Generative Adversarial Neural Networks

April 19, 2025

Authors:

Fabio Cozzuto Student ID: 002214965

Johan Mogollon Student ID: 002359844

Contributions:

- Fabio Cozzuto: All code, experiments, and analysis

- Johan Mogollon: All code, experiments, and analysis

Course: CS551 - Deep Learning

Professor: Dr. Mohammed Ayoub Alaoui Mhamdi

1.1 Abstract

In this report, we present the implementation and evaluation of two Generative Adversarial Network (GAN) architectures. In the first part we focused on the development of a Deep Convolutional GAN (DCGAN) to generate images of grumpy cats from random noise inputs. We implemented and analyzed key components, such as data augmentation, the discriminator, the generator, and the training loop for this study. Additionally, we evaluated the impact of data augmentation strategies (basic and deluxe) on the training process, with visual comparisons of generated samples at different training stages.

In the second part, we explored the CycleGAN architecture for unpaired image-to-image translation, by transforming images between two distinct cat types: Grumpy and Russian Blue. The CycleGAN implementation includes encoding, transformation, and decoding stages in the generator, as well as the introduction of cycle consistency loss that guarantees that the translated images retain their original features. We conducted experiments to compare results with and without cycle consistency loss, highlighting its impact on training stability and image quality.

The report consolidates all results and analyses, accompanied by visualizations of training losses, generated samples, and loss curves. This work gave us practical experience in implementing and training GANs, providing insights into their capabilities and limitations in image generation and translation tasks.

1.2   Introduction

Generative Adversarial Networks (GANs) have become a powerful framework for generative modeling in deep learning, enabling us to create realistic images and data samples across various domains. In this report, we investigate two distinct GAN architectures; Deep Convolutional GAN (DCGAN) and CycleGAN to explore their capabilities in image generation and translation tasks.

At the core of GANs, lies a competitive game between two neural networks: the Generator, whose job is to produce images that appear authentic, and the Discriminator, that tries to discern real images from fake ones. This adversarial game, drives both networks to improve continuously, but also leads to the generation of more and more realistic images. In our study, the DCGAN is tasked with creating images of grumpy cats from random noise inputs, while CycleGAN is used for unpaired image-to-image translation between two distinct cat types: Grumpy and Russian Blue. CycleGAN uses a dual-generator setup and incorporates cycle consistency loss to ensure that transforming an image from one style to another, and back, preserves its original features.

In this stydy, we also examine data augmentation techniques. In particular, we compare a "basic" augmentation strategy with a more elaborate "deluxe" approach to understand how the quality and variability of the training data affect the performance and stability of both GAN models.

Our report consolidates the findings through detailed visualizations of training losses, generated samples, and loss curves, offering a comprehensive look at the strengths and limitations of GANs in creative and transformative applications.

1.3   Methodology

In this section, we describe the methodology that we used for this study, can be subdivided and described as follows:

1.3.1   Deep Convolutional GAN (DCGAN)

- **Data Preparation:**
  - The grumpy cat images dataset was preprocessed and augmented using two strategies: basic and deluxe augmentations.
- **Model Architecture:**
  - We implemented the DCGAN architecture with a generator and discriminator, designed using convolutional and deconvolutional layers.
  - We performed padding calculations to make sure that the spatial dimensions were halved at each layer of the discriminator.
- **Training:**
  - We implemented a training loop to optimize the generator and discriminator, using adversarial loss.
  - We used TensorBoard to log training metrics and visualize loss curves.
  - We conducted experiments with both basic and deluxe augmentations, to evaluate their impact on the model's performance.
- **Evaluation:**
  - We analyzed the images generated at different training iterations, to examine the quality and progression of the model.
  - We compared the loss curves for the generator and discriminator, to understand the training dynamics.

