Gabriel Augusto Santos Silva

Aumento de Dados e Classificação Profunda com Redes Adversárias Generativas

Data Augmentation and Deep Classification with Generative Adversarial Networks

Gabriel Augusto Santos Silva

Aumento de Dados e Classificação Profunda com Redes Adversárias Generativas

Data Augmentation and Deep Classification with Generative Adversarial Networks

"*But still try, for who knows what is possible?*"

— Michael Faraday

Universidade de Aveiro
2021

Gabriel Augusto Santos Silva

Aumento de Dados e Classificação Profunda com Redes Adversárias Generativas

Data Augmentation and Deep Classification with Generative Adversarial Networks

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia de Computadores e Telemática, realizada sob a orientação científica do Doutor Filipe Miguel Teixeira Pereira da Silva, Professor auxiliar do da Universidade de Aveiro, e da Doutora Pétia Georgieva, Professora associada do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro.

Dedico este trabalho à minha família e amigos.

o júri / the jury

presidente / president

Prof. Doutor Joaquim João Estrela Ribeiro Silvestre Madeira
Professor Auxiliar do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro

vogais / examiners committee

Prof. Doutora Catarina Helena Branco Simões Silva
Professora Auxiliar do Departamento de Engenharia Informática da Faculdade de Ciências e Tecnologia da Universidade de Coimbra

Prof. Doutor Filipe Miguel Teixeira Silva
Professor Auxiliar do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro

agradecimentos / acknowledgements

Quero deixar um agradeciemnto aos professores Filipe Silva e Pétia Georgieva pela orientação e ajuda dada durante este trabalho. Quero também agradecer à equipa do LAR, do departamento de Engenharia Mecânica da Universidade de Aveiro, por me facilitarem os recursos computacionais que usei ao longo da dissertação. Finalmente, deixo um agradecimento aos meus familiares e amigos que me acompanharam durante o meu percurso letivo.

Palavras Chave

Generative Adversarial Networks (GAN), Auxiliary Classifier GAN (AC-GAN), BigGAN, lesões de pele, CIFAR-10, aumento de dados

Resumo

Aprendizagem automática tem visto bastantes melhorias em anos recentes. Um tipo de modelo que tem evoluído bastante são as Generative Adversarial Networks (GANs). Estes modelos têm a capacidade de criar dados falsos que se assemelham aos dados em que foram treinados. O interesse por estes modelos tem vindo a crescer desde a sua criação, em 2014. Tem-se vindo a provar que a possibilidade de criar dados falsos pode ser bastante útil, especialmente, em áreas com pouca abundância de dados, como é o caso da imagem médica. As GANs têm sido usadas, com bastante sucesso, nesse tipo de áreas para aumentar o tamanho dos datasets existentes de modo a melhorar a qualidade dos classificadores usados. Esta dissertação faz um estudo com um tipo específico de GAN, a Auxiliary Classification GAN (AC-GAN), para perceber se existem novas formas para as GANs melhorarem a qualidade de classificadores. Para isso, uma experiência de três partes foi desenhada, sendo cada parte designada como um Cenário. No Cenário 1 um classificador isolado foi treinado, no Cenário 2 foi treinado um classificador igual após uma GAN ter sido usada para fazer aumento de dados e, finalmente, no Cenário 3 usou-se uma AC-GAN em vez de um classificador. Foram considerados dois problemas distintos. O primeiro foi o CIFAR-10, que é um problema bastante conhecido e bem estruturado, usado com muita frequência em problemas relacionados com GANs. O segundo problema usado foi um de lesões de pele. Isto serviu dois propósitos: aumentar significativamente a dificuldade do problema em mão e aproximar o trabalho feito aqui com um dos maiores usos práticos das GANs, que tem sido o uso de GANs para fazer o aumento de datasets em problemas de imagem médica. Os modelos desenvolvidos foram baseados na AC-GAN original e na BigGAN, que, quando foi apresentada, era a melhor GAN conhecida e era capaz de produzir imagens de alta qualidade com resoluções de até 512x512. Adaptar a BigGAN para uma AC-GAN resultou na melhor AC-GAN conhecida treinada no dataset CIFAR-10. O estudo feito nesta dissertação pode servir como uma base sólida para que mais estudos sejam feitos neste âmbito, visto que os resultados obtidos aqui sugerem firmemente que o uso de AC-GANs pode ser uma forma efetiva de atingir classificadores melhores.

Keywords Generative Adversarial Networks (GAN), Auxiliary Classifier GAN (AC-GAN), skin lesion, CIFAR-10, data augmentation

Abstract Machine learning has seen many advances in recent years. One type of model that has evolved a lot recently is Generative Adversarial Networks (GANs). These models have the ability to create fake data that resembles the data on which they were trained on. The interest for these models has been ever growing since their creation, in 2014. The ability to create fake data has also been found to be quite useful, especially, in data starved areas, like medical imaging. GANs have been used, with positive results, in areas like these to increase the size of the datasets available, as a way to improve the quality of classifiers. This dissertation makes a study with a specific type of GAN, the Auxiliary Classification GAN (AC-GAN), to understand if there may be new ways in which GANs can improve classification tasks. For this, a three parted experimented was designed, with each part being denominated as a Scenario. In Scenario 1 a standalone classifier was trained, in Scenario 2 that same classifier was trained after data augmentation was done with a GAN and, finally, in Scenario 3 an AC-GAN was used instead of the classifier. Two distinct problems were considered here. The first was the CIFAR-10 problem, which is a well known and structured problem, quite often used as a benchmark in GAN related works. The second problem used here was a skin lesion one. This served two purposes: significantly increasing the difficulty of the problem at hand and approximating the work done here to, possibly, the biggest practical usage of GANs, which has been data augmentation for medical imaging problems. The models developed were based on the original version of the AC-GAN and on the BigGAN, which, when presented, was the best performing GAN known, able to produce high quality images of resolutions of up to 512x512. Adapting the BigGAN into an AC-GAN resulted in the best known performing AC-GAN on the CIFAR-10 dataset. The study made in this dissertation can serve as a solid backbone for further studies on this matter to be made, since the results obtained here strongly suggest that the use of AC-GANs can be an effective way to achieve superior classifiers.

Contents

List of Figures

List of Tables

Acronyms

AC-GAN	Auxiliary Classifier GAN
ADA	Adaptive Discriminator Augmentation
AdaIN	Adaptive Instance Normalization
AK	Actinic Keratosis
AUC	Area Under Curve
BCC	Basal Cell Carcinoma
BK	Benign Keratosis
BMA	Balanced Multi-Class Accuracy
CGAN	Conditional GAN
CIFAR-10	Canadian Institute For Advanced Research
ciGAN	Conditional Infilling GAN
CNN	Convolutional Neural Networks
CT	Computed Tomography
DCGAN	Deep Convolutional GAN
DF	Dermatofibroma
DIL	Dermoscopic Image Library
FFHQ	Flickr Faces HQ
FID	Frechet Inception Distance
GAN	Generative Adversarial Network
GPU	Graphics Processing Unit
IS	Inception Score
ISIC	International Skin Imaging Collaboration
KL	Kullback-Leibler
MEL	Melanoma
M-GAN	Multi-Channel GAN
MLP	Multi Layer Perceptron
MNIST	Modified National Institute of Standards and Technology
MRI	Magnetic Resonance Imaging
MS-SIM	Multi-Scale Structural Similarity
NLP	Natural Language Processing
NV	Melanocytic Nevus
PET	Positron Emission Tomography
PGAN	Progressive Growing GAN
SA-GAN	Self-Attention GAN
SCC	Squamous Cell Carcinoma
SL-StyleGAN	Skin Lesion StyleGAN
SN-GAN	Spectral Normalization GAN
SRGAN	Super-Resolution GAN
TAC-GAN	Twin Auxiliary Classifier GAN
TFD	Toronto Face Database
TTUR	Two Time Update Rule
UAC-GAN	Unbiased Auxiliary Classifier GAN
VASC	Vascular Lesion
WGAN	Wasserstein GAN

CHAPTER 1

Introduction

Machine learning is a very popular subject in computer science that has been growing every year and that has seen its abilities improve dramatically, both with the increase in computational power driven by the more capable hardware that is developed and by the creation of innovative mechanisms.

One of such mechanisms is convolutions, introduced by LeCun in 1989 [1], which showed to be very promising in computer vision tasks, *i.e.*, computer problems that involve images. Soon after their creation, neural networks based on convolutions, Convolutional Neural Networks (CNN), became the top performing models on computer vision machine learning competitions.

Later, with the rise of the Graphics Processing Unit (GPU), it was possible to massively improve the implementation of machine learning models (Oh and Jung [2]), and that made it much more feasible to further increase their complexity, which lead to the creation of increasingly bigger models, with increasingly better performances. Model complexity increased so much that a subfield of machine learning was born: deep learning.

Recently, one of the most interesting inventions in machine learning is the GAN. These models, introduced in 2014 by Goodfellow [3], have the ability to learn the distribution of a given dataset, and create fake data that resembles the original one. To this day, their improvement has been massive: models have gone from producing realistic 28x28 images of simple numbers in black and white to producing 1024x1024 images of highly realistic human faces in color.

1.1 Motivation

GANs were created with the purpose of generating data. They have, to this day, been used for things such as increasing the resolution of images [4] (Figure 1.1), photo inpainting (Figure 1.2), *i.e.*, filling in an area of a photo that was removed [5] or simply to generate realistic images of common entities, like animals or human faces (Figure 1.3) [6] [7].

Figure 1.1: From left to right: using bicubic interpolation to increase resolution of an image, using the SRGAN for the same purpose and the original higher resolution image. Adapted from [4].

Figure 1.2: An example of photo inpainting, taken from [5].

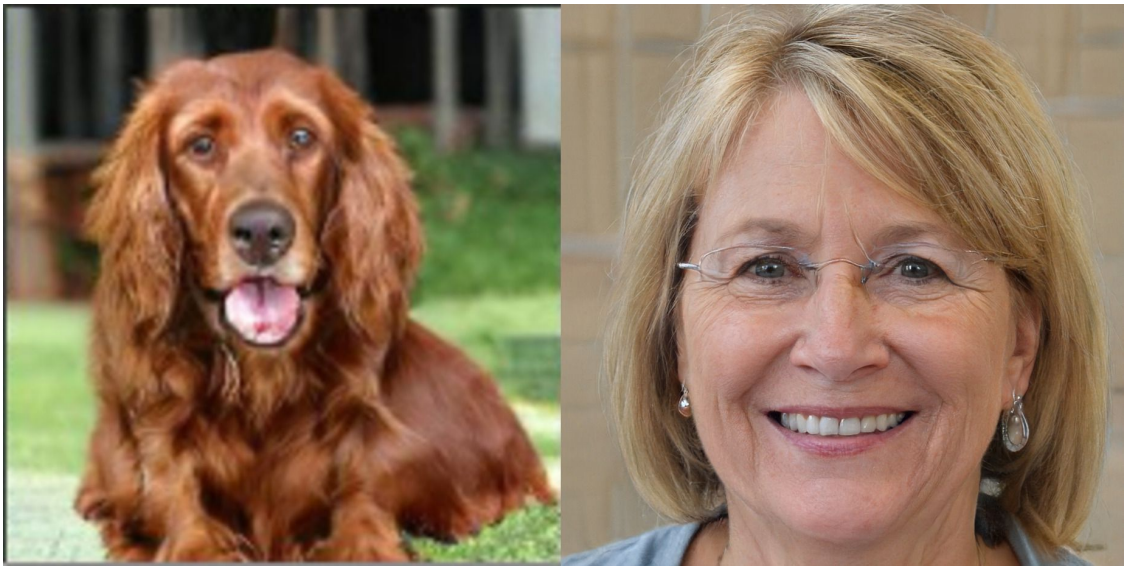

Figure 1.3: High resolution images of a dog and a person, created by the BigGAN [6] and the StyleGAN2 [8], respectively.

One other use of GANs that has received a lot of attention has been data augmentation. A GAN is trained to produce images of a certain problem and then those images are used to enhance the dataset with which another neural network will be trained, in an attempt to improve the later model. This gives GANs a very broad application, since it can be applied to any problem. Still, GANs can not, at least to this day, perfectly mimic the data distribution of a dataset, and, so, traditional data augmentation techniques, such as flips, rotations and color changes, tend to lead to better improvements than data augmentation through GANs [9]. Still, there are areas were data is so scarce that data augmentation through GANs became a quite explored subject. One of those areas is medical imaging problems. The biggest issue in medical imaging problems is, perhaps, that they require an expert so that images can be accurately labeled, which limits, a lot, the amount of people available to assemble a big dataset. Furthermore, it is hard already on its own to acquire a large amount of images, since medical information is often private and medical images are not abundant (some medical images, for example, come from expensive exams that are only done when really necessary). As a result of this lack of data, a lot of work on data augmentation for medical imaging has been done regarding different kinds of images: lung cancer images (Positron Emission Tomography (PET) scans) [10], mammograms [11], skin lesions [12] [13], brain images [14], liver lesions [15], prostate lesions [16] and brain tumors [17].

However, there may be other ways to help models through the use of GANs that have yet to be explored. In particular, GANs have already been used to do the classification themselves, instead of being used to perform data augmentation. Yi *et al.* [18] worked with a Categorical GAN and used it as the classifier for a skin lesion dataset and Rashid *et al.* [19] proposed a GAN architecture that could make the classification task, on top of generating data, and used it for another skin lesion dataset. These two works gave promising results for a novel way to use GANs as a means to improve the classification task of a problem. This dissertation will be heavily focused on the use of a type of GAN which is very similar to the one proposed on the later work mentioned: the AC-GAN.

1.2 Objectives

A GAN is a combination of two models that work together (as will be explained ahead in Chapter 2) : a generator, which creates fake data, and a discriminator, which learns fake data from real data so that the generator can learn how to reproduce real data. Generally, when using GANs, the focused element, after training the GAN, is the generator. However, the discriminator of an AC-GAN is a little different and has the ability to perform the classification task, and the use of GANs this way has had some promising results [18] [19]. The goal of this work is to explore the performance of the discriminator of this GAN architecture and compare it with the performance of standalone classifiers and the more traditional use of GANs, which is to train a GAN first and then use it to perform data augmentation so that a classifier can then be trained.

The models developed for this purpose will be tested on the CIFAR-10 dataset and on a dataset of skin lesions. The CIFAR-10 dataset is a very standard dataset in GAN works and

is a well structured dataset with small images. This allows for less complex models, faster training while maintaining an interesting challenge. Then, the dataset of skin lesions comes to increase the complexity of the problem and approximate this study to one of the areas where GANs have been having the most impact.

1.3 Dissertation Outline

The rest of this dissertation is organized as follows:

- Chapter 2 talks about concepts that are relevant to the work done in this dissertation;
- Chapter 3 makes a study on the classification abilities of AC-GANs on the CIFAR-10 dataset;
- Chapter 4 presents a similar study to the one in Chapter 3, except on a dataset of skin lesions;
- Chapter 5 gives an overview of the work that was done and draws some final conclusions.

CHAPTER 2

Related Work

This chapter introduces related work, mainly focused on GANs. It will introduce several GANs, and how they work, data augmentation techniques, and how GANs fit in that theme, and finally will talk about evaluation metrics for GANs.

2.1 GAN Introduction

A GAN, as proposed by Goodfellow [3], consists of two models that compete with each other. On the one side is a discriminator, responsible for classifying data as real or fake, and on the other side is a generator, responsible for turning noise, $\boldsymbol{z}$, into data that matches the real data's distribution, making it seem real to the discriminator. When training begins the generator is only going to be able to produce data that makes little to no sense, making it very easy for the discriminator to see it as fake. This means that the generator's loss will be high, forcing it to make changes in its approach. Eventually the generator starts producing data that the discriminator can no longer easily distinguish from real data. This means that the discriminator will start miss-classifying data, making it's loss increase, changing the discriminator until it can again detect fake data. This special interaction between the models is the reason why it is said that the models compete, or are adversarial.

The loss function for the discriminator stems from the classic binary cross-entropy. If we consider $\boldsymbol{x}$ as the real data and $\hat{\boldsymbol{x}}$ as the fake data, the discriminator loss is:

$$\log(D(x)) + \log(1 - D(\widehat{x})) \tag{2.1}$$

where $D(x)$ is the discriminator's prediction on real data and $D(\widehat{x})$ is the discriminator's prediction on fake data. Yet, since we know that fake data comes from the generator, G, and that the fake data is mapped from noise, $\boldsymbol{z}$, we can represent synthetic data as $G(z)$ and update the discriminator's loss to:

$$\log(D(x)) + \log(1 - D(G(z))) \tag{2.2}$$

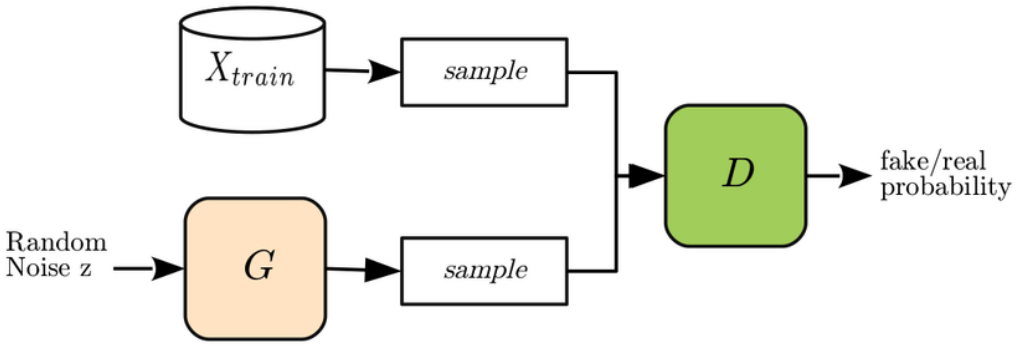

Figure 2.1: The classical GAN diagram. Taken from [20].

During training the discriminator maximizes this loss, while the generator attempts to minimize the second part of the loss, $\log(1 - D(G(z)))$. This means that the discriminator attempts to maximize the probability of assigning the correct label to real and synthetic samples, while the generator tries to minimize the probability of the discriminator assigning the correct label to fake examples.

This first GAN was tested on the Modified National Institute of Standards and Technology (MNIST), Toronto Face Database (TFD) and CIFAR-10 datasets. In Figure 2.1 there is a diagram of this classic GAN.

2.2 GAN Variants

Since the introduction to GAN, many new models have been presented. These models often introduce either a change in the GAN's architecture, or a change to the training of the model (generally modifying the model's objective function). Following are some of these models that have been relevant to the development of this work.

2.2.1 Conditional GAN

Shortly after the GAN was presented by Ian Goodfellow, Mirza and Osinero proposed a new GAN model, the Conditional GAN (CGAN) [21]. This model introduces the possibility to control the data generated. This means that for multi-class problems it is possible to make a GAN create data for the classes we want. To do this, the authors follow a suggestion left by Ian Goodfellow at the end of his paper [3] on how to create this conditional model, which is to add an extra information $\boldsymbol{c}$ to the generator and discriminator models of the GAN. $\boldsymbol{c}$ can be, for example, labels of the classes of the data. This model's architecture can be seen in Figure 2.2. To test their approach, the authors used the MNIST dataset and a collection of images from the Flickr website, using user tags as labels. More recently, a new dataset based on human face images from Flickr was created, and has been used as a benchmark for GANs, called Flickr Faces HQ (FFHQ).

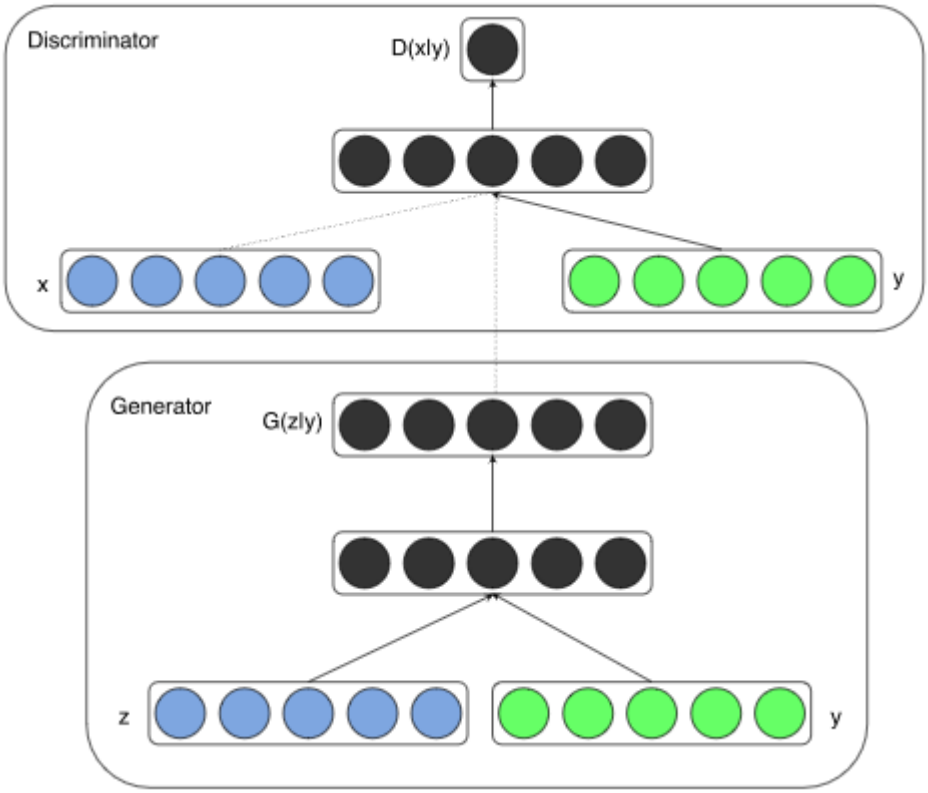

Figure 2.2: Conditional GAN architecture. Taken from [21].

