Face Generation with Generative Adversarial Networks

by

Eduardo Jimenez, Kevin Mai, Ratul Nandy & Cole O'Connell

University of San Diego

Professor Addison

15 August, 2022

# Abstract

Modern neural network architectures are producing models capable of generating and altering images to such a degree that they are largely indiscernible from genuine images. These types of network are called Generative adversarial networks, otherwise known as GANs. GANs use a dueling network architecture that introduces a form of reinforcement learning without requiring a human intervention within the programmed loop. Using a GAN architecture and a dataset consisting of many pictures such as ten thousand unique celebrities, a model was trained to generate realistic images of faces. The generator's latent space is then leveraged to interpolate intermediary faces between two generated faces, and perform vector arithmetic to combine features. The output can transform a face with no smile to smiling.

*Keywords:* Generative adversarial network, Stochastic gradient descent,

# Face Generation with Generative Adversarial Networks

## Introduction

Generative adversarial networks have proved useful in a plethora of tasks, but have shown immediate upside in the content generation space. Perhaps, the most well-known image generation model is the StyleGAN, which came from NVIDIA's research team and was the first to focus on utilizing the latent space in the vector representation of (Karras et al., 2019). Other advanced variations of this technique includes Cycle GAN, Pix to Pix image translation, unpaired image to image translation etc (Brownlee, 2020).

Image generation and alteration has a tremendous research potential for the entertainment industry. This technique has been depicted as Deepfakes where it is used to generate background actors for films, realistic contents for games, anime generation, and within fashion industries. This makes it an attractive tool for creators as it reduces the time spent on set and refocuses funds to the visual content team. As well as, the research in GAN can help provide solutions across malicious usage of content generation applications.

# 

# Scope

The focus is to train GANs where a Generator will generate fake synthetically produced human pictures. The following steps depict exploring the latent space for linear vector interpolation between two random points and use those points as seeds to generate twenty images to foresee the change in features. Finally, the ten latent vectors of images of neutral man will be captured to find the mean and leverage the mean vector as a seed to generate an image of a new neutral man. The same experiment is repeated for neutral women with a generation of smiling women. Next, vector arithmetic is performed where it subtracts the latent vector of the neutral women from the neutral man and adds the vector of the smiling women. The resultant vector will be used as a seed to generate an image of a smiling man.

# Data

## Exploratory Analysis:

The CelebFaces Attributes (CelebA) Dataset was used to train the model. This dataset includes over 200,000 images of over 10,000 unique individuals, with each image annotated with binary attribute tags for 40 features. The images cover a variety of poses, accessories, and backgrounds. The dataset is not balanced as far as ethnic makeup or other diversity factors, it is simply ~10,000 unique "celebrities".

Link - <https://www.kaggle.com/datasets/jessicali9530/celeba-dataset>

## Data Preprocessing:

Due to the limitation in compute capacity, the dataset was reduced from 200K to around 30K images. With an incredible amount of images in the dataset, the data needs to be screened for images that do not have a presentable face. In other words, the human faces need to be detected and images cropped to reduce noise. One of the best ways to perform this quickly is to implement a Haar cascade detector. OpenCV library uses a Haar cascade detector or the Viola-Jones algorithm, arguably the most common computer vision algorithm. The detector uses a sliding window with a decision tree to determine face or not and leverages image pyramid technique which downsampled images with neighboring pixels (Google, 2012). As the algorithm parses through the tree, it uses the Haar filters for facial features.