1.3.2   CycleGAN

- **Data Preparation:**
  - We used unpaired datasets of grumpy and Russian Blue cat images.
  - Preprocessing steps included resizing and normalization of the images.
- **Model Architecture:**
  - We implemented the CycleGAN architecture with two generators and two discriminators.
  - We incorporated cycle consistency loss, to ensure that translating an image to another domain and back retains its original features.
- **Training:**
  - We implemented a training loop to optimize the generators and discriminators, using adversarial and cycle consistency losses.
  - We experimented with and without cycle consistency loss, to evaluate its impact on training stability and image quality.
- **Evaluation:**
  - We analyzed the images generated at different training iterations, to assess the quality of the translations.
  - We compared the loss curves, to understand the effect of cycle consistency loss on training dynamics.

1.4   Results and Discussion

1.4.1   Results Overview

In this section, we summarize the results obtained from training the DCGAN and CycleGAN models. We analyze the generated images, loss curves, and training dynamics to provide insights into the performance of the models. Finally, we compare the results of different augmentation strategies and the impact of cycle consistency loss on CycleGAN.

1.4.2   DCGAN Results

Generated Images   Basic Augmentation: - Iteration 200: The generated images are mostly noise and lack any discernible structure resembling a grumpy cat.
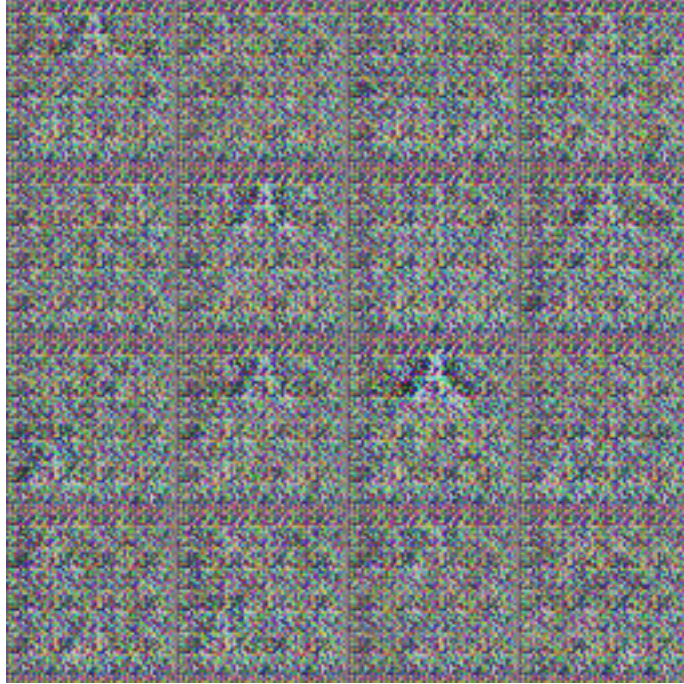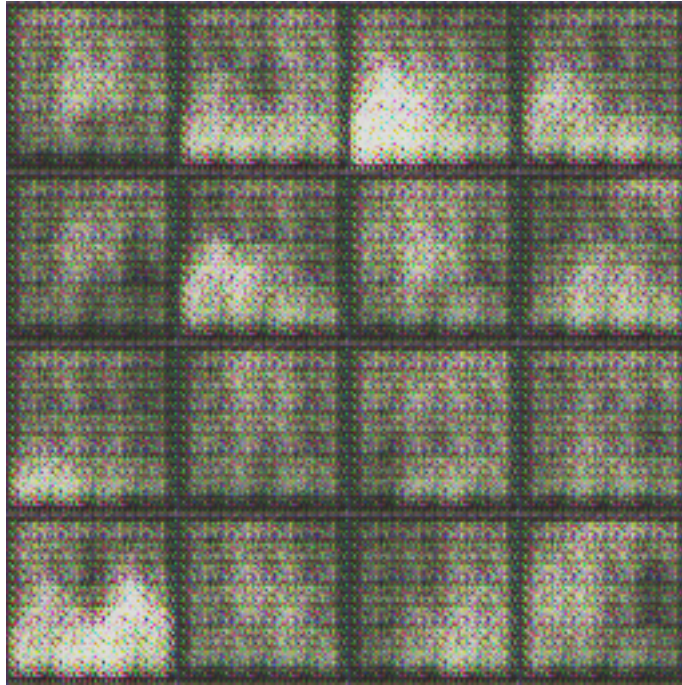
- **Iteration 1200:** The images show some improvement, with vague shapes and colors resembling cats, but they remain blurry and lack fine details.
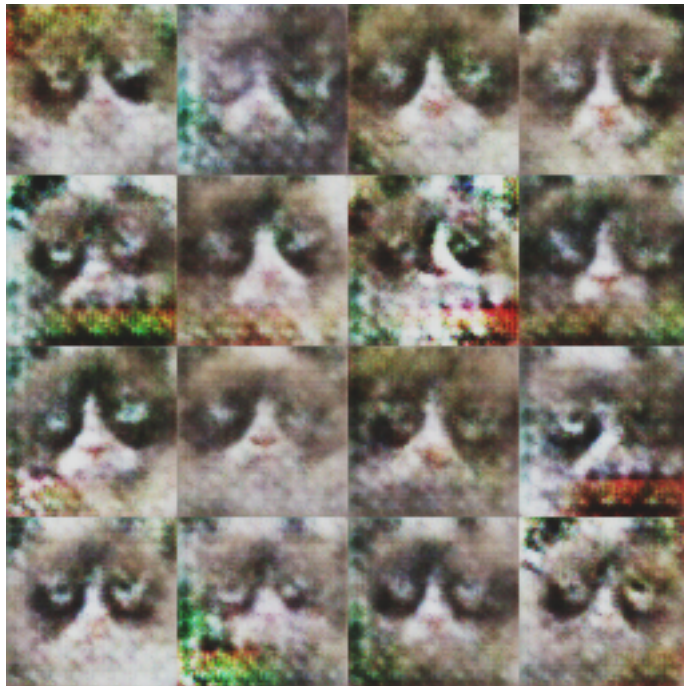

