Synthetic Data Generation To Better Perceive Classifier Predictions

Frederico Vicente

Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa, Quinta da Torre,
2829-516 Monte de Caparica, Portugal
fm.vicente@campus.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 evidence on the role a set of pixels play in an image. Are all regions of pixels relevant, or are there some more relevant than others? Is the model able to extract more meaningful information from specific pixels? Some approaches regarding this problem take Saliency Maps on differential Classifiers to extrapolate possible meaningful visual representations [1]. These methods require the calculation of gradients, which are associated with some issues, such as saturation (e.g., small magnitude values). We propose the use of Generative Models to better understand the frontier of each class through the eyes of a Classifier. We intend to build a Generative Adversarial Network [2] (GAN) with a pre-trained Classifier to influence the Generator into generating images of a certain class, so one can have a visual portrayal of each class from the classifier's perspective. We also expect to experiment with a trained Variational Autoencoder [3] (VAE) to analyse the region of the latent space which resembles the frontier between two different classes, to better understand the Classifiers view of what each class visually represents.

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

1 Introduction

1.1 Objective

We want to better perceive the predictions of an image Classifier using Generative Models. To this extent, a library will be developed to aid the study of the interpretation of a Classifier. This library shall result in a user-friendly tool-set ready to generate artificial image datasets based on a set of real images.

1.2 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 to 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 typologies.

Image Classifiers are a sub-type of Classifier 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 Integrated Gradients [4]. Even though these methods are able to provide some visual intuitions on the behaviour of a Deep Image Classifier, unfortunately, they face some issues worth pointing out. In the case of Saliency Maps, saturated gradients may occur and threaten the visual insights produced. As for Integrated Gradients, they rely on the adjustment of a hyper-parameter, named *baseline*, and depending on its value/set of values it may lead or not to a useful representation of the Deep Neural Network's cognitive process. Therefore, we share a different approach by feeding images generated by Generative Models to an image Classifier, to have a better sense of its functioning. The motivation for utilising Generative Deep Models comes from their ability to generate new artificial data points, which create a useful playground to experiment with.

Another relevant aspect to mention is that both the Saliency Maps and the Integrated Gradients methods rely only on the images available from the training set to interpret the Classifier's behaviour. In contrast, by generating new data points we can explore, in depth, not only the region in space belonging to one class but also the frontiers between classes. These regions and frontiers can be found somewhere "in between" the examples in 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 aren't possible with the methods previously discussed: the mapping of a manifold, analysis of the classes and a search for inter class frontiers.

As there are many Generative Models, the choice of applying a GAN and a 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 as the GAN. With the VAE, we want to create a latent representation of a dataset and then use this representation 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.

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 the architecture's reasoning.

Some approaches to the interpretation of Convolutional Neural Networks have relied on Saliency Maps and Integrated Gradients. Saliency Maps can be regarded as a tool for computing the sensitivity of the Classifier's decision with respect to each input (e.g., pixel) dimension. On another hand, we have the Integrated Gradients method, whose main focus is understanding the features' importance in a data point. (See [4] for an insightful and interactive article). These methods are able to provide a visual understanding on the functioning of a Deep Image Classifier, however they suffer from some issues. Saliency Maps 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. Integrated Gradients is an attempt to overpass the saturation of local gradients by integrating the gradients over a path. However, the introduction of a hyperparameter, the *baseline*, originates a new question to answer, which is to find the right value/set of values which provide a helpful representation of a Deep Neural Network's cognitive process.

In 2014, the Generative Adversarial Network (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 [5] 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 [6] 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 64x64 sized images. A technique named Progressive growing GAN [7] 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 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.

3 Tools

Spyder IDE will be used to develop the library and to write the documentation, whereas Google Colab will be used to experiment with the models. This project will be developed having in mind code clarity, flexibility, structure quality and best practices.

The library will be written in Python using the novelties of the newest version of Tensorflow (2.1) and other relevant frameworks for manipulating and storing data. Tensorflow is an open-source library which provides powerful tools to build and deploy Deep Neural Networks.

