De Filippis Stefano, 1707247

Conditional GAN - Image-to-Image Translation Using Conditional Adversarial Networks Report

1. Introduction

During this practical assignment I was able to reproduce a neural network architecture that could in some way solve the problem of the automatic *image-to-image translation.* In this kind of problems of image processing we need to solve the task of translating a possible representation of the input image into another one.
The general approach requires the use of CNN networks where we try to minimize a loss function, but we still need to put a lot of effort in order to build losses that are effective for the problem at hand. Another approach, instead, make use of the GANs architecture, where the network learn a loss that tries to classify if the output image is real or fake, while simultaneously training a generative model to minimize this loss.
In particular, the solution that I re implemented explored GANs in the conditional setting, where the generative model produces the desired output given a random noise and conditioned on the input image.

2. Conditional GANs

Therefore, conditional GANs try to learn a mapping from observed image *x* and random noise vector *z*, to *y*, *G : {x,z}→y*. The objective of a conditional GAN can be formulated in the following way:

$$\mathcal{L}_{cGAN}(G, D) = \mathbb{E}_{x,y}[\log D(x, y)] + \mathbb{E}_{x,z}[\log(1 - D(x, G(x, z))]$$

where *G* tries to minimize the objective against an adversarial *D* that tries to maximize it.
Moreover, it is beneficial to add more traditional loss to the objective just showed. Therefore, the discriminator loss remains the same while the generator needs not only to to fool the discriminator but also to generate outputs as close as possible to the ground truth and this is achieved through the L1 distance:

$$\mathcal{L}_{L1}(G) = \mathbb{E}_{x,y,z}[\|y - G(x, z)\|_1]$$

and the generator loss became:

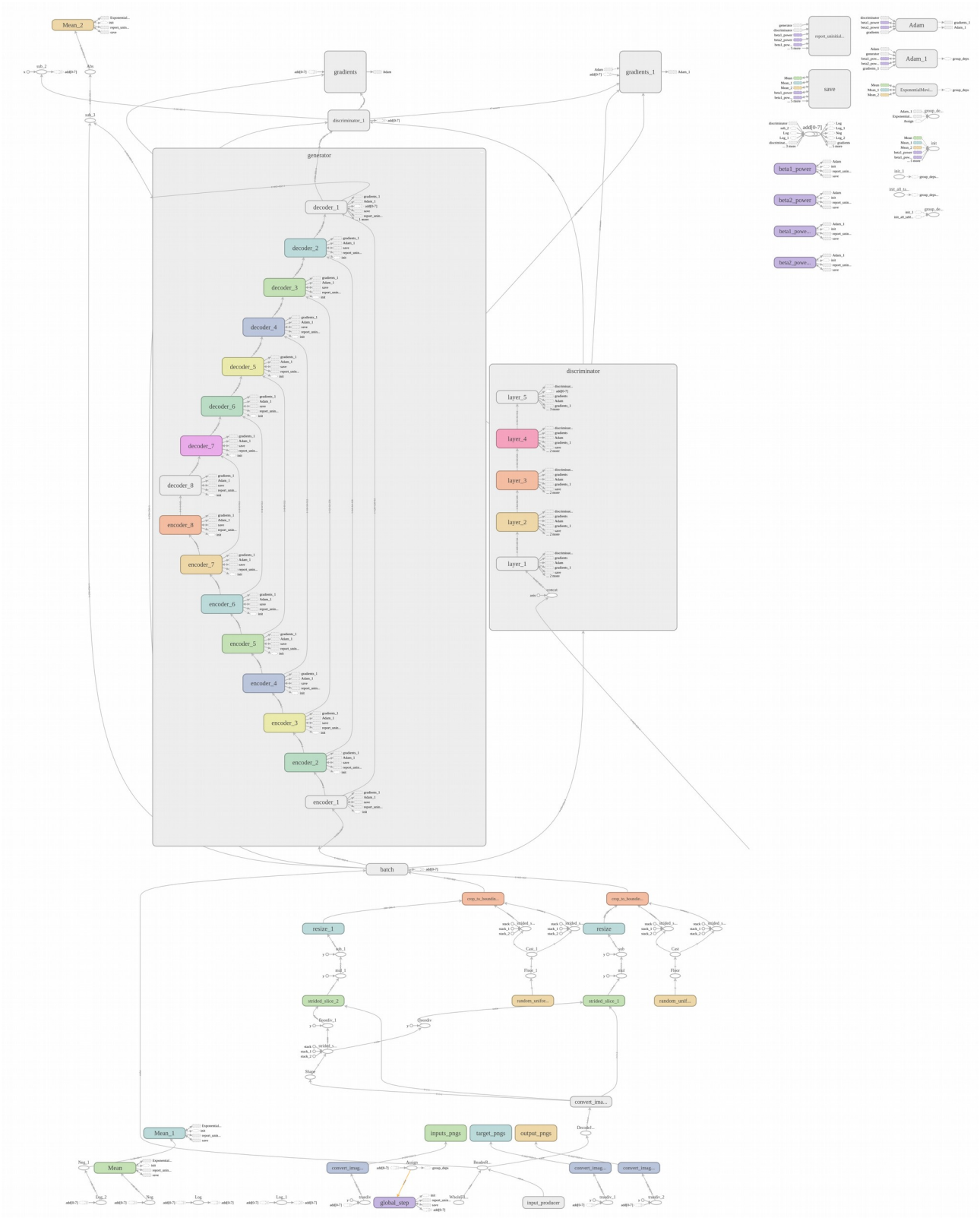
$$G^* = \arg \min_G \max_D \mathcal{L}_{cGAN}(G, D) + \lambda \mathcal{L}_{L1}(G)$$

