# Synthetic Data Generation To Better Perceive Classifier Predictions

Frederico Vicente
Mentor: Ludwig Krippahl

Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa, Quinta da Torre, 2829-516 Monte de Caparica, Portugal
fm.vicente@campus.fct.unl.pt, ludi@fct.unl.pt

**Abstract.** Understanding the underlying reasons behind the predictions of Neural Networks remains an active research topic in the field of Deep Neural Networks. The interpretation of the results of Neural Networks is highly necessary due to its impact on ethical, legal and security problems. Image Classifiers are a kind of Deep Learning model whose predictions can have a real impact in some day-to-day activities, therefore there is interest in providing intuition on what each class visually means to an Image Classifier. Can we meaningfully introspect the n-dimensional space that represents each class in a way that is easily comprehensible? In this paper, in contrast to some more common approaches that use Saliency Maps, we propose the use of Generative Models to better understand the classes and class boundaries of a previously trained Image Classifier. We have built a Generative Adversarial Network (GAN) embedded with a pre-trained Classifier to visually create an image collage for each class and for describing the inter-frontier space of classes. We also built a Variational Autoencoder (VAE) in an attempt to visually explore the frontiers between classes. The experiments' results show that using Generative Models can indeed be beneficial in mapping biases of Classifier's perception of classes. This project may contribute to a possible follow up research consisting in assessing whether this methodology can allow humans to guide the training of a Classifier by the selection of generated images.

**Keywords:** Artificial Intelligence · Deep Learning · Generative Models · Classifier · Interpretation of Neural Networks · GAN · VAE.

## 1 Introduction

### 1.1 Walk-through

The domain of Deep Learning interpretability has never been so relevant. Most industries are moving towards a digital revolution, applying Artificial Intelligence anywhere it can create value. This rampant zenith of Intelligent Systems creates an extra load of responsibility for the AI research community. How can we develop systems, tools and models that the organisations seek, while also guaranteeing confidence in their results? The field of interpreting Neural Networks

focuses their work on providing and developing tools, such as visual interfaces, as a means to bring some intuitive reasoning to such complex topologies.

The continuous development of architectures, such as the object detection YOLOv4 [1], prove how well Deep Learning helps us solve difficult problems, but should also evoke attention concerning ethical and safety questions. The possibilities such tools can provide are remarkable, but should be taken with caution, as they are also a good reminder of the importance of interpretative systems. When creating such models, the ill-use and unfortunate side effects can be fought with introspective approaches. Building tools which try to fill holes in the understanding of complex technologies, such as Neural Networks, should be a community effort. This methodology of using introspective tools not only eases the ethical responsibilities one carries when striving to innovate, but also helps focus the research on important breakthroughs. In line with this idea, in this project we think of means and ways to provide visual representations of a Classifier's view of its classes, by classifying the images generated from the Generative Models.

Delving into Deep Learning, Neural Networks can generally be categorised as either being discriminative or generative. These categories, are substantially different from both theoretical and empirical points of view. Whereas the main idea behind Generative Networks is to create new data points, in our case images, with Discriminative Networks the goal is to distinguish different kinds of data instances. This project makes use of both types, in a cooperative manner, to extract insights of a Discriminative Model (Image Classifier) by using beneficial characteristics of a Generative Model. Generative Models work by learning the distribution of real images provided as input. After training, they generate images that mirror the distribution/perception of the dataset the model learnt. (See an abstract overview of a Generative Model Fig. 1)

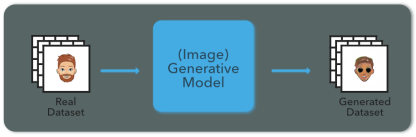


Fig. 1: Generative Models - Abstraction

Image Classifiers are a sub-type of Discriminative Models which take images as input and provide the most probable class for those images. These types of Classifiers confront us with an arduous challenge, which is to interpret their decisions. To this date, there are several interpretation methods to better comprehend deep models, such as Saliency Maps and Attribution Methods [2]. Even though these methods are able to provide some visual intuitions on the behaviour of a Deep Image Classifier, they unfortunately face some issues worth pointing