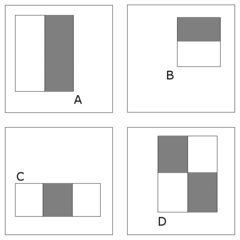
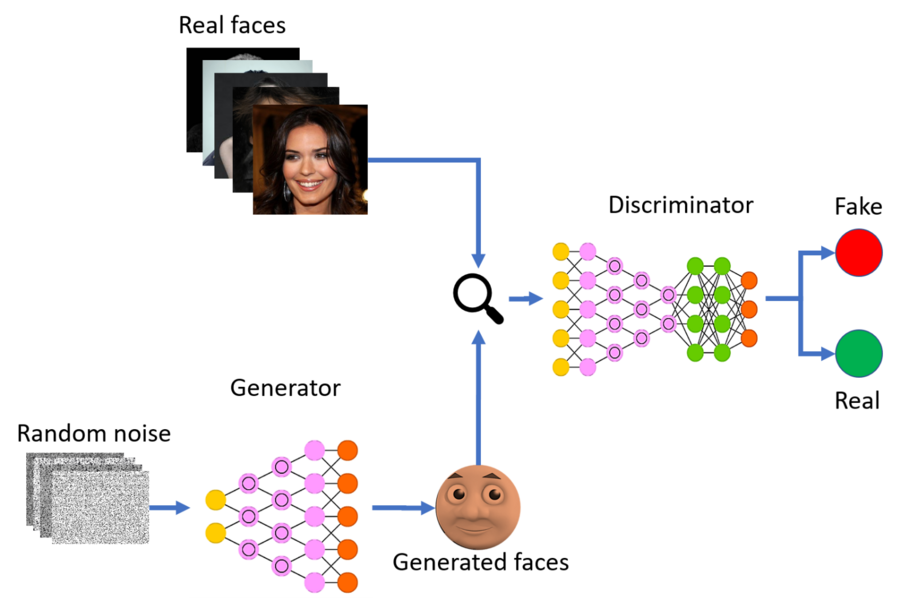


Figure 1. A, B, and C are subregions of the Haar features called edge and line features while D contains four subregions (Google 2012).

Haar Cascade uses Adaboost or Adaptive boosting which learns from weak learners and uses forest stumps, meaning that the first learner it gathers info from influences the next decisions within the tree. With the cascade classifier function, detectMultiscale function is used in combination with cv2.cvtColor() and cv2.COLOR\_BGR2GRAY for grayscale image conversion. Within the detectMultiscale, two parameters were adjusted, scaling factor and nearest neighbors. The scaling factor when adjusted to 1.3, the resize step will reduce up to 30% to find a face. While, the nearest neighbors or minNeighbors increase from 3 to 5 is to decrease the amount of false-positives. After performing this activity, the new dataset was around 20K which was resized to 32\*32 size.

## 

## Solution Approach



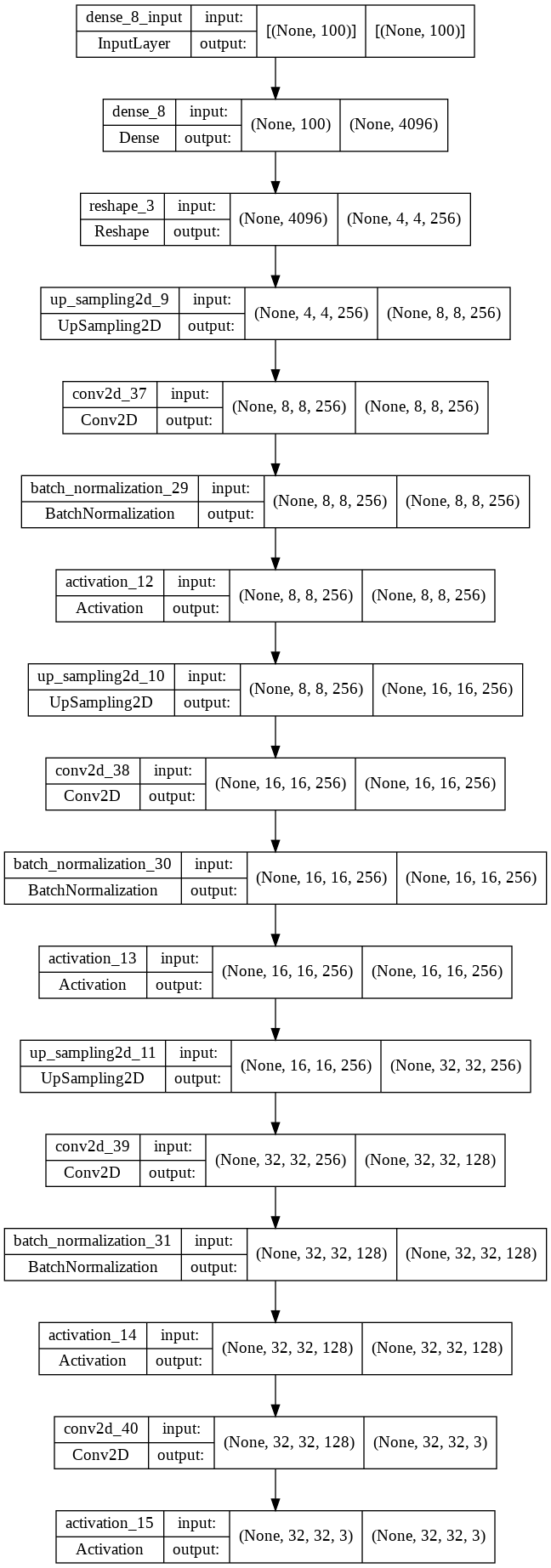
Source : [https://miro.medium.com/max/1400/1\*TKr1dtcNgJCA8uYY1OhmSg.png](https://miro.medium.com/max/1400/1*TKr1dtcNgJCA8uYY1OhmSg.png)