2.2.2 Auxiliary Classifier GAN

The AC-GAN [22], like the conditional GAN, uses extra information to condition the behavior of the generator. The discriminator though, unlike in the conditional GAN, is forced to classify the class of the data, as well as if that data is real or not. The idea of turning the discriminator into a discriminator and a classifier at the same time makes this model very interesting, and this property will be crucial for the work done in this dissertation. The architecture for this model is depicted in Figure 2.3. This means that there is a new loss to consider for this new classification goal, which is a simple cross-entropy loss.

This model was tested on the ImageNet and CIFAR-10 [23] datasets to produce 128x128 images and 32x32 images. The authors show that the higher resolution images have more complete and distinguishable class information, yet higher resolution images have their own issues, like higher memory requirements and a higher amount of features to learn, which can difficult the generator's training.

Moreover, this paper approaches an important issue in GANs: data variability. A data generator holds little interest if it can only generate a single piece of data, *i.e.* the same data every time. It is important that the generator can create data from the entire distribution of the original data. Commonly, it was believed that to achieve more accurate data, variability was sacrificed in GANs. This paper, though, presented results that suggested otherwise. In their experiments, the authors trained 100 models, each one on 10 different classes of the ImageNet dataset, and the models that had the lowest diversity of images generated where the

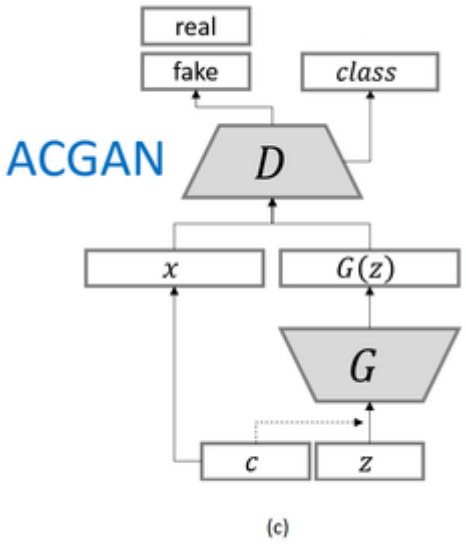

Figure 2.3: AC-GAN architecture. Adapted from [25].

ones that had the least similar to real images. The opposite was true as well, *i.e.* models with higher diversity produced better images. Image quality was evaluated using the Inception accuracy [24] and diversity was measured with a method proposed by the authors themselves, called Multi-Scale Structural Similarity (MS-SIM).

2.2.3 BigGAN

The BigGAN [6] was presented in 2018 and was, at that time, the generative model achieving the best IS and FID for the ImageNet [26] dataset at a resolution of 128x128. Additionally, it was used to generate images of up to 512x512 resolution. It is based on the Self-Attention GAN (SA-GAN) [27] architecture, which has 3 main architectural aspects: an architecture based on the ResNet model [28], the use of Spectral Normalization [29], which has the goal to normalize the spectral norm of the weight matrix of a layer, and a self-attention mechanism, inspired by the non-local model of Wang *et al.* [30], that allows networks to model long range dependencies in images, *i.e.*, the relationship between distant parts of an image.

The self-attention mechanism consists of running 3 different 1x1 convolutions through the result of a previous layer, **x**, creating 3 new feature maps: **f**, **g** and **h**. **f** and **g** are multiplied and the result of that is passed through a softmax activation, producing an attention-map. This attention-map is then multiplied with **h** and the result of this second multiplication is used on another 1x1 convolution, producing **v**. Finally, **v** is added with the input **x**. Figure 2.4 illustrates this process (the addition at the end of the process is not displayed in the image).

As already mentioned, a ResNet-like GAN is used to build the BigGAN. Figure 2.5 shows the general layout of a BigGAN generator and the composition of the residual blocks that make the generator and discriminator. Class conditioning is applied to the generator using an embedding that is concatenated to the latent space z and then used in all residual blocks via conditional Batch Normalization layers, following on the work of Dumoulin[31] and de Vries

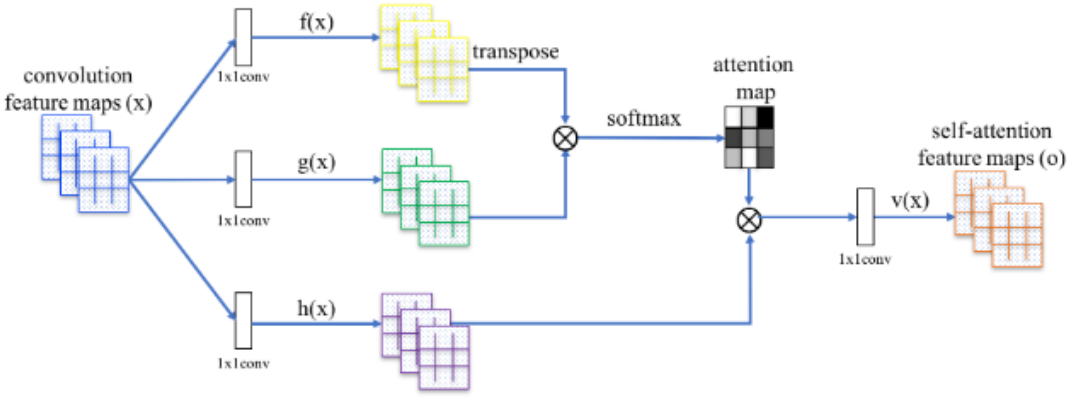

Figure 2.4: Self-Attention mechanism. Taken from [27].

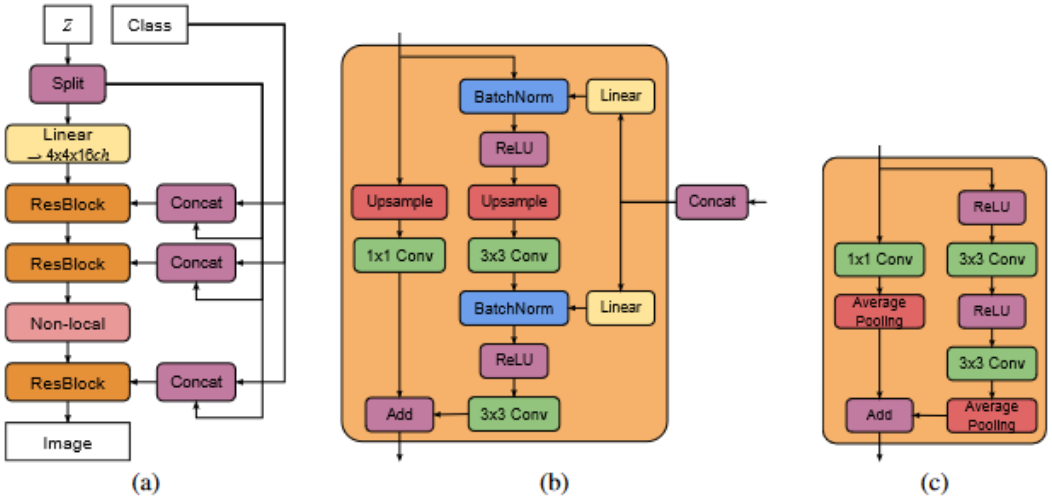

Figure 2.5: (a) Typical layout for BigGAN generator. (b) A residual block for the generator. (c) A residual block for the discriminator. Taken from [6].

[32]. This way, the class information propagated to the later layers of the network too, which is referred to as *skip-z* connections. As for the discriminator, the class information is applied using an embedding as well that is added to the final linear layer responsible for classifying data as real or fake.

All layers in the GAN are initialized using Orthogonal initialization, as opposed to the typical Xavier initialization or a normal distribution. The Adam optimizer is used with $\beta_1 = 0$ and $\beta_2 = 0.999$. The learning rates used were $2 \cdot 10^{-4}$ for the discriminator and $5 \cdot 10^{-5}$ for the generator in models generating images with a resolution of 128x128 and $2.5 \cdot 10^{-5}$ for the discriminator and generator in models generating images with 256x256 and 512x512 resolutions. The discriminator is updated twice before updating the generator. The loss used for the optimization problem is the Hinge loss for GANs, which was the same as the one used for the SA-GAN model.

Two other really simple techniques that improved the results obtained were to increase the batch size during training up to 2048, which resulted in a 46% increase in IS and to increase

the number of channels in each layer by 50%, which resulted in 21% increase in IS. This is directly related to the goal of exploring *"the performance benefits of larger models and larger batches"*.

2.2.4 Progressive Growing GAN

Generating high resolution data, and in particular images, is a big challenge for GANs. There are many things to learn, from the small details, to the general shape images, or data, should have. Progressive growing GANs [33] attend this issue by starting the training process with a generator and discriminator of low resolutions, like 4x4 for images, and progressively increasing that resolution. This means that the generator will start by learning hot to reproduce the large-scale structure of the data and only then learn the smaller details. By doing so, the difficulty of the task at the hands of the generator is always much simpler. This is, in fact, very similar to what humans themselves do: when facing hard problems we break them down into steps/phases and progressively improve our solution to attend the different new issues of the problem.

During the training process all layers in the generator and discriminator are trainable. To avoid sudden abrupt changes in already trained layers when increasing resolution, the new layers are added in a smooth way. This is done with a weight, α, that begins at 0 and grows to 1 as training progresses. While the new, higher resolution, layer's weight is α, the smaller resolution layer's weight is $1 - \alpha$. Since these two layers will have different output/input sizes, nearest neighbour interpolation and average pooling are used, respectively, to increase the output of the previous output layer of the generator and to diminish the input size at the discriminator (again, for the previous layer), until α becomes 1. This process is depicted in Figure 2.6.

As already mentioned, data variability is very relevant for GANs. To improve the variability of their generator, the authors of the progressive growing GAN add a new layer to the end part of the discriminator. This layer is built by computing the standard deviation of each

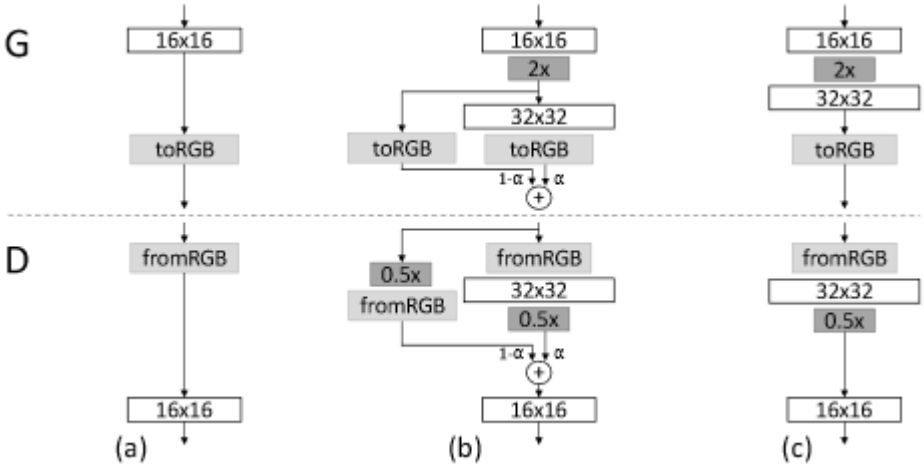

Figure 2.6: The smooth introduction of new, higher resolution, layers. Taken from [33].

feature over all examples in a batch of data and then averaging everything into a single value. That value is replicated for every feature, creating the mentioned layer.

This model was tested on the CIFAR-10, LSUN (on the bedroom class) and CelebA datasets. To test the model on high resolution settings, the authors created an improved version of the CelebA dataset, CelebA-HQ, containing 30000 samples of 1024x1024 images of human faces.

2.2.5 Style GAN

The Style GAN, by Karras *et al.* [34], is an approach to improve the quality of the results of GANs heavily focused on the generator instead of the discriminator, which had been done in multiple other papers, with multiple different approaches like self-attention [27] or the use of multiple discriminators [35].

In traditional approaches the generator's input is a latent code, $\boldsymbol{z}$. The Style GAN changes this by using $\boldsymbol{z}$ as the input to a 8 layer, fully connected, Multi Layer Perceptron (MLP) which outputs a new code, $\boldsymbol{w}$. Instead of using $\boldsymbol{w}$ as the input for the generator, the generator has a fixed input, and $\boldsymbol{w}$ is modified into a *style* with use of affine transformations, and that style will then control a new normalization technique introduced in the paper, called Adaptive Instance Normalization (AdaIN). AdaIN normalizes feature maps into standard scores and then scales and biases that result based on the style information. This normalization is applied at the end of every convolution. Additionally, noise is introduced, right before this normalization, in the form of a one channel image of Gaussian noise. This helps improve the diversity of the data created. This is depicted in Figure 2.7. One final aspect developed in Style GAN is style mixing. This means that two latent codes, z_1, z_2, are used instead of just one. The point in the network at which the second code starts being used is selected randomly. The authors reveal very interesting results of this feature and show the impact that the second latent code has on the resulting images based on what point in the network it begins being used. When used in the earlier stages of the generator (smaller resolutions), it makes it so that the high-level aspects of the data are affected by z_2 (for example, for human faces generation, high-level aspects are the pose of the head, face shape or general hair style) and, when applied further into the generator, the smaller details (in human face generation, small details could be small parts of the hair or how open/closed the eyes are) are what is impacted by z_2.

This method was tested on the CelebA-HQ and the FFHQ datasets. The FFHQ dataset was created by the authors and is a collection of images of human faces from the Flickr website at a resolution of 1024x1024.

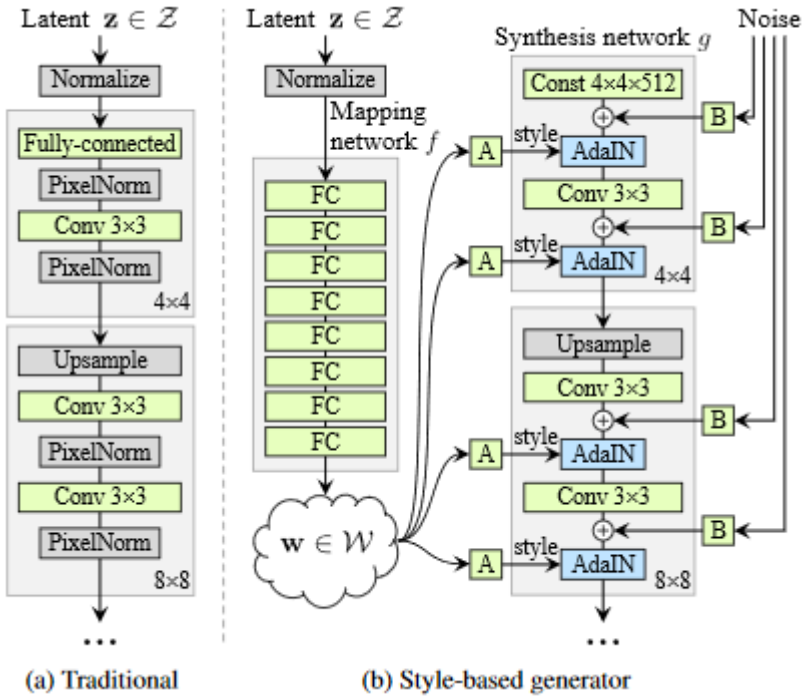

Figure 2.7: Style GAN architecture. Taken from [34].

Style GAN 2

Karras *et al.* further improved the quality of the Style GAN, creating Style GAN 2 [8]. This new model redesigned the generator normalization and the concept of progressive growing. Moreover, biases and noise are now applied after a convolution has been normalized, instead of before.

The reason why normalization was changed was because the scaling of weights based on the style applied could amplify weights by one order of magnitude or more, which would hurt subsequent layers. To solve this, after scaling weights based on the style being applied, weights are again scaled by a factor of $1/\sigma$, where σ is the standard deviation of the output of a convolution. This second scaling operation was called demodulation and it helps removing artifacts.

The progressive growing aspect of the previous model was associated with some artifacts. However, the fact that the generator could focus on high-level details when on low resolutions and only then on smaller details was too good of a property too just abandon. The solution was to employ a generator with skip-connections (inspired by the MSG-GAN [36]) and a residual discriminator (inspired by residual networks [28]). The networks used now do not grow in size as train progresses, however the skip-connections allow the generator to output lower resolution data, which allows the generator to still be able to focus on high-level details before anything else.

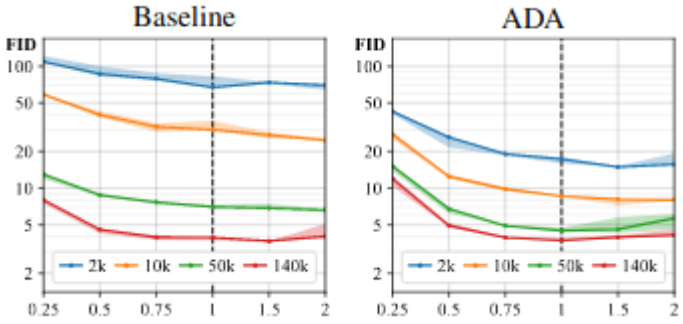

Figure 2.8: Impact of the ADA technique. Taken from [7].

Style GAN 2 ADA

Machine learning algorithms learn from the data they use and typically the more the better. Training models with small amounts of data can cause models to overfit, and GANs are no exception. The Style GAN 2 model was further improved with a technique called Adaptive Discriminator Augmentation (ADA) [7] to attend this issue. A series of possible augmentations, such as pixel blitting (x-flips, 90^o rotations, translations), geometric transformations and color transformations are used. The advantage of using augmentations is that they can prevent the discriminator from overfitting to the training examples (which would make the feedback used from the discriminator to train the generator quite meaningless). Whenever training the discriminator or generator all the data used is transformed according to these operations. They are selected randomly and applied given a probability p. The reason why the method is called adaptive is because the value of p that controls the probability of augmentations happening, is variable depending on how much the discriminator is overfitting. This is measured using a metric created by the authors, following their observations that when overfitting on the training data the discriminator then behaves on the validation data similarly to how it behaves on the generated data (this discriminator output cannot go over the output of the training set neither the output of the generated set). Using $E[.]$ as the mean of the discriminator outputs and $D_{train}, D_{val} and D_{gen}$ as the outputs of the generator for the training, validation and generated data, respectively, the overfitting is measured like so:

$$r = \frac{E[D_{train}] - E[D_{val}]}{E[D_{train}] - E[D_{gen}]} \tag{2.3}$$

Knowing that when the validation data outputs are similar to generated data outputs we have overfit, we can see that for values of r near 1 there is overfit and when r is close to 0 there is no overfit.

This method allowed to achieve similar results to those obtained with the Style GAN 2 on large datasets with limited datasets, of less than 10 thousand images. Figure 2.8 displays the impact of the ADA methodology. Quality of the model was measured using the FID metric (lower is better).

2.3 Data Augmentation and GANs

As already mentioned in Section 2.2.5 (concerning Style GAN 2 ADA), machine learning algorithms tend to overfit when trained on small amounts of data, especially deep ones. That same section is, in fact, about a data augmentation technique. Data augmentation techniques have been used in machine learning since the appearance of some of the first convolutional neural networks [37], such as LeNet-5 [38] and AlexNet [39]. They can bring variance into the data and help models generalize better. The techniques presented here will be focused on image data augmentation. Note, although, that data augmentation is a much broader subject and is used with more kinds of data, such as textual [40] and audible [41] data, used, for example, in Natural Language Processing (NLP) problems.

2.3.1 Basic Image Manipulation Augmentations

Shorten and Khoshgoftaar [37] separate basic image manipulation augmentations into 5 different types of augmentations: geometric transformations, color transformations, kernel filters, mixing images and random erasing. Since these methods focus on altering an image that belongs to the data, it is crucial that the resulting image preserves the label of the original image, because if it does not we may be introducing confusion to our model. As an example, if we rotate an image of a number 6 by 180^{o} it becomes a 9. This new, augmented, 9 would be considered, mistakenly, as a 6, since it is the result of manipulating an image with that label. However rotating circles or squares in a circle vs. square classification problem is not an issue. This means that the preservation of labels depends on the classification problem itself and it is up to developers to choose the fitting augmentations.

Geometric Transformations

Geometric transformations are perhaps the simplest to implement. They are composed of operations such as translations, flips, rotations, cropping and noise injection. These transformations are very good at handling positional bias in the training data, *i.e.* pixels that have the same value in most images. Some transformations, like the translation and cropping, may require special attention, because there may be no absolute rule to determine how much a given image can be cropped or moved.

Color Space Transformations

These transformations aim at making models tolerant to lighting and color changes. Possible changes to images are transforming images to grayscale, which results in faster computation with the drawback of resulting in less accurate models [42], applying color filters or transforming images to a different colorspace among RGB, HSV, YUV or CMY. This later concept was studied by Jurio *et al.* [43]. In some application these methods may not preserve labels, in the case that color conceals important information for the classification problem.

Kernel Filters

Kernels are n x n filters that are slid across an image. Depending on the values of the kernel it is possible to blur an image, extract its edges locations and highlight them, called

sharpening, or detect color change. Blurring can make the model tolerant to motion blur. Another interesting use of kernels is PatchShuffle regularization [44]. This method uses a kernel that slides across the image and shuffles the pixels inside of the kernel, with a certain probability. The authors of this method used a 2 x 2 kernel and a 5% chance of shuffling pixels and reduced the error rate on the CIFAR-10 dataset.

Mixing Images

Mixing images is a very unintuitive mechanism to create new images. There is is more than one way to do it. One technique is to average the pixels from 2 images and consider the resulting image as belonging to the label of the first. Another way to randomly crop parts of several images to create a new one. Both these methods were shown to be effective by Ionue [45] and Takashi & Matsubara [46], respectively.

Random Erasing

This technique grabs a random n x m section of the picture and removes it by giving it a constant, mean or random value or even by filling the empty part using a GAN to paint those pixels [47]. It was created to help with image recognition problems for the case when images had objects partly occluded. This method can suffer from not preserving labels.