out. In the case of Saliency Maps, saturated gradients may occur and threaten the visual insights produced [2] (explained in the state of the art section). As for Attribution Methods, some rely on the adjustment of new hyper-parameters, which hinder the end result of mirroring the Classifier's classification process, and others have their visuals constrained to some architectures. Another relevant aspect to mention is that both the Saliency Maps and the Attribution Methods rely only on the images available from a training set to interpret the Classifier's behaviour. Furthermore, these methods lack immediate comprehension, making it hard to be convincing to a regular human being. We would like to demystify, in an intuitive way, why the multiple chaining of mathematical expressions, which encompass Image Classifiers, helps us solve difficult problems. Therefore, is there a way to introduce a more intrinsic approach into analysing these complex models, by sampling the classes of an Image Classifier? Maybe by using collections of generated images we can visualise what each class represents from the Classifier's perspective. This could be seen as a more intuitive procedure to analyse the predictions a Classifier is providing, rather than looking at feature maps which requires more expertise. (See Fig 2 for an abstraction view of the spacial idea of classes within a Classifier)

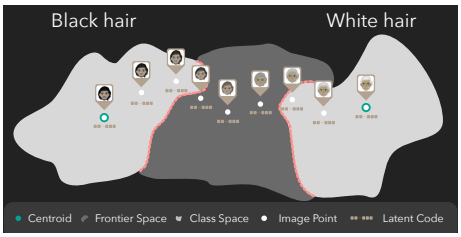


Fig. 2: Classes spaces - 2D Abstraction

To pursue the idea presented, we propose a different approach of feeding images generated by Generative Models to an image Classifier, to have a better sense of its functioning. The motivation for using Generative Deep Models comes from their ability to generate new artificial data points (images), which create a useful playground to experiment with. With this playground of images we can explore in depth, not only the region of image space belonging to one class but also the frontiers between classes. These regions and frontiers can be found somewhere "in between" the examples of the training set and the new generated images from the Generative Models. The strength of this approach is that it enables some key features that are not possible with the methods previously discussed: the mapping of a manifold [1] , analysis of the classes, and a search for inter class frontiers.

When an Image Classifier receives an image as input and outputs a distribution of probabilities of belonging or not to the classes it was trained to dis-

[1] manifold: compact representation of a higher dimensional space

tinguish, it is portraying to us a partial idea of what it means to belong to each class, within its n-dimensional space. In search of an easier way to understand the Image Classifier's dimensionality (an n-dimension representation of its classes), a simple approach would be to continuously feed it with different images, and analyse the output. However, this restricts the insights that can be produced. The bottom line is that we want to understand what the Classifier is internally doing to reach a certain prediction. Thus, it would be useful to have at our disposal more insightful images, which could be obtained by generating artificial images mirroring what we expect from the Image Classifier. Understanding what images belong to a class or what makes the Classifier hesitant between classes it what, ultimately, allows us to understand its view of the classes. This is a question this project tries to answer, while using a Classifier cooperatively with Generative Models.

As there are many Generative Models, the choice of applying a Generative Adversarial Network (GAN) and a Variational Autoencoder (VAE) in this work comes from the quality of their results, and the meaningful latent representation of the latter. In using GANs, we intend to explore the outputs, and even find unintended biases of the images generated. The training of the GAN will be assisted by a Classifier pre-trained on the same dataset. With the VAE, we want to create a latent representation of a dataset and then use it to explore the regions that the Classifier assigns to each class. The usage of the Classifier with the VAE will only take part after the training process of the Generative Model. The actual implemented versions of both architecture's were a WGAN-GP and a vanilla VAE.

### 1.2 Objective

Summing up, we want to determine whether Generative Models can help us understand what led to the predictions of an Image Classifier. The motivation for this lies in providing a more intuitive methodology to understand the Classifiers Predictions, with the aid of generated images. Also provided Generative Models help, this approach could be useful not only in detecting inconsistencies in the Classifier's predictions, but also in improving Image Classifier architectures by feeding them generated images from our models. This idea could act as a replacement, or even as a complement, to other well-known visual alternatives regarding the interpretation of Neural Networks. To this extent, a library has been developed to aid the study of the problem in hand. This library is intended to be a user-friendly tool-set, prepared to generate artificial image datasets based on a set of real images and equipped with tools that tried to give insights on whether Generative Models were helpful in understanding Image Classifiers. For further improvements in the quality of the generated images and to allow the exploration of higher resolution images, we set ourselves the goal to preliminarily implement both an Introspective Variational Autoencoder and a Progressive Growing GAN.

## 2 State of the art

Deep Convolutional Neural Networks allow Image Classifiers to find patterns that otherwise would be extremely hard, if not impossible, to find. The use of convolution helps to preserve spacial information so as to find 2D Patterns. However, these layers stacked on top of each other, allied with fully connected layers and mid term operations, make it extremely hard to understand how these networks work.