## 

## 

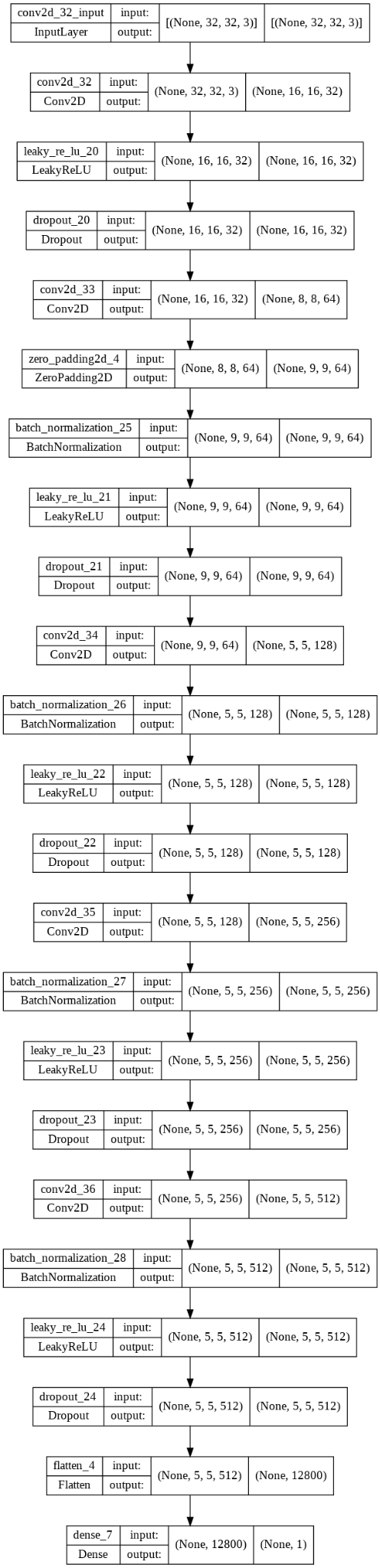
## As part of the solution, two neural networks were designed, a generator and discriminator. A random vector of size 100 is fed to the generator to create output. These outputs are fed to the discriminator as fake and at the same time real images from the dataset are also fed to the discriminator as real images. The discriminator is trained on these real and fake images. Once the discriminator is trained and the weights of the discriminator are kept constant, the generator is trained to fool the discriminator. This process is repeated for multiple epochs and in the process both the generator and the discriminator get better at its job. One of the important prerequisites is that weights of the discriminator are kept as constant when training the generator and vice versa. Once a desired result is achieved, the generator is used to produce fake images.

## Generator Architecture:



The first layer of the Generator is a dense layer of 4\*4\*256 units with a relu activation function. This is followed by a series of three layers of upsampling, convolution2D and batch normalization. All the layers have relu as activation function. The last layer is convolution layer with tanh function function which produces as 32\*32 image.