2.3.2 GAN Augmentations

The use of GANs for data augmentations has become a quite popular subject in medical imaging problems, given how starved these problems are of data.

On a study with lung cancer images, from PET scans, Bi *et al.* [10] worked to understand the quality that a regular classifier can attain after being trained only with generated images from a GAN, when compared to an equal classifier trained on real data. They proposed their own GAN model, called the Multi-Channel GAN (M-GAN). The particularity of this GAN is that it has an embedding at the beginning of the generator for more than just labels. They used this embedding to feed both labels and CT images to the generator so it would learn how to produce PET images. CT and PET scans were from 50 PET-CT studies from 50 patients with lung cancer, that gave their dataset a total of 876 PET and CT images, with resolutions of 200x200 and 512x512, respectively. Despite not having done data augmentation with a

Authors	Task	Images
Bi *et al.* [10]	Lung cancer	PET
Frid-Adar *et al.* [15]	Liver lesion	Computed Tomography (CT)
Wu *et al.* [11]	Mammograms	Magnetic Resonance Imaging (MRI)
Bowles *et al.* [14]	Brain segmentation	MRI
Han *et al.* [17]	Brain tumor	MRI
Moradi *et al.* [48]	Cardiovascular anomalies	X-ray
Sandfort *et al.* [49]	Segmentation tasks	CT
Shin *et al.* [50]	Brain segmentation	MRI

Table 2.1: Overview of several works that use GAN data augmentations for medical imaging.

GAN, the competitive results they achieved with their classifier trained only on generated data had them suggest, as part of their final remarks, that using a GAN to augment the real dataset would be a highly likely way to improve the classifier.

Frid-Adar *et al.* [15] worked with a very small dataset of CT images of liver lesions. The dataset had a total of 182 samples: 53 cysts, 64 metastases and 65 hemangiomas. They increased the dataset through traditional data augmentations (translations, scaling, flips and rotations) to train a CNN that achieved 78.6% sensitivity and 88.4% specificity. They, then, further increased the dataset with images generated by a Deep Convolutional GAN (DCGAN) and were able to improve both sensitivity and specificity to 85.7% and 92.4%, respectively. Furthermore, authors studied the limits of data augmentation: for both classical and GAN augmentations, the authors evaluated until what amount of increase in images their classifier improved. The quality of samples was evaluated by two expert radiologists. The experiment that they were a part of did not conform with their normal working environment, still both specialists had the same results, regarding the amount of images that were miss-classified as real or fake.

A similar study to the previously mentioned one was made with mammograms, by Wu *et al.*. The authors were able to improve the Area Under Curve (AUC) of a ResNet-50 classifier by performing data augmentation with a GAN they called the Conditional Infilling GAN (ciGAN). Knowing that GANs had more trouble when generating higher resolution images, the authors decided to use infilling to simplify the problem. The generator was not actually trained to create full images, but instead to complete images with gaps (making the resolution of what the generator had to output smaller), and fill in the gaps with either a lesion or healthy tissue. The classifier, trained with the dataset with no augmentations had an AUC of 0.882, with traditional augmentations had an AUC of 0.887, and by adding generated images on top of the traditional augmentations the classifier achieved an AUC of 0.896.

There are other medical areas were this type of work has already found an interest. Table 2.1 shows some other references of similar work to the ones described above.

GAN Augmentations for Skin Lesion

Another medical imaging area where the interest for GANs has been rising is skin lesion. This area has a particular interest in this work, since it is an area where this work will focus on. One of the things that has been pushing these studies forward is the existence of public datasets, like the ones that the International Skin Imaging Collaboration (ISIC) publishes every year for their challenges. Table2.2 shows several papers that used GANs to perform data augmentation in skin lesion problems. Notice how most papers used an ISIC dataset for their work. Other datasets mentioned here include the Dermoscopic Image Library (DIL) [51] and PH^2 [52].

Bissoto *et al.* [13] worked with three different models: the DCGAN, a conditional version of the Progressive Growing GAN (PGAN) and the pix2pixHD GAN [57] [58]. The pix2pixHD GAN is an image-to-image translation model, which means that it does not produce images from a random vector of numbers (latent space), but instead from other images. The pix2pixHD

Authors	Dataset
Yi *et al.* [18]	ISIC2016+PH^2
Bissoto *et al.* [13]	ISIC2017+ISIC2018+PH^2+DIL
Bisla *et al.* [53]	ISIC2017+PH^2
Rashid *et al.* [19]	ISIC2018
Ali *et al.* [12]	ISIC2018
Ghorbani *et al.* [54]	Custom [55]
Qin *et al.* [56]	ISIC2018

Table 2.2: Overview of several works that use GAN data augmentations for the skin lesion problem.

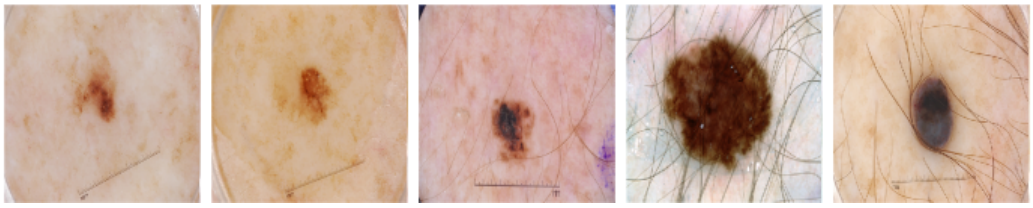

Figure 2.9: Examples of occlusions in skin lesion images, like ruler marks and hair. Taken from [53].

GAN model was trained on the ISIC 2018 dataset, while the other two models where trained on images form several datasets: the ISIC 2017, the ISIC Archive, the DIL and the PH^2. After training all models, the authors trained the Inception-v4 model, which was the top performing model on the ISIC 2017 challenge, on the same data as the DCGAN and PGAN to analyse the impact of performing data augmentation with the GAN models trained before. An improvement of 1.3% on the AUC was achieved when applying data augmentation with both the PGAN and pix2pixHD GAN, with each GAN producing an amount of images equal to the size of the original training dataset.

Bisla *et al.* [53] developed a two parted system to improve a ResNet-50 classifier pre-trained on the ImageNet dataset. The two parts consist in, first, purifying images and then generating images, both with classical data augmentation techniques and with GANs. Purifying images consists in removing things like hair and ruler marks, that are common appearances in skin lesion datasets (Figure 2.9 shows some examples). Their method to remove these occlusions is based on the hair removal algorithm of Saugeon *et al.* [59]. Images were generated at resolutions of 256x256 with a DCGAN. The images of the dataset on which the classifier was trained were then purified and after that the dataset was augmented with vertical flips and random cropping and with the use of the DCGAN developed. Over a quarter of the images in the final dataset (26%) were created by the GAN. Results showed that the combination of image purification and data augmentation with traditional techniques and a GAN improved the performance of the ResNet-50 classifier from an AUC of 0.873 to 0.915, which outperformed the winning model of the ISIC 2017 challenge.

Ali *et al.* [12] proposed a model named APGAN+TTUR to boost the performance of a classifier with the use of data augmentation via a GAN. Their GAN makes use of three main concepts. First, it is based on the PGAN of Karras [33], which allowed the generation of images with very high resolutions and very good quality. Second, it uses the Self-Attention

mechanism, which is used in several areas of machine learning, that gives models the ability to comprehend long range dependencies in the data. And third, it uses a technique called Two Time Update Rule (TTUR) which is a stabilization technique that is as simple as training, generally, the discriminator multiple times per generator step. For their work, though, the authors decided instead to increase the discriminator learning rate by 5 times, compared to the one used on the generator, while maintaining a 1:1 ratio on training steps between both parts of the GAN. The dataset used was the ISIC 2018 and it was augmented with traditional techniques like skewing, scaling and flipping for the GAN training. The classifier used was the ResNet-18, pre-trained on the ImageNet dataset. To test their model, the authors selected 100 real images from each class of the ISIC 2018 and augmented them up to 1000 either with their own GAN model or with the traditional techniques mentioned before. The use of standard augmentation improved the classifier's accuracy by 1.4% and by using images from the APGAN+TTUR they were able to improve the classifier by 2.8%.

Ghorbani *et al.* [54] worked with a custom dataset, provided a teledermatology service existent in some states of the United States of America. The dataset is composed of almost 50000 images, belonging to 26 different classes of skin lesions. Like other datasets for skin lesion, this one suffers heavily from data imbalance too, with some classes like melanoma having less than 200 images, in the midst of 50000. The dataset is presented/explained in detail in [55]. The GAN used in their work is based on the pix2pix GAN, however the authors made three adaptations: they reduced checkerboard effects, used a condition specific loss and a feature matching loss. The resulting model was called DermGAN. The GAN was used to generate 20000 images to increase the dataset. A MobileNet [60] classifier was trained on both the base dataset and the augmented one, and the results showed an improvement on the classification accuracy from 49.6% to 56%, with the largest increases in accuracy, class wise, being on the least represented classes.

Qin *et al.* [56] proposed the Skin Lesion StyleGAN (SL-StyleGAN) to produce skin lesion images. It changed, comparatively to the orignal StyleGAN by Karras *et al.* [34], how the random noise fed to the generator is processed and the progressive growth system. The model was built to produce images with resolutions of 256x256. When trained on the ISIC 2018, the proposed SL-StyleGAN performed better than the original StyleGAN, based on the FID metric. Images produced seemed to be more realistic, diverse and contained less artifacts. To test the impact of their GAN for data augmentation, authors trained a ResNet-50, pretrained on the ImageNet dataset, on the ISIC 2018 dataset. The GAN trained before was used to produce 800 melanomas that led the ResNet-50 model to improve its Balanced Multi-Class Accuracy (BMA) from 80.4% to 83.2%.

There are two other studies of particular interest for this dissertation. All studies presented here have made efforts in the direction of improving classifiers for skin lesion with the use of GANs for data augmentation. Still, maybe, GANs can be used to improve classifiers in a different way. The work of Yi *et al.* [18] and Rashid *et al.* [19] take a different approach on how to improve classifiers with GANs and instead of using to perform data augmentation, they use GANs to make the classification themselves. This is not very far from the previous

data augmentation techniques, since the discriminator does train on generated data and will be used as a classifier. This type of process will be a subject of study in this dissertation as well.

Yi *et al.* [18] suggested the combination of a Categorical GAN with a Wasserstein GAN (WGAN) to perform the classification of skin lesion with the ISIC 2016 dataset in a unsupervised/semi-supervised setting. The proposed GAN is composed of one generator and two discriminators. The first discriminator is like the one in the Categorical GAN, which, instead of outputting if images are real or not, outputs the likelihood of the input to belong to any class of the problem. The second discriminator is equal to the one of the WGAN, which is a variation of the GAN that modifies the learning objective to one based on the Wasserstein distance. The inclusion of this second discriminator served the purpose of stabilizing the training of the Categorical GAN. Both the discriminators provide feedback to the generator. The model was trained on the ISIC 2016 dataset, while only using 140 labeled samples, and achieved an average precision of 42.4%.

Rashid *et al.* [19] used a GAN that was very similar to the AC-GAN [22], despite that work not being mentioned. It is just like a normal GAN where the discriminator has 2 outputs: one to tell if images are real or nor, and another to tell the class the images belong to. For each output there is a loss. The output responsible with identifying images as real or fake follows the same objective as the original GAN. The output responsible with classifying images is associated with a standard cross-entropy loss. The model was trained on the ISIC 2018 dataset and was built to produce images with a resolution of 224x224, and, in this case, to classify said images too. Then, a DenseNet and a ResNet-50 were trained on the ISIC 2018 dataset. For all the models, the dataset was increased with traditional data augmentation techniques. The models were compared based on the balanced accuracy score, and the GAN developed was the best classifier of the three, beating the DenseNet by over 5%, which is very considerable.

2.4 Quality metrics for GANs

It is, naturally, very important to have ways to evaluate models, so that models can be compared to each other. There are some very common metrics in machine learning, like accuracy, recall, precision or the AUC. These metrics, though, were designed to evaluate models that were trained to interpret a certain kind of data. However, GANs are commonly used with another purpose, which is to create data, and not interpret it. So, for GANs, these metrics are a not a good way to evaluate and compare models. To compare GANs with each other, since their goal is to generate data, it is necessary to use metrics that evaluate the quality of the data produced.

One simple way to evaluate the quality of fake data is to use ourselves. However, humans are biased and subjective, so while it would, maybe, be possible to tell models that generate good data from those that do not, it would still be very hard to reach consensus on which models are the best out of the better ones. This approach would probably be very costly, especially in problems that would require specialized opinions.

Today, there are two metrics that are extensively used to evaluate GAN models. They are the IS and the FID. which will be explained ahead.

2.4.1 Inception Score (IS)

The IS was introduced in 2016, by Salimans *et al.* [61]. It uses the Inception network, which is a classifier that takes in images and outputs a probability distribution, *i.e.*, a number between 0 and 1 for each class that represents how much the Inception network believes the given image belongs to that class. Since it is a probability distribution, the numbers of all classes sum to a total of 1.

The goal of the IS is to capture two properties from images: if the images are varied and if their contents look clear. If both of these conditions are verified the score should be high, and if one or both are missing the score should be low.

To do this, the method uses the probability distributions that the Inception returns. If the contents of the image are clear, than the Inception should have no issues in classifying it, and, so, the returned distribution should be narrow, meaning that one particular class has a high value, while others have a small one. However if there are several objects in the image, or if they are indistinguishable, the distribution will no longer be narrow, which means that there will be several classes with high values. This verifies if images have clear contents, which is one of the conditions stated above.

To verify if images have variety, the probability distributions that the model outputs for all images are summed together, which is a marginal distribution. If images are varied the marginal distribution will have multiple classes with a high value, corresponding to the different classes that those images belong to. If only a few or one class is being represented in the images, the marginal distribution will be narrow.

With a way to verify both conditions it is just necessary to find a method to combine them and turn them into a score. Since the label distribution should be narrow and the marginal distribution should be closer to uniform, these distributions should be very different from one another, in the case that images are of good quality. The method, then, proposed by the authors is to calculate the Kullback-Leibler (KL) divergence [62], which is a measure of how similar two distributions are. Results are then exponentiated so that they are easier to compare. For better images the score should be larger.

Mathematically, the IS corresponds to Equation 2.4:

$$IS = exp(\mathbb{E}_x KL(p(y|x)||p(y))) \tag{2.4}$$

where $p(y|x)$ is the label distribution and $p(y)$ is the marginal distribution, which is equal to $\int p(y|x = G(z))dz$, where G(z) is a generated image, from the latent vector z.

Still this scoring metric suffers from some issues. It is susceptible to the Inception model's weights, meaning that the randomness associated with the start of the training process of the model has an impact in the IS, even though that does not affect the performance of the Inception model, and its calculation could be improved, as shown by Barratt and Sharma [63].

Some other problems of the metric are associated with the data on which the Inception network was trained, which was the ImageNet dataset [64]. If the GAN under evaluation was trained on a dataset different than the ImageNet it is possible that there are no images of the same kind in the ImageNet, and that will cause the Inception model's predictions uncertain, creating the possibility for a low IS, regardless of the images generated being good. It is also possible that the Inception model can not "see" the variety in the images, for example, if one is generating images of different kinds of apples and in the ImageNet dataset there is no differentiation of apples, the Inception will see all images as apples and the marginal distribution will be narrow, even though the GAN may be generating different kinds of apples effectively. It would make sense to only use the IS when the Inception model and the GAN being evaluated where trained on the same data, however this may not always be convenient or even doable.

Other limitations are the fact that if a GAN only produces one image per class (no intra-class variety), from several classes, it can score highly, and the fact that this method does not account for overfit, since a GAN can score highly if it just copies real images.

2.4.2 Fréchet Inception Distance (FID)

The FID was introduced by Heusel [65], and is named after the Inception model and the Fréchet Distance, which is a method to measure similarity between curves that can be extended to distributions. In this case, the Fréchet Distance is used to measure the difference between real and generated images. To do this, a multivariate normal distribution is fit to both real and fake images, which is possible through the use of the Inception model.

This is what the Inception model is used for. A high number of samples from real and generated images are given to the Inception model to get the feature embeddings of each image, given by a layer of the model. Then a multivariate distribution is fit to each group of embeddings, **i.e.**, the mean and covariance matrix are calculated for the real and generated images' embeddings. Finally, it is just necessary to apply the Fréchet Distance formula, shown in Equation 2.5, where μ_r, Σ_r amd μ_f, Σ_f are the mean and covariance matrix of real and generated data. Since the measure is a distance between real and fake images, the smaller the better.

$$FID(r, f) = ||\mu_r - \mu_f||^2 + Tr(\Sigma_r + \Sigma_f - 2\sqrt{\Sigma_r \Sigma_f}) \tag{2.5}$$

This method can, unlike the IS, detect the lack of intra-class variety. Even so, the FID still has some issues. It requires a large amount to samples to be reliable, which can make it computationally slow, it only uses the first two moments of a distribution (the mean and covariance), which are not all the relevant statistics of a distribution, and it is susceptible to the features that the Inception model retrieves from images (there is no guarantee that the Inception model will retrieve the meaningful features of images, which will then, in turn, make the distributions used not representative of the images).

2.5 Summary

This chapter focused on making a review of previous work related to the one made in this dissertation. It began with an introduction to the ideas behind the first ever GAN and how that model works, and following that, several otherGANs were introduced. Some of those GANs will be of major importance, like the AC-GAN and the StyleGAN2-ADA, as they will be used later in this dissertation. Other GANs were discussed given their relevance in the creation of the two others mentioned before.

Then, an overview of data augmentation was made. That section looked at the traditional methods for data augmentation, like flips and mirrors, and afterwards it focused on a more recent and relevant (for this thesis) matter: the use of GANs for data augmentation. The review of the literature on this subject focused on the use of GANs for data augmentation in medical imaging problems, and then, more specifically, on skin lesion problems. Two studies, in that subject, are of special importance for this work, given their different approach on how to use GANs to improve classifiers, which is a big part of what is done later in this dissertation.

Finally, this chapter took a look at performance metrics for GANs. These models are quite different from conventional models, and the typical evaluation metrics do not apply to them. However it's still very important to be able to evaluate GANs in a quantitative manner. In this chapter, the two most broadly used metrics were introduced: the IS and the FID.

CHAPTER 3

AC-GAN for CIFAR-10

One very interesting feature of the AC-GAN is the fact that the discriminator does not just output its prediction on whether data is real or fake. An AC-GAN's discriminator predicts what the class of the data is as well. This chapter explores that feature, to understand how powerful the discriminator of an AC-GAN can be at classifying images. Two architectures are used for this, one based on the original AC-GAN and another based on the BigGAN, which was modified into an AC-GAN.

3.1 CIFAR-10 Dataset

The dataset chosen to train the models developed throughout this chapter was the CIFAR-10 dataset. It is composed of 50000 images for training and 10000 for testing, totaling 60000 images belonging to 10 different classes, 6000 for each class. Such classes are: airplane, automobile, bird, dog, deer, cat, frog, horse, ship and truck. Classes are mutually exclusive, meaning images belong to one class, and one class only. There is no overlapping between the classes automobile and truck. The automobile class only includes small cars, mostly sedans, and the truck class only includes big trucks. There are no pick up trucks in the dataset, which could belong to any or both classes.

The CIFAR-10 dataset presented itself a great candidate to be used in these experiments. One big factor that directs the decision of which dataset to use is the computational power available. At disposal, there were 4 NVIDIA®GeForce®RTX 2080 Ti GPU's. For reference, the longest experiments of this work, ran on one GPU, with the Auxiliary Classifier BigGAN training on the CIFAR-10 dataset, took over 12 hours to finish. Making experiments take longer to complete would injure the ability to explore and investigate with the models used, given the limited time setting under which this work was done.

Another factor that helped choosing the CIFAR-10 as the dataset to use was the fact that it is a broadly used dataset in GAN projects. Even though in more recent GAN projects the main goal is not to produce images equal to the CIFAR-10 ones, this dataset is still commonly used in testing, making it a very good comparison tool.

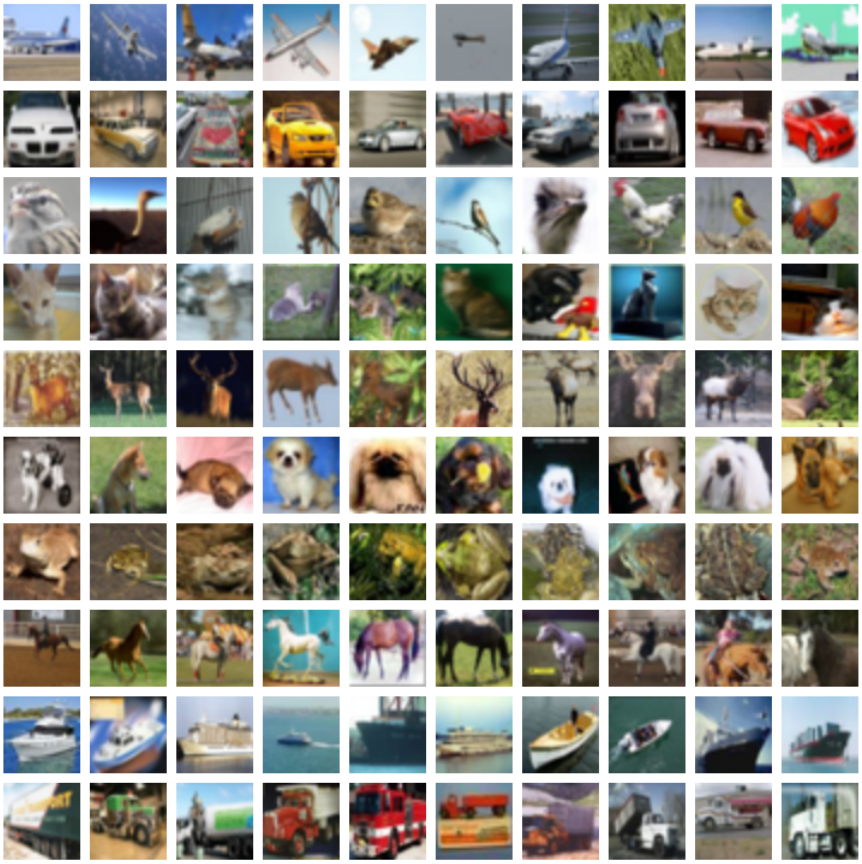

Figure 3.1: 100 images from the CIFAR-10 dataset. Each row has 10 images of the same class. Classes, from top to bottom rows, are: airplane, car, bird, cat, deer, dog, frog, horse, ship and truck.