Some approaches to the interpretation of Convolutional Neural Networks have relied on Saliency Maps and Attribution Methods. Saliency Maps can be regarded as a tool to highlight which set of pixel regions are relevant to the prediction of a classification. More specifically, Saliency Map is a method which calculates the gradients of some input features in a search for insights on the direction of maximum increase. The problem arises when these inputs are ranged over small scaled values and, therefore, changing pixel values by a minuscule amount results in no apparent change in what a network learns. This method is also reckoned to be extremely noisy, though it is thought that this noise may be regarded as a positive noise, also known as informative noise. On the other hand, we have the Attribution Methods, a more general approach to Saliency Maps whose main focus is assigning an attribute/relevance value to each input feature of a network. These Attribution Methods are normally visualised as heatmaps, with the same shape as the original image. The Attribution Methods can be categorised in some key variants: some have attempted to alleviate saturation by estimating the global importance of each pixel (eg: IntegGrad [3]), while others have attempted to smooth discontinuous gradient with a Gaussian kernel (eg: SmoothGrad [4]). Finally, some attempted to remove the negative gradient during backpropagation of the models (eg: Guided BP [5]). Like Saliency Maps, Attribution Methods are able to provide a visual understanding on the functioning of a Deep Image Classifier. Nonetheless, they suffer from some issues too. The Attribution Methods alternatives presented actually produce some solid results in providing some visual guidance in what a Classifier is internally using to classify an image, but they end up facing other problems. One of these problems is the additional workload of feature engineering new hyper-parameters, and another is that they simply fail to provide meaningful results when compared with other methods.

In 2019, a new state of the art Saliency Map method was introduced, the Rectified Gradient [6]. In contrast with a basic Saliency Map method, RectGrad is able to provide a much sharper idea of the feature maps. However, this method faces a niche problem, which is its dependence on the usage of the ReLU activation function between layers. To this date, several activation functions are popular and used interchangeably (eg: LeakyReLU, ELU, Tanh, etc), so the applications for this method may run quite short on many networks.

In 2014, the Generative Adversarial Network [7] (GAN) was introduced as a new Generative Model, reflecting outstanding results on image datasets of lower resolution. Generative Adversarial Networks rely on an adversarial game in which two players co-exist. The players, a generator and a Discriminator/Critic,

play against each other with opposite goals. The Critic's objective is to learn techniques to better distinguish a real distribution of data from a synthetic one, whereas the Generator's goal is to learn techniques to fool the Critic into thinking the synthetic distribution is the real one. In Game theory, this kind of game is known as a minimax game. Wasserstein GAN [8] was later introduced with the main goal of providing a better mathematical meaning to the optimisation of the approximation between the distribution of real images and the generated ones. In order to ensure the strong constraints necessary to the Wasserstein distance, the vanilla WGAN sacrifices the learning of complex functions. As such, an improvement was suggested named WGAN-GP [9] which penalises the Critic if the gradients calculated do not conform to the norm of one. Regular GANs yield promising results only when constrained to lower resolution images such as 128x128 sized images. A technique named Progressive growing GAN [10](PGGAN) has introduced a new approach based on training a Critic and a Generator together on a small resolution image, and then increasingly adding and training Generator/Critic combos on larger and larger images of the same dataset. This last method is quite popular nowadays when dealing with large scale images due to its outstanding performance.

GANs, however, lack in the relevance of their latent space. The Variational Autoencoder [11] (VAE) is more interesting in that respect. As the VAE model learns a lower dimension representation of the real data, it is able to extract the main features which define the real data distribution. By interpolating these features and feeding them into the Generator of a VAE, it is somewhat possible to have control over the generated data points.

As stated previously, vanilla Variational Autoencoders are great at providing a meaningful latent space to explore the attributes of a certain domain. However, they lack in the sharpness of the images generated and even in the disentanglement, which often could be better. Disentanglement could be regarded as all the dimensions of the latent space being independent, this is, each unit of the latent vector being responsible for only one generative factor. Introspective VAE [12] (IntroVAE), introduced in late 2018 provides sharp and vivid results. It could be seen as an hybrid model between GANs and VAEs, taking advantage of the more prosperous characteristics of both and merging them into a simple, but effective, model. To be more specific, the model is inspired by the generative component of a normal GAN and by the inference component of a regular VAE. This way, it can not only create stunning artificial images, but also provide tools to easily manipulate the images to be generated.

