Paris City University

Perception, acquisition and image analysis

Image-to-Image Translation with Conditional Adversarial Networks

---

Student :
Mohamed El Mehdi MAKHLOUF

Professor :
Jonathan Vacher

September 23, 2023

Contents

1 Introduction

The paper "Image-to-Image Translation with Conditional Adversarial Networks" by Isola et al. addresses a generative modeling problem, which has become an important focus in the field of AI for its utility and significance in various applications. Especially in computer vision and image processing tasks, where several problems require translating between paired images, such as depth estimation from an image. The main motivation of the authors in this paper is to propose a method for image-to-image translation that can generate high-quality and realistic images in a variety of different domains. The authors aim to address the limitations of previous methods, which were unable to produce visually convincing results or required a significant amount of training data.

The proposed method can generate realistic images of objects in different styles or from different viewpoints, which has a wide range of applications such as style transfer, object transfiguration, and scene understanding. It can also improve the computer's understanding and interpretation of visual information, and contribute to the development of new technologies in areas such as art, media, and entertainment, as well as the creation of more realistic computer-generated imagery in fields such as movies, video games, and advertising.

The authors make a significant contribution to the field of image-to-image translation by introducing the use of a conditional adversarial network (cGAN) to learn the mapping between the input and output images. At a time when GANs were already gaining attention, particularly due to the performance demonstrated by deep convolutional GANs (DCGANs) on various data generation or image tasks. The authors' approach allows for the generation of high-quality and realistic images in a variety of different domains, which had not been achieved by previous methods. This research has had a significant impact on the literature in the field of image-to-image translation, as it has provided a new approach for solving this problem. Many subsequent papers have built upon the ideas presented in this paper, and the use of cGANs has become a widely adopted method for image-to-image translation. Additionally, it also has an impact on the broader field of computer vision, as the ability to translate images can greatly improve the ability of computers to understand and interpret visual information.

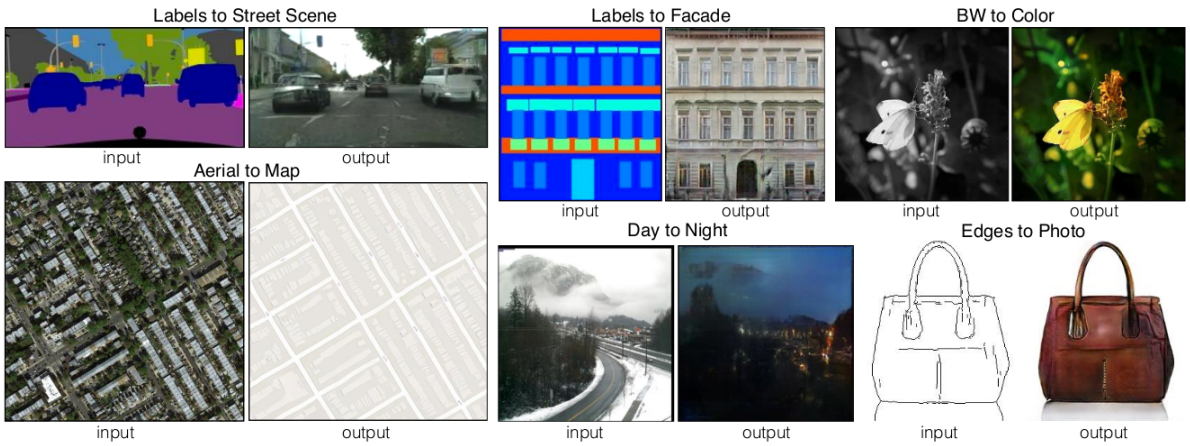

Figure 1: The figure illustrates different applications of the proposed method for various tasks, it was picked from the original article [2].