Above it was mentioned that the dataset has 50000 training images and 10000 test images. Given that the dataset does not have a validation set, 5000 images (500 of each class) were taken from the training set to form a validation set. This leaves the dataset with 45000 training images, 5000 validation images and 10000 test images. Figure 3.1 shows samples of images from the CIFAR-10 dataset of every class.

3.2 AC-GAN

Auxiliary Classifier GANs still have not been explored much, since the release of the original AC-GAN. Apart from the original one, there are not many other architectures of GANs that adopt the Auxiliary Classifier technique. Two known AC-GAN architectures are the Twin Auxiliary Classifier GAN (TAC-GAN) [66] and the UAC-GAN [67]. According to the quality metrics reported in [67], these two different architectures are not too much better than the original AC-GAN if we compare those performance metrics with the ones achieved by state-of-the-art GANs. The FIDs of above 40 of the AC-GAN models mentioned are much higher (which means worse) than the FIDs of around 20 and below of the top GAN models.

Since there was no need to use all three of the models mentioned above, and since all three are close from each other, performance wise, the original AC-GAN was selected for being

the simplest and probably easiest to reproduce. Furthermore, its architectural layout is well detailed on the paper where it was presented. However, there were difficulties in recreating the AC-GAN and its results, which motivated changes, in order to obtain a stable model that was similar to it.

3.2.1 Replicating the AC-GAN

The AC-GAN's [22] architecture and hyper-parameters are well described in the paper were it was presented. One thing that is not as clear in the paper is how the input of the generator is handled. Note that the generator has two inputs, and they can not be simultaneously used as the input of the first layer of the generator. So somehow, they have to be joined, using, for example, arithmetic operations or a concatenation. The technique used was to pass the class input through an embedding layer and then performing the element-wise multiplication, known as Hadamard product, between the two inputs. An embedding layer transforms positive integers into vectors of the wished size. Transforming integers to vectors means that the output of an embedding has on extra dimension, so, in order to multiply the output of the embedding and the latent space, z, input, it is first necessary to flatten the output of the embedding layer.

Another important aspect is how the training was done. The discriminator was not trained more times than the generator and was trained on real and generated images separately. For every iteration, the discriminator was first trained with a batch of real images, followed by generated images and then the generator was trained. Training the discriminator is fairly simple: it is given images to make predictions on, *i.e.*, the indication if images are real or not, plus the class that they belong to. Then it is up to calculating the losses and gradients to move the weights.

The generator, however, updates based on the feedback of the discriminator. This means that, to train, the generator has to create images, feed them to the discriminator and update based on the discriminator's losses. For these losses to be correct it is necessary to, when training the generator, tell the discriminator that the generated images it is seeing are in fact real (when they are not). This will make it so that when the discriminator predicts that the images are real, the loss will be low (because the target of the prediction is that images are real), and that is exactly what the generator seeks. If the discriminator sees generated images and says that they are real, then the generator is tricking the discriminator, and the loss should, in fact, be low. The same line of thought can be made for when the discriminator predicts that the generated images are fake. Then, because the target of the predictions was that images were real, the loss will be big, which is exactly what the generator wants, since it was unable to fool the discriminator.

Remember that there are two discriminator outputs and, consequently, two loss functions: one source output (to predict the source of the image: real or fake), which has a loss resembling a binary cross-entropy; and a class output (to predict the class of the image), which can have either a binary cross-entropy loss or a categorical cross-entropy loss, depending on whether the problem is multi-class or not. To simplify, the losses shown here will assume a binary

class problem. The general loss for the discriminator is, then, the sum of the these two losses.

This summed loss is shown in Equation 3.1, where x is real data, the fake data, mapped from the noise z, is $G(z)$, D_s is the source output of the discriminator, D_c is the class output of the discriminator, c is the target class for a given image, and m is a sample from either real or fake data (can either be x or $G(z)$, whichever one applies).

$$[\log(D_s(x)) + \log(1 - D_s(G(z)))] + [c \log(D_c(m)) + (1 - c) \log(1 - D_c(m))] \tag{3.1}$$

As said above, though, the discriminator is trained on real and fake images separately, and so the source part from Equation 3.1 can be split to form two new losses, one for when training on real data, Equation 3.2, and another for when training on fake data, Equation 3.3. For the generator it was said that it needs to feed fake data to the discriminator, while at the same time telling the discriminator that such data is real (when it is not), and use the discriminator's loss to update it's own weights. This means that the loss of the generator is equal to the one of the discriminator on real data, with the exception that fake data is used, creating Equation 3.4.

$$[\log(D_s(x))] + [c \log(D_c(m)) + (1 - c) \log(1 - D_c(m))] \tag{3.2}$$

$$[\log(1 - D_s(G(z)))] + [c \log(D_c(m)) + (1 - c) \log(1 - D_c(m))] \tag{3.3}$$

$$[\log(D_s(G(z)))] + [c \log(D_c(m)) + (1 - c) \log(1 - D_c(m))] \tag{3.4}$$

To achieve this training schema, a combined model was built. The generator and discriminator were put together into one model, with the generator's output as the input of the discriminator and the losses being applied just like in any other case. Note, though, that if nothing is done, the gradients after each iteration will move the weights of the discriminator in this combined model, which is not correct, since the weights of the discriminator would move based on the opposite goal. So, the most crucial detail is to freeze the weights of the discriminator during the generator's training. Then, training the generator corresponds to training this combined model.

3.2.2 Training the AC-GAN Replica

Still, as mentioned before, there were problems when attempting to replicate such architecture, namely a collapse/failure to converge. In normal neural networks, failing to converge means that the loss is not dropping, however, with GANs, failure to converge refers to the inability of finding the equilibrium between the generator and discriminator, leading to one or both losses to drop to zero. Thinking about the training of GANs helps understand why losses dropping to zero is bad in GANs.

When training begins both models are bad: the generator outputs very bad images, and the discriminator does not yet know the difference between real and generated images. The

generator improves based on the feedback of the discriminator, so only when the discriminator can tell real images from fake ones will the generator improve. So the discriminator gets better at recognizing real and fake images, which will decrease it's loss. This will, in turn, increase the generator's loss. Now the generator will make larger updates to its weights, and will get better at producing images, making the discriminator less sure of which images are real or fake. This will make the discriminator's loss higher and the generator's smaller and now the discriminator is the one making larger updates and improving. This creates a cycle, and it is expectable that losses increase and decrease continuously, while remaining at around the same values while the models improve (this is exemplified in Figure 3.2, at the right).

Losses dropping to zero means that one or both of the models are performing badly, making the other very sure of what it is doing, independently of it being good or not. That is what is seen in Figure 3.2, at the left. The fact that models are not performing well is supported by the evolution of the FID score across training. The FID score measures how good generated images are when compared to the images of the dataset. The score gets lower as images get closer and closer to looking like the original ones. Because at the start the generator is unable to make decent images the score should be very high. Still, as training progresses the value should decrease. Figure 3.3 shows that the FID remained very high during the entire training, confirming that the generator performed very poorly from the start. The ultimate proof that the generator did not perform is the images that it generated, which made no sense and had no variability too, making it easy to identify what images are fake. Figure 3.4 shows 100 images created by the generator, 10 of each class of the CIFAR-10 dataset.

From looking at both graphs, in Figure 3.2, there is one loss that looks the same in both and that does not obey the idea of remaining with a similar value during the whole training. The loss of the discriminator on real images, even though it wiggled up and down during the entire training, moved down steadily until stagnating, similar to how the loss on a normal neural network would progress.

In the AC-GAN there are two losses: one loss for the discrimination objective (whether or not images are real) and another for the classification objective. Each one of the three losses shown in the graphs of Figure 3.2 is the sum of those two losses. To understand why the loss of the discriminator on real images drops, one can look at that loss (the discriminator's loss on real images from the left graph of Figure 3.2) decomposed into its two objectives, shown in Figure 3.5, at the left. In the graph, it is visible that the discriminator's auxiliary (classification) loss drops, while the other remains stable, which is what causes the the drop on the summed loss. This seems natural, because the discriminator needs time to learn how to classify real images. It is worth noting that it is possible for the discrimination loss to rise in a smoother way (in Figure 3.5, at the left, there is a very abrupt increase), making the curve look symmetrical to the auxiliary loss one, and, in turn, causing the summed loss to be straight.

The reason why this does not happen with the other losses is because they refer to the usage of generated images. What happens, in those cases, is that since fake images are fairly

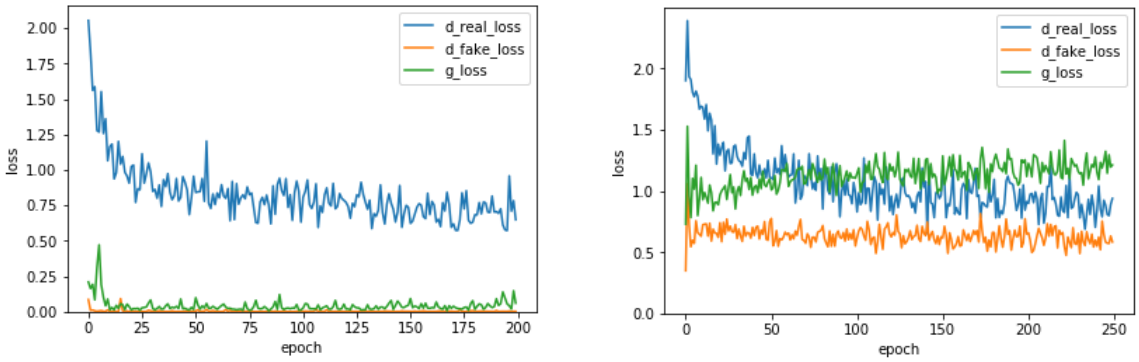

Figure 3.2: Example of failure to converge, at the left, and stable training, at the right.

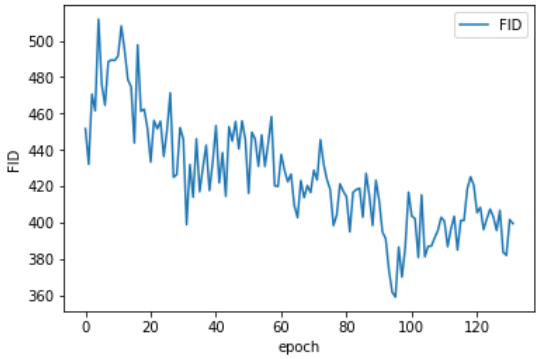

Figure 3.3: FID score evolution during training of AC-GAN that collapsed.

simple (and bad) at the start, the discriminator can easily, and quickly, learn how to classify them. Then, as the generator improves, so does the discriminator, so the classification fraction of the loss, when using fake images, is never high (it remains stable and low for the entire training). This is shown in Figure 3.5, at the right.

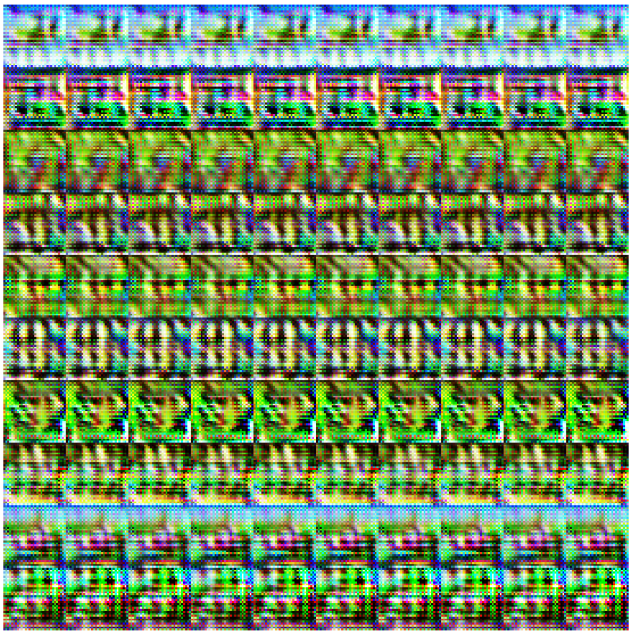

Figure 3.4: Images produced by the generator of an AC-GAN that collapsed. Each row has 10 images of one class of the CIFAR-10 dataset. Notice that, not only do the images make no sense, they are all equal when of the same class, which is very bad as well.

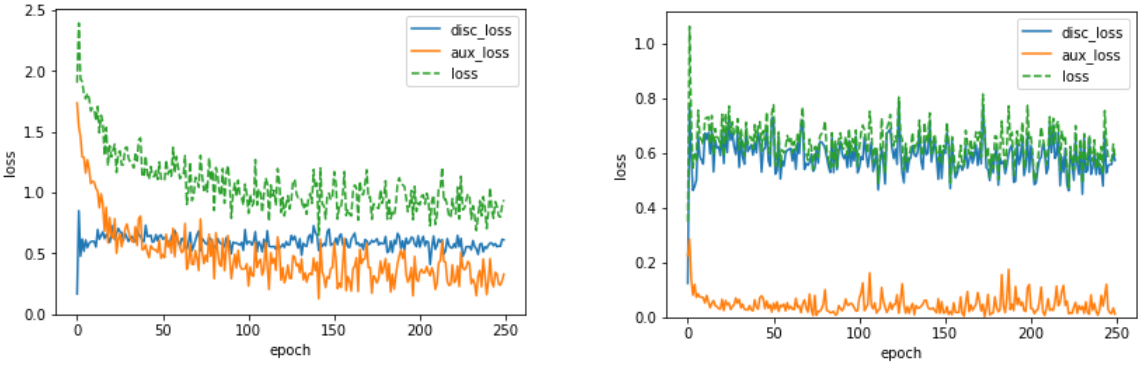

Figure 3.5: The loss of the discriminator on real (at the left) and generated (at the right) images decomposed into its two parts. *disc_loss* corresponds to the loss of the discrimination objective, *aux_loss* to the loss of the classification objective and *loss* is the summed loss. Notice how low the auxiliary loss is for generated images (at the right) and how that value remains at around the same value for the whole training, contrarily to what happens on the left image.

3.2.3 Modifying the Original AC-GAN

Discovering what lead to the attempt at reproducing the original AC-GAN fail would prove to be a arduous task, so, in order to achieve a stable model, the original architecture was changed. To mention the changes done, one should first take a more detailed look at the original AC-GAN's architecture. Table 3.1 shows this architecture. The slope for the Leaky ReLU activation is 0.2 and, for all layers, biases are initialized at 0 and weights are initialized according to an Isotropic Gaussian with $\mu = 0$ and $\sigma = 0.02$. Adam was the optimizer of choice, with $\beta_1 = 0.5$ and $\beta_2 = 0.999$. Learning rates were equal for the generator and discriminator and the values used were 1x10^{-4}, 2x10^{-4} and 3x10^{-4}. Table 3.2 shows the architectural changes made to the model.

Since training showed that the generator's abilities were not on a par with the discriminator's, and given that the discriminator is deeper than the generator, it was decided to remove the layers that had a stride of 1 in the original architecture. Because of having less layers now, another adaptation was made to the number of feature maps of the layers that remained. On top of this, batch normalization was entirely removed from the discriminator and kernel size was increased to 5 in convolution layers. As for the generator, changes include the use of the Leaky ReLU activation, with a slope of 0.2, just like in the discriminator, the introduction of batch normalization after the first layer and the reduction of the feature maps in each layer. Table 3.2 summarizes the architecture of this different model.

This model showed to be stable and trained well. It will be referred to as the baseline model, in the next Section.

Table 3.1: The original AC-GAN architecture used on the CIFAR-10. n is the number of classes in the dataset, 10 for the CIFAR-10 dataset. T stands for transpose and BN for Batch Normalization.
Generator on the left, Discriminator on the right.

z $\epsilon \mathbb{R}^{110}$ $\mathcal{N}(0, I)$
ReLU
Linear $\rightarrow$ 4x4x384
5x5 T Conv (Stride 2) $\rightarrow$ 192
BN, ReLU
5x5 T Conv (Stride 2) $\rightarrow$ 96
BN, ReLU
5x5 T Conv (Stride 2) $\rightarrow$ 3
Tanh

RGB image $x \epsilon \mathbb{R}^{32x32x3}$
3x3 Conv (Stride 2) $\rightarrow$ 16
Leaky ReLU, Dropout(0.5)
3x3 Conv (Stride 1) $\rightarrow$ 32
BN, Leaky ReLU, Dropout(0.5)
3x3 Conv (Stride 2) $\rightarrow$ 64
BN, Leaky ReLU, Dropout(0.5)
3x3 Conv (Stride 1) $\rightarrow$ 128
BN, Leaky ReLU, Dropout(0.5)
3x3 Conv (Stride 2) $\rightarrow$ 256
BN, Leaky ReLU, Dropout(0.5)
3x3 Conv (Stride 1) $\rightarrow$ 512
BN, Leaky ReLU, Dropout(0.5)
Linear $\rightarrow$ 1 | Linear $\rightarrow$ n

Table 3.2: The modified AC-GAN architecture for the CIFAR-10 dataset. n is the number of classes in the dataset, 10 for the CIFAR-10 dataset. T stands for transpose.
Generator on the left, Discriminator on the right.

z ϵ $\mathbb{R}^{110}$ $\mathcal{N}(0, I)$
Linear $\rightarrow$ 4x4x256
BN, **Leaky ReLU**
5x5 T Conv (Stride 2) $\rightarrow$ **128**
BN, **Leaky ReLU**
5x5 T Conv (Stride 2) $\rightarrow$ **64**
BN, **Leaky ReLU**
5x5 T Conv (Stride 2) $\rightarrow$ **3**
Tanh

RGB image x ϵ $\mathbb{R}^{32x32x3}$
5x5 Conv (Stride 2) $\rightarrow$ 64
Leaky ReLU, Dropout(0.5)
5x5 Conv **(Stride 2)** $\rightarrow$ 128
Leaky ReLU, Dropout(0.5)
5x5 Conv (Stride 2) $\rightarrow$ 256
Leaky ReLU, Dropout(0.5)
5x5 Conv **(Stride 2)** $\rightarrow$ 512
Leaky ReLU, Dropout(0.5)
Linear $\rightarrow$ 1 | Linear $\rightarrow$ n

3.2.4 Optimizing the Modified AC-GAN

After attaining a stable model, the baseline model, some attempts to optimize it were made. Changes to the model include: modifying the batch size (which was initially 64), increasing the number of feature maps of the generator (back to the same as in the original AC-GAN) and halving the leaning rate of the discriminator. Table 3.3 summarizes the experiments done and the performance of the models. By looking at the FID scores, it is visible that changing the learning rate of the discriminator had the biggest impact. Even the auxiliary accuracies generally increased when that change took place. The other two changes, however, seem to have had little impact, since models with the same learning rate on the discriminator have quite similar performances. There is, although, the exception of model number 7, which collapsed.

The model with the highest auxiliary classification accuracy, *i.e.*, its accuracy at predicting images' classes (dog, cat, car horse, etc.), will now be selected to take part in another experiment, to test its discriminator, and before doing so, it is a good idea to check the training progress of said model (model number 6), which has an auxiliary classification accuracy of 79.32%.

Model	D_LR	Batch Size	IS ↑	FID ↓	Aux Acc(%)
(1) Baseline	2x10^{-4}	64	5.45	66.07	76.96
(2) Baseline	2x10^{-4}	32	5.17	65.16	76.14
(3) Baseline	2x10^{-4}	128	5.26	60.31	77.6
(4) Baseline w/ +50% fmaps in G	2x10^{-4}	64	5.38	63.31	78.15
(5) Baseline	1x10^{-4}	64	5.75	46.98	78.96
(6) Baseline	1x10^{-4}	32	5.80	**45.67**	**79.32**
(7) Baseline	1x10^{-4}	128	3.48	126.97	78.02
(8) Baseline w/ +50% fmaps in G	1x10^{-4}	64	**6.03**	48.11	78.90

Table 3.3: Performance of different AC-GAN models. D_LR is the discriminator's learning rate. The generator's learning rate is 2x10^{-4} for all models.Aux Acc is the auxiliary classification accuracy of the discriminator.

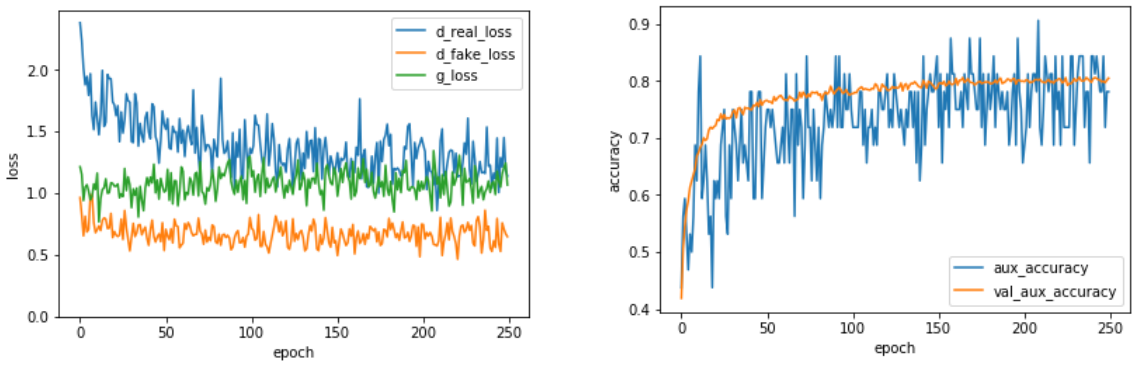

Figure 3.6: The losses of the best AC-GAN model achieved, at the left, and the auxiliary classification accuracy of its discriminator on real images from the training and validation sets, at the right.