## 3 Datasets

Throughout the process of this project, we tested and worked with several different datasets to help us achieve the intended expressiveness of our goals. We used Fashion MNIST dataset (First image of Fig. 3), at a first stage, to learn about generative modelling and to make simple visual sampling experiments so as to have an idea on what could be possible to achieve with more complex

datasets. The LSUN-bedroom dataset (Second image of Fig. 3) proved itself as a fundamental dataset as it was able to give us solid guidelines on whether our architectures were performing correctly or not. The reason for this, is because most papers regarding Generative Models use this dataset as a benchmark. So, visually one can compare our results with those from other papers and adapt the models accordingly. The cartoon dataset (Third image of Fig. 3) was relevant due to having many different labelled attributes, facilitating the creation of a Classifier to study and also to generate images by manipulating or filtering those attributes. We used this dataset to both study the convergence of classes and frontiers exploration between classes, as it will be shown in following sections. We also used a dataset of trains (Fourth image of Fig. 3), which was developed at our institution and which has deserved attention by several students researching interpretation of neural networks. This was used prior to the Cartoon Dataset, as some pre-trained Classifiers already existed and because the dataset was well organised and ready to feed the models without the need to work on preparing the dataset. This allowed us, at a first stage, to work on what was relevant to our project such as focusing on the Generative Models and the correspondent mutations to them to better understand the Classifiers.

Regarding the image pre-processing aspect for these datasets, we prepared the images to be fed to the Generative Models. This consisted in rearranging the datasets into sets of zip files easier to digest and read from. Once the data was loaded and before feeding it to the models, standardisation in the case of GAN, and normalisation in case of VAE would be needed. The deep convolutions and dense layers take care of mapping and extracting the important features.

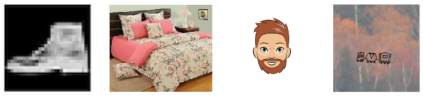

Fig. 3: Image samples from the datasets used

For further information about the datasets see attachment in section 1.

## 4 Experiments and Results

Our approach is based on a playground of Generative Models, namely GANs and VAEs, and pre-trained Classifiers. In the following subsections we describe in more detail our experiments with both the Generative Models and Classifiers. For some experiments we had to train our own Classifiers, for others we used pre-trained Classifiers on the datasets.

Regarding the GAN, to explore the understanding of each class and what stood between them through the eyes of a Classifier we tweaked the standard loss function of the generative component in two different ways. One way was to

add a penalisation to the generator if the image it generated belonged to a class which was not our target. The other approach involved rewarding the model if the prediction tagged to the image generated followed a certain distribution we had defined as target. The first method is insightful, not only in terms of enabling the model to generate images of a certain class, but also to visually understand what a Classifier thinks a certain class is. The second method, on the other hand, allows a visual exploration of the latent space where the frontiers of different classes meet, or even other complex spacial relationships.

With the VAE we propose two different approaches to visually represent the frontiers between two or more classes. One was to optimise the latent space so that when feeding it to a generator it would give us images of a certain class. The other was to interpolate from a region of a class to another, mapping this transition with images so it could easily be spotted what was visually meant to be in the frontier of those classes.

Lastly we present a Graphical User Interface (GUI) application created to manually classify generated images. We suggest that cycling between the training of a Generative Model, manually classifying the images generated using this GUI and retraining the Classifier with these new images might end up with a Classifier whose classes correspond to a closer idea of what they should be.

### 4.1 Class Convergence with a GAN

On this experiment, we wanted to produce images of a certain class, depicted by a Classifier. To do so, a Classifier was positioned in parallel with the normal architecture of a WGAN-GP. Thus, every image generated during the Generator's training process would be classified. The class predicted by the Classifier and the realness score assigned by the Critic would influence the model towards images that are both realistic and of a certain class. (See Fig 4).