Discriminator Architecture:



The first layer of the Discriminator is a convolution2D layer of 32 units with kernel size 3 and strides as 2. This is followed by a LeakyRelu layer. The next layer starts with a dropout layer followed by a convolution layer, zero padding, batch normalization and LeakyRelu. The next three layers consist of dropout, convolution2D, zero padding, batch normalization and LeakyRelu. The last layer consists of a dropout layer followed by a flatten and dense layer. The dense layer has a sigmoid activation function as this is a binary classifier.

The models are trained using Adam optimizer and Cross Entropy is used to compute loss function.

**Tools**

IDE : Google Colab Pro

Storage : Google Drive

Libraries & Packages : PIL, MatplotLib, OS, Keras, Tensorflow, Numpy, CV2.

These packages were selected as they offer powerful APIs which were used for data preprocessing, model building, training and evaluation.

# 

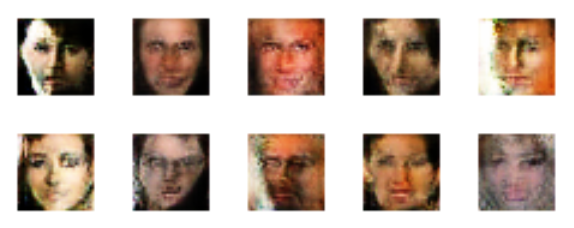
# Training

GANs are often difficult to train because of their competitive nature. In this particular task the discriminator learns quickly since its only task is to identify whether an image belongs to the dataset. This task is a relatively simple task compared to that of the generator, as is often the case with GANs where there are large datasets. This imbalance in model improvement prevented many of the early generator models from finding adequate solutions, which required that we tune the hyperparameters and decrease optimization weights for the discriminator. Still, problems would continue to arise late in training, where the generator output would begin to deteriorate since it didn't have enough positive reinforcement to learn from. To combat this, we let the generator train with multiple less-skilled discriminators. Training with the resized 128 pixel images took approximately 14 hours.

# Results

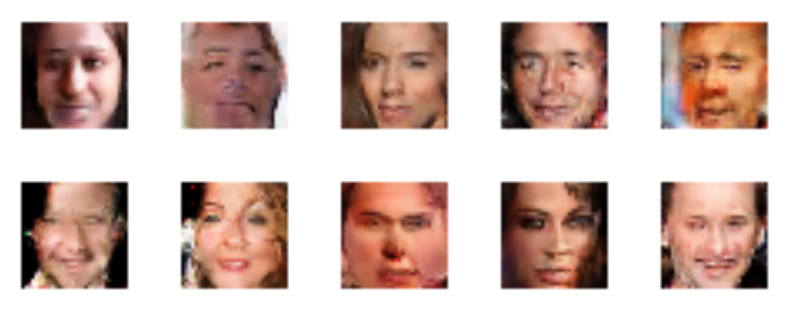
# Generator Examples

In early training, the model struggled with some of the more granular features like the eyes and facial contours. There's also very significant discoloration in at least part of the image in each attempt.



*Figure X: Generator results after 500 epochs*

Further training yielded much better results with much less discoloration and a more accurate map of the relevant features. Even in cases where the image is clearly generated, all the facial features are included in the image and placed in the appropriate positions. The model simply struggles to stitch the features together in a believable manner, which may be resolved by further training or perhaps training with higher resolution images.



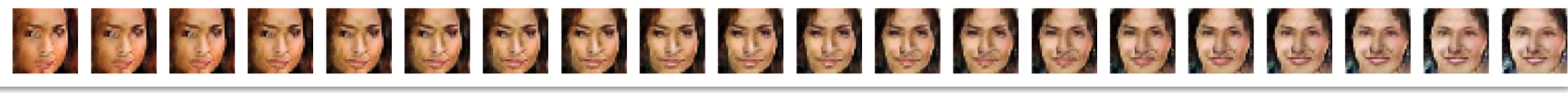
*Figure X: Generator results after 10,000 epochs*

When comparing the dataset examples with generated images that managed to fool the discriminator, it's clear the model is at least *occasionally* producing believable outputs. These images include detailed features and facial contours. Still, additional training is needed to accurately match the resolution of the input data.

