Findings/Analysis of QGAN on CIFAR-10

Seidenberg School of Computer Science and Information Systems
Pace University, New York City, New York, USA

Jash Shah
Jersey City, New Jersey
Js92482n@pace.edu

Understanding CIFAR-10.

The CIFAR-10 dataset contains 60,000 32x32 color pictures, grouped into ten classes of 6,000 images each. These classes are a balanced combination of animals (e.g., bird, cat, deer, dog, frog, and horse) and vehicles (e.g., airplane, automobile, ship, and truck), providing a wide range of options for generative modeling problems.

1. Introduction

Generative Adversarial Networks (GANs) have emerged as a game changer in the field of artificial intelligence, notably in synthetic picture synthesis. Their innovative adversarial training technique, in which a generator and a discriminator network battle, allows for the development of very realistic pictures. This work conducts a rigorous comparison of three advanced GAN architectures - Deep Convolutional GAN (DCGAN), Conditional GAN (CGAN), and Quantum GAN (QGAN) - using the CIFAR-10 dataset, a standard benchmark in machine learning for image-related tasks.

The CIFAR-10 dataset, with its broad collection of pictures covering ten unique classes, provides a hard yet fascinating canvas for studying GAN capabilities. Our research not only investigates the architectural complexities and training dynamics of DCGAN, CGAN, and QGAN, but also provides insight on their potential to produce pictures that blur the barrier between synthetic and genuine. Each model contributes a unique set of inventions and solutions to the challenge of picture generation.
DCGAN transforms previous GANs by including convolutional layers, considerably increasing the model's capability to generate detailed pictures that establish new standards for realism and diversity.

CGAN extends the GAN architecture to include class labels in the generation process, enabling exact picture production across many categories. This improvement not only increases the variety of GANs, but also their ability to create different, category-specific pictures.

QGAN, at the cutting edge of GAN innovation, uses quantum computing concepts to increase the efficiency and quality of produced pictures. This technique aims to address some of the scalability and complexity challenges associated with standard GANs, with the promise of faster and more authentic picture production.

2. Distribution and Diversity

Our exploratory data analysis (EDA) began with a thorough evaluation of the class distribution in the CIFAR-10 dataset, which revealed an equitable distribution across all ten classes. This balanced representation is critical for limiting any model bias by ensuring that our generative models are exposed to and learn from the whole range of categories in the dataset. Refer Fig 2.2. Running the example loads the dataset and prints the shape of the input and output components of the train and test splits of images. We can see that there are 50K examples in the training set and 10K in the test set and that each image is a square of 32 by 32 pixels.