2 Contribution overview

The authors propose a novel method for image-to-image translation using conditional GANs (cGANs) which has been proven to generate high-quality and realistic images in various domains and has become widely adopted in the field. The method not only has been applied to various applications such as style transfer, object transfiguration, and scene understanding, but also has an impact on the broader field of computer vision by improving the ability of computers to understand and interpret visual information. The authors use cGANs to learn the mapping between the input and output images, enabling the generation of high-quality and realistic images without requiring significant amounts of training data. Additionally, the proposed method is simpler than most other methods as it does not require application-specific modifications and uses a U-Net-based architecture for the generator and a convolutional PatchGAN classifier for the discriminator, which only penalizes structure at the scale of image patches, this approach is shown to be effective on a wide range of problems.

3 Mathematical and technical details of the contribution

This section we will start by a quick remained of the mathematical and technical details of the Generative Adversarial Networks (GANs) and the conditional Generative Adversarial Neworks (cGANs). Then we will go through the key points and details of the Pix2Pix architecture as proposed in [2].

3.1 Generative Adversarial Networks

Generative Adversarial Networks (GANs) was presented for the first time in [1], where the authors define the **Adversarial Modeling Framework**. Which is used to learn the generator's distribution p_g over data x. A prior is defined on input noise variables $p_z(z)$, and a mapping to data space is represented as $G(z;\theta_g)$, where G is a differentiable function represented by a multilayer perceptron with parameters θ_g. A second multilayer perceptron $D(x;\theta_d)$ is defined that outputs a single scalar, representing the probability that x came from the data rather than p_g. The goal is to train D to maximize the probability of assigning the correct label to both training examples and samples from G, while simultaneously training G to minimize $\log(1-D(G(z)))$. This results in a two-player minimax game with value function $V(G,D)$, defined as:

$$\min_G \max_D V(D,G) = E_{x\sim pdata(x)}[\log D(x)] + E_{z\sim p_z(z)}[\log(1-D(G(z)))]$$

In other words a GAN consists of two parts: a generator and a discriminator. The generator learns to generate new data samples that are similar to the training data, while the discriminator learns to distinguish between real and generated data.

3.2 Conditional GANs

A conditional GAN (cGAN) is a variant of GANs that allows for the generation of samples conditioned on additional input. In [2] cGANs are used to learn the mapping between input and output images. The main difference between a cGAN and a simple GAN is the addition of an additional input to the generator and discriminator networks. The mathematical definition of a cGAN is similar to that of a simple GAN but with the addition of an additional input y. The objective of a conditional GAN is expressed as:

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

Where x represents real data samples, z represents random noise samples, and y represents the additional input or lets we call it the target,as you can see in Figure 2.
In the same manner as in GANs, G will tries to minimize this objective against an adversarial D that tries to maximize it, i.e $G^* = \arg\min_G \max_D \mathcal{L}_{cGAN}(G, D)$
To test the importance of conditioning the discriminator, the authors of [1] compared the previous loss function to an unconditional variant in which the discriminator does not observe x:

$$\mathcal{L}_{GAN}(G, D) = E_y[\log D(y)] + E_{x,z}[\log(1 - D(G(x, z))] \tag{2}$$

This equation represents the objective function of an unconditional variant of GANs, where the discriminator "D" does not observe the input x, and it's compared to the cGAN objective function. This comparison is done to test the importance of conditioning the discriminator.