By comparing the images that Vanilla GAN generated, we can see that at the beginning, around step 200, the images are just messy noise and do not look like anything specific, not even a cat. This is normal because the network that creates the images is just starting to learn and being as basic as Vanilla_Gan is, it may not be as fast in generating good results from the beginning. However, if we look at step 1200, the images improve a bit. Even so, they are still blurry and although you can

see some shapes and colors that maybe look a bit like cats they are not of the best quality. It is as if the network is slowly realizing what a grumpy cat looks like, but it is still not very clear or real, it is clear at this stage that the network requires many more steps to learn how to create cat images, or, as seen in the following part of the results, we need to use data augmentation techniques, to obtain better images.

Deluxe Augmentation: - **Iteration 200:** Similar to the basic augmentation, the images are noisy and lack structure.


- **Iteration 1200:** The images show better quality compared to the basic augmentation, with more defined shapes and textures.
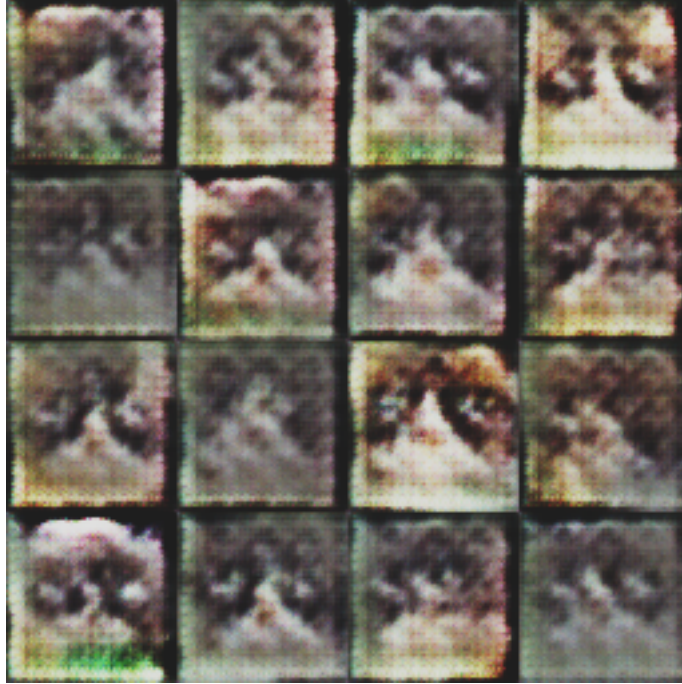
Loss Curves