The losses of this model, shown in Figure 3.6 (at the left) moved as expected from a stable model. There is a lot of jiggling, however losses remain stable for the whole training, with the exception of the loss of the discriminator on real images, which, again, has a noticeable drop at the start. These are both signs of a good training progress. Generally there is not much concern about the discriminator, however, given the scope of this chapter it is important to check if it is performing well and discard problems such as over and underfitting. At the right of Figure 3.6, the auxiliary classification accuracy of the discriminator on real data from the training and validation sets is shown. The fact that the accuracies are not both really low refutes the existence of underfitting, while the fact that the validation accuracy did not suddenly start decreasing midway trough training indicates that there is no overfitting. One interesting thing about these curves is that, while the train accuracy wiggles a lot, the validation one does not, which suggests that the confusion that the generator causes does not injure the discriminator's ability to classify real images. For the generator, it is a good idea to, just like what was done for the model that collapsed, take a look at the evolution of the FID during training. This is shown in Figure 3.7, which shows a curve that resembles the training of a normal neural network. The FID of the generator steadily decreased until stagnating, meaning that the quality of images progressively improved during training. There is a huge contrast between this graph and the graphs with losses of GAN's, and this is a very good way to visualize the progress of the GAN, since by looking at the losses it is impossible to tell how the model improved (the losses tell, mainly, whether the model collapsed or not). Finally, Figure 3.8 shows some samples of images generated by this model.

It would now be interesting to compare this modified model with the original AC-GAN, which, as reported in the original paper, achieved an IS of 8.25, on the CIFAR-10 dataset. The FID metric was not measured. It was already mentioned above that, in the experiments for the UAC-GAN, the FID of the AC-GAN was measured, and the reported value was 47.75. However, in that same work, the IS of the AC-GAN was measured, and the reported value for that was 4.71, which is different from the 8.25 reported in the original work. Regardless, the FID scores suggest that the AC-GAN created has a similar performance to the original one.

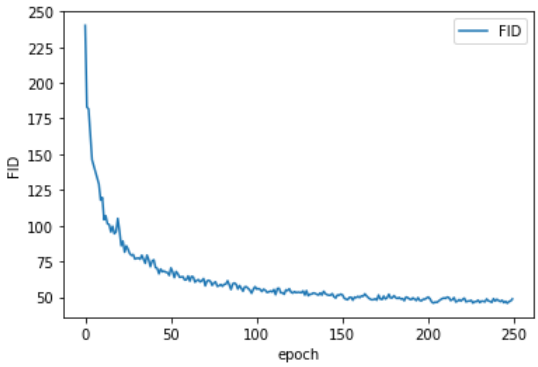

Figure 3.7: FID evolution across training for the best AC-GAN model achieved.

Figure 3.8: Samples of generated images of the best AC-GAN model achieved. Each row has 10 images of one of the 10 classes of the CIFAR-10 dataset. Classes are, from top to bottom row: airplane, car, bird, cat, deer, dog, frog, horse, ship, truck.

3.2.5 Scenarios Classification with the Modified AC-GAN

To analyse the abilities of the discriminator of an AC-GAN a three parted experiment was set up. Each of these parts will be referred to as a Scenario.

- Scenario 1 corresponds to training a normal classifier on the CIFAR-10 dataset.
- Scenario 2, the same classifier will be trained with the difference that the CIFAR-10 dataset will be augmented with images generated by a GAN, doubling the amount of images in the training set.
- Scenario 3, instead of using a classifier, an AC-GAN is trained and its discriminator is used to classify images of the CIFAR-10 dataset.

For comparison purposes, it is important that the classifier used in Scenarios 1 and 2 is as similar as possible to the discriminator of the AC-GAN trained in Scenario 3. The only difference between the classifier and the AC-GAN's discriminator should be the fact that the latter has two outputs, instead of one. The AC-GAN to be used, as already mentioned, is the one with the best auxiliary classification accuracy, from table 3.3.

Scenarios 1 and 3 are enough to see if there are improvements in classification caused by AC-GANs. Scenario 2, though, is also very important. Its purpose is to compare the use of AC-GANs with the use of data augmentation via GANs. This will help understand the impact of the AC-GAN as well as its usefulness. In this Scenario, generators from two GANs will be used: the generator from StyleGAN2-ADA, a state-of-the-art model, and the generator of the AC-GAN used in Scenario 3. The use of the same generator as the one in Scenario 3 will help understand if there are benefits in using an AC-GAN. If Scenario 2 with the AC-GAN's generator ends up producing the same, or better, result as Scenario 3, then it is clear that there is no benefit, so this is another important experiment. For convenience purposes, the Scenario 2 experiments with the StyleGAN2-ADA and the AC-GAN's generators will be simply referred to as Scenario 2 with StyleGAN2-ADA and Scenario 2 with AC-GAN.

The training progress of the classifier is shown in Figure 3.9 for Scenario 1, in Figure 3.10 for Scenario 2 with the StyleGAN2-ADA, and in Figure 3.11 for Scenario 2 with the AC-GAN. It does not seem like there is over or underfitting in any case, however one thing to note is how the gap between training and validation losses/accuracies varies in these experiments, caused by the change in the losses. The fact that the validation loss decreased in Scenario 2 with the StyleGAN2-ADA, comparing with Scenario 1, means that the generated images helped the classifier, while the fact the validation loss did not really change and the training loss decreased in Scenario 2 with the AC-GAN, comparing with Scenario 1, means that the images from the AC-GAN are still lackluster and potentially injured the classifier. This is confirmed by Table 3.4, which shows the accuracies obtained in each experiment. Confusion matrices for all Scenarios are shown in Figure 3.12.

From Gonçalves' work[9] it was already known that using GANs for data augmentation can lead to classification improvements, so the difference between Scenarios 1 and 2 with the StyleGAN2-ADA was somewhat expected. Scenario 3 seems to improve classification as well. The fact that it improved classification almost as much as Scenario 2 with the

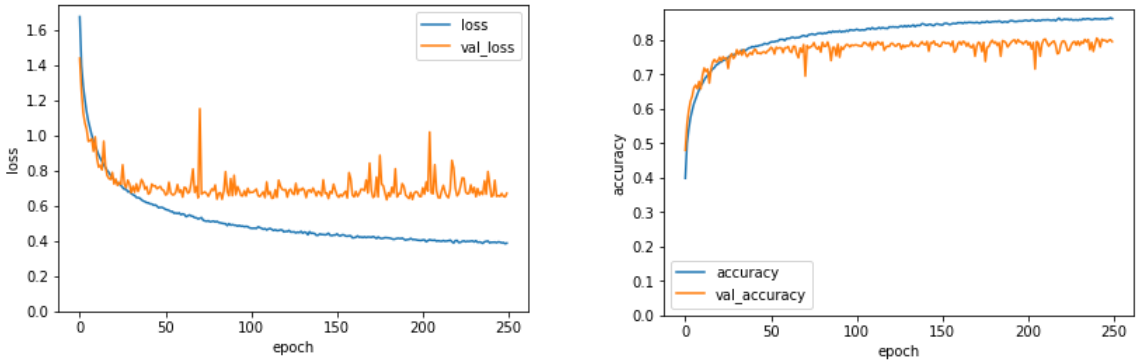

Figure 3.9: Loss along with validation loss (at the left) and accuracy along with validation accuracy (at the right) for Scenario 1.

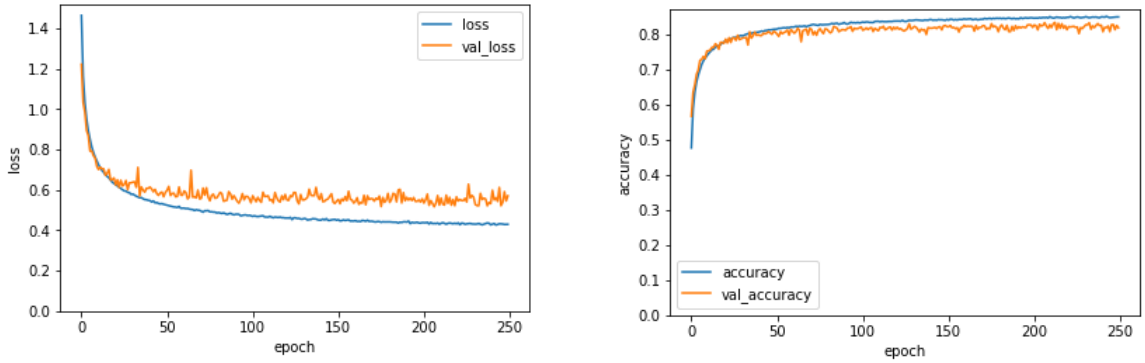

Figure 3.10: Loss along with validation loss (at the left) and accuracy along with validation accuracy (at the right) for Scenario 2 with the StyleGAN2-ADA.

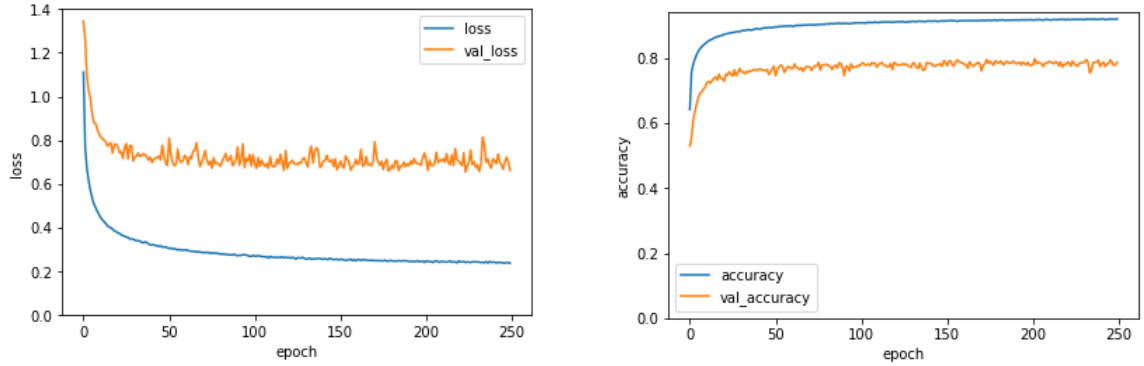

Figure 3.11: Loss along with validation loss (at the left) and accuracy along with validation accuracy (at the right) for Scenario 2 with the AC-GAN.

Scenario	Acc (%)	FID ↓
Scenario 1	78.53	N/A
Scenario 2 w/AC-GAN	77.31	45.67
Scenario 2 w/StyleGAN2-ADA	**79.57**	**2.42**
Scenario 3	79.32	45.67

Table 3.4: Accuracies obtained on the different Scenarios.

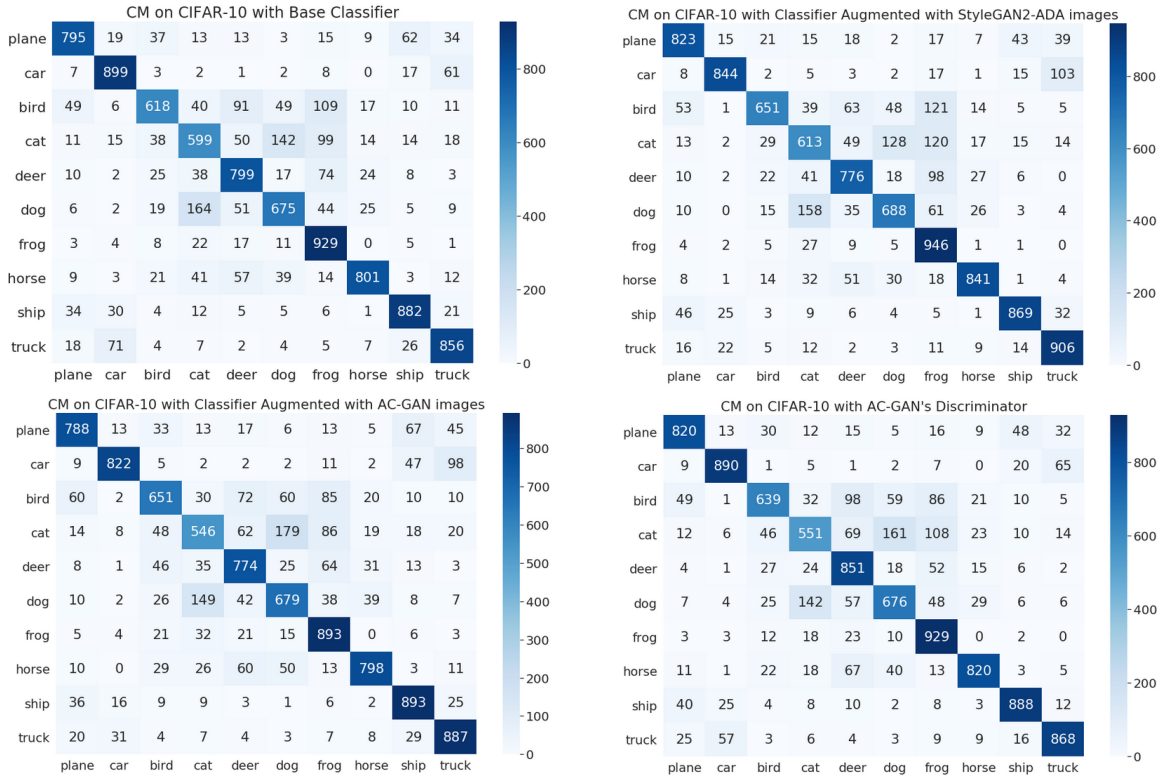

Figure 3.12: Confusion matrices for classifier/discriminator of all Scenarios. Top left: Scenario 1; Top right: Scenario 2 w/ Stylegan2-ADA; Bot left: Scenario 2 w/ AC-GAN; Bot right: Scenario 3;

StyleGAN2-ADA, in which data augmentation was done by using a very superior GAN, evidenced by the difference in FIDs, is very interesting, because of two things: first, for a good part of the training in Scenario 3, the generator is still learning how to create decent images, so the images fed to the discriminator are far from the real images and probably do not help the discriminator get better at its classification objective, and second, the best images that the generator can produce in Scenario 3 are still quite far from the ones that the StyleGAN2-ADA's generator can produce. The fact that worse images lead to a similar amount of classification improvement suggests that there may be benefits in using AC-GANs to obtain better classifiers. Moreover, Scenario 2 with the AC-GAN made the classifier worse. Note that, in that Scenario, the generator that produced images was the same as the one in Scenario 3, with the difference that it was already fully trained, and the classifier was the same as the discriminator in Scenario 3. The fact that from Scenario 2 with the AC-GAN to Scenario 3 the classifier went from worse than in Scenario 1 to as good as in Scenario 2 with the StyleGAN2-ADA further complements the idea that AC-GANs may have the ability to produce better classifiers than by using data augmentation with GANs.

3.3 BigGAN

This section will cover the same aspects as the previous one, with the exception that the model used will be based on the BigGAN. The BigGAN, however, is not capable of performing

the classification task with its discriminator so first it is necessary to transform it into an AC-GAN, which is the subject of the subsection ahead.

3.3.1 Auxiliary Classifier BigGAN

Transforming a conditional GAN into an AC-GAN requires only a few changes: the conditional input given to the discriminator is removed, a new output is inserted in the discriminator and the loss function is adapted to consider the GAN's new objective. These changes can, however, make a model unstable and have it collapse during training. Because of this, some adaptations were made to original BigGAN model to transform it into a stable AC-GAN.

The BigGAN was mainly used to generate images with resolution of 128x128 and above. The paper where it was presented shows the general layout of networks used to produce images of different resolutions (Table 3.5 shows the architecture for 128x128 images). However, even though the BigGAN was tested on images of the CIFAR-10 dataset with resolution 32x32, the architecture layout used was not included in the paper. In this work, the CIFAR-10 dataset was used, since it is computationally light while holding a bigger complexity and interest than datasets like the MNIST one. The architecture adopted follows the idea of what was done in the Spectral Normalization GAN (SN-GAN) [29], which was tested on the CIFAR-10 dataset. Table 3.6 shows the SN-GAN architecture and Table 3.7 shows the architecture for this Auxiliary Classifier version of the BigGAN for the CIFAR-10 dataset, the baseline model for this part of the work. Note that while both tables use the "ResBlock" notation, the residual blocks of the SN-GAN are different from the ones of the BigGAN and this Auxiliary Classifier version of the BigGAN. For reference, Figure 3.13 shows the residual block of the SN-GAN.

Other changes made include the hyper parameters of the Adam optimizer, which were changed to follow the ones used in the AC-GAN ($\beta_1 = 0.5$, $\beta_2 = 0.999$) with the difference that the learning rate of the discriminator was halved, making it $1 \cdot 10^{-4}$, while the generator's learning rate is $2 \cdot 10^{-4}$; the generator was initialized with a normal distribution with $\sigma = 0.05$ and the discriminator was initialized with the Xavier initialization, instead of the Orthogonal initialization for both networks; the introduction of dropout layers (with a probability of 50%) after ReLU activations in the discriminator's residual blocks; and the use of the binary cross-entropy, instead of the Hinge loss, and a categorical cross-entropy for the novel GAN objective (outputting class labels too, instead of just whether the image is fake or real) making the optimization objective equal to the one in the AC-GAN.

Table 3.5: BigGAN architecture for 128x128 images. *ch* is the channel width multiplier (multiplier to calculate output dimension of layers). Values for *ch* used were 64 and 96.
Generator on the left, Discriminator on the right.

$z \in \mathbb{R}^{120} \mathcal{N}(0, I)$ Embed(y) $\in \mathbb{R}^{128}$
Linear (20 + 128) $\rightarrow$ 4x4x16ch
ResBlock Up $\rightarrow 16ch$
ResBlock Up $\rightarrow 8ch$
ResBlock Up $\rightarrow 4ch$
ResBlock Up $\rightarrow 2ch$
Non-Local Block
ResBlock Up $\rightarrow 1ch$
BN, ReLU, 3x3 Conv $\rightarrow$ 3
Tanh

RGB image $x \in \mathbb{R}^{128x128x3}$
ResBlock Down $\rightarrow 1ch$
Non-Local Block
ResBlock Down $\rightarrow 2ch$
ResBlock Down $\rightarrow 4ch$
ResBlock Down $\rightarrow 8ch$
ResBlock Down $\rightarrow 16ch$
ResBlock $\rightarrow 16ch$
ReLU, GlobalSumPooling
Embed(y)$\cdot$h + (Linear $\rightarrow$ 1)

Table 3.6: Spectal Normalization GAN architecture for 32x32 images.
Generator on the left, Discriminator on the right.

$z \in \mathbb{R}^{128} \mathcal{N}(0, I)$
Linear $\rightarrow$ 4x4x256
ResBlock Up $\rightarrow$ 256
ResBlock Up $\rightarrow$ 256
ResBlock Up $\rightarrow$ 256
BN, ReLU, 3x3 Conv $\rightarrow$ 3
Tanh

RGB image $x \in \mathbb{R}^{32x32x3}$
ResBlock Down $\rightarrow$ 128
ResBlock Down $\rightarrow$ 128
ResBlock $\rightarrow$ 128
ResBlock $\rightarrow$ 128
ReLU, GlobalSumPooling
Linear $\rightarrow$ 1

Table 3.7: Auxiliary Classifier version of BigGAN used on 32x32 images. n is the number of classes in the dataset, 10 for the CIFAR-10 dataset.
Generator on the left, Discriminator on the right.

$z \in \mathbb{R}^{128} \mathcal{N}(0, I)$ Embed(y) $\in \mathbb{R}^{128}$
Linear $\rightarrow$ 4x4x256
ResBlock Up $\rightarrow$ 256
ResBlock Up $\rightarrow$ 256
Non-Local Block
ResBlock Up $\rightarrow$ 256
BN, ReLU, 3x3 Conv $\rightarrow$ 3
Tanh

RGB image $x \in \mathbb{R}^{32x32x3}$
5x5 Conv $\rightarrow$ 128
ResBlock Down $\rightarrow$ 128
Non-Local Block
ResBlock Down $\rightarrow$ 128
ResBlock $\rightarrow$ 128
ResBlock $\rightarrow$ 128
ReLU, GlobalSumPooling
Linear $\rightarrow$ 1 | Linear $\rightarrow n$

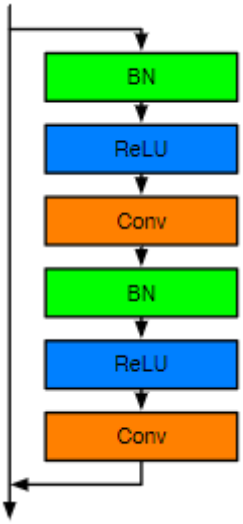

Figure 3.13: Residual block for the generator of the SN-GAN. The discriminator block is the same except without Batch Normalization. Up sampling and down sampling are done with the convolutions. Taken from [29].

3.3.2 Optimizing the Proposed BigAC-GAN

Once again, similarly to what was done with the AC-GAN, some small variations were made to the baseline model to try to improve it. These variations include, like before, modifying the batch size (originally 64) and halving the discriminator's learning rate. Table 3.8 sums up the experiments done and the performance of the resulting models. No variation in specific seem to surely improve the model and model performances are very similar all around.

As before, the model with the highest auxiliary classification accuracy (88.36%, of model number 1) was used in another experiment, and, once more, before doing so, it is important to check this model's training progress.