To test the library models we will be working with a dataset of trains created at FCT NOVA. These images have 3 colour channels and their size is 152x152.(see Fig. 1)

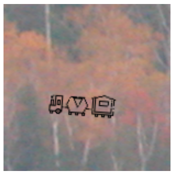

Fig. 1: Sample image from the train dataset

4 Methodologies

Our approach will be based on a playground of Generative Models, namely a GAN and a VAE, and a pre-trained Classifier.
At first, the experiments will involve manipulating a WGAN-GP. A Classifier will be positioned in parallel with the normal architecture of a GAN. Thus, every image generated during the Generator's training process will be classified. The class predicted by the Classifier and the realness score assigned by the Critic will influence the model towards images that are both realistic and of a certain class. (see Fig. 2).

Whereas with a GAN we intend to put the Classifier in the training process, when using a VAE we will first train it and later explore the generated output with the pre-trained Classifiers' predictions. Due to its architecture, the trained VAE will allow us to explore the regions in space that have equal probabilities of belonging to one class or another. This approach enables us to gain a visual

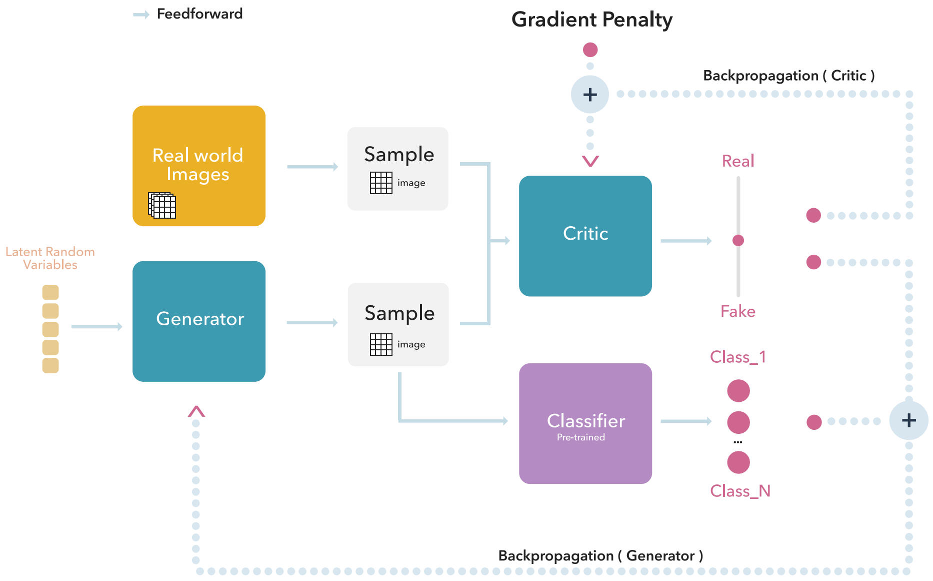

Fig. 2: WGAN GP Architecture with a pre-trained Classifier

intuition of the classes' frontier and build a perceivable representation of it. This can be done by fixing a latent code, which will correspond to a class, and then moving through the hyperplane of the latent code until we intersect a different class. With this information we might be able to map the classes' frontiers and extract the meaning of each class through the eyes of the Classifier. (see Fig. 3).

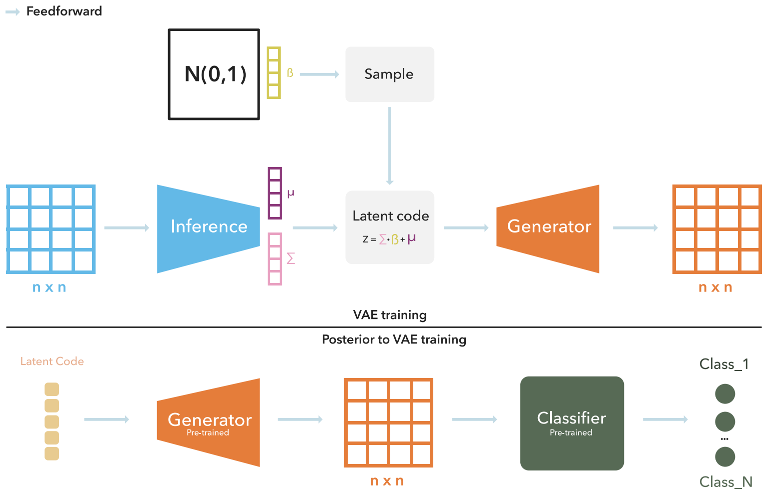

Fig. 3: VAE Architecture and its usage with the Classifier

5 Work Plan and Scheduling

The project will be split in 4 main stages which are described bellow:

1. First Stage, Part I (Prepare personal research workstation):
   (a) Trello - Research repository;
   (b) Github - Library repository;
   (c) Overleaf - Notes and final report repository.