- **Basic Augmentation:** On the Generator losses we can see an increase trend over the training steps. This indicates that the discriminator is getting better at separating false images from real ones, making it more difficult for the generator to "fool" it. This increase in loss may suggest that the generator requires additional effort to achieve good images, and that the discriminator is indeed getting better, outperforming the generator. However, it may also indicate that the generator is not performing as well as the training progresses. Looking at the generated images is important to understand which scenario is happening.

- The discriminator losses show a decreasing trend in both false images and real images. This indicates that, as training progresses, the discriminator fails to differentiate between false and real images. Now, it is possible to expect that D/false_loss is reduced as the generator gets closer to producing better images, in the same way that a reduction in D/real_loss can be seen. Both reductions would be indications that the generator is managing to "fool" the discriminator, however, it could also indicate that there is a learning problem and that therefore the discriminator is losing the ability to effectively differentiate between the two groups of images.
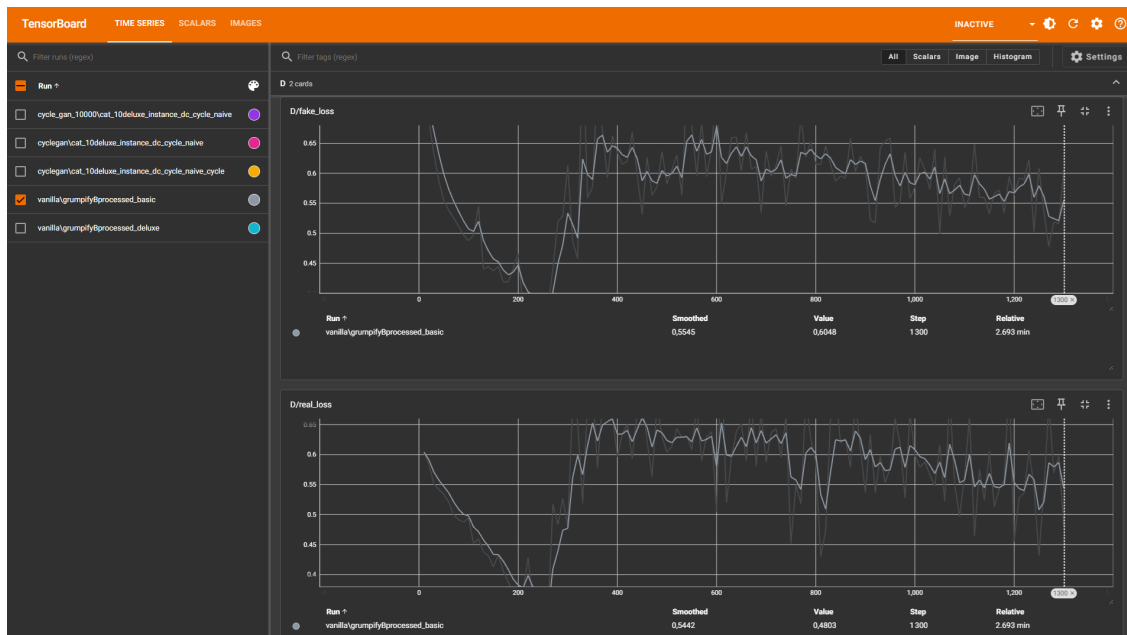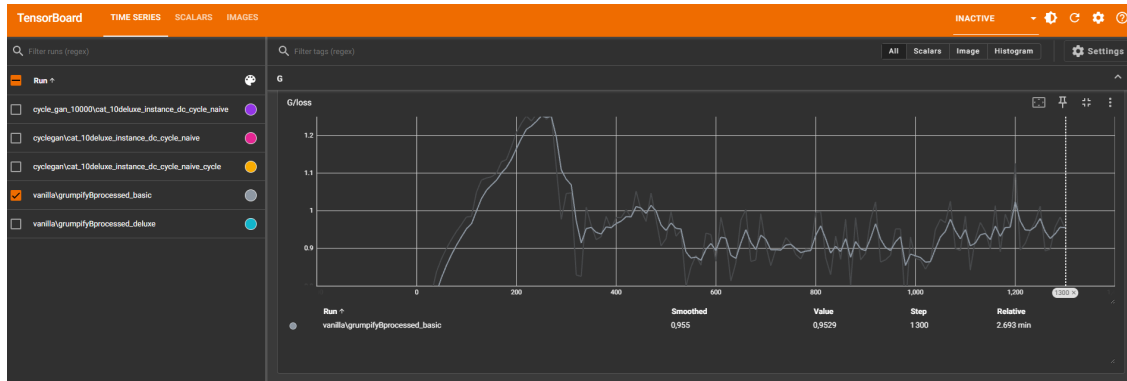
- **Deluxe Augmentation:** The curve D/false_loss starts high and decreases as the training progresses, showing some fluctuations during the process. Again, at the beginning, the discriminator manages to identify the generated images as false, but its accuracy decreases as the training advances, suggesting that the generator improves its performance. Now, the fluctuations that we see could suggest moments when the discriminator adapts to the strategies of the generator to produce more realistic false images. The same way D/real_loss shows a dropping tendency from a high starting point, this suggests that the performance of the discriminator in trying to classify images decreases over the training process. The presence of the augmentations is an important component for this to happen, as the discriminator learns to identify real images even under various transformations, but the decreasing loss indicates that the generator also improves. The fluctuations could represent the Discriminator continued attempts to learn features and correctly identify images.