There are a few small details in the losses of this BigAC-GAN (at the left of Figure 3.14) that are different from what was seen in the losses of the AC-GAN model achieved before. The most obvious one is, perhaps, the larger gap between the generator and discriminator losses. This is, probably, caused by the more capable discriminator. Another detail is the discriminator's loss on fake images, which now drops at the start, like its loss on real images

Model Nr.	D_LR	Batch Size	IS ↑	FID ↓	Aux Acc(%)
(1)	$2x10^{-4}$	64	7.72	25.33	**88.38**
(2)	$2x10^{-4}$	32	7.73	27.96	86.83
(3)	$2x10^{-4}$	128	7.65	25.81	84.53
(4)	$1x10^{-4}$	64	7.74	26.8	86.52
(5)	$1x10^{-4}$	32	7.71	29.91	85.39
(6)	$1x10^{-4}$	128	7.81	**23.98**	87.91

Table 3.8: Performance of different AC-GAN models. D_LR is the discriminator's learning rate. The generator's learning rate is $2x10^{-4}$ for all models. Aux Acc is the auxiliary classification accuracy of the discriminator.

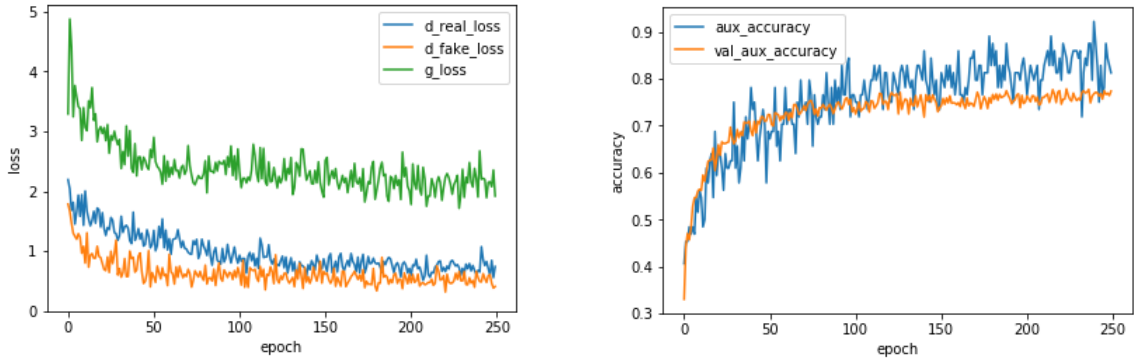

Figure 3.14: The losses of the best BigAC-GAN model achieved, at the left, and the auxiliary classification accuracy of its discriminator on real images from the training and validation sets, at the right.

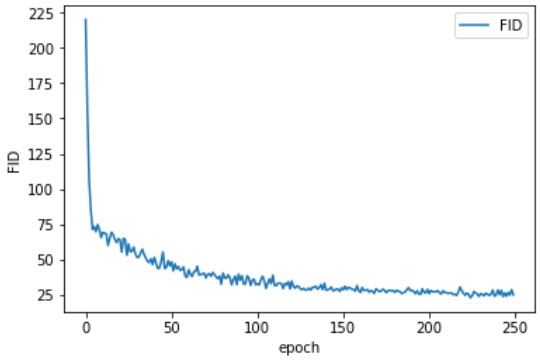

Figure 3.15: FID evolution across training for the best BigAC-GAN model achieved.

(this is the same thing that makes the generator's loss drop at the start as well). This is a consequence of the generator being better from the start, making it harder for the discriminator to learn what class each fake image belongs to. More importantly, though, the losses still maintain a similar value throughout the entire training process, which is the main sign that the GAN trained well. In the same Figure, at the right, the auxiliary accuracy on real and validation data are illustrated. This graph looks very much like the one in Figure 3.14, and, again, shows no signs of under or overfitting. The FID evolution across epochs is shown in Figure 3.15, showing the progress of the generator. Finally, in Figure 3.16 there are some samples of images produced by the BigAC-GAN. Curiously, this model seems to have struggled more than the previous one at modelling automobiles (second row in Figure 3.16), while, at the same time, it seems to model other classes better, like cats and dogs (third and fifth rows of the same image).

Figure 3.16: Samples of generated images of the best BigAC-GAN model achieved. Each row has 10 images of one of the 10 classes of the CIFAR-10 dataset. Classes are, from top to bottom row: airplane, car, bird, cat, deer, dog, frog, horse, ship, truck.

3.3.3 Scenarios Classification with the Proposed BigAC-GAN

It is now time to recall the three Scenarios that were used above, with AC-GAN, as part of an experiment:

- In Scenario 1, a normal classifier is trained on the CIFAR-10 dataset;
- In Scenario 2, the same classifier as in Scenario 1 is trained on the CIFAR-10 dataset augmented (to double its size) with images generated by a GAN;
- In Scenario 3, the discriminator of an AC-GAN trained on the CIFAR-10 dataset, in this case the BigAC-GAN, is used instead of a normal classifier;

Yet again, in Scenario 2 more than one generator will be used to augment the data set (in separate): the generator from the StyleGAN2-ADA and the generator from the best achieved BigAC-GAN. For convenience purposes, the Scenario 2 experiments with the StyleGAN2-ADA and the BigAC-GAN's generators will be simply referred to as Scenario 2 with StyleGAN2-ADA and Scenario 2 with BigAC-GAN.

The training progress of Scenarios 1, 2 with the StyleGAN2-ADA and with the BigAC-GAN are in Figures 3.17, 3.18 and 3.19, respectively. By looking at the losses, it is visible that the validation loss (on all three cases) does increase, which is a sign of overfitting, but since the accuracy was unarmed and the increase in the validation loss is so small this was not considered a severe issue. Furthermore, it is important to notice that, since these classifiers

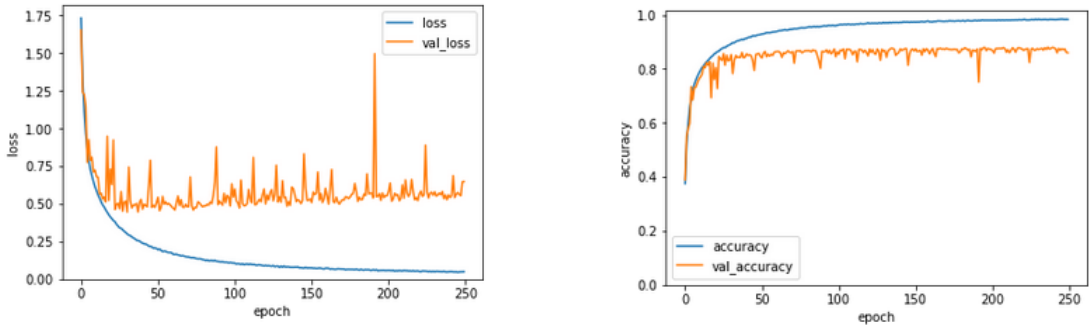

Figure 3.17: Loss along with validation loss (at the left) and accuracy along with validation accuracy (at the right) for Scenario 1.

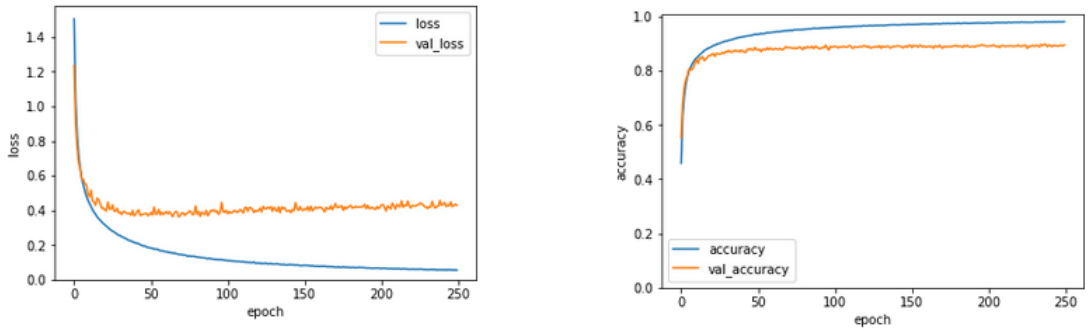

Figure 3.18: Loss along with validation loss (at the left) and accuracy along with validation accuracy (at the right) for Scenario 2 with the BigAC-GAN.

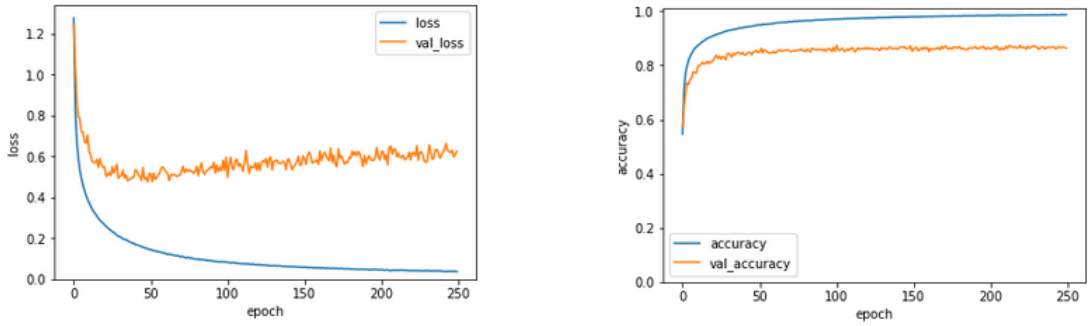

Figure 3.19: Loss along with validation loss (at the left) and accuracy along with validation accuracy (at the right) for Scenario 2 with the BigAC-GAN.

and the discriminator of the GAN in Scenario 3 should be equal, a small change in these classifiers to attempt to remove this little bit of overfitting could easily make the training of the GAN for Scenario 3 unstable, and it is important to keep in mind that the training of a GAN like the one used on Scenario 3, with the available hardware, takes over 8 hours. There is, though, the exception of Scenario 2 with the BigAC-GAN, where the validation loss increases a bit more aggressively, which is, possibly, an indication that the images from the generator were still not good enough to help the classifier. This thought is confirmed by the accuracies showed in Table. 3.9. Confusion matrices for all Scenarios are shown in Figure 3.20.

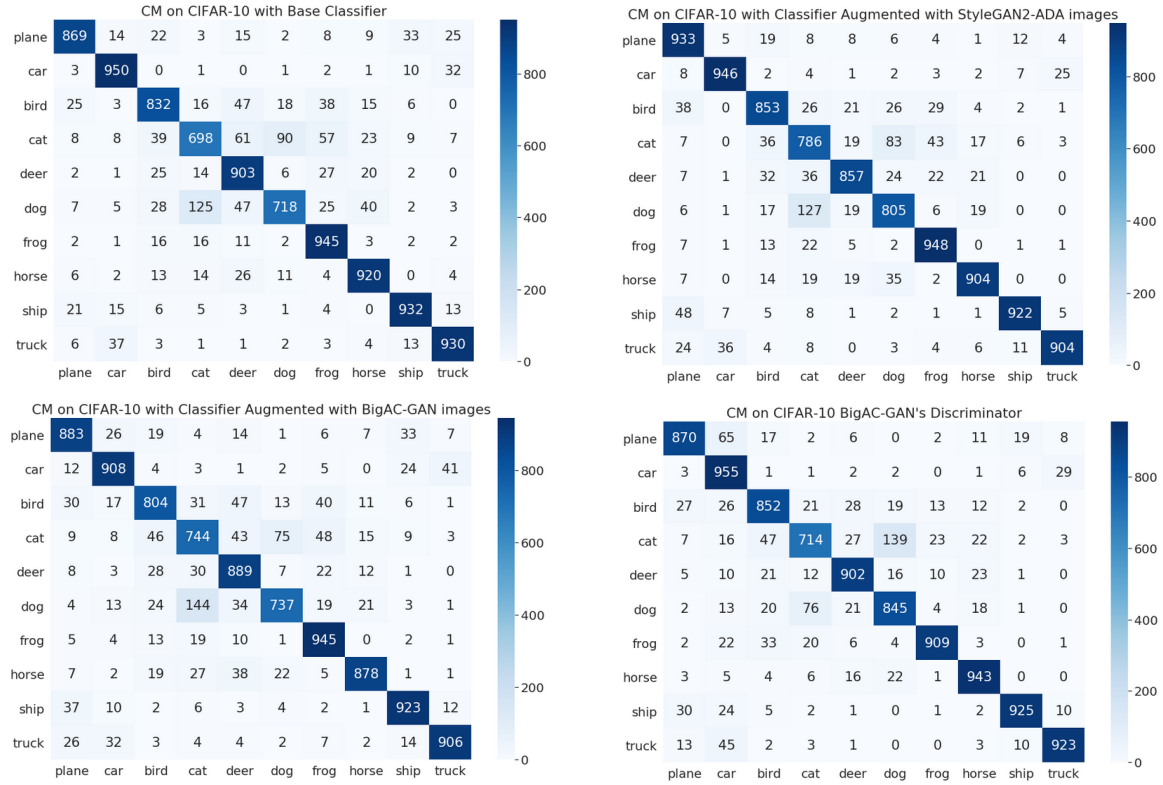

Figure 3.20: Confusion matrices for classifier/discriminator of all Scenarios. Top left: Scenario 1; Top right: Scenario 2 w/ Stylegan2-ADA; Bot left: Scenario 2 w/ BigAC-GAN; Bot right: Scenario 3;

Scenario	Acc (%)	FID ↓
Scenario 1	86.97	N/A
Scenario 2 w/BigAC-GAN	86.17	25.33
Scenario 2 w/StyleGAN2-ADA	**88.58**	**2.42**
Scenario 3	88.38	25.33

Table 3.9: Accuracies obtained on the different Scenarios.

The results in Table 3.9 are quite similar to what happened in the previous part of this Chapter, with the AC-GAN. Again, using the StyleGAN2-ADA to perform data augmentation helped the classifier, while doing data augmentation with the BigAC-GAN still did not improve the classifier, despite the much lower FID (almost half of the one of the AC-GAN used before). Moreover, like before, even though the BigAC-GAN did not improve the classifier when doing data augmentation, in Scenario 3, the BigAC-GAN's discriminator was able to perform as well as the classifier improved by the StyleGAN2-ADA. The same thought stems from this result: the fact that when using a GAN with an FID ten times larger it was possible to achieve the same classification accuracy, by exploiting the properties of the AC-GAN architecture, suggests that there may be an advantage in using AC-GANs as a way to achieve better classifiers.

Model	IS ↑	FID ↓
AC-GAN*	4.71	47.75
TAC-GAN*	4.17	54.91
UAC-GAN*	4.92	43.04
BigAC-GAN	**7.81**	**23.98**

Table 3.10: IS and FID of different AC-GANs on the CIFAR-10 dataset. * indicates that the metrics for those models are the ones measured in the experiments of the UAC-GAN.

3.4 Summary

In this chapter, two different architectures were used to study the quality of the discriminator of an AC-GAN. The first architecture was based on the original AC-GAN and the second was based on the (quite better) BigGAN, which had to be modified to accommodate the two objectives of the AC-GAN.

For both architectures, the use of AC-GANs showed to bring improvements to a standalone classifier. Additionally, those improvements were very similar to the ones provided by data augmentation via a state-of-the-art GAN, the StyleGAN2-ADA, which still has a quite superior generator, as according to the FID metric. This suggested that AC-GANs can improve a classifier and, perhaps, it may even be possible that this type of architecture can bring improvements better than the ones from data augmentation via GANs. It is, although, not possible to say this for sure with only the results present in this chapter. However, it is tempting to think that if one were to successfully transform the StyleGAN2-ADA into an AC-GAN, perhaps better improvements could be achieved. Transforming models into AC-GANs can, although, be a hard task. GANs are sensitive models and even small changes can cause models to collapse during training.

Finally, it is of worthy mention, that the modifications made to the BigGAN to transform it into an AC-GAN resulted in the best AC-GAN known, on the CIFAR-10 dataset. Table 3.10 shows the performance of the other AC-GANs on the CIFAR-10 dataset, as reported by [67]. The reason why the performances displayed are as in [67] is because the original AC-GAN did not measure its FID and the TAC-GAN did not do tests on the CIFAR-10 dataset.

CHAPTER 4

AC-GAN for Skin Lesion

In this chapter, the ideas explored in Chapter 3 are applied to a skin lesion problem, supported by the ISIC 2019 dataset. This makes the task of the models to be tested harder both by making images bigger and by making the aspects that differentiate classes harder to understand.

4.1 Motivation

Skin cancer is the most common kinds of cancer, and melanoma is the deadliest kind of skin cancer. Dermoscopy is a non-invasive skin imaging technique that has been proven to help diagnose melanoma, when used by an experienced professional. With the rise of machine learning, and specifically of models based on convolutions and their effectiveness with images, dermoscopy can now be further used as a support for research toward automated analysis of medical imaging, in particular, of skin lesions.

The results from Chapter 3 were promising. However, in that Chapter the CIFAR-10 dataset was used, which despite being a good benchmark and initial challenge, holds little practical use. Moreover, the CIFAR-10 is still a relatively simple task, since the images are so small (32x32 pixels) and since classes are so distinct. A problem based on skin lesion images comes to give a very practical use to the work done here and makes the problem at hand considerably harder, given both the underlying complexity of the images, as well as their size (128x128, in this case).

4.2 The Data

The data used in this Chapter comes from the ISIC 2019 dataset. The ISIC has been creating challenges since 2016 with the purpose of melanoma classification using machine learning models. The dataset for the 2019 challenge is composed of 25331 dermoscopic images. The images belong to 8 different classes: Melanoma (MEL), Melanocytic Nevus (NV), Basal Cell Carcinoma (BCC), Actinic Keratosis (AK), Benign Keratosis (BK), Dermatofibroma (DF),

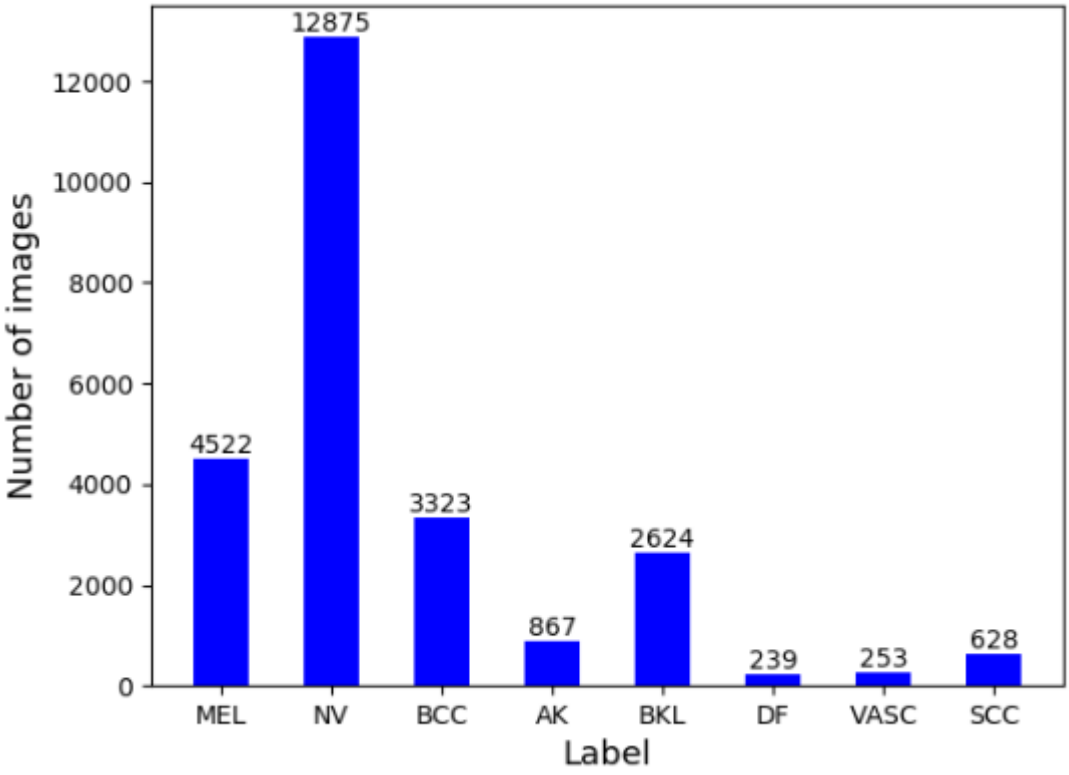

Figure 4.1: Amount of images per class in the ISIC 2019 dataset.

Vascular Lesion (VASC) and Squamous Cell Carcinoma (SCC). Figure 4.1 shows the amount of images per class. It is noticeable, in the same Figure, how imbalanced the dataset is, with one class making almost half of the dataset (NV).

These images are provenant from three different sources: the HAM10000 [68], BCN_20000 [69] and MSK [70] datasets. These datasets have images in different sizes and, so, to work with the ISIC dataset it is necessary to preprocess the images, at least so that they have equal sizes. Images were all cropped in the center and then resized to a resolution of 128x128. Cropping the center removes part of the surroundings of the lesions in the images, since images are already centered on the lesions, and the resolution chosen to resize the images to was motivated by two factors: first, it is already a considerably high resolution and a widely used one in GAN projects and, second, higher resolutions (like 256x256 or 512x512) resulted in memory issues when training models.

