Deep Learning Project report: Conditional DCGANs

Filippo Bordon
University of Trento

Olivier Nicolini
University of Trento

Simone Zamboni
University of Trento

Abstract

Deep Learning project course focused on the implementation of a conditional DCGAN able to generate faces based on labels.

1. INTRODUCTION

1.1. GANs

Since their introduction in 2014[1] , Generative Adversatial Networks(GANs) have became one of the most popular generative models thanks to their flexibility and their ability to create visually appealing results.
A GAN is composed by two networks that are trained together: a generator that takes as input random noise and outputs a sample (in our case an image), and a discriminator, that takes as input both real and generated samples and has to determine if the sample is fake or not. The objective of the generator is to fool the discriminator and the one of the discriminator is to correctly label the data that is fed with. We can observe all of this in the loss function of a GAN:

$$\min_{\theta_g} \max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$

Figure 1. The loss of a Generative Adversarial Network.

1.2. Conditional GANs

Datasets of real samples are used to train GANs. Some of of them have also labels, that could be used to generate samples with predetermined characteristics: architectures of this type are called *conditional GANs*.
Conditional GANs have the same architecture and the same loss of a standard GAN, the only difference resides in the fact that label information is added both to the discriminator and generator input. Therefore, the generator has to create samples that respect the labels it is fed with in order to better fool the discriminator, which is fed with the same labels.

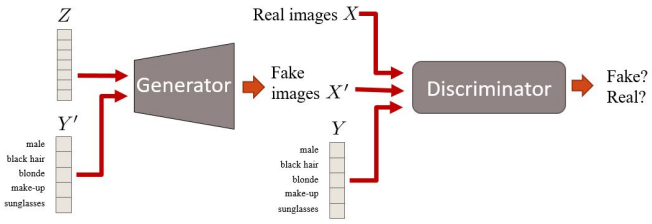

Figure 2. General Conditional GAN architecture.

2. Challenge definition

2.1. Objectives

Our objective in this project was to design a GAN able to produce images of faces conditioned with different labels. More specifically we wanted to take an already existing architecture, modify it and obtain better results generating faces conditioned by some specifics attributes, like male/female or black hair/non black hair.

2.2. CelebA dataset

For our project we used CelebA, a widely used dataset for faces, that contains more than 200k faces of famous people. The images used are cropped on the faces, and re-sized at 64x64 pixels. Each one of these images has 40 binary labels, for example:

- Male
- Attractive
- Blonde hair
- Smiling

This is a very challenging dataset for the variety of the faces orientation and occlusion.

2.3. Starting Architecture

As starting architecture a modified version of the DCGAN that uses one label in the one-hot encoding was used.
The generator takes as input the noise vector and a vector of two elements where the label is encoded in one-hot fashion.

Both inputs are deconvoluted separately, concatenated and then deconvoluted together until a 64x64 image is obtained.

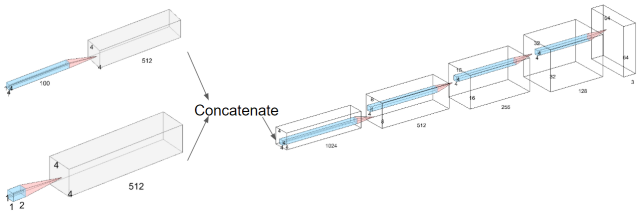

Figure 3. Generator of the starting architecture.

The discriminator similarly takes as input an image and the label, which is encoded as a 64x64 image with two channels. Each channel contains only ones or zeros depending on the label: for example 01 for male and 10 for female, in a one-hot fashion. Again, these inputs are convoluted independently, then concatenated and convoluted until we have a single number: the probability of the image being real.

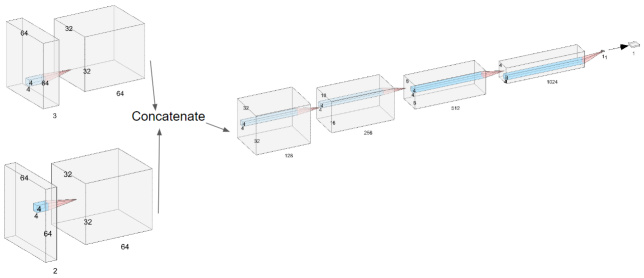

Figure 4. Discriminator of the starting architecture.