Finally, passing as input a Gaussian noise *z* the results were not particularly good, and the performances increased when the noise was provided in the form of dropout, applied on several layers of the generator at both training and test time.

3. Network Architecture

It is important to notice that every layer of the generator and discriminator is build with modules that consist of a convolutional layer, a batch normalization layer and a final non linear layer with a ReLu activation function.

Moreover, in a image-to-image translation problem the aim is to map a high resolution grid of pixels to another high resolution grid of pixels and the input and output, even if different, are both representations of the same underlying structure. Due to this consideration, an encoder-decoder network can be used. The input is passed trough a series of downsampling layers until a bottleneck layer and then the process is reversed. This structure is useful since the input and the output still represents the same images even if in different ways. Anyway, in image translation problems there are a lot of low-level information shared between the input and output which would be better if directly passed through the network. For this purpose, the implemented architecture makes us of skip connections between each layer i and layer $n - i$, where n is the total number of layers. Each skip connection simply concatenates all channels at layer i with those at layer $n - i$.
Finally, since the L1 loss on its own generates blurry results, it is necessary to tweak the network in order to also model the high frequency structures. This can be done by restricting the attention to the structure in local image patches and, therefore, the discriminator tries to classify if each NxN patch in an image is real or fake by running convolutionally across the image and averaging all responses to provide the final output of D.
In the following page I will report the tensorflow graph of the model.


Mean_2
gradients
gradients_1
discriminator_1
generator
discriminator
batch
resize_1
resize
convert_ima...
Mean_1
Mean
inputs_pngs
target_pngs
output_pngs
global_step
input_producer
report_uninitial...
save
Adam
Adam_1
ExponentialMovi...
beta1_power
beta2_power
beta1_powe...
beta2_powe...
add[0-7]
decoder_1
decoder_2
decoder_3
decoder_4
decoder_5
decoder_6
decoder_7
decoder_8
encoder_8
encoder_7
encoder_6
encoder_5
encoder_4
encoder_3
encoder_2
encoder_1
layer_5
layer_4
layer_3
layer_2
layer_1

4. Experiments

In the following section I will report the result on the experiments conducted.

4.1 Dataset

For the experiments I used 2 different dataset. The first one is the facade dataset containing input images of building facades blueprints and the target images are actual building facades . The training dataset contains 400 images, while the test dataset contains 106 images.

The second dataset, instead, is the cityscapes dataset whose input images are semantic segmentation of city life scenes while the target images are the actual city scenes. In this case the training dataset contains 2975 images while the test dataset contains 500 images.

4.2 Preprocessing

An essential step in order to get good results was the image preprocessing. In my experiments all the input images were normalized in the range {-1,+1}. Successively, I scaled up the images with a filter in order to have images of size 286x286 and finally I cropped down the input back to 256x256 but choosing randomly the offset from the corner from which to start the bounding box.