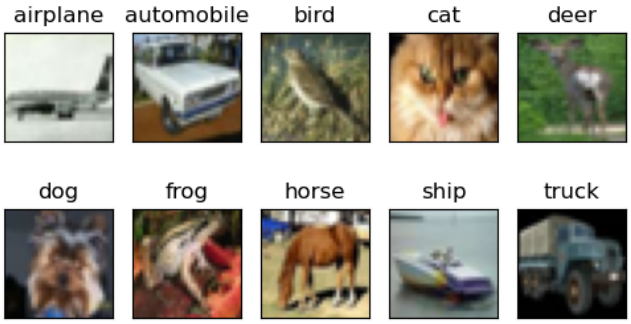

Figure 2.1 The CIfar-10 Classes

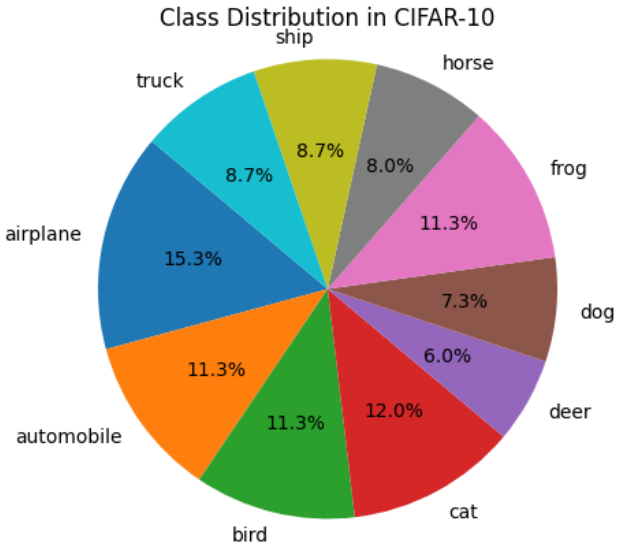

Figure 2.2 Class Distribution

The investigation delves further into the diversity and intricacy of each class. We found considerable differences in image backgrounds, ranging from basic, monochrome images to sophisticated, multi-element compositions. Such variance in backdrops presents a unique challenge for generative models, which must learn to effectively recreate this variability in order to produce images that are not only realistic in their depiction of the subject but also in their environmental context.

Our EDA began with a distribution analysis, which confirmed an equal representation of classes, an important component in preventing model bias. The variety within each class was also seen, with significant differences in picture backgrounds, orientations, and topic scales. This variety highlights the problem for GANs in generating not just recognized but also different and realistic pictures for each class.

3. Pixel Intensity and Image Quality

Our exploratory voyage through the CIFAR-10 dataset revealed the intricacy and diversity of pixel intensity distributions across its pictures. Using Python packages such as Matplotlib, Seaborn, and TensorFlow, we conducted a multifaceted study to discover the dataset's rich color variety and its implications for synthetic image synthesis.

The analysis found a wide range of pixel values throughout the RGB channels, indicating the dataset's rich color diversity. This variation is critical for developing generative models capable of producing a diverse range of realistic visuals. For example, certain classes exhibited a preference for brighter or darker intensity distributions, indicating significant differences in visual luminance and contrast across categories. Refer Fig 3.1.

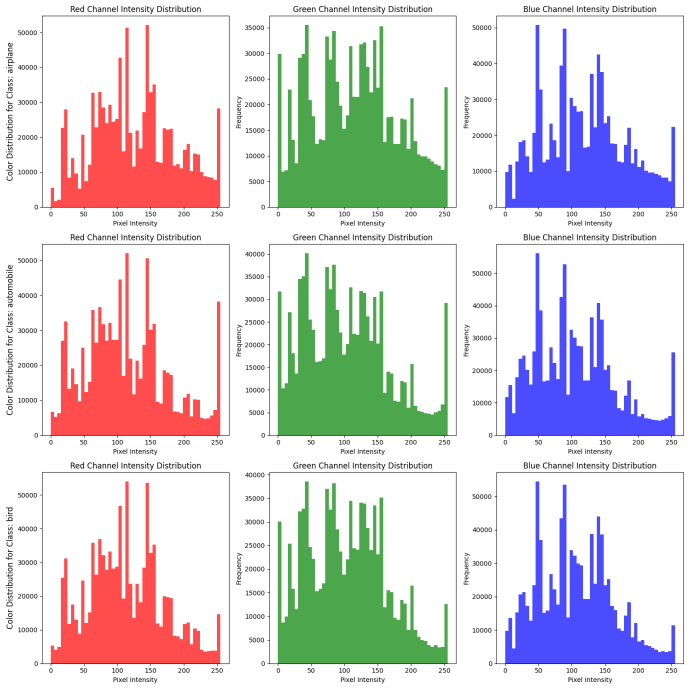

Figure 3.1 RGB Channels

The CIFAR-10 dataset's pixel intensity distributions for the RGB channels of the car, bird, and airplane

classes are shown in the histograms. In aircraft, the green and blue channels indicate a balanced presence of these colors with a propensity towards mid-range intensities, while the red channel displays a broad distribution with a tilt towards higher intensities, indicating of bright reds. The car class, on the other hand, has a bimodal distribution in the blue channel that indicates fluctuations in the presence of blue hues and a prominent peak in the mid-range for both the red and green channels, suggesting a moderate intensity preponderance of these colors.