7
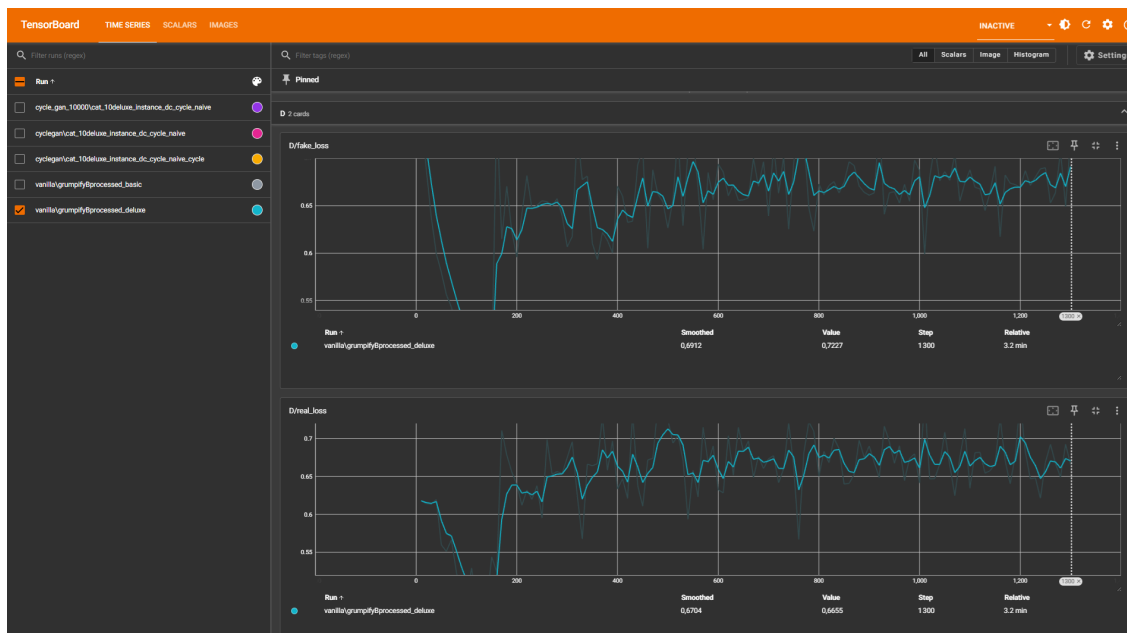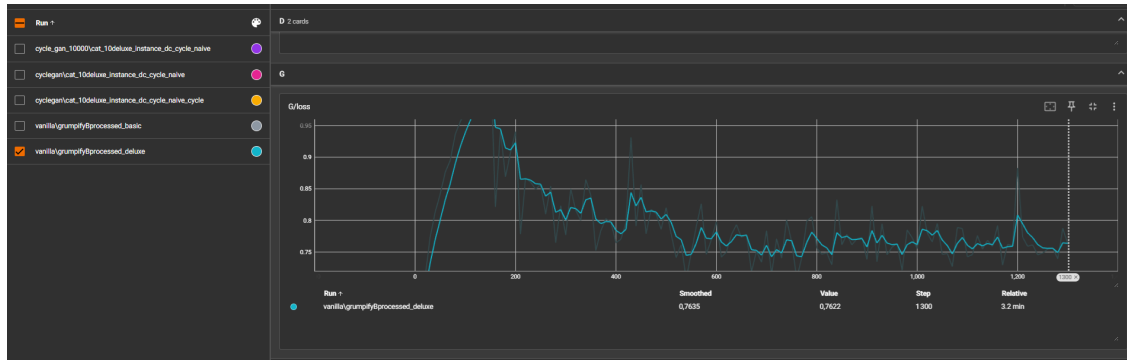
1.4.3 CycleGAN Results