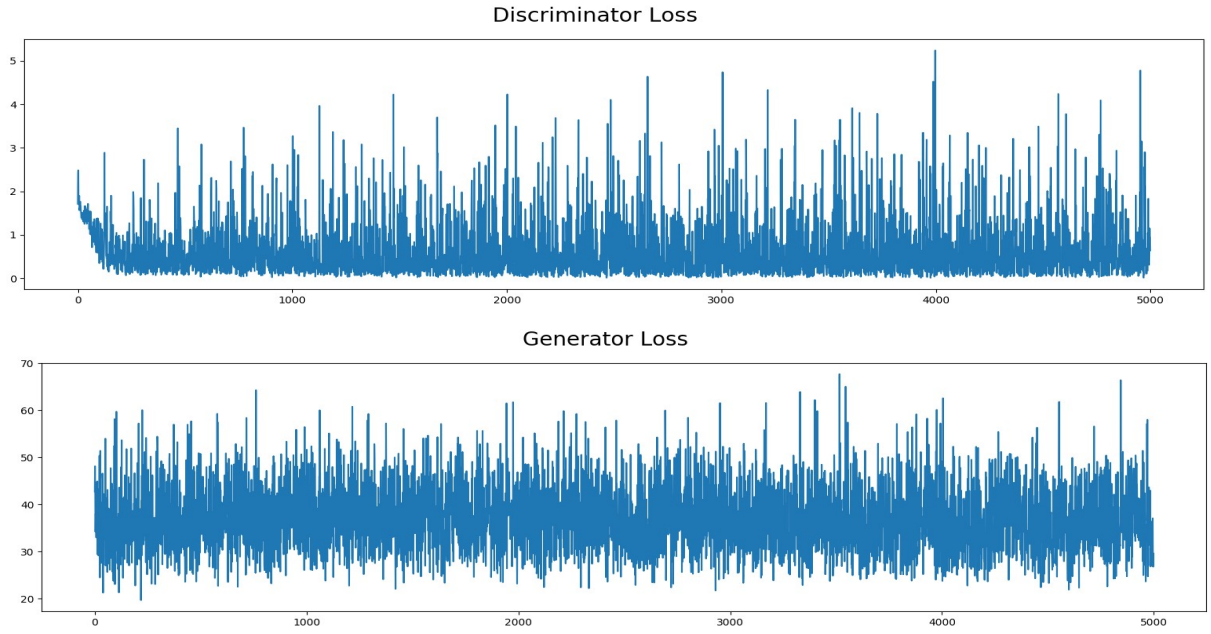
4.3 Experimental Set Up

I will now report the parameter chosen for the training

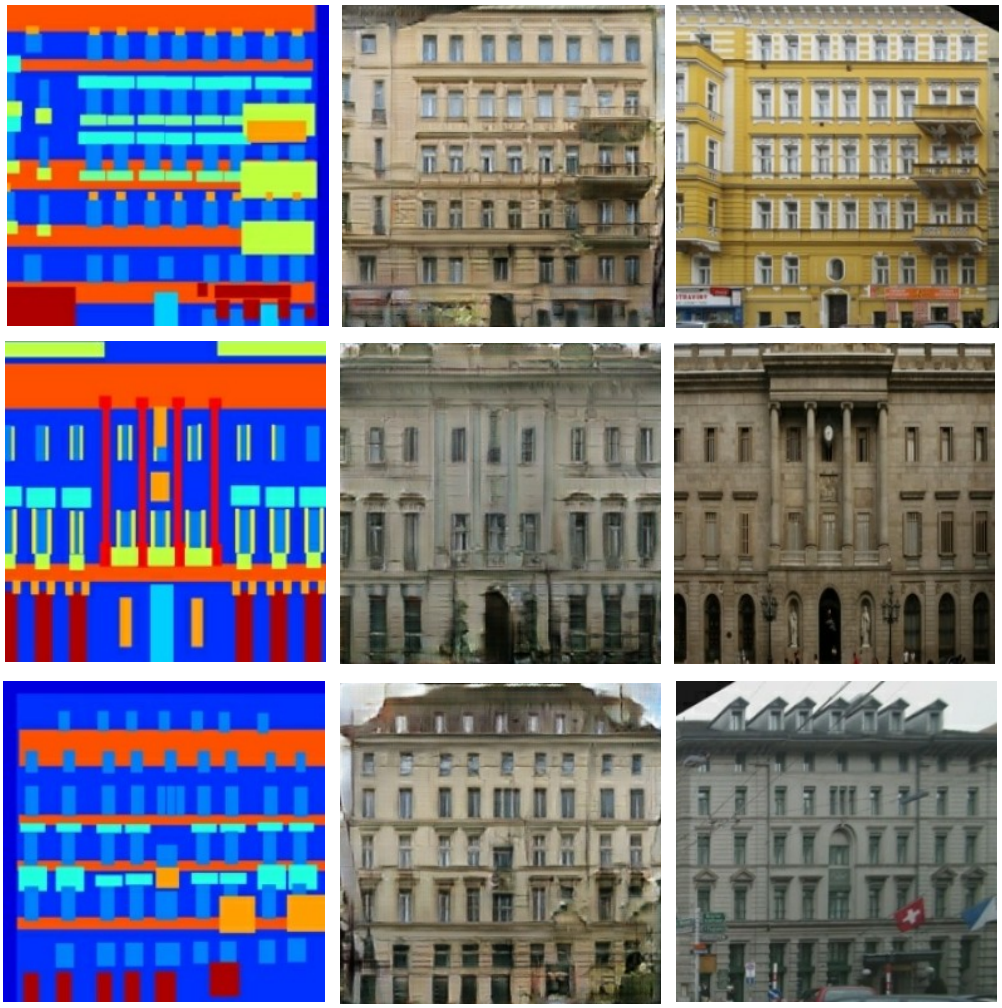
Dataset	Batch	Epochs	Dropout	Gan Weight	L1 Weight	Optimizer	Learning Rate	Beta value
Facades	1	200	0.5	1	100	Adam	0.0002	0.5
Cityscapes	32	200	0.5	1	100	Adam	0.0002	0.5

4.5 Facades Result

I will now report the graph for the discriminator and generator losses during the training and some output of the final model obtained