In the case of the bird class, the blue distribution has more mid-to-high intensity values, suggesting the prominence of sky or water backgrounds that are typically associated with this class; the green shows a more even spread; and there is a noticeable peak at lower red intensities, suggesting images with subtler reds.

4. Intensity Distribution

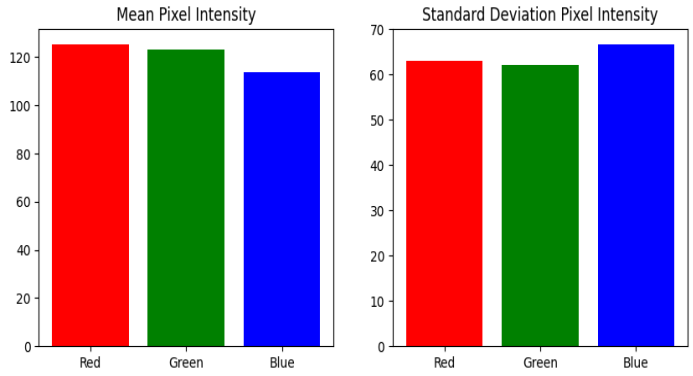

Figure 4.1 Mean and STD intensity

The mean pixel intensity for each color channel is displayed in the first bar chart, which shows a balanced brightness with a small leaning towards the red spectrum. This suggests that, on average, the photos in the collection have a warmer tone. The standard deviation graphic indicates a consistent distribution of color intensities within the dataset by showing a reasonably uniform variability throughout the red, green, and blue channels. Generative Adversarial Networks (GANs) benefit from this kind of color variability distribution since it offers a wide range of tones from which the network may learn, guaranteeing the production of realistic and varied pictures. Refer Fig 4.1