With this architecture generated images were blurry, contours were not well defined and worked only with very discriminative labels like male/female. We tried to modify it in order to use two labels instead of one, increasing the dimension of the label channels from 2 to 4, but the network did not converge and the output was only noise.

2.4. Technical details

We created and trained our network on Google Colab, and all the results that we show here are obtained after 10 epochs on the CelebA dataset, which corresponding to 6-9 hours of training on Colab.

3. Failed attempts

3.1. One number encoding for a single label

One of the first thing that we tried was to substitute the two channels of the label encoding with only one channel. This channel contained one number: 0.5 or 1 based on the value of the label. With this approach results were very similar to the starting architecture. Thus, multilabeling was still an issue.

Figure 5. Results with the base architecture for the label male/female

3.2. Information in the noise

We also tried to embed the label information in the noise input of the generator by feeding it with only a vector of 256 numbers, 128 of random noise and 128 represent the label in the one-hot encoding: the first 64 numbers to 0 and the other 64 to 1 or viceversa based on the label. In this case the network was not able to produce any results, and therefore this route was abandoned.

4. Improvements

4.1. More convolutions

One of the first successful modifications of the original architecture was the addition of two more deconvolutional layers to the label input in the generator. This did not produce significantly better results but created better results earlier: after only one epoch was possible to understand if the network was learning to differentiate between labels or not.

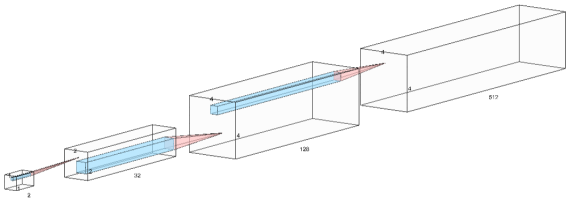

Figure 6. The convolutions of the label in the modified generator.

4.2. One number encoding for multilabeling

Following the idea of using only one number for the encoding of a label we tried to see if it was possible to use a single number to embed the information of two labels. Therefore, we used a single channel for the label in both networks with the following encoding:

- 0.25 for male with non black hair
- 0.5 for female with black hair
- 0.75 for male with non black hair
- 1 for male with black hair

With this structure we were able for the first time to produce images with multiple labels. The downside is that this approach is not scalable with more labels because the numerical difference will be too low and the images were still not very defined and of a not satisfying quality for us.

4.3. Spectral normalization

In order to increase the quality of the images and the stability of the training we introduced spectral normalization in the discriminator[2]. Spectral normalization is a type of normalization that takes advantage of the Lipschitz continuity. It works by normalizing every weight W with W/σ(W) where σ(W) is the spectral norm of W (which is the square root of the largest eigenvalue of W).

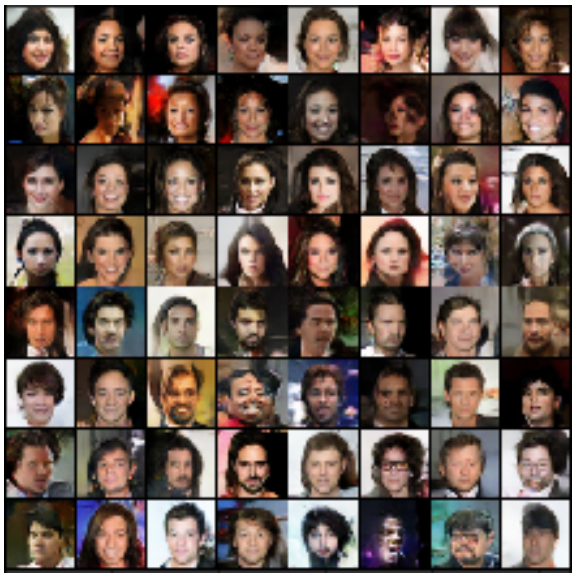

Figure 7. Spectral normalization results, in the first two rows there should be female with non black hair, then female with black hair, then male with non black hair and then male with black hair. Images are well defined but the black hair label/non black hair label is not represented correctly.