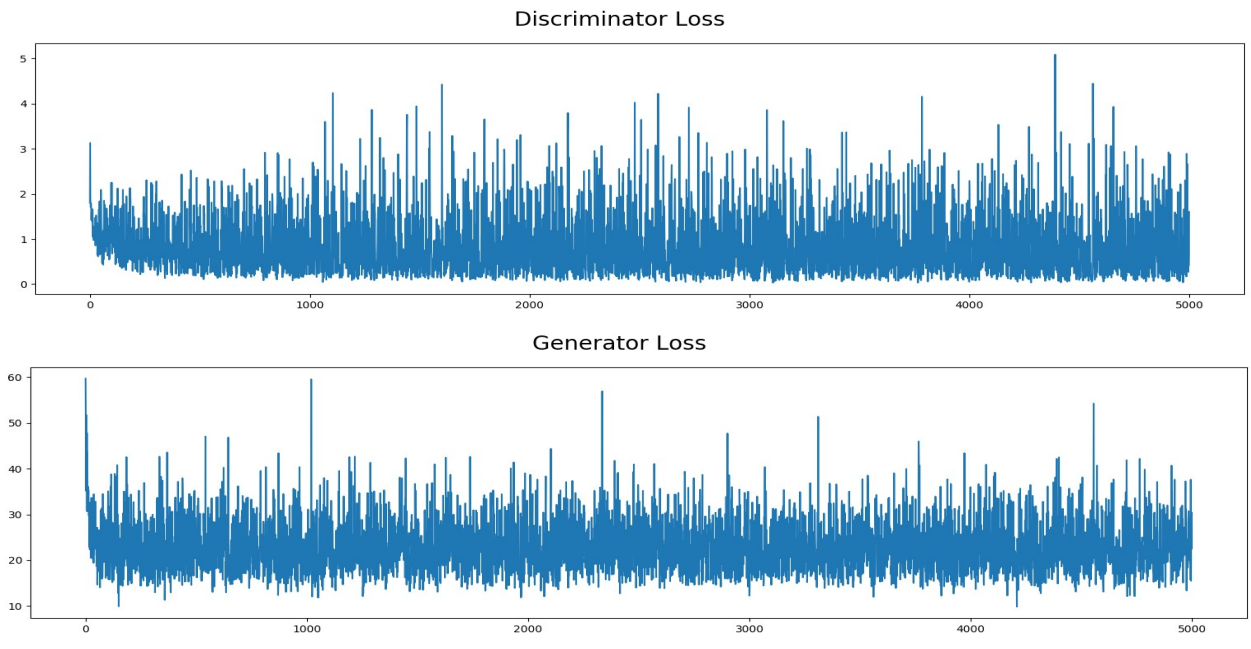


in both cases I can see a very high variance in the losses value but in any case for the discriminator the loss started high and its final mean value is around 0.5. The generator loss, instead, has its mean value around 30.
Now I will report some of the input, output and target sequences of the final model



4.6 Cityscapes Result

I will now report the graph for the discriminator and generator losses during the training and some output of the final model obtained with the cityscapes dataset

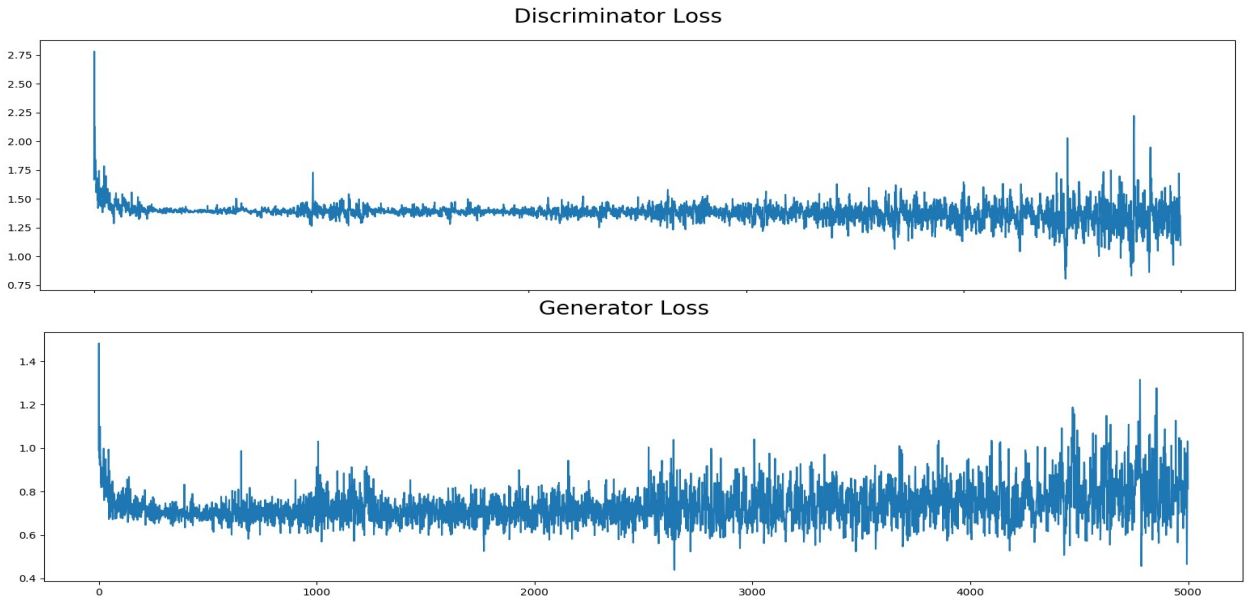


Now I will report some of the input, output and target sequences of the final model