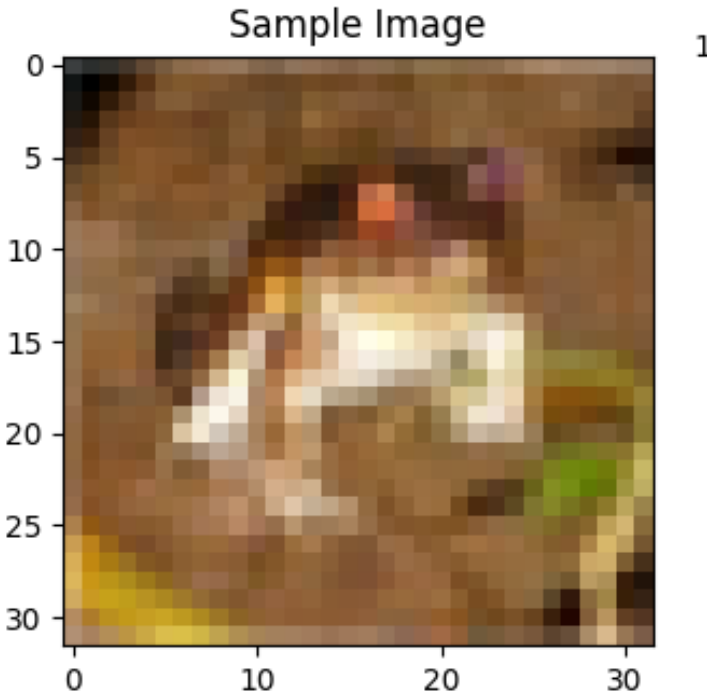

Figure 4.2 Sample Image

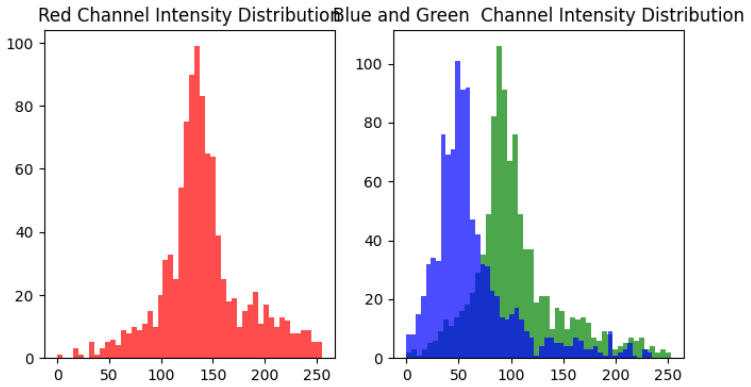

Figure 4.3 RGB channel for Sample Image

Red Channel Intensity Distribution: This image's red hue is well-balanced, neither too bright nor too dark, according to the red channel (middle) histogram, which displays a bell-shaped distribution centered on the mid-range intensity values. Refer 4.3.

Blue and Green Channel Intensity Distribution: The frequency of pixel intensity values ranging from 0 to

255 is displayed in the combined histogram for the blue (right) and green (left) channels. It suggests that there are a good number of both hues in this image, with a range of intensities that encompasses both low and high values. While the blue channel has a minor bias towards higher intensity values, the green channel indicates a peak in the mid-range. Refer Fig 4.2.

5. Augmentation

Data augmentation is a strategy employed to artificially expand the diversity of a dataset without actually collecting new data. This is achieved by applying various transformations to the original images that simulate the variations that could naturally occur. These transformations can include geometric modifications like rotation, translation (shifts in height or width), zooming, flipping (horizontal or vertical), and more. They may also extend to color augmentations like adjusting brightness, contrast, or saturation to mimic different lighting conditions. Refer Fig 5.1

Rotation and Flipping: A few photos have been flipped horizontally or rotated to different degrees. In order to create a model that is resilient to changes in orientation, it is necessary to mimic the item being viewed from several angles or orientations.
Translation: Some have been moved, either vertically or horizontally. This illustrates how items naturally vary in their placement inside a camera's range of vision.
Zooming: To reflect an object's look from varying distances, some photographs may be zoomed in or out to varying degrees.
Shearing: Shearing is the distortion of an image along one axis that mimics a shift in perspective in some cases.