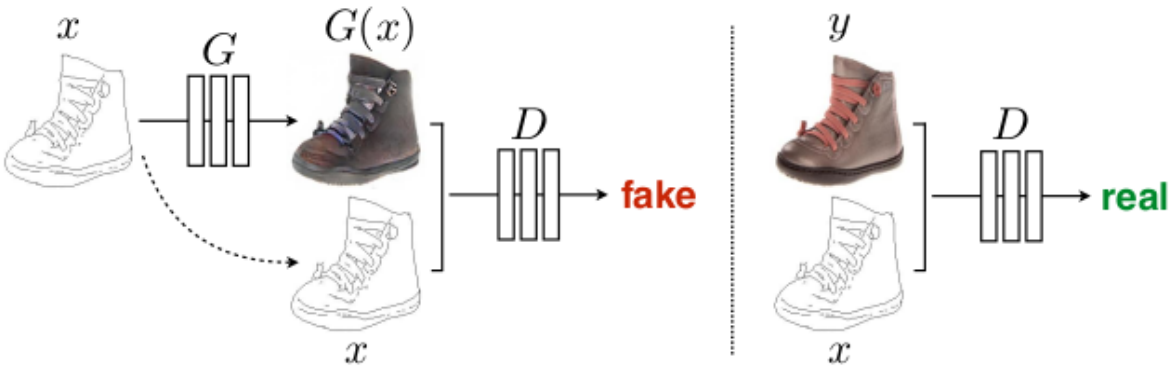

Figure 2: An illustration of a cGAN where the generator, represented as "G," takes input "x" and generates a new sample based on the input and embedded random noise samples "z" and the discriminator, represented as "D," is conditioned with input "x" and attempts to detect generated samples.

The authors propose a new objective function for the generator in a conditional GAN (cGAN) that includes an additional term, $\mathcal{L}_1(G)$, to encourage the generator to produce outputs closer to the target. The new objective function for the generator is defined as:

$$\mathcal{L}_{L1}(G) = E_{x,y,z}[|y - G(x, z)|_1] \tag{3}$$

This equation represents the L1 norm of the difference between the generator's output and the target.

The final objective for the generator is to minimize the sum of this new term and the original cGAN objective, with a weighting factor , it is represented as :

$$G^* = \arg\min_G \max_D \mathcal{L}_{cGAN}(G, D) + \lambda \mathcal{L}_1(G) \tag{4}$$

This equation represents the final objective function for the generator, it's a combination of the cGAN objective and the added term $\mathcal{L}_1(G)$ with a weighting factor λ.

The authors also mention that in their experiments, they use dropout noise instead of Gaussian noise as an input to the generator. They found that this strategy was more effective than using Gaussian noise. However, they acknowledge that designing cGANs that produce highly stochastic output is an important question that has not been addressed by their work.

3.3 Network architecture

3.3.1 Generator

The authors chose to use a U-Net-based architecture for the generator in their image-to-image translation method. This architecture allows for the efficient transfer of low-level information between the input and output by incorporating skip connections between layers. This design decision was made to take into account that image-to-image translation requires mapping high-resolution input to high-resolution output and that the input and output images share the same underlying structure, which is particularly important for image translation problems where a significant amount of low-level information is shared between the input and output.

The authors used several techniques to improve the training of the Pix2Pix model, such as using instance normalization and dropout in the generator network.

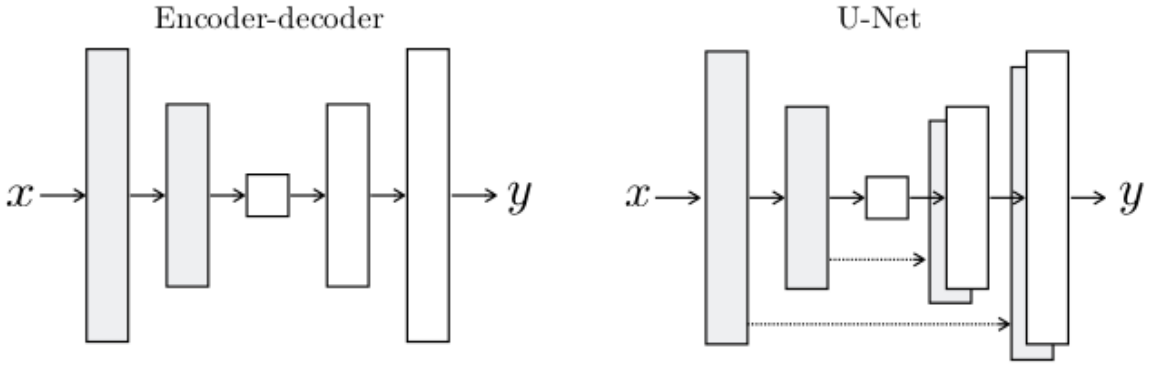

Figure 3: An illustration of an encoder-decoder architecture and the U-Net architecture, which is an encoder-decoder with added skip connections.