Because of how imbalanced the dataset is, and since the primary goal of the ISIC dataset is to train models that accurately classify melanomas, the dataset was turned into a 2 class dataset, with the two classes being melanoma and non-melanoma (all other 7 classes). This resulted in a dataset with 4522 images for one class and 20809 for the other. To even out the classes, it was decided to augment the least represented class (melanomas), with traditional data augmentation techniques. Before performing any data augmentation, 1000 melanomas were set aside to be a part of the validation and test sets (500 images for each). This leaves a total of 3522 melanomas left to be a part of the train set. Each one of the 3522 melanomas was

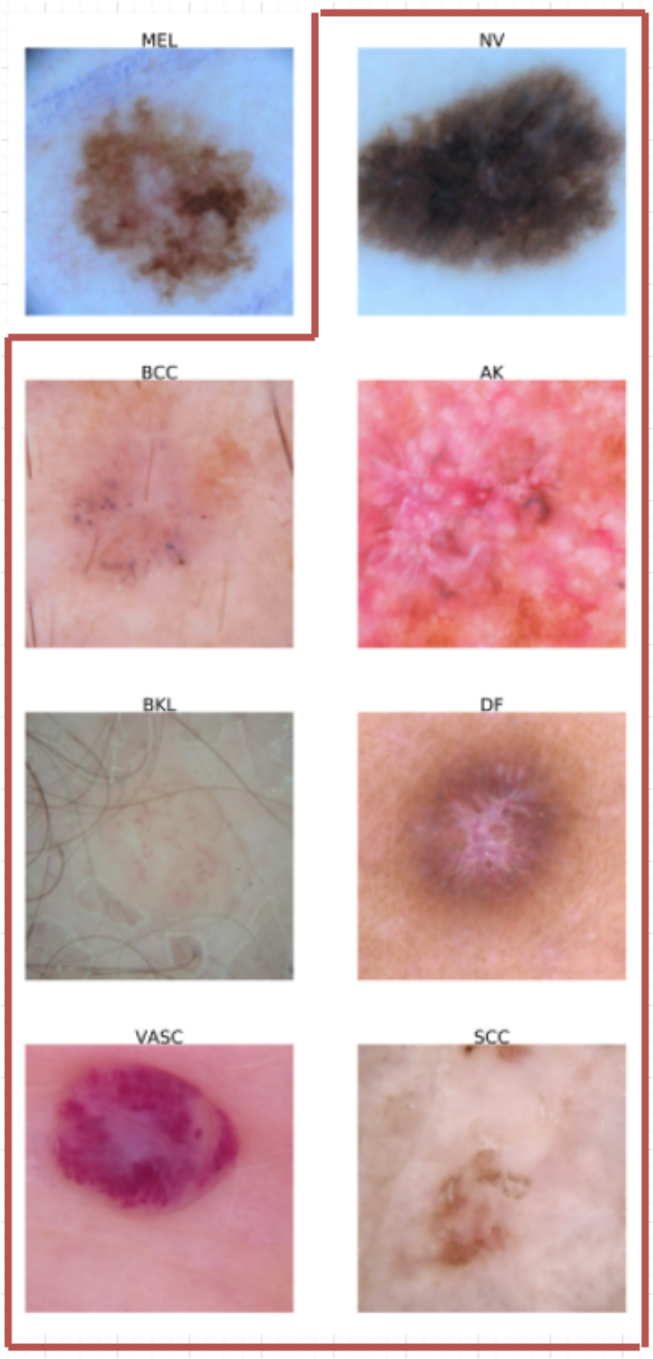

Figure 4.2: A sample from each of the classes of the ISIC 2019 dataset after cropping the center and resizing to 128x128. The red line separates melanomas from non-melanomas.

flipped and mirrored, totaling 10566 melanomas to be used for training. The same amounts of images were used for the non-melanomas: 10566 for the train set and 500 for both the validation and test sets. From the 7 classes that compose non-melanomas, images were chosen so that the ratio of images between classes remained the same as in the ISIC 2019 dataset, *i.e.* NV is still 62% of non-melanomas, BCC is still 16%, etc. Figure 4.2 shows one sample from each class of the ISIC 2019 dataset after the preprocessing operations.

Table 4.1: Auxiliary Classifier version of BigGAN used on 128x128 images. n is the number of classes in the dataset, 2 for the skin lesion dataset arranged here.
Generator on the left, Discriminator on the right.

z ϵ $\mathbb{R}^{128}$ $\mathcal{N}(0, I)$ Embed(y) ϵ $\mathbb{R}^{128}$
Linear $\rightarrow$ 4x4x192
ResBlock Up $\rightarrow$ 192
ResBlock Up $\rightarrow$ 192
ResBlock Up $\rightarrow$ 192
ResBlock Up $\rightarrow$ 192
Non-Local Block
ResBlock Up $\rightarrow$ 192
BN, ReLU, 3x3 Conv $\rightarrow$ 3
Tanh

RGB image x ϵ $\mathbb{R}^{32x32x3}$
5x5 Conv $\rightarrow$ 128
ResBlock Down $\rightarrow$ 128
Non-Local Block
ResBlock Down $\rightarrow$ 128
ResBlock Down $\rightarrow$ 128
ResBlock Down $\rightarrow$ 128
ResBlock Down $\rightarrow$ 128
ResBlock $\rightarrow$ 128
ReLU, GlobalSumPooling
Linear $\rightarrow$ 1 | Linear $\rightarrow$ n

4.3 Optimizing the BigAC-GAN for Skin Lesion

For this section, an AC-GAN similar to the one used in the previous Chapter was used. Some changes to that model (Table 3.7, on the previous Chapter) had to be made, given the change in size of the images. Typically, in GANs, generators produce images of certain resolutions by continuously doubling the size of an initial matrix. So if images were 32x32 before and are now 128x128, that means that the generator has to increase the size of its output twice, which means that there are additional layers. Adding layers in the generator means that layers have to be added on the discriminator too, otherwise, there is a good chance the GAN will collapse. This increase in layers is what caused memory issues when trying to use images of higher resolutions than 128x128. The increase in memory consumption by the network also meant that one of the aspects that showed improvements in the original BigGAN had to be changed/discarded, namely the increase in batch size. Batch size had to be reduced to 16, which is a move in the opposite direction of what led to improvements in the BigGAN and makes the training process quite slower. The optimizer used was the same as the one used in the experiments of the previous Chapter (Adam optimizer, with learning rates that changed for the different experiments made, specified in Table 4.2, and with a $beta_1 = 0.5$ and $beta_2 = 0.999$). The loss functions were the same as well. The architectural layout of this baseline model is depicted in Table 4.1. Table 4.2 shows the different variations made to this model and the results that each one achieved.

Once more, the model with the best auxiliary classification (model number 4) will be used in further experiments and, before doing so, it is important to look at the model's training process to make sure everything is alright.

Figure 4.3 shows the training of this Auxiliary Classifier version of the BigGAN. The most noticeable thing is the increase in the generator's loss, seen in the graph with the losses of discriminator and generator, at the left. This is generally a sign that the model is going to collapse. However, that loss stopped increasing, which indicates that the generator and discriminator found a new equilibrium. At the right, in the graph with the discriminator's

Model Nr.	LR	Fmap Inc.	FID ↓	Aux Acc(%)
(1)	$1x10^{-5}$	No	**19.51**	92.3
(2)	$7x10^{-6}$	No	23.76	91.9
(3)	$1x10^{-5}$	Yes	25.25	90.1
(4)	$7x10^{-6}$	Yes	19.68	**92.5**

Table 4.2: Performance of different BigAC-GAN models. LR is the learning rate for both the discriminator and generator in each model. Fmap Inc. is whether or not an increase in feature maps was done. In case of a feature map increase, the feature maps of the layers of the generator where increased from 192 to 256, on all layers. Aux Acc is the auxiliary classification accuracy of the discriminator.

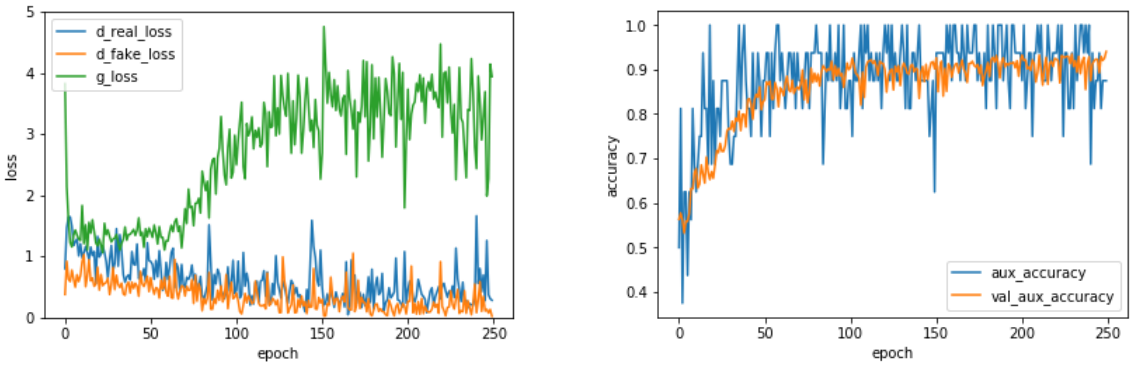

Figure 4.3: The losses of the BigAC-GAN model trained, at the left, and the auxiliary classification accuracy of its discriminator on real images from the training and validation sets, at the right.

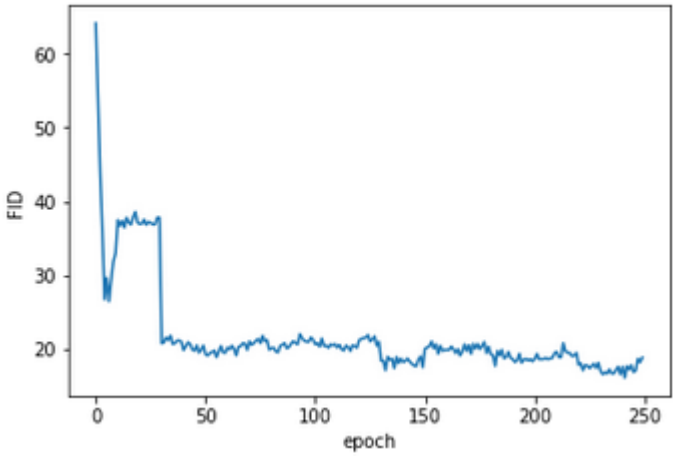

Figure 4.4: FID of the BigAC-GAN trained over epochs.

accuracy on real images of the training and validation sets, there is a clear, continuous, increase in accuracy, on both training and validation images, until it reaches values around the 90% mark. In Figure 4.4 the FID measure of the generator across the training process is shown. Its drop to low values (near 20) give further indication that the training procedure was effective. Figure 4.5 shows some samples of images created by the model's generator. Their quality gives a final reassurance that the training process went well.

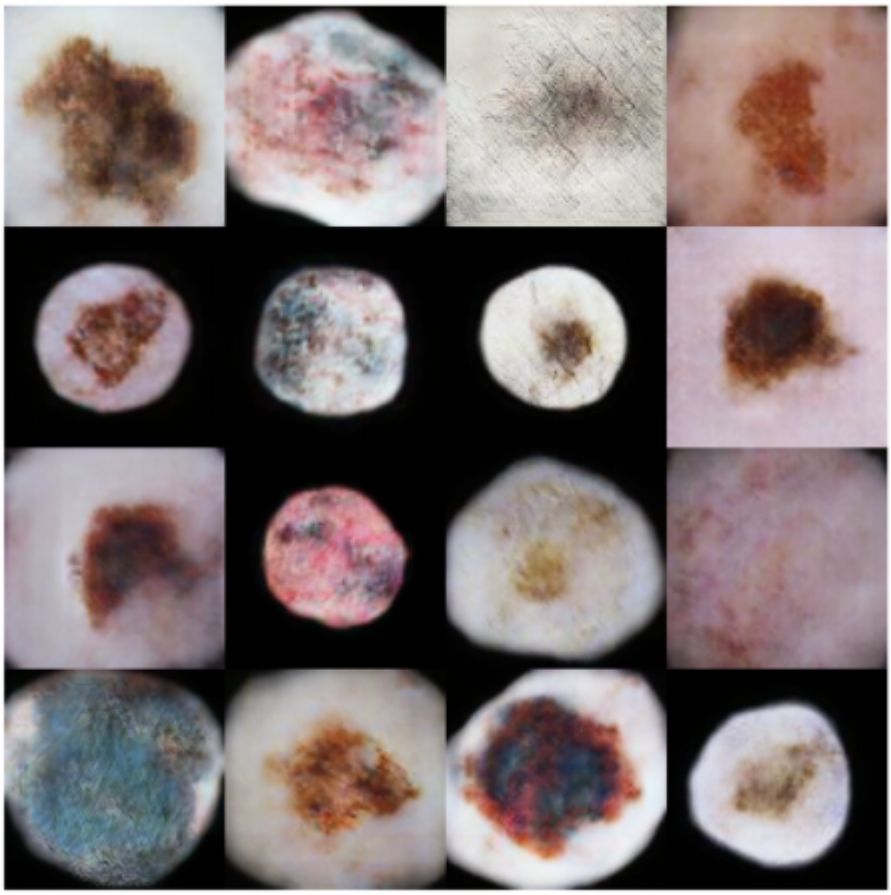

Figure 4.5: Samples from the BigAC-GAN model trained. Top 2 rows are melanomas, and bottom 2 rows are non-melanomas.

4.4 Scenarios Classification with BigAC-GAN for Skin Lesion

Experiments for this Chapter will be similar to the ones of the previous Chapter 3. The 3 Scenarios used to evaluate the AC-GAN and the Big-ACGAN on the CIFAR-10 will now be used to test the latter model on the skin lesion dataset, based on the ISIC 2019 dataset, described above.

Once more, this is a good time to recall the three Scenarios that were used before:

- In Scenario 1, a normal classifier is trained on the dataset;
- In Scenario 2, the same classifier as in Scenario 1 is trained on the dataset augmented with images generated by a GAN;
- In Scenario 3, the discriminator of an AC-GAN trained on the dataset, in this case the BigAC-GAN, is used instead of a normal classifier;

It is important to mention how Scenario 2 was set up. Since there are a lot of unused non-melanoma images in Scenario 1, the increase in the dataset for Scenario 2 was made with real images for non-melanomas, and with fake images for melanomas, generated by the StyleGAN2-ADA. The amount of generated images was equal to the total of melanomas in the ISIC 2019 dataset, 4522. The same amount of images was added to non-melanomas, once

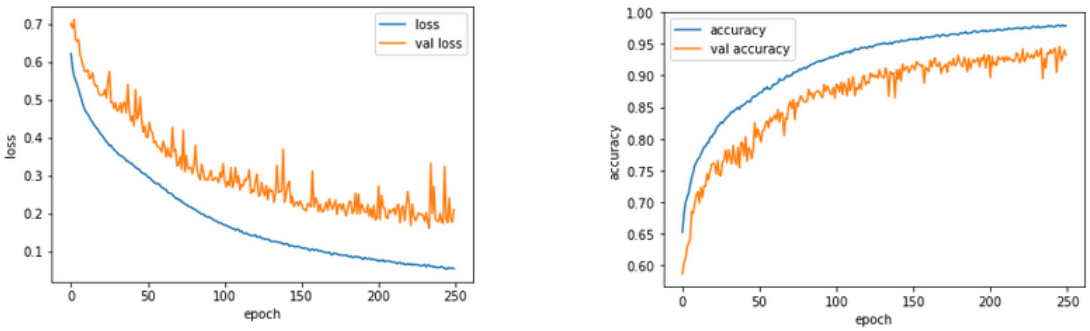

Figure 4.6: Loss along with validation loss (at the left) and accuracy along with validation accuracy (at the right) for Scenario 1.

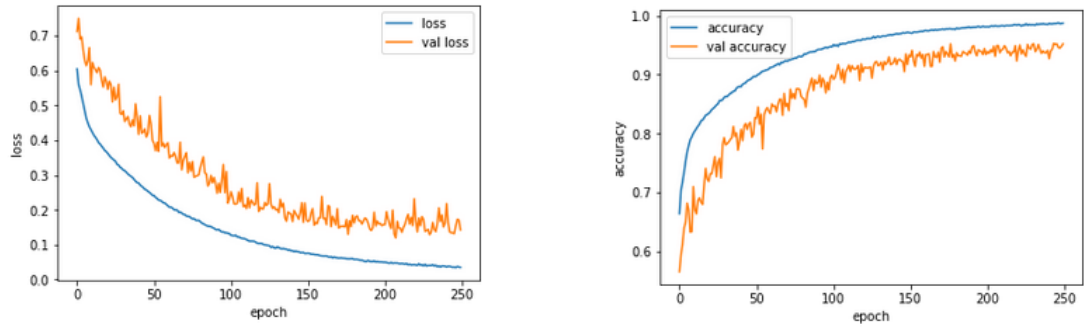

Figure 4.7: Loss along with validation loss (at the left) and accuracy along with validation accuracy (at the right) for Scenario 2.

more, in a way that maintained the ratio of images between the 7 classes of the ISIC 2019 dataset that compose the non-melanomas. The resulting dataset has, therefore, for each class, 15088 images for training, 500 images for validation and 500 for testing.

The training progress of Scenarios 1 and 2 is illustrated in Figures 4.6, 4.7, respectively. For both Scenarios, the training losses drop smoothly and stabilize at small values. The validation losses follow the training ones from very close, which discards under and overfitting. The accuracy of these models increased steadily to values around 90%, for both training and validation accuracies.

The confusion matrices for all Scenarios' classifiers/discriminator are in Figure 4.8. As expected from all of them, given that they achieved accuracies of around 90%, miss-classifications are rare, yet there is a tendency for models to confuse non-melanomas for melanomas more than the other way around. This tendency, though, is not present in Scenario 2, which indicates that the increase in the amount of melanomas coming from the StyleGAN2-ADA helped the model fix this.

The results shown on Table 4.3 are very similar to the ones of the previous Chapter. The AC-GAN brought improvements to a standalone classifier, even though those improvements are still not as good as the ones achieved from performing data augmentation with the StyleGAN2-ADA. Like before, though, the StyleGAN2-ADA still had a better generator than the one of the AC-GAN developed, based on the BigGAN. These results come, first, to show

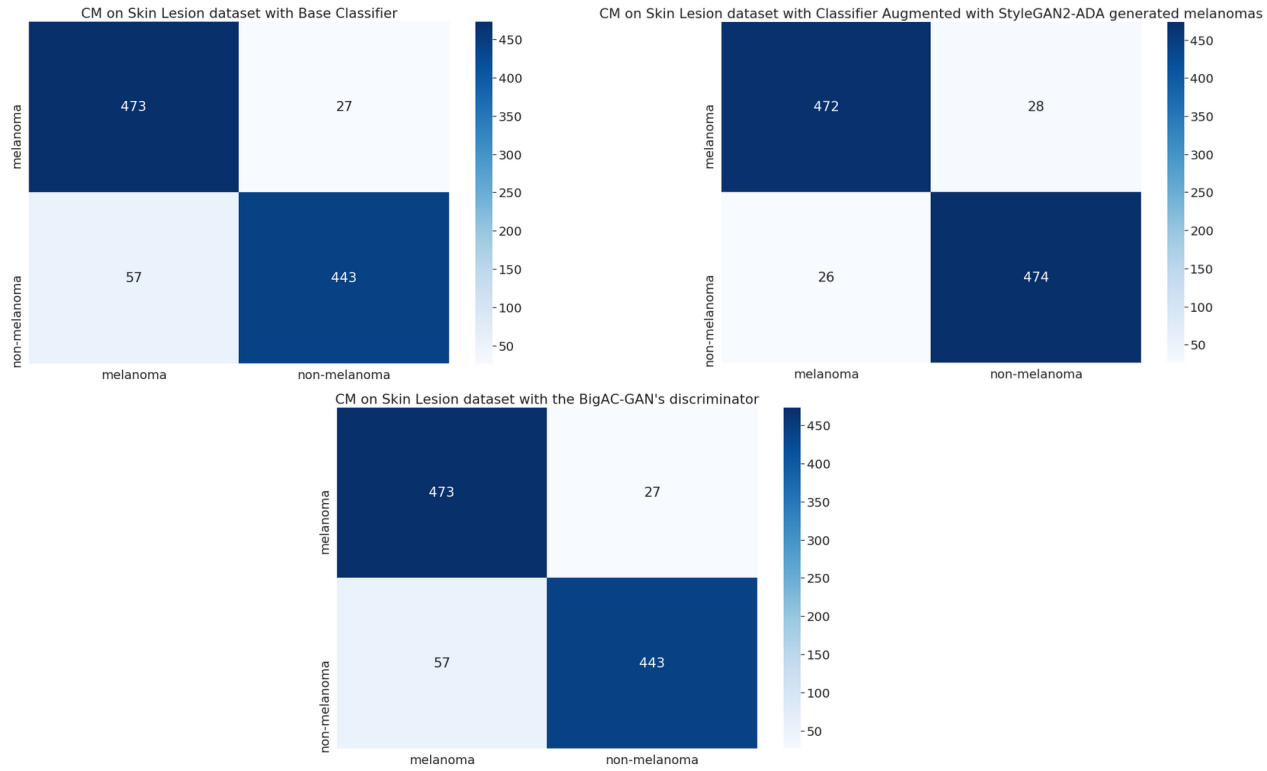

Figure 4.8: Confusion matrices for classifier/discriminator of the three Scenarios. Top left: Scenario 1; Top right: Scenario 2; Bottom: Scenario 3;

Scenario	Acc (%)	FID ↓
Scenario 1	91.1	N/A
Scenario 2 w/StyleGAN2-ADA	**94.6**	**14.3***
Scenario 3	92.3	19.51

Table 4.3: Accuracies obtained on the different Scenarios. *Stylegan FID corresponds to the FID for generating melanomas only, while the FID for Scenario 3 is for generating both melanomas and non-melanomas.

that AC-GANs can have a good performing discriminator on more complex problems and, second, to cement the results and thoughts that were collected in the previous Chapter.

4.5 Summary

The goal of this chapter was to increase the complexity of the problem that the AC-GANs used before had to face. For that purpose, the dataset was changed to one of skin lesions. The dataset used is based on the ISIC 2019, which was changed from a multi-class problem into a binary problem, with melanomas and non-melanomas. Given the imbalance present on the dataset, some traditional data augmentation techniques were used to create a balanced dataset.