Generated Images

- **Without Cycle Consistency Loss:**
  - **Iteration 400:** The generated images show some resemblance to the target domain but lack fine details and consistency.

– **Iteration 700:** The images improve slightly, but the quality remains suboptimal.

– **Iteration 10000:** The images improve, with acceptable quality.

- **With Cycle Consistency Loss:**
  - **Iteration 400:** The images are more consistent and retain some features of the original domain.

  - **Iteration 700:** The images show noticeable improvement, with better textures and colors.

– **Iteration 10000:** The images show substantial improvement, with even better textures and colors.

Looking at the images, we can see that the ones from step 400 may be somewhat similar to the ones from step 700, although they change and look somewhat better. This is probably because when the training has just started the generator has not learned very well how to change the first type of image to look like the second type. In this case, both images may not be very well created and we could at a glance identify which are fake and which are real, since if we look at step 700, even though the network tries to get better at making the images look like the other type, it may not be enough to have better textures, match colors and look more real. So we may still see some problems or things that don't look quite right because the model is still learning and trying to improve in the next steps. Something that could be improved if we revisit images in later steps in the training, as seen in the 10000 run.

Loss Curves

- **Without Cycle Consistency Loss:** The loss curves show fluctuations, indicating instability in the training process.
- **With Cycle Consistency Loss:** The loss curves are a little bit smoother and stable, suggesting that the cycle consistency loss helps in regularizing the training.
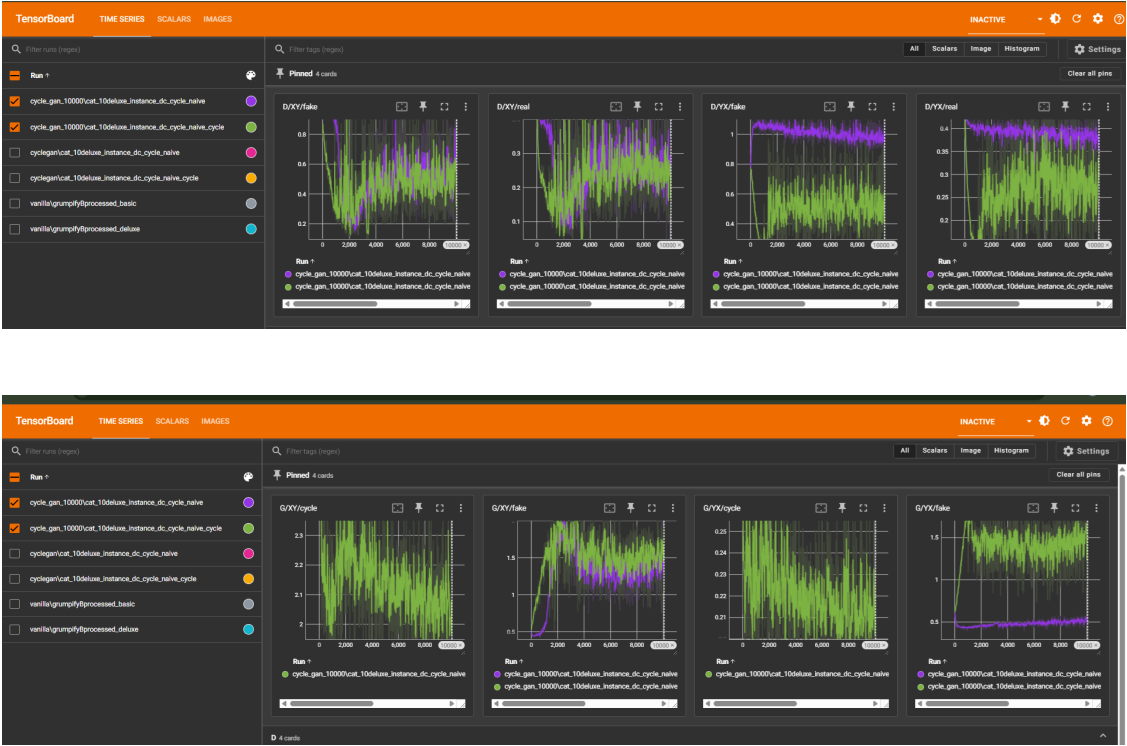
1.4.4 Results Summary

1.5 Results Table

Model	Augmentation	Iterations	Insights	Loss Curve Insights