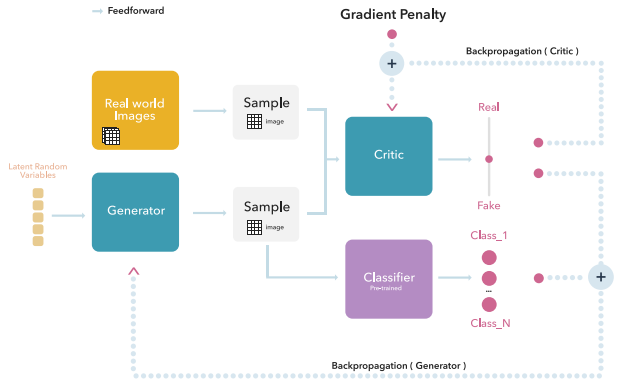


Fig. 4: Training process of WGAN-GP with Classifier embedded

The generator loss can therefore be written with the equation:

$$L_G = \frac{1}{n}\sum_{n=1}^{n}(y * y\_gen) + k * \frac{1}{n}\sum_{n=1}^{n}\sum_{c=1}^{C} 1_{y_i \in Cc} log P_{model}[y_i \in Cc] \quad (1)$$

Where: n is the number of images; C is the number of classes; k is a hyper-parameter to control the impact of the penalisation of generating an image of the wrong class; y corresponds to a real image; y_gen corresponds to a generated image. The left term represents the standard WGAN-GP generator loss; The right term is the categorical cross entropy representing the penalisation if the predicted classification is different from the target class.

This experiment had three main goals: we wanted to have a visual idea of each class through the eyes of an Image Classifier; find if the generator would indeed converge to images of a target class; and possibly find generated images wrongly classified (bias). For design and creative purposes this can be helpful, granted there are no biases in the concept that the Classifier holds on each class.

It is key to highlight that using a Classifier to influence the generator, will not necessarily lead the generator into generating images a regular human would expect from a certain class, but rather give us a set of images the Classifier places in that class. These can either be in accordance or not with our concept of a certain class, but ultimately, it provides us evidence for some bias. Supporting these claims, in our experiments we found that the grand majority of images generated fall in the spectrum of unwanted bias. The perception of the classes the Classifiers hold is not as straightforward as the training metrics sometimes suggest. This finding might raise awareness of introspective approaches to building models such as Image Classifiers. Building Image Classifiers has become facilitated with existing frameworks such as Tensorflow or Pytorch. This is desirable as it enables more people to work with powerful tools, without a high level of expertise. However, deploying biases models in the real world can have a real impact in the lives of people, therefore, using methodologies such as this one might help analyse whether a model is ready for production or still needs some tuning.

We trained a Classifier over the hair colour of cartoon figures, from the cartoon dataset. This Classifier predicts the probability of a figure having a certain hair colour over 10 possible colours. Choosing the Generator to converge to the class of white hair, we have the visuals seen in Fig. 5:

Looking at Fig. 5 we see that all images generated follow the class chosen (White hair). However, whereas we would be expecting the cartoon figures to have white hair, in this figure we see several images classified as white hair, but having other hair colours. This happens because the Generative Model is exactly mirroring the Classifier perception of the white colour, which as it seems fails to match ours perception. This example presents itself as a good example to show how useful this method can be in finding biased in a Classifier and even have a concise idea of what a Classifier thinks each class is.

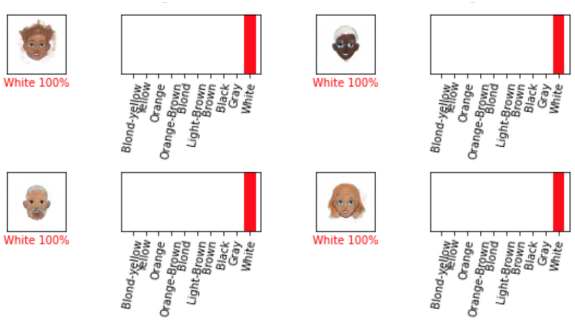


Fig. 5: Images generated and Classifier predictions

### 4.2 Frontier Exploration with a GAN

To help visually understand the frontiers between classes we used a GAN and defined a target distribution we wanted the Classifier to output. We would then penalise or reward the generator component whether the image would follow the target distribution or not. To this extent, we added the Kullback–Leibler Divergence to our generator loss function, because it is a measure of how one probability distribution is different from a second, depicting the mathematical distance between the target distribution and the predicted one.

Using the Kullback-Leibler Divergence, the generator loss of the GAN can be written with the equation:

$$L_G = \frac{1}{n}\sum_{n=1}^{n}(y * y\_gen) + k * \sum_{i=1}^{n} p(x_i)\log(\frac{p(x_i)}{q(x_i)}) \tag{2}$$

where: $p(x_i)$ represents the target distribution and $q(x_i)$ the real distribution; n is the number of images; y corresponds to a real image; y_gen corresponds to a generated image; k is a hyper-parameter to control the impact of the reward of generating an image with the right probability distribution. The left term represents the standard WGAN-GP generator loss; The right term is the Kullbacl-Leibler Divergence representing the penalisation if classification prediction is different from the target distribution.

