeakyReLU in the discriminator model as activation function.

The **conditional generative adversarial network** (CGAN) is a type of GAN that involves the conditional generation of images by a generator model. And this condition is based on the class label. The architecture of the conditional GAN is composed of generator and discriminator model. As mentioned before that the generator is responsible for generating fake images that are very close to original image to deceive discriminator .The target of the discriminator is to distinguish between real and fake images.

The model is trained in adversarial manner. In other words, if the discriminator model shows improvement while discriminating between real and fake images, this will affect negatively the cost of the generator.

The main targets from using the class labels are to improve the GAN (stable, faster training and better quality images) and targeted image generation.

Conditional GAN is trained in a way that both the generator and the discriminator models are conditioned by the class label which means that when the trained generator model is used as a standalone model to generate images in the domain, these generated images should be of a given type, or class label.

***References***

1. Goodfellow, Ian; Pouget-Abadie, Jean; Mirza, Mehdi; Xu, Bing; Warde-Farley, David; Ozair, Sherjil; Courville, Aaron; Bengio, Yoshua (2014). [Generative Adversarial Nets](https://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf) (PDF). Proceedings of the International Conference on Neural Information Processing Systems (NIPS 2014). pp. 2672–2680.
2. Salimans, Tim; Goodfellow, Ian; Zaremba, Wojciech; Cheung, Vicki; Radford, Alec; Chen, Xi (2016). "Improved Techniques for Training GANs". [arXiv](https://en.wikipedia.org/wiki/ArXiv_(identifier)):[1606.03498](https://arxiv.org/abs/1606.03498) [[cs.LG](https://arxiv.org/archive/cs.LG)].
3. ^Isola, Phillip; Zhu, Jun-Yan; Zhou, Tinghui; Efros, Alexei (2017). ["Image-to-Image Translation with Conditional Adversarial Nets"](https://phillipi.github.io/pix2pix/). Computer Vision and Pattern Recognition.
4. Ho, Jonathon; Ermon, Stefano (2016). ["Generative Adversarial Imitation Learning"](http://papers.nips.cc/paper/6391-generative-adversarial-imitation-learning). Advances in Neural Information Processing Systems. 29: 4565–4573. [arXiv](https://en.wikipedia.org/wiki/ArXiv_(identifier)):[1606.03476](https://arxiv.org/abs/1606.03476). [Bibcode](https://en.wikipedia.org/wiki/Bibcode_(identifier)):[2016arXiv160603476H](https://ui.adsabs.harvard.edu/abs/2016arXiv160603476H).
5. Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434, 2015.
6. Ian J Goodfellow. On distinguishability criteria for estimating generative models. arXiv preprint arXiv:1412.6515, 2014.
7. Donggeun Yoo, Namil Kim, Sunggyun Park, Anthony S Paek, and In So Kweon. Pixel-level domain transfer. arXiv preprint arXiv:1603.07442, 2016.
8. Arthur Gretton, Olivier Bousquet, Alex Smola, and Bernhard Scholkopf. Measuring statistical depen- ¨ dence with hilbert-schmidt norms. In Algorithmic learning theory, pages 63–77. Springer, 2005.
9. Kenji Fukumizu, Arthur Gretton, Xiaohai Sun, and Bernhard Scholkopf. Kernel measures of conditional ¨ dependence. In NIPS, volume 20, pages 489–496, 2007.