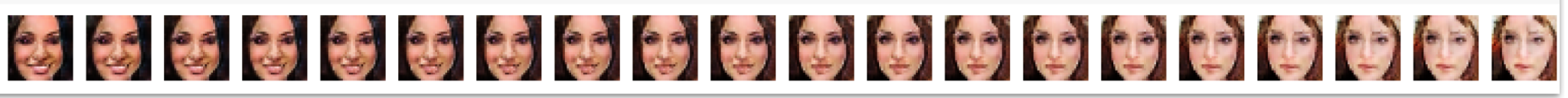


# Vector Interpolation :

Two random vectors were selected in the latent space and a linear interpolation was done with 20 points. Each point was used as seeds to generate new images. Here, we can see that in each example one of the features is gradually changing .







*Figure X: Latent vector interpolation examples*

Vector Arithmetic :

We generated 100 random images with the model. We selected 10 images of neutral men and extracted the corresponding latent vector. We calculated the mean of these vectors and used the mean as a seed to generate an image. The mean vector also generated an image of a neutral man. We carried out the same experiment for neutral women and smiling women and got similar results.

Seed generating ‘Neutral Man’ :



Seed generating ‘Neutral Woman’ :



Seed generating ‘Smiling Woman’ :



We then performed vector arithmetic, using the corresponding vectors and used the results as seed to generate an image.

Neutral Man - Neutral Woman + Smiling Woman = Smiling Man



# Discussion

The model we created lived up to its expectations given the processing power we had to work with. The real images we used to train the discriminator were heavily pixelated due to processing limitations, thus the generated pictures also came out pixelated. However, facial features from the fake images that were generated were easily distinguishable and tied in nicely together. This achievement was crucial because performing vector arithmetic on the generated images would have been impossible without having distinguished facial features. The result of the vector arithmetic was also shown to be successful, as the expectation was that the output showed a smiling man.

## 

## Improving the Model

After poor results early on, we implemented a face-finding algorithm to identify regions of interest in each image so that we could eliminate clothing and other background objects from the algorithm input. Results were also improved by training the final generator model against multiple discriminators. Due to the constraint in the storage and compute resource, the images had to be cropped to 32\*32 size. Images with larger pixel resolution will yield better looking results. Also, with more compute power and time, the model could be trained with more datasets to combat biases and trained with more epochs to further refine the quality of the output images.

## Ethical Considerations

It is important to discuss ethical concerns surrounding the deployment of AI models in any field, but it is of particular concern when working with face data with the expressed purpose of creating realistic transformations. Realistic transformations can be used to depict presidential or important figures making false speeches or appearances in areas that would otherwise be dangerous. Further research within this field may develop into harmful use-cases. However, researchers have found that a simple solution to discern deep fakes from reals is asking the user to turn sideways (Macaulay, 2022). Once the user turns sideways, there is a visual problem with the face manipulation. Therefore, it is imperative to uphold ethical standards to research and develop AI to make preventative tools.

A second ethical concern with generating faces for media would be if the model associated a race with a feature, incorrectly. For the CelebA dataset for example, many celebrities have the extra income to spend on luxuries such as jewelry, tattoos, and facial feature alterations. This may be a trend that celebrities may partake in, so it doesn’t mean that it has to be associated with any particular race. To remedy this, data with unwanted features can either be removed or alternate data can be introduced.

Another concern for bias in the data can be an unbalanced representation of races. This can cause a similar issue that speech to text had in its early stages, which is lower quality performance for certain accents. If the use-case of the model was avatar generation, then the model should output similar results amongst races. If there is in fact a bias, then it means that the data was likely trained with an unequal number of images for the underrepresented race. Bias concerns can be found in early stages of development and can be shown through extensive exploratory data analysis. The fix would be to balance out the training data with additional data.