This experiment is a natural follow up to the previous one. It is not an unthinkable scenario that when thinking about two classes we often conceive a mental image of how a middle ground could be depicted. Providing an example, when reflecting upon the colours black and white, it can be acknowledged that conceiving the colour grey as being somewhere in between both colours is quite likely regarded as being correct. So the motivation for this technique is an introspection of whether a certain model is preserving a cohesive relationship between classes one would be expecting. Also, apart from preserving any relationship between the inter-class frontiers, it can be a valuable exercise to self

introspect what visuals make the most sense being depicted in the transition of classes.

We ran some experiments using several datasets, trying to depict and analyse what it means to be 50% of one class and 50% of another. The following result represents the visuals when finding a meeting point between t-shirts and trousers:

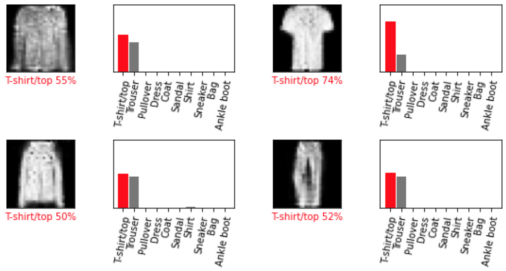


Fig. 6: Frontier between t-shirt class and trouser class

Our findings show that this approach does not guarantee a perfect fit to the target distribution, in this case scenario, of belonging 50% to one class and 50% to another one. Nevertheless, even though not all images generated fall on the target distribution, the ones that do either show insightful biases, depicting something we would not be expecting, or they do show a creative morphing of both classes proving a valuable source of analysis and artistic work. In Fig. 6 it is interesting to see that a shirt with long sleeves seems to be closer to trousers, even though this is just an assumption.

### 4.3 Frontier Exploration with a VAE

Regarding the Variational Autoencoder, we had two different ideas in mind for attempting to portray the frontier space separating 2 or more classes. Both ideas relied on pre-training the VAE on a real dataset of images, for it to build a meaningful latent space. This latent space is essential when working with VAEs, because it represents the images attributes in a condensed form, making it easier and more intuitive to tweak and understand the images generated (See Fig. 7 to visualise the VAE experiment process)

The first idea consisted in discovering the latent space mapping the frontier between two classes, by optimising the latent vector into providing us an image, which when classified would give us, for instance, 50% chance of belonging to one class or another. As an example, using the cartoon dataset to classify the gender(male/female) of the cartoon figures, we could mathematically think of the equation (3) as being the function we are trying to solve.

$$Gender\_Classifier(vae.generator(latent\_vector)) - 0.5 = 0 \tag{3}$$

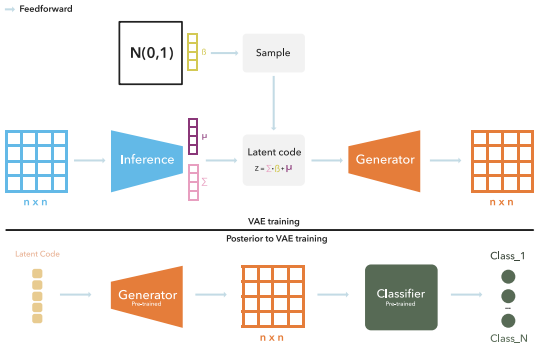


Fig. 7: Training process of VAE and Classifier usage

Generally speaking, the idea is to find the latent vector space, which generates images conforming to a certain distribution of probabilities (the probability of belonging to each featured class), as it can be seen below:

$$Target = [.0, .0, ..., .33, .33, .0, .33]$$

This approach sounds solid and useful to create a visual mapping of the classes' frontier in theory, however, mathematically speaking it is not straightforward. We tried at first using SciPy optimisation toolkit to compute this problem, but unfortunately no solid results were achieved. A reason for this is that optimising a complex function such as the equation (3) is not computationally feasible, due to the derivatives needed to be calculated. Consequently we did not pursue this idea, though theoretically promising.