Using the base architecture with the one-hot encoding on multiple labels with the addition of the spectral normalization in the discriminator we were able to obtain images more defined. Faces were better defined, but the network was not able to distinguish properly between labels: we needed a way for the network to differentiate more between labels.

4.4. Categorical Batch normalization

Categorical batch normalization [3] is a type of batch normalization that changes the normalization parameters based on the input category. It is applied to the generator and for every combination of labels we assign a different category; for example the generation of females with black hair and of females with non black hair will use different batch normalization parameters.

$$y_i = \gamma \overline{x_i} + \beta$$

Figure 8. Batch normalization scaling and shifting, y_i is the batch normalization output and x_i is the normalized output of the previous layer.

This approach has the advantage that the network differentiate more between the various labels but it also adds more parameters to compute, slowing down the training. Using the base architecture with the one-hot encoding on multiple labels and the conditional batch normalization we obtained results respecting more the labels, but images were not as defined as when using spectral normalization, but these approaches can be combined.

5. Final Architecture

Our final architecture is the base architecture modified to use more than one label both in the generator and in the discriminator using the one-hot encoding, with some additional features:

- More convolutions of the label in the generator
- Categorical batch normalization in the generator
- Spectral normalization in the discriminator.

This architecture has all the advantages we were looking for: it is more stable (in a sense that the loss of the discriminator evaluating the generated images is more or less constant), it is able to produce good quality faces with well defined traits and is able to correctly differentiate between labels.

6. Conclusion

6.1. Key takeaway

1. We found out that one of the main indicator to look during the training of a GAN is the loss of the discrim-

Figure 9. Conditional normalization results, in the first two rows there should be female with non black hair, then female with black hair, then male with non black hair and finally male with black hair. Images are not well defined like the spectral normalization ones but the difference the labels is clear (look at the blonde women for female with non black hair).

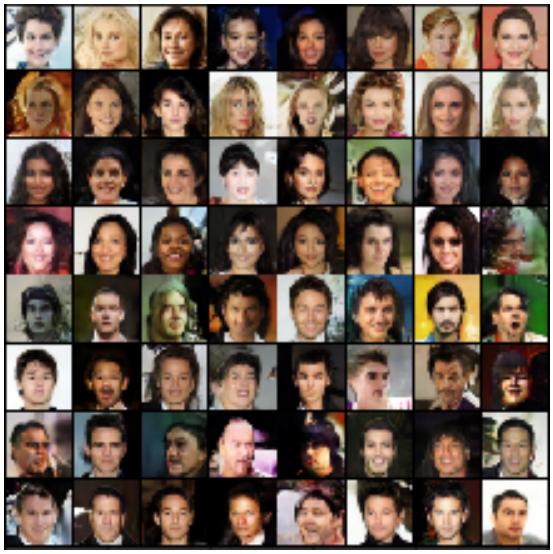

Figure 10. Final results, in the first two rows there should be female with non black hair, then female with black hair, then male with non black hair and finally male with black hair.

inator evaluating the images of the generator: if this number is close to 0 the generator will not improve.

2. Not all labels are created equal: labels like make/female are very discriminative and the network learns pretty easily to differentiate between them, while others are not, and therefore it is more challenging to create images with that labels.

Figure 11. Final results, in the first two rows there should be female non smiling, then female smiling, then male non smiling and finally male smiling.

6.2. Results

We are proud to say that the objectives of the project are achieved, we took an architecture that was not able to generate images using more than one label and we made it generate better images using more than one label. We achieved these results mainly thanks to the spectral normalization to stabilize the training and have better images and thanks to categorical batch normalization that makes the network differentiate more between labels.

References

[1] Generative Adversarial Nets - 2014 - Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio

[2] Spectral Normalization for Generative Adversarial Network - 2018 - Takeru Miyato, Toshiki Kataoka, Masanori Koyama, Yuichi Yoshida

[3] Modulating early visual processing by language - 2017 - Harm de Vries, Florian Strub, Jrmie Mary, Hugo Larochelle, Olivier Pietquin, Aaron Courville