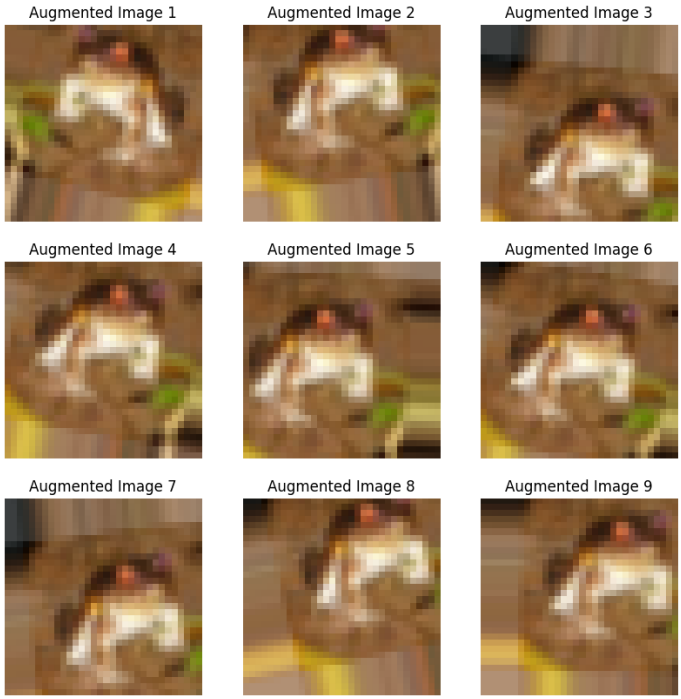

Figure 5.1 Data Augmentation

The t-SNE The CIFAR-10 dataset visualization offers a useful representation of the structure of the data in a condensed two-dimensional space. The scatter plot, which shows significant overlap between multiple groups while also clustering data points based on similarity, highlights the dataset's inherent complexity. Because of the tight grouping and intermingling of information, the different colors that correspond to the 10 different classes highlight the difficult work that generative models must do. This kind of visualization shows that although the data may be somewhat separated, there aren't any obvious class borders; instead, the differences are gradual. This implies that in order for Generative Adversarial Networks (GANs) to effectively identify and produce the subtle characteristics of each class, advanced feature detection is required. Refer 5.2

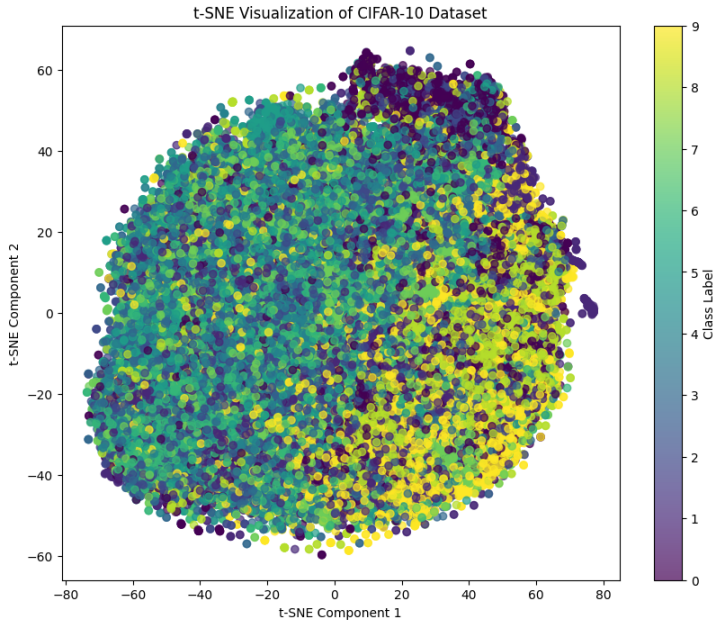

Figure 5.2 t-SNE visualization.

The t-SNE figure highlights the significance of resilience and complexity in the model architecture while training GANs using the CIFAR-10 dataset. While several picture groups exhibit remarkable similarities, the overall image space appears to be a tapestry of gentle transitions between classes, indicating a wide variety of related qualities, as indicated by the small clusters seen in the figure. Therefore, a GAN trained on this dataset needs a generator that can capture the essence of each class and a discriminator that is precisely tailored to differentiate these minute deviations. In order to generate varied and class-consistent images, the GAN may need to enhance certain parts of its generative skills. These regions are highlighted by the t-SNE analysis, which serves as a guide.

6. Model Findings

The Generative Adversarial Network (GAN) structure used for producing bird pictures from the CIFAR-10 dataset is Considering the preprocessed CIFAR-10 data, the model selects only photos belonging to the bird class and normalizes the pixel values. The discriminator and the generator are the two primary parts of the GAN architecture.

In order to produce fresh pictures, the Generator is intended to map an earlier noise distribution to the data space. It upscales the produced noise to the final image size gradually by using an initial dense layer followed by reshaping and upsampling layers. To improve the quality of the produced pictures, convolutional layers are alternated with batch normalization and ReLU activations.

A convolutional neural network with leaky ReLU activations and dropout for regularization is called the discriminator. This design decision makes gradient flow easier, which is important for learning. It makes a distinction between the false pictures created by the generator and the actual photos from the dataset. In order to trick the discriminator into thinking that artificial pictures are genuine, the generator plays a min-max game throughout the training process of this GAN. The model incorporates a number of advances from recent research, including batch normalization in nearly every layer, the use of distinct optimizers (Adam and SGD, respectively) for the generator and discriminator, the addition of noise to labels and inputs, and pre-training of the discriminator. Refer Fig 6.1
Images shown as amorphous blobs of color with only a passing similarity to the avian subjects of the CIFAR-10 dataset early in the training phase lacked clarity and variation. Nevertheless, after more training, distinguishable traits started to show. By the latter phases, the pictures had become more identifiable as birds, with features like beaks, wings, and plumage patterns showing through, albeit still being a little fuzzy and lacking in fine detail.