The second idea was an approach consisting in gradually generating a set of images from a certain class to another, thus mapping the visual space between two or more classes. To achieve this idea, we based this methodology on the usage of a Centroid, which is the classes dataset's latent space centre of mass. To do this we would find a Centroid of a class or multiple Centroids from samples from a certain class and then move the latent vector in the direction of another Centroid/set of Centroids. Using several Centroids is more powerful as it enables the creation of a denser visual frontier mapping, making it a more meaningful and insightful representation of the image space of the classes' frontiers.

It is important to note that there are variants of Variational Autoencoder, which grant fairly well independent segregation of the images' multiple attributes. Having independent features makes it easier to navigate the latent space and ultimately meticulously scrutinise the classes in a more intuitive manner. This is possible because only by changing a certain value in the n-dimensional vector of the latent space you might be moving over the space of a class. With a network such as ours we cannot guarantee this attribute's independence, therefore for topologies such as this one, our approach may be a good way to visualise the classes.

The following figures attempt to portray the transitions between classes. These examples make use of a hair Classifier pre-trained on the dataset and the classes chosen were the white colour and black colour. (Fig. 8) shows the interpolation between black and white hair colour and (Fig. 9) shows the correspondent probabilities of belonging to each class chosen during the same interpolation.

Fig. 8: Interpolation of classes: Black hair colour → White hair colour

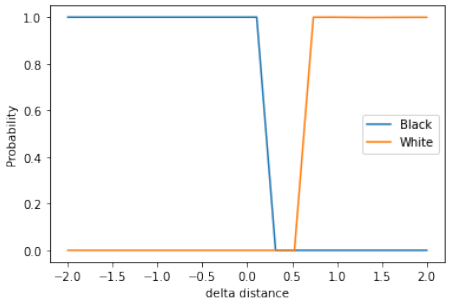


Fig. 9: Probability of the images belonging to Black colour vs White colour during interpolation

To sum up, in this experiment we wanted to have a better idea of what kind of figures would stand between black hair colour figures and white hair colour ones. To do this we find the centre of a class and then move through a hyperplane of the latent code until we intersect the centre of another class. As we lack independence in the attributes' mapping, though the hair colour changes consistently, the facial visuals vary with a certain entropy. This is expected because of the lack of independence within the latent space created by the VAE. It is also worth noting, that the colour which stands between the black and white one is grey like we would expect.

### 4.4 GUI - Manual Classification

In order to try to prove that our system not only helps finding biases in a Classifier, but can also can be a tool to improve a Classifier's classification process, we created a visual interface to manually classify the generated images. The interface prompts to the user images generated by a Generative Model and asks the user to classify these images according to the possible classes associated to the

Classifier in question. Once all images are classified, two zip files will be created. One will contain the images, whose classification of the user did not match the predicted one by the Classifier. The other one will store the correspondent new labels to the images on the other file. The goal with this output is to re-train the Classifier with these new images. Supposedly, this retraining could end in a better approach to what we believe a certain set of classes represent to us.

To test this hypothesis, as a proof of concept, we had a Classifier trained on the hair colour of Cartoon Dataset. Then we fed the GAN, embedded with an hair Classifier, with a set of images from the dataset. We trained the GAN for 200 Epochs targeting the convergence of the images to the white hair. After training, we extracted 500 generated images and manually classified the images using the simple GUI application we created (See Fig. 10). From manually classifying the artificial images, we analysed that from those 500 images 437 of them did not have white hair. Collecting these new 437 conflicting images and their new labels, we trained the hair Classifier with these new images until we reached a similar accuracy to the one the Classifier had before. After the hair Classifier retraining process we moved back to the GAN and repeated the training process but with this new Classifier. Then we manually classified the generated images again and compared the number of images which were wrongly classified by the Classifier. This second time, we only got 420 images classified as not white hair. There is evidence of some improvement (17+ images classified as white), however a more intense experiment would be necessary to extract any conclusion. It could be the case that this is helping guide the Classifier to our conception of the classes, or it could just have been by chance. It is worth it to highlight that this experiment was just a preliminary test, as this was not the main focus of our research. It is possible that the Classifier is suffering from overfitting, but even if it is this methodology helps spotting errors in the Classifier.

Multiple iterations might help the Classifier's classifications to start sharing a more similar idea to the general perception of the classes being evaluated. We did not explore this concept much further than this, thus leaving this hypothesis open to further work and follow up experiments on the topic.

