# Final Exam Report

Exam Title: Generative Adversarial Neural.

Course Name: CS 551 – Deep Learning.

Professor's Name: Dr. Alaoui Mhamdi.

University Name: Bishop’s University.

Students:

Johan Stevenson Mogollon Arcos / Student ID: 002359844, Contribution: All code, experiments, and analysis.

Fabio Cozzuto / Student ID: 002214965. Contribution: All code, experiments, and analysis.

## Abstract

## In this report, we analyze the results of the final exam, in which we conducted an investigation of generative adversarial networks (GANs) and their different configurations. Two architectures, Vanilla\_GAN and CycleGAN were developed and then trained with a dataset of “grumpy cats”. the idea is to compare the performance of the models with two augmentations: basic and deluxe. We analyzed the loss curves of the networks, both for the Generator and the Discriminator, by visual inspection of the generated image at different iterations. in the same way, we evaluated how the various configurations influenced the learning process. The results lead us to interesting conclusions, providing knowledge, ideas, and learning about strategies for training GANs for image transformation tasks.

## Introduction

Generative Adversarial Networks (GANs) are important models used in deep learning. In this particular case, they are used to create images. The main idea is to develop two neural networks that compete against each other, like in a game: one player (Generator) tries to create fake images that look real, and the other player (Discriminator) tries to identify which ones are real or fake. This game allows both players to improve their performance, and in the case of the Generator, to learn how to create more and more images that look realistic.

One thing that we can do with GANs is image-to-image translation. This is done when we want to change an image to make it look like another type or style of image. CycleGAN is a useful tool to do so; it uses two Generators to go from one style to the other, and vice versa, and uses something called cycle consistency loss to make sure that if we change an image to another style and then back, it looks like the original.

The quality of the images that GANs produces, depends on the training set of images and how we modify them to get more variety. This modification is called data augmentation, and it is done by using one the different methods to modify the images, so that the GANs learn better and help improving the quality of the images generate by the model.

In this paper, we are going to see what happens when we use two different ways of augmenting the data: a “basic” way and a more elaborate, “deluxe” way. We will use both GANs, Vanilla\_GAN and CycleGAN, to change images of “grumpy cats” from one type to another and then compare the results. We will see how these augmentations affect how the networks learn and how realistic the images they create are. The goal of this exercise is to better understand how GANs works and acquire knowledge in this field.

## Part 1: Deep Convolutional GAN

* **Implement Data Augmentation [10 points]**

.

* **Implement the Discriminator of the DCGAN [10 points]**

.

* **Generator [10 points]**

.

* **Training Loop [10 points]**
* **Experiment [10 points]**

.

## Part 2: CycleGAN

* **Data Augmentation**
* **Generator [20 points]**
  + Encoding Stage
  + Transformation Stage
  + Decoding Stage
* **CycleGAN Training Loop [20 points]**

We use L1

* **CycleGAN Experiments [15 points]**

.

**Can you account for these differences?**

We can see that when we use cycle consistency, the the loss curves appear more stable, as their slopes are generally gradual, without any major peaks. This could happen because the consistency loss acts as a rule that tells the networks not only to create fake images that "fool" the other network, but also to make sure that if you change an image and then change it back again, the it looks like the original image you started from. This rule helps the training to be more stable and helps the networks learn in a more organized way, rather than in a disorganized way, trying to "fool" each other all the time. The generator seems to learn better when we this rule applies.

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

Analyzing the images, one possible explanation as to why we don't see a big difference in the images in the end, is because the two types of “grumpy cats” we are using are not that different in style. If the change we want to make isn't too significant, the network can probably learn to do it right, even without the cycle consistency rule. Also, the rule has a weight (lambda), that if we don't adjust correctly, it may not help the network in its learning process. Additionally, in this case we use 𝐿1 loss, as the original paper suggested. Perhaps, if we trained 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. This indicates that the rule helps the learning process to be more stable. However, for the images to look much better, we could apply further augmentations or train for longer.

## Improvement or Further Experimentation

.

## Conclusions

.