DCGAN	Basic Augmentation	200	No discernible structure, mostly noise.	Increasing generator loss, decreasing discriminator loss.
		1200	Vague shapes resembling cats, blurry images. (wrapped for clarity)	Stable training dynamics.
	Deluxe Augmentation	200	No discernible structure, mostly noise.	Similar to basic augmentation.
		1200	Better-defined shapes and textures compared to basic augmentation.	More stable loss curves compared to basic augmentation.
CycleGAN	Without Cycle Consistency	400	Some resemblance to the target domain, but lacks fine details.	Fluctuating loss curves, indicating instability.

Model	Augmentation	Iterations	Insights	Loss Curve Insights
		700	Slight improvement in quality, but still suboptimal. (wrapped for clarity)	Fluctuations persist.
		10000	Most noticeable improvement in overall quality, textures and colors.	Stable training dynamics, showing that more iterations help to the result.
	With Cycle Consistency	400	More consistent images, retaining features of the original domain.	Smoother and more stable loss curves.
		700	Noticeable improvement in textures and colors, better quality overall.	Stable training dynamics, indicating the effectiveness of cycle consistency loss.
		10000	Most noticeable improvement in overall quality, textures and colors.	Stable training dynamics, indicating the effectiveness of cycle consistency loss.

1.5.1 Discussion

1. **Impact of Augmentation on DCGAN:**
   - The deluxe augmentation strategy leads to better image quality and more stable training compared to the basic augmentation. This highlights the importance of diverse and enriched training data in improving GAN performance.
2. **Effectiveness of Cycle Consistency Loss:**
   - The inclusion of cycle consistency loss in CycleGAN significantly improves the stability of the training process and the quality of the generated images. This demonstrates the importance of regularization techniques in unpaired image-to-image translation tasks.

1.6 Questions and Answers

1.6.1 Padding Calculation for DCGAN Discriminator

Question: With kernel size (K=4) and stride (S=2), what padding (P) halves the spatial dimensions?