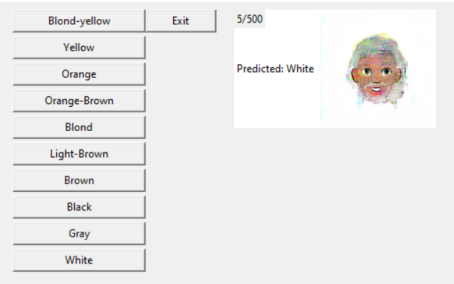


Fig. 10: Cartoon dataset using a hair colour Classifier — GUI application Example

## 5 Library - Generate_Datasets

One of goals of this project was to create a friendly level abstraction regarding the generation of artificial images so anyone could play and use this library to generate images and even extract visual insights of their Classifiers. The library was named **generate_datasets** and is open source, available on github:

– Github repository: `https://github.com/Mr-Vicente/generate_datasets`

It contains all the files which helped to process images, generate artificial images, explore the Classifiers and more. The Github Repository also contains some tips when working with generative models, tensorflow, google colab and processing data. Along with the python files, .ipynb files were also made available so one can easily jump on to Google Colab and run the code without any computational constraint. The readme file tries to portray how the library can be used and present a brief summary of its architecture and results that can be achieved with it.

We did the experiments solely with a WGAN-GP and a vanilla VAE, however we provide other GAN and VAE implementations. Having studied the state of the art regarding Generative Models, we were aware of much more robust models able to generate high definition images. Adding such models to this library would make it much more flexible, open to a bigger audience of datasets to scrutinise, along with Classifiers. To this extent, we made an approach to implement both a PGGAN and an IntroVAE, providing first preliminary versions of them.

It would be interesting to publish our library on the Pypi library database, so it can be easily installed using pip and therefore making it easier to use.

## 6 Conclusions and Future Work

We propose that image generators can be trained to provide insights into the workings of image Classifiers by targeting the generator to specific Classifier outputs, answering the main hypothesis presented. The results show that classification errors can be visualised using this methodology and we suppose that using a collection of chosen generated images we can improve Classifiers' classifications. We provide a user-friendly library to generate synthetic images and explore Classifiers' reasoning. We would like to remember the potential that this cooperative work between a GAN and a Classifier can create. It enables a more intuitive approach in providing useful visual information about the behaviour of a Classifier, when compared with other alternatives. It is worth mentioning that, ultimately, our method and the alternatives combined might provide a more complete analysis on a Classifier reasoning.

Future work regarding this project can come in many forms. One could be the study of whether a cycling approach of retraining a Classifier with corrections provided from the Generative Models and their labels can help the Classifier improve its classifications. Also, it would be interesting to take this work and expand it to higher resolutions so it could have a further improved outcome in

the generation of content. We explored Classifiers using the media of images, but it would be relevant to try to achieve similar results with different types of datasets such as music, text, or even other unstructured data. Another thing to mention is that there are some evaluations that can be done to Generative Models to evaluate and analyse the quality of their output samples. One recent popular metric is the *Frechet Inception Distance*, which calculates the distance between latent vectors distribution between the real and generated images. To ensure the quality of the results, one could analyse the scores of the models we presented and improve our models accordingly to the scores achieved.

## 7 Acknowledgements

I thank Professor Ludwig Krippahl, Matthias Knorr, Ricardo Gonçalves and João Leite for their continued help and feedback over the course of the development of the project to this date. Finally, I thank Manuel Ribeiro for providing the Trains Dataset and the corresponding Classifiers pre-trained on the dataset.

## References

1. Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934*, 2020.
2. Pascal Sturmfels, Scott Lundberg, and Su-In Lee. Visualizing the impact of feature attribution baselines. *Distill*, 2020. https://distill.pub/2020/attribution-baselines.
3. Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks, 2017.
4. Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. Smoothgrad: removing noise by adding noise, 2017.
5. Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net, 2014.
6. Beomsu Kim, Junghoon Seo, Seunghyeon Jeon, Jamyoung Koo, Jeongyeol Choe, and Taegyun Jeon. Why are saliency maps noisy? cause of and solution to noisy saliency maps. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pages 4149–4157. IEEE, 2019.
7. Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014.
8. Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan, 2017.
9. Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. Improved training of wasserstein gans, 2017.
10. Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation, 2017.
11. Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2013.
12. Huaibo Huang, Ran He, Zhenan Sun, Tieniu Tan, et al. Introvae: Introspective variational autoencoders for photographic image synthesis. In *Advances in neural information processing systems*, pages 52–63, 2018.