The accuracy metrics of the discriminator were originally high, but they gradually converged to a level that was closer to 50% - the threshold of uncertainty at which the discriminator can no longer accurately discern between genuine and false. This implies that the generator is creating visuals that are getting more and more realistic. Refer 6.2
0 [D loss: 1.080248] [G loss: 0.529628]

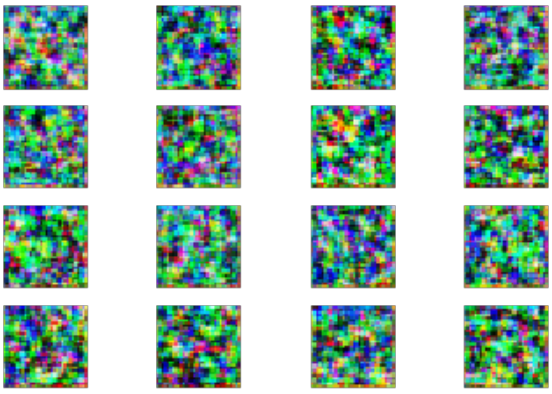

Figure 6.1 Epoch 0

25000 [D loss: 0.356242] [G loss: 2.008193]

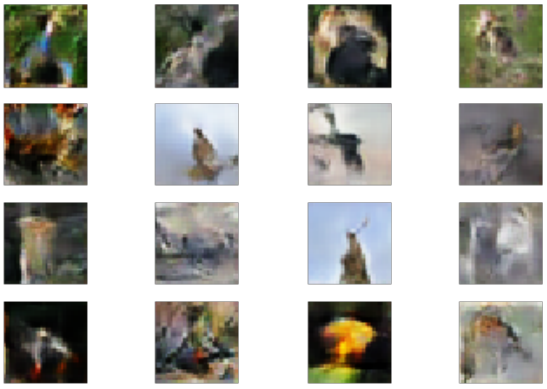

Figure 6.2 Epoch 25000

6.1 CGAN Model

The Conditional Generative Adversarial Network demonstrated promising results in generating the images. The model gave significant performance by maintaining balance between the generator and discriminator to prevent the collapse.

A key discovery was the model's improved capacity to produce pictures that match particular class labels and also represent the features of the dataset. This development highlights the model's ability to follow conditional inputs, which allows for the accurate creation of class-specific pictures with a high level of diversity and accuracy. After running the first epoch it gave a discriminator loss: 0.3560 and generator loss: 4.0177 and on the 120th epoch discriminator loss: 0.6427 – generator loss: 0.8364.

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 32, 32, 64)	7,552
leaky_re_lu (LeakyReLU)	(None, 32, 32, 64)	0
conv2d_1 (Conv2D)	(None, 16, 16, 128)	73,856
leaky_re_lu_1 (LeakyReLU)	(None, 16, 16, 128)	0
conv2d_2 (Conv2D)	(None, 8, 8, 256)	295,168
leaky_re_lu_2 (LeakyReLU)	(None, 8, 8, 256)	0
flatten (Flatten)	(None, 16384)	0
dropout (Dropout)	(None, 16384)	0
dense (Dense)	(None, 1)	16,385