Answer: We want each layer to reduce the spatial dimensions by a factor of 2, without clipping important features. That means that we want to control the padding. So, we have the convolution output formula:

$$O = \left\lfloor \frac{I + 2P - K}{S} \right\rfloor + 1$$

Where: - (I) = input size - (O) = output size - (K = 4) (kernel size) - (S = 2) (stride) - (P) = padding

We want to obtain this:

$$\text{output\_size} = \frac{\text{input\_size}}{2}$$

So we solve as follows:

$$\left\lfloor \frac{I + 2P - 4}{2} \right\rfloor + 1 = \frac{I}{2} \Rightarrow 2P = 2 \Rightarrow P = 1$$

### 1.6.2 Can you account for these differences?

Answer: We can see that when we use cycle consistency, the loss curves tend to be more stable, with gradual slopes and fewer extreme flactuations. This probably happens because the consistency loss acts as a guiding principle that ensures that the networks not only create fake images to "fool" the other network, but also thet they maintain coherence when an image is modified and then reverted to its original state. This guiding principle helps the training to be more stable and helps the networks to learn in a more organized way, instead of simply trying to "fool" each other all the time. The generator seems to learn better when following this principle.

1.6.3 Provide explanations as to why there might or might not be a noticeable difference between the two sets of results.

Answer: Analyzing the images, we do not see a big difference in the final images. This could be because the two types of "grumpy cats" we are using are not different in style. If the transformation we want to make is not too significant, the network can probably still learn to perform it correctly, even without the cycle coherence rule. Also, the rule depends on its weight parameter (lambda), and if this parameter needs to be tuned.

In this case, we used L1 loss, as it was recommended in the original paper. Perhaps, if we trained the model longer or used larger networks, or if the difference between cat styles was more noticeable, we would see a significant improvement in the images produced when we use the cycle coherence rule.

Any differences appear to be difficult to detect and would probably require closer examination; at a glance the images generated are the same. This visual similarity supports the explanation given above about the possible reasons for a possible large difference in the final result for this particular data set and training configuration.
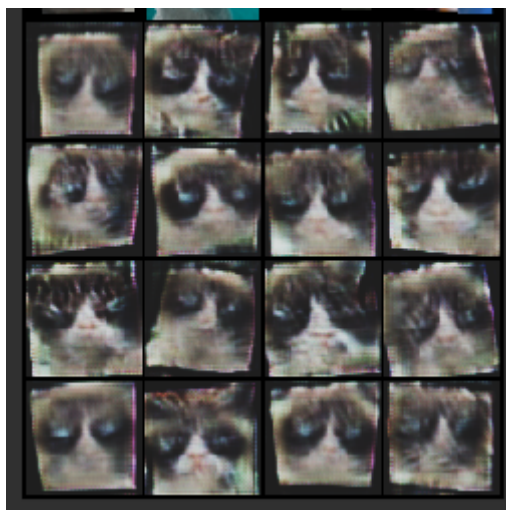
To observe a noticeable difference, we can compare the images generated at iteration 1900 with the one from the final iteration. We notice that the images generated in this iteration, show a significant improvement in quality, which leads us to conclude that the Generator has learned to create better images. Finally, this comparison allows us to conclude that CycleGan performs much better that VanillaGan at generating high-quality images.

1.7   Conclusion

In this report, we explored the implementation and evaluation of two Generative Adversarial Network (GAN) architectures: Deep Convolutional GAN (DCGAN) and CycleGAN. Through several experiments, we analyzed the impact of data augmentation strategies and the inclusion of cycle consistency loss on the performance and stability of these models.

For DCGAN, we can see that the deluxe augmentation strategy led to better image quality and more stable training dynamics compared to the basic augmentation. This highlights the importance of diverse and enriched training data in improving the performance of GANs. However, the generated images still lacked fine details, indicating the need for further optimization or more advanced architectures.

For CycleGAN, the inclusion of cycle consistency loss significantly improved the stability of the training process and the quality of the generated images. This demonstrates the importance of regularization techniques in unpaired image-to-image translation tasks. While the generated images showed noticeable improvements with cycle consistency loss, the differences between the two cat styles used in this study may have limited the visual impact of the results.

Overall, this study provided us with valuable insights into the capabilities and limitations of GANs in image generation and translation tasks. Future work could focus on exploring more complex architectures, longer training durations, and additional data augmentation techniques to further enhance the quality of the generated images.