## Further Study

The final generated model successfully generated fake images and altered features on the fake images such as gender and facial expressions. A natural extension of this work would be to work with video data and maintain facial alterations throughout entire video scenes. This may be a problem where it is particularly difficult to prevent the skill gap between the discriminator and generator during training, as maintaining the facial features in motion would be a much more challenging task for the generator. It would perhaps have the best likelihood of success with many very short clips, say around 1 second. Before commercializing any similar model, an audit should be performed to ensure accurate representation across various ethnic and social groups, which may require another round of annotations, followed by controlled testing to compare model performance across each group. In the case the model underperforms for any group, it may be necessary to revisit the data preprocessing steps or provide supplemental annotated data.

# 

# References

Brownlee, J. (2020, September 01). How to develop a cyclegan for image-to-image translation

with keras. Retrieved August 15, 2022, https://machinelearningmastery.com/cyclegan-tutorial-with-keras/

Géron Aurélien. (2022). Hands-on machine learning with scikit-learn, Keras, and tensorflow

concepts, tools, and techniques to build Intelligent Systems. O'Reilly.

Karras, T., Laine, S., & Aila, T. (2019). A style-based generator architecture for generative

Adversarial Networks. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). https://doi.org/10.1109/cvpr.2019.00453

Lienhart, Rainer. (2000). Stump-based 24x24 discrete(?) adaboost frontal face detector. https://raw.githubusercontent.com/opencv/opencv/4.x/data/haarcascades\_cuda/haarcascade\_frontalface\_default.xml

Google. (2012). Viola-Jones Face Detection. https://sites.google.com/site/5kk73gpu2012/assignment/viola-jones-face-detection#TOC-Image-Pyramid

Ioffe, S. and Szegedy, C. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. https://arxiv.org/pdf/1502.03167.pdf

Macaulay, Thomas. (2022). Want to spot a deepfake video caller? Ask the suspect to turn sideways. https://thenextweb.com/news/to-detect-deepfake-video-call-ask-suspect-to-turn-sideways

Maistre, Ray Le. (2022). How many hyperscale data centres does the world need hundreds more it seems. https://www.telecomtv.com/content/digital-platforms-services/how-many-hyperscale-data-centres-does-the-world-need-hundreds-more-it-seems-44015/

Aggarwal, A., Mittal, M., Battineni, G. (2021). Generative adversarial network: An overview of theory and applications. https://www.sciencedirect.com/science/article/pii/S2667096820300045

Brouton Lab. (2020). Three Cool Applications of GANs in eCommerce. https://broutonlab.com/blog/applications-of-gans-in-ecommerce

O’Brien, Chris. (2020). How Pixar uses AI and GANs to create high-resolution content. https://venturebeat.com/business/how-pixar-uses-ai-and-gans-to-create-high-resolution-content/

Guptha M, N., Guruprasad, Y. K., Teekaraman, Y., Kuppusamy, R., & Thelkar, A. R. (2022). Generative Adversarial Networks for Unmanned Aerial Vehicle Object Detection with Fusion Technology. Journal of Advanced Transportation, 1–13. https://doi-org/10.1155/2022/7111248.

Rehman, Junaid. (2022) Advantages and disadvantages of generative adversarial networks. https://www.itrelease.com/2020/06/advantages-and-disadvantages-of-generative-adversarial-networks-gan/

Yasha Jain, Y. (2020). GAN + Latent Space. https://medium.com/@jain.yasha/gan-latent-space-1b32cd34cfda

Viola, P. and Jones, M. (2001). Rapid Object Detection using a Boosted Cascade of Simple Features. https://www.cs.cmu.edu/~efros/courses/LBMV07/Papers/viola-cvpr-01.pdf

Sainburg, T.(2018). Generative Adversarial Interpolative Autoencoding(GAIA) https://timsainburg.com/gaia.html

Li, Z., & Luo, Y. (2017). Generate Identity - Preserving faces by generative adversarial networks.

Shen, Y., Gu, J,. Tang, X., & Zhou, B. (2019). Interpreting the latent space of GANs for semantic face editing

Behera, Girija Shankar. (2020). Face Detection with Haar Cascade. https://towardsdatascience.com/face-detection-with-haar-cascade-727f68dafd08