Figure 6.1.1 CGAN Discriminator model Representation

Layer (type)	Output Shape	Param #
dense_1 (Dense)	(None, 4096)	454,656
leaky_re_lu_3 (LeakyReLU)	(None, 4096)	0
reshape (Reshape)	(None, 4, 4, 256)	0
conv2d_transpose (Conv2DTranspose)	(None, 8, 8, 128)	524,416
leaky_re_lu_4 (LeakyReLU)	(None, 8, 8, 128)	0
conv2d_transpose_1 (Conv2DTranspose)	(None, 16, 16, 128)	262,272
leaky_re_lu_5 (LeakyReLU)	(None, 16, 16, 128)	0
conv2d_transpose_2 (Conv2DTranspose)	(None, 32, 32, 128)	262,272
leaky_re_lu_6 (LeakyReLU)	(None, 32, 32, 128)	0
conv2d_3 (Conv2D)	(None, 32, 32, 3)	3,456

Figure 6.1.2 CGAN Generator model Representation

The training loss curves for the discriminator and generator in a CGAN are displayed on the supplied graph across a number of epochs. The generator's loss initially decreases dramatically, indicating quick early learning and development. It stabilizes after this dip, suggesting the generator is generating reliable, consistent data. The discriminator may not be improving much or may already be operating at peak efficiency if its loss is still minimal. The losses converge as training goes on, which might indicate that an equilibrium has been established where the generator's fakes are plausible enough to be mistaken for actual data at random times. On the other hand, issues like as overfitting or a lack of complexity in the model may be indicated if the discriminator's loss is regularly too low. Refer Fig 6.1.3

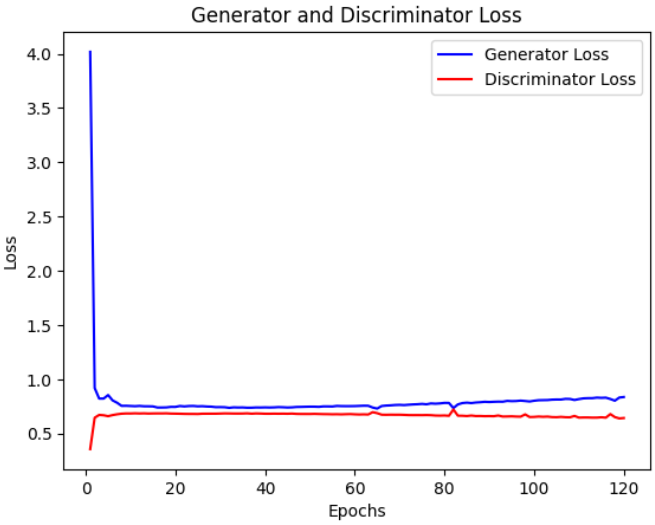

Figure 6.1.3 Loss percentage.

6.2. DCGAN Model

The generator's loss first spiked and then steadied as the training epochs went on, showing its enhanced adaptability and ability to produce visually compelling pictures. This version demonstrates how the generator may be trained to generate pictures that progressively defy the discriminator's evaluation standards. The discriminator's ability to discern between authentic and fraudulent photos varied noticeably. This unpredictability is a result of the discriminator's repeated learning process and its progressive fine-tuning in the identification of subtle differences between produced and actual pictures. The discriminator and generator losses peaked in the latter epochs, indicating an equilibrium in the adversarial learning process. This stability indicates that the generator has mastered the art of creating pictures that are challenging for the discriminator to correctly identify, fulfilling the stated purpose of GAN.