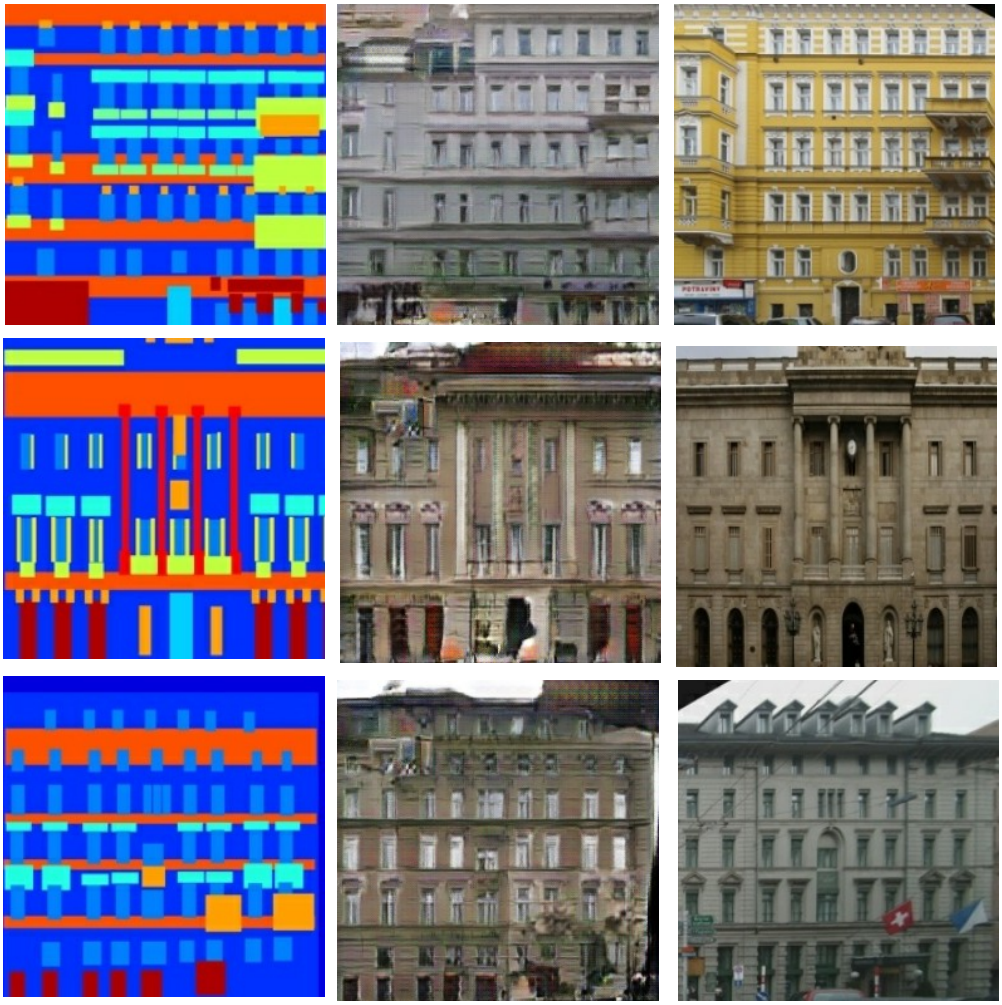
In this case I can see that the results are less accurate with respect to the outputs of the previous dataset and this is probably due to the fact that the model needs more epochs to train. Indeed, 200 epochs for this dataset are not enough but the time required was too much and I needed to stop the training.

4.7 Facades Result without L1 Loss

I will now report the graph for the discriminator and generator losses during the training and some output of the final model obtained with the facades dataset but omitting the L1 loss term in the generator overall loss.



Now I will report some of the input, output and target of the final model



From the losses' plots, I can see that the discriminator loss presents a lot less variance during the training with the respect to the first experiment conducted on the facades dataset. Nonetheless, its final mean value is almost 1.30 which is way higher than the 0.5 of the first experiment. This means that the discriminator still can distinguish the fake images from the real ones and, instead with GANs we would like the discriminator to not be able to do that. Also the generator loss presents a smaller variance with respect to the first experiments but the values can not be compared since there is the L1 term missing.
The consideration just made are directly reflected on the model outputs. Indeed, the output images quality is considerably deteriorated with respect to the output of the first model.

5. Conclusion

With this assignment I was able to re implement a new kind of architecture based on the concept of GANs results. I tested the architecture on 2 different dataset and the model behaved quite well in both cases even if the problem posed by the cityscapes is a little bit harder to tackle and the outputs quality is not very sharp and accurate. Moreover, an important step in order to get better result was the preprocessing of images and the presence of the L1 loss in the generator final loss. In particular, I was able to study the differences in the model output with and without the presence of L1 term and the quality was decisively better when the additional term was present. This fact, finally, was also evidenced by the

higher final mean value of the discriminator loss when the L1 loss term was not present and by the qualities of the results in the 2 cases.