3.3.2 Markovian discriminator (PatchGAN)

The authors propose a new method for image generation that utilizes the L1 loss in the generator to capture low-frequency information effectively. However, it is known that the L1 loss does not capture high-frequency information. To address this issue, they introduce a new discriminator architecture called PatchGAN, which is designed to model high-frequency structure. The PatchGAN architecture only penalizes structure at the scale of small image patches, which results in a more efficient network with fewer parameters and faster running times. Additionally, it can be applied to arbitrarily large images. The method is based on the assumption that images can be modeled as a Markov random field, and the PatchGAN can be understood as a form of texture/style loss. This discriminator tries to classify if each N $\times$ N patch in an image is real or fake. We run this discriminator convolutionally across the image, averaging all responses to provide the ultimate output of D.

In their experimentation's they tested different values for the patch size, in addition they indicate that the optimal patch size can be estimated using the receptive field.

The receptive field of a convolutional neural network (CNN) refers to the region of the input image that a particular neuron in the network is able to "see" or be affected by. In other words, it is the portion of the input image that contributes to the output of a particular neuron cf 4.

The receptive field can be calculated by considering the spatial extent of the convolutional filters and the strides used in the network. A simple formula to calculate the receptive field of a CNN neuron located at position (x, y) in a particular layer can be represented as:

$$RF(x,y) = \bigcup_{i=1}^{n}(x \times s_x + k_x \times (i-1), y \times s_y + k_y \times (i-1))$$

Where RF(x,y) is the receptive field of the neuron, n is the number of layers, sx and sy are the strides of the filters in x and y axis respectively , kx and ky are the size of the kernel in x and y axis respectively and i is the layer number.Note that this formula is an approximation and assumes that the padding and dilation are zero in each layer.

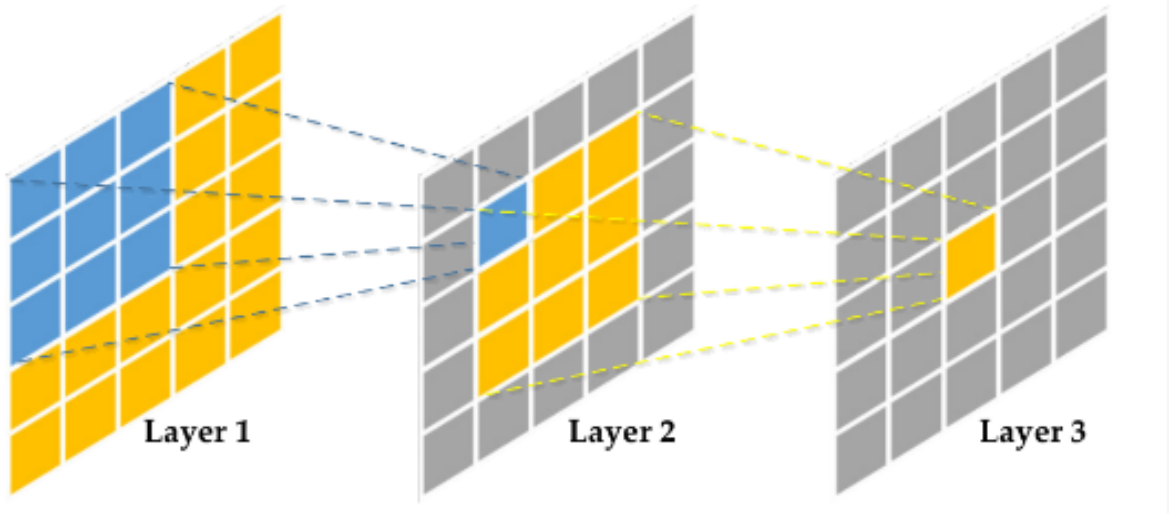

Figure 4: Schematic diagram of the receptive field in CNNs.