Then, given the new images and their new resolution, some adaptations were made to the AC-GAN based on the BigGAN that was used on the previous chapter. Unfortunately, this lead to hardware limitations and forced the models used here to be trained on very small

batch sizes, which goes directly against one of the methods that lead to improvements for the BigGAN: increasing the batch size.

The results attained with the models trained were very identical to the ones of the previous chapter. The developed AC-GAN is better than a baseline classifier, yet not as good as that base classifier when using data augmentation coming from the StyleGAN2-ADA. It is, again, necessary to consider that the StyleGAN2-ADA still had a better generator than the one of the AC-GAN developed, based on the FID measure. These results additionally support the idea that AC-GANs can be a novel way to bring improvements to general use classifiers.

CHAPTER 5

Conclusions

This dissertation's main goal was to explore GANs and understand if there may be new ways in which they can help classifiers improve. These models have been used a lot to perform data augmentation in data hungry problems, like medical imaging related problems, to assist classifiers with their learning process. Results of this approach have been fairly positive across different areas and give proof that GANs are in fact useful and can have a decisive role in improving the performance of other models.

One GAN architecture was of huge importance to this work, because of one very interesting property. The AC-GAN's discriminator has two outputs, instead of one. It outputs whether an image is fake or real, as well as what class it belongs to. When designed, this property was only supposed to be a method that would improve the generator of the GAN, yet for this work, this property was the main subject of attention. The question was whether or not the discriminator of an AC-GAN could be used as a classifier and if that could be better than having a normal classifier with and without GAN data augmentation.

First, an AC-GAN was developed. There were difficulties in recreating the original model, so a new model was built from scratch, for the CIFAR-10 dataset. To test how well the discriminator of this AC-GAN could perform, a classifier as similar as possible to the AC-GAN's discriminator was trained with and without data augmentation from one of the state-of-the-art GANs, the StyleGAN2-ADA, and from an AC-GAN equal to the one created. As expected, the classifier with GAN data augmentation performed better than the one without it. The more interesting results, though, were: how close the accuracy of the classifier with data augmentation from the StyleGAN2-ADA was to the discriminator of the AC-GAN, which was trained on data with no augmentations of any kind; and the fact that the discriminator of the AC-GAN performed better than a classifier with data augmentation coming from an equal GAN. The fact that the performances of the AC-GAN and of the classifier with data augmentation from the StyleGAN2-ADA were so competitive suggested that it may be possible that AC-GANs can become a way to create better classifiers than just doing data augmentation, especially if one considers the very large gap between the abilities of

the generator of the AC-GAN used and of the StyleGAN2-ADA used for data augmentation.

The same experiments were then made, for the same dataset, with a new AC-GAN. For these experiments, a considerably better GAN than the original AC-GAN, the BigGAN, was adapted into an AC-GAN. A lot of the ideas that made the BigGAN remained intact. The goal was to have a discriminator that has the two outputs that the AC-GAN has, while maintaining a GAN that can train in a stable manner. Results showed the same as in the previous situation. The performance of a classifier with data augmentation from the StyleGAN2-ADA was only slightly better than the performance of the discriminator of this new, better, AC-GAN, and the AC-GAN performed better than a classifier with data augmentation coming from an equal GAN. These results did not bring any new conclusions, however they reinforced the ones taken before, since there was still a considerable gap between the quality of the generator of the StyleGAN2-ADA and of the AC-GAN built. Together, the results suggest that maybe, an AC-GAN version of the StyleGAN2-ADA could result in a discriminator that would outperform a classifier with the benefit of data augmentation from the StyleGAN2-ADA.

Finally, the experiments were done once again, this time with a different dataset: a skin lesion dataset. This skin lesion dataset was used to increase the difficulty of the problem and to approximate this work with one of the biggest practical uses GANs have been having recently. Even with the increase in complexity, results were coherent with what happened on the earlier experiments. The AC-GAN developed performed better than a standalone classifier and did not perform better than an equal classifier improved with data augmentation coming from the StyleGAN2-ADA, which can be explained by the fact that the StyleGAN2-ADA still created better images than the AC-GAN used.

The results gathered here leave very strong reasons to believe that it is possible for AC-GANs to become a novel better way to improve the performances achieved on classification tasks. The work done here can become an important pillar to guide further studies in this direction, that come to develop and further solidify these results, and can definitely be a motivation for other new approaches to be experimented with GANs.

5.1 Future Work

The following are some ideas for future works that could result in improvements:

- The implementation of an Auxiliary Classifier StyleGAN2-ADA would, almost for sure, improve the results achieved here with the use of different AC-GANs.
- The use of techniques that select only the better images created by the generator, like discriminator rejection sampling [71] and other similar ones [72], [73], [74], can additionally improve the results achieved here, since these techniques have been shown to improve the performance of GANs.
- The adaptation to a multi-class problem based on skin lesions is a natural path of progression for this work too.

References

[1] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Computation*, vol. 1, no. 4, pp. 541–551, 1989. DOI: 10.1162/neco.1989.1.4.541.

[2] K.-S. Oh and K. Jung, "Gpu implementation of neural networks," *Pattern Recognition*, vol. 37, pp. 1311–1314, Jun. 2004. DOI: 10.1016/j.patcog.2004.01.013.

[3] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, *Generative adversarial networks*, 2014. arXiv: 1406.2661 [stat.ML].

[4] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. P. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi, "Photo-realistic single image super-resolution using a generative adversarial network," *CoRR*, vol. abs/1609.04802, 2016. arXiv: 1609.04802. [Online]. Available: http://arxiv.org/abs/1609.04802.

[5] D. Pathak, P. Krähenbühl, J. Donahue, T. Darrell, and A. A. Efros, "Context encoders: Feature learning by inpainting," *CoRR*, vol. abs/1604.07379, 2016. arXiv: 1604.07379. [Online]. Available: http://arxiv.org/abs/1604.07379.

[6] A. Brock, J. Donahue, and K. Simonyan, "Large scale GAN training for high fidelity natural image synthesis," *CoRR*, vol. abs/1809.11096, 2018. arXiv: 1809.11096. [Online]. Available: http://arxiv.org/abs/1809.11096.

[7] T. Karras, M. Aittala, J. Hellsten, S. Laine, J. Lehtinen, and T. Aila, *Training generative adversarial networks with limited data*, 2020. arXiv: 2006.06676 [cs.CV].

[8] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, "Analyzing and improving the image quality of stylegan," *CoRR*, vol. abs/1912.04958, 2019. arXiv: 1912.04958. [Online]. Available: http://arxiv.org/abs/1912.04958.

[9] G. Gonçalves, *A comparative study of data augmentation techniques for image classification: Generative models vs. classical transformations*, 2020.

[10] L. Bi, J. Kim, A. Kumar, D. Feng, and M. J. Fulham, "Synthesis of positron emission tomography (PET) images via multi-channel generative adversarial networks (gans)," *CoRR*, vol. abs/1707.09747, 2017. arXiv: 1707.09747. [Online]. Available: http://arxiv.org/abs/1707.09747.

[11] E. Wu, K. Wu, D. D. Cox, and W. Lotter, "Conditional infilling gans for data augmentation in mammogram classification," *CoRR*, vol. abs/1807.08093, 2018. arXiv: 1807.08093. [Online]. Available: http://arxiv.org/abs/1807.08093.

[12] I. S. Ali, M. F. Mohamed, and Y. B. Mahdy, *Data augmentation for skin lesion using self-attention based progressive generative adversarial network*, 2019. arXiv: 1910.11960 [eess.IV].

[13] A. Bissoto, F. Perez, E. Valle, and S. Avila, "Skin lesion synthesis with generative adversarial networks," *CoRR*, vol. abs/1902.03253, 2019. arXiv: 1902.03253. [Online]. Available: http://arxiv.org/abs/1902.03253.

[14] C. Bowles, L. Chen, R. Guerrero, P. Bentley, R. N. Gunn, A. Hammers, D. A. Dickie, M. del C. Valdés Hernández, J. M. Wardlaw, and D. Rueckert, "GAN augmentation: Augmenting training data using generative adversarial networks," *CoRR*, vol. abs/1810.10863, 2018. arXiv: 1810.10863. [Online]. Available: http://arxiv.org/abs/1810.10863.

[15] M. Frid-Adar, I. Diamant, E. Klang, M. Amitai, J. Goldberger, and H. Greenspan, "Gan-based synthetic medical image augmentation for increased CNN performance in liver lesion classification," *CoRR*, vol. abs/1803.01229, 2018. arXiv: 1803.01229. [Online]. Available: http://arxiv.org/abs/1803.01229.

[16] A. Kitchen and J. Seah, "Deep generative adversarial neural networks for realistic prostate lesion MRI synthesis," *CoRR*, vol. abs/1708.00129, 2017. arXiv: 1708.00129. [Online]. Available: http://arxiv.org/abs/1708.00129.

[17] C. Han, L. Rundo, R. Araki, Y. Furukawa, G. Mauri, H. Nakayama, and H. Hayashi, "Infinite brain MR images: Pggan-based data augmentation for tumor detection," *CoRR*, vol. abs/1903.12564, 2019. arXiv: 1903.12564. [Online]. Available: http://arxiv.org/abs/1903.12564.

[18] X. Yi, E. Walia, and P. S. Babyn, "Unsupervised and semi-supervised learning with categorical generative adversarial networks assisted by wasserstein distance for dermoscopy image classification," *CoRR*, vol. abs/1804.03700, 2018. arXiv: 1804.03700. [Online]. Available: http://arxiv.org/abs/1804.03700.

[19] H. Rashid, M. A. Tanveer, and H. Aqeel Khan, "Skin lesion classification using gan based data augmentation," pp. 916–919, 2019. DOI: 10.1109/EMBC.2019.8857905.

[20] J. Hayes, L. Melis, G. Danezis, and E. De Cristofaro, "Logan: Evaluating privacy leakage of generative models using generative adversarial networks," May 2017.

[21] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *CoRR*, vol. abs/1411.1784, 2014. arXiv: 1411.1784. [Online]. Available: http://arxiv.org/abs/1411.1784.

[22] A. Odena, C. Olah, and J. Shlens, *Conditional image synthesis with auxiliary classifier gans*, 2017. arXiv: 1610.09585 [stat.ML].

[23] A. Krizhevsky, "Learning multiple layers of features from tiny images," *University of Toronto*, May 2012.

[24] D. P. Kingma, T. Salimans, R. Jozefowicz, X. Chen, I. Sutskever, and M. Welling, *Improving variational inference with inverse autoregressive flow*, 2017. arXiv: 1606.04934 [cs.LG].

[25] A. Mino and G. Spanakis, *Logan: Generating logos with a generative adversarial neural network conditioned on color*, Oct. 2018.

[26] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, *Imagenet large scale visual recognition challenge*, 2015. arXiv: 1409.0575 [cs.CV].

[27] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena, *Self-attention generative adversarial networks*, 2019. arXiv: 1805.08318 [stat.ML].

[28] K. He, X. Zhang, S. Ren, and J. Sun, *Deep residual learning for image recognition*, 2015. arXiv: 1512.03385 [cs.CV].

[29] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, *Spectral normalization for generative adversarial networks*, 2018. arXiv: 1802.05957 [cs.LG].

[30] X. Wang, R. Girshick, A. Gupta, and K. He, *Non-local neural networks*, 2018. arXiv: 1711.07971 [cs.CV].

[31] V. Dumoulin, J. Shlens, and M. Kudlur, "A learned representation for artistic style," *CoRR*, vol. abs/1610.07629, 2016. arXiv: 1610.07629. [Online]. Available: http://arxiv.org/abs/1610.07629.

[32] H. de Vries, F. Strub, J. Mary, H. Larochelle, O. Pietquin, and A. C. Courville, "Modulating early visual processing by language," *CoRR*, vol. abs/1707.00683, 2017. arXiv: 1707.00683. [Online]. Available: http://arxiv.org/abs/1707.00683.

[33] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of gans for improved quality, stability, and variation," *CoRR*, vol. abs/1710.10196, 2017. arXiv: 1710.10196. [Online]. Available: http://arxiv.org/abs/1710.10196.

[34] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," *CoRR*, vol. abs/1812.04948, 2018. arXiv: 1812.04948. [Online]. Available: http://arxiv.org/abs/1812.04948.

[35] I. Durugkar, I. Gemp, and S. Mahadevan, *Generative multi-adversarial networks*, 2017. arXiv: 1611.01673 [cs.LG].

[36] A. Karnewar and O. Wang, *Msg-gan: Multi-scale gradients for generative adversarial networks*, 2020. arXiv: 1903.06048 [cs.CV].

[37] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of Big Data*, vol. 6, no. 1, 2019, ISSN: 2196-1115. DOI: 10.1186/s40537-019-0197-0. [Online]. Available: https://doi.org/10.1186/s40537-019-0197-0.

[38] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998. DOI: 10.1109/5.726791.

[39] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, ser. NIPS'12, Lake Tahoe, Nevada: Curran Associates Inc., 2012, pp. 1097–1105.

[40] X. Zhang, J. J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," *CoRR*, vol. abs/1509.01626, 2015. arXiv: 1509.01626. [Online]. Available: http://arxiv.org/abs/1509.01626.

[41] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "Specaugment: A simple data augmentation method for automatic speech recognition," *Interspeech 2019*, 2019. DOI: 10.21437/interspeech.2019-2680. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2019-2680.

[42] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, "Return of the devil in the details: Delving deep into convolutional nets," *CoRR*, vol. abs/1405.3531, 2014. arXiv: 1405.3531. [Online]. Available: http://arxiv.org/abs/1405.3531.

[43] A. Jurio, M. Pagola, M. Galar, C. Lopez-Molina, and D. Paternain, "A comparison study of different color spaces in clustering based image segmentation," *Communications in Computer and Information Science*, vol. 81, pp. 532–541, Jun. 2010. DOI: 10.1007/978-3-642-14058-7_55.

[44] G. Kang, X. Dong, L. Zheng, and Y. Yang, "Patchshuffle regularization," *CoRR*, vol. abs/1707.07103, 2017. arXiv: 1707.07103. [Online]. Available: http://arxiv.org/abs/1707.07103.

[45] H. Inoue, "Data augmentation by pairing samples for images classification," *CoRR*, vol. abs/1801.02929, 2018. arXiv: 1801.02929. [Online]. Available: http://arxiv.org/abs/1801.02929.

[46] R. Takahashi, T. Matsubara, and K. Uehara, "Data augmentation using random image cropping and patching for deep cnns," *CoRR*, vol. abs/1811.09030, 2018. arXiv: 1811.09030. [Online]. Available: http://arxiv.org/abs/1811.09030.

[47] A. Mikołajczyk and M. Grochowski, "Data augmentation for improving deep learning in image classification problem," May 2018, pp. 117–122. DOI: 10.1109/IIPHDW.2018.8388338.

[48] M. Moradi, A. Madani, A. Karargyris, and T. Syeda-Mahmood, "Chest x-ray generation and data augmentation for cardiovascular abnormality classification," Mar. 2018, p. 57. DOI: 10.1117/12.2293971.

[49] V. Sandfort, K. Yan, P. Pickhardt, and R. Summers, "Data augmentation using generative adversarial networks (cyclegan) to improve generalizability in ct segmentation tasks," *Scientific Reports*, vol. 9, Nov. 2019. DOI: 10.1038/s41598-019-52737-x.

[50] H.-C. Shin, N. A. Tenenholtz, J. K. Rogers, C. G. Schwarz, M. L. Senjem, J. L. Gunter, K. P. Andriole, and M. Michalski, "Medical image synthesis for data augmentation and anonymization using generative adversarial networks," *CoRR*, vol. abs/1807.10225, 2018. arXiv: 1807.10225. [Online]. Available: http://arxiv.org/abs/1807.10225.

[51] L. Ballerini, R. B. Fisher, B. Aldridge, and J. Rees, "A color and texture based hierarchical k-nn approach to the classification of non-melanoma skin lesions," *Color Medical Image Analysis Lecture Notes in Computational Vision and Biomechanics*, pp. 63–86, 2013. DOI: 10.1007/978-94-007-5389-1_4.

[52] T. Mendonça, P. M. Ferreira, J. S. Marques, A. R. S. Marcal, and J. Rozeira, "Ph2 - a dermoscopic image database for research and benchmarking," in *2013 35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2013, pp. 5437–5440. DOI: 10.1109/EMBC.2013.6610779.

[53] D. Bisla, A. Choromanska, J. A. Stein, D. Polsky, and R. S. Berman, "Skin lesion segmentation and classification with deep learning system," *CoRR*, vol. abs/1902.06061, 2019. arXiv: 1902.06061. [Online]. Available: http://arxiv.org/abs/1902.06061.

[54] A. Ghorbani, V. Natarajan, D. Coz, and Y. Liu, *Dermgan: Synthetic generation of clinical skin images with pathology*, 2019. arXiv: 1911.08716 [cs.CV].

[55] Y. Liu, A. Jain, C. Eng, D. H. Way, K. Lee, P. Bui, K. Kanada, G. de Oliveira Marinho, J. Gallegos, S. Gabriele, and et al., "A deep learning system for differential diagnosis of skin diseases," *Nature Medicine*, vol. 26, no. 6, pp. 900–908, May 2020, ISSN: 1546-170X. DOI: 10.1038/s41591-020-0842-3. [Online]. Available: http://dx.doi.org/10.1038/s41591-020-0842-3.

[56] Z. Qin, Z. Liu, P. Zhu, and Y. Xue, "A gan-based image synthesis method for skin lesion classification," *Computer Methods and Programs in Biomedicine*, vol. 195, p. 105 568, May 2020. DOI: 10.1016/j.cmpb.2020.105568.

[57] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," *CoRR*, vol. abs/1611.07004, 2016. arXiv: 1611.07004. [Online]. Available: http://arxiv.org/abs/1611.07004.

[58] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro, "High-resolution image synthesis and semantic manipulation with conditional gans," *CoRR*, vol. abs/1711.11585, 2017. arXiv: 1711.11585. [Online]. Available: http://arxiv.org/abs/1711.11585.

[59] P. Schmid-Saugeon, J. Guillod, and J.-P. Thiran, "Towards a computer-aided diagnosis system for pigmented skin lesions," *Computerized Medical Imaging and Graphics*, vol. 27, no. 1, pp. 65–78, 2003, ISSN: 0895-6111. DOI: https://doi.org/10.1016/S0895-6111(02)00048-4. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0895611102000484.

[60] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, *Mobilenets: Efficient convolutional neural networks for mobile vision applications*, 2017. arXiv: 1704.04861 [cs.CV].

[61] T. Salimans, I. J. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," *CoRR*, vol. abs/1606.03498, 2016. arXiv: 1606.03498. [Online]. Available: http://arxiv.org/abs/1606.03498.

[62] S. Kullback and R. A. Leibler, "On Information and Sufficiency," *The Annals of Mathematical Statistics*, vol. 22, no. 1, pp. 79–86, 1951. DOI: 10.1214/aoms/1177729694. [Online]. Available: https://doi.org/10.1214/aoms/1177729694.

[63] S. Barratt and R. Sharma, *A note on the inception score*, 2018. arXiv: 1801.01973 [stat.ML].

[64] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*, Ieee, 2009, pp. 248–255.

[65] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, G. Klambauer, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a nash equilibrium," *CoRR*, vol. abs/1706.08500, 2017. arXiv: 1706.08500. [Online]. Available: http://arxiv.org/abs/1706.08500.

[66] M. Gong, Y. Xu, C. Li, K. Zhang, and K. Batmanghelich, "Twin auxiliary classifiers GAN," *CoRR*, vol. abs/1907.02690, 2019. arXiv: 1907.02690. [Online]. Available: http://arxiv.org/abs/1907.02690.

[67] L. Han, A. Stathopoulos, T. Xue, and D. N. Metaxas, "Unbiased auxiliary classifier gans with MINE," *CoRR*, vol. abs/2006.07567, 2020. arXiv: 2006.07567. [Online]. Available: https://arxiv.org/abs/2006.07567.

[68] P. Tschandl, C. Rosendahl, and H. Kittler, *The ham10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions*, 2018. DOI: 10.1038/sdata.2018.161.

[69] M. Combalia, N. C. F. Codella, V. Rotemberg, B. Helba, V. Vilaplana, O. Reiter, A. C. Halpern, S. Puig, and J. Malvehy, *Bcn20000: Dermoscopic lesions in the wild*, 2019.

[70] N. C. F. Codella, D. Gutman, M. E. Celebi, B. Helba, M. A. Marchetti, S. W. Dusza, A. Kalloo, K. Liopyris, N. Mishra, H. Kittler, and A. Halpern, *Skin lesion analysis toward melanoma detection: A challenge at the 2017 international symposium on biomedical imaging (isbi), hosted by the international skin imaging collaboration (isic)*, 2017.

[71] S. Azadi, C. Olsson, T. Darrell, I. Goodfellow, and A. Odena, *Discriminator rejection sampling*, 2019. arXiv: 1810.06758 [stat.ML].

[72] S. Arora and Y. Zhang, "Do gans actually learn the distribution? an empirical study," *CoRR*, vol. abs/1706.08224, 2017. arXiv: 1706.08224. [Online]. Available: http://arxiv.org/abs/1706.08224.

[73] R. Turner, J. Hung, E. Frank, Y. Saatci, and J. Yosinski, *Metropolis-hastings generative adversarial networks*, 2019. arXiv: 1811.11357 [stat.ML].

[74] T. Che, R. Zhang, J. Sohl-Dickstein, H. Larochelle, L. Paull, Y. Cao, and Y. Bengio, "Your GAN is secretly an energy-based model and you should use discriminator driven latent sampling," *CoRR*, vol. abs/2003.06060, 2020. arXiv: 2003.06060. [Online]. Available: https://arxiv.org/abs/2003.06060.