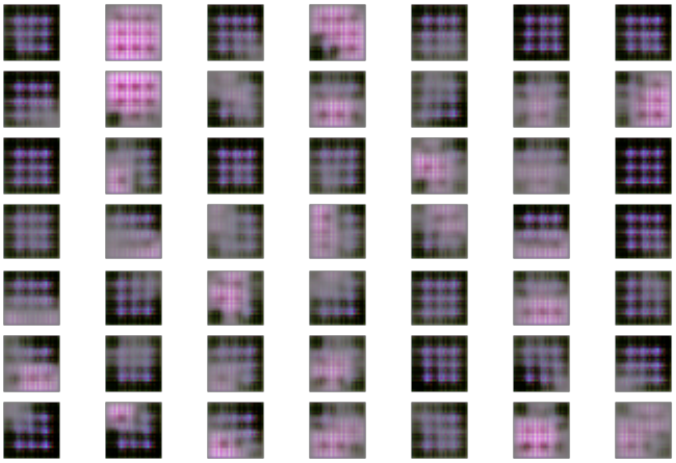

Figure 6.2.1 Image at epoch 200

The generator loss and the discriminator loss d1 and d2 were 0.35 and 0.18 respectively and the generator was 2.73 with having the Accuracy of being a real image 89% and fake 92% respectively. Refer fig 6.2.1.

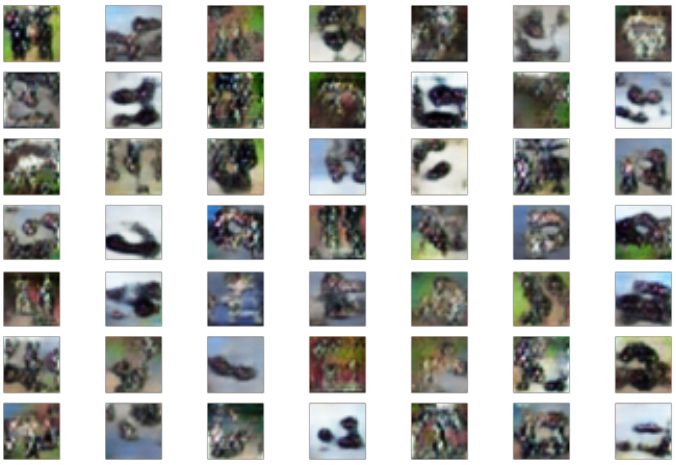

Figure 6.2.1 Image at epoch 5000

The generator loss and the discriminator loss d1 and d2 were 0.62 and 0.59 respectively and the generator was 21.267 with having the Accuracy of being a real image 49% and fake 83% respectively. Refer fig 6.2.2.

The generator's increasing performance over time is demonstrated by the accuracy percentages in identifying bogus pictures. Accuracy increases show that the generator can produce pictures with more

fidelity, getting closer to the distribution of the original dataset.

These findings underscore the effectiveness of the DCGAN architecture in learning to generate high-quality images from a complex dataset.

7. References

1. https://pennylane.ai/qml/demos/tutorial_quantum_gans/
2. https://www.mdpi.com/2079-9292/12/4/856
3. https://ieeexplore.ieee.org/abstract/document/10264175
4. https://machinelearningmastery.com/how-to-develop-a-generative-adversarial-network-for-a-cifar-10-small-object-photographs-from-scratch/
5. https://machinelearningmastery.com/how-to-implement-the-frechet-inception-distance-fid-from-scratch/
6. https://machinelearningmastery.com/how-to-implement-the-inception-score-from-scratch-for-evaluating-generated-images/
7. https://mafda.medium.com/gans-generative-adversarial-networks-101-8bf8e304585c
8. https://www.cs.toronto.edu/~kriz/cifar.html
9. https://arxiv.org/pdf/1511.06434.pdf
10. https://www.nature.com/articles/s41534-019-0223-2
11. https://pubs.acs.org/doi/10.1021/acs.jcim.3c00562
12. https://www.tensorflow.org/tutorials/generative/dcgan
13. https://paperswithcode.com/method/dcgan
14. https://www.analyticsvidhya.com/blog/2021/07/deep-convolutional-generative-adversarial-network-dcgan-for-beginners/
15. https://towardsdatascience.com/cgan-conditional-generative-adversarial-network-how-to-gain-control-over-gan-outputs-b30620bd0cc8
16. https://machinelearningmastery.com/how-to-develop-a-conditional-generative-adversarial-network-from-scratch/
17. https://www.educative.io/answers/what-is-a-conditional-gan-cgan