2. First Stage, Part II (Search for relevant material which will help achieve the objective):
   (a) Scientific Papers;
   (b) Mathematics and Probability insights;
   (c) Textbooks with theory that is easier to digest;
   (d) Information on how to develop a Python library using best practices.

3. Second Stage (Prepare models):
   (a) Understand the theory behind GANs and VAEs;
   (b) Learn and gain intuition regarding variational inference, latent space, other network architectures which might be useful later, overall machine learning theory;
   (c) Explore State of the Art Generative Models;
   (d) Primordial Design of the library architecture (name, modules, classes);
   (e) Design a diagram of the models' architecture;
   (f) Learn Tensorflow 2 while trying to implement both Generative Models in their simplest form, but always trying to use best practices;
   (g) Improve previous Generative Models using ideas from State of the Art papers;
   (h) Build and train a Classifier for Fashion MNIST;
   (i) Test models with Fashion MNIST dataset;
   (j) Write first part of final report.

4. Third Stage (Experiment with the models):
   (a) Study the Dataset of Trains (152x152 images);
   (b) Upgrade models' architecture to load the Dataset of Trains;
   (c) Improve previous models if necessary. For instance, improve WGAN-GP to a Progressive Growing GAN;
   (d) Write down all experiment plans;
   (e) These experiments should attempt to better understand the predictions of the Classifiers;
   (f) For all experiment plans write and analyse their results;
   (g) Experiment 1 (example with GAN): For every epoch of training, store a set of images the classifier assigned to the chosen class, so as to see the evolution of the perception of a class;

   (h) Experiment 2 (example with VAE): Explore the frontier between two pairs of classes, adjusting the latent representation values, so as to find the region in space that corresponds to 50% of one class and 50% of another;
   (i) Update final report with further work completed.

5. Fourth Stage (Finishing touches):
   (a) Clean up the code;
   (b) Write the documentation for the library;
   (c) Finish final report.

The schedule can be seen in Fig. 4.

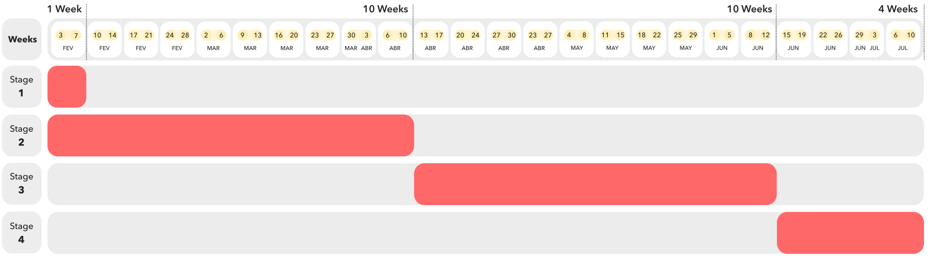

Fig. 4: Predicted Schedule

6 Acknowledgements

I would like to thank Ema Vieira for her help in reviewing this intermediate report. I also thank Professor Ludwig Krippahl, João Leite, Matthias Knorr and Ricardo Gonçalves 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. Chang C, Creager E, Goldenberg A, Duvenaud D.: Explaining Image Classifiers by Counterfactual Generation. arXiv preprint arXiv:1807.08024v3 (2019)
2. Goodfellow, I.: NIPS 2016 Tutorial: Generative Adversarial Networks. arXiv preprint arXiv:1701.00160v4 (2017)
3. Kingma D, Welling M.: Auto-Encoding Variational Bayes. arXiv preprint arXiv:1312.6114v10 (2014)
4. Distill Journal Article, https://distill.pub/2020/attribution-baselines. Last accessed 3 Mar 2020

5. Arjovsky M, Chintala S, Bottou L.: Wasserstein GAN. arXiv preprint arXiv:1701.07875v3 (2017)
6. Gulrajani I, Ahmed F, Arjovsky M, Dumoulin V, Courville A.: Improved Training of Wasserstein GANs. arXiv preprint arXiv:1704.00028v3 (2017)
7. Karras T, Aila T, Laine S, Lehtinen J.: PROGRESSIVE GROWING OF GANS FOR IMPROVED QUALITY, STABILITY, AND VARIATION. arXiv preprint arXiv:1710.10196v3 (2018)