3.4 Optimization and inference

The authors use the standard approach of alternating one gradient descent step on the discriminator and one step on the generator. In addition, they divide the objective by 2 while optimizing discriminator to slow down the rate at which D learns relatively to G . They train the generator and the discriminator using the Adam optimizer with a learning rate of 0.0002 and momentum parameters of $\beta_1 = 0.5$ and $\beta_2 = 0.999$. At inference time, they use dropout and apply batch normalization using the statistics of the test batch rather than the training batch. They also use a batch size between 1 and 10 in their experiments.

4 Experimentation and result analysis

The authors tested the Pix2Pix model on various data sets, including day and night city images, city model and real city images, and drawings and wildlife photos.

The authors evaluate the quality of the synthesized images using two methods. The first is through "real vs. fake" perceptual studies on Amazon Mechanical Turk (AMT), where human observers are presented with real and fake images and asked to determine which is fake. On each trial, each image appeared for 1 second. The results of this study were used to validate the perceptual realism of their results for example for the Map-aerial data set.

The second method used the FCN-8s architecture for semantic segmentation, and score synthesized photos by the classification accuracy against the labels of these photos were synthesized from.

The authors also use the experimentation section to study and demonstrate the effect of conditioning the discriminator by x, $\mathcal{L}_1$ normalization, and the effect of patch size. They ultimately show that all the choices made previously, as explained above, help to improve the results of the method. It should be noted that there are examples among the datasets used in the experimentation section that are of lower quality than those presented in the article. Additionally, when attempting to reproduce the exact same experiment on the Map-aerial data set, using the same training parameters, after 200 iterations the results were still far from those presented in the article.

It is worth noting that the results may vary depending on the specific data-set and use case, and the authors of the paper proposed a specific use case for Pix2Pix which is image-to-image translation. However, the Pix2Pix architecture and GANs in general have been applied in many other fields such as background removal and image synthesis see Figure 5.

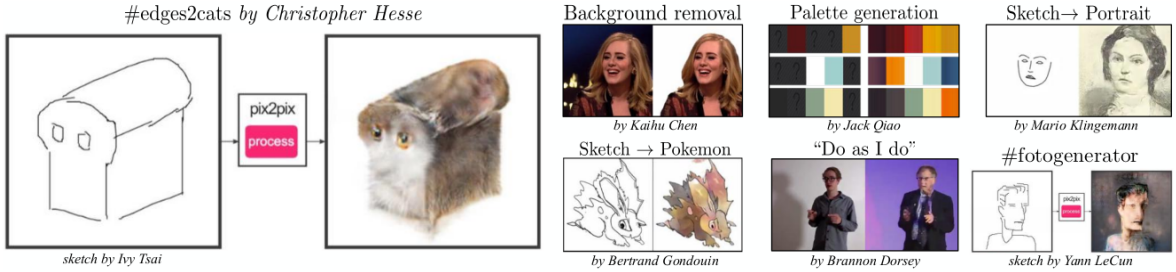

Figure 5: Pix2Pix applications.

5 Discussion

In the "Image-to-Image Translation with Conditional Adversarial Networks" paper, the authors propose the use of the Pix2Pix model, a specific implementation of cGANs, for image-to-image translation. The results of the experiments conducted on several map data sets, including aerial maps and maps of cities, show that the Pix2Pix model is able to generate high-quality and realistic images that are coherent with the input images. The model has been able to generate high-resolution images of buildings and roads for example.

However, it's worth noting that the Pix2Pix model is a specific implementation of cGANs designed for a specific task, image-to-image translation. The model may not perform well on other tasks or data sets, and more research is needed to investigate its generalization capabilities. Also, the authors used specific metrics such as FCN-8, and user study to evaluate the quality of the generated images, which may not be applicable to other tasks or data sets.

In terms of limitations, one of the main limitations of GANs in general is the instability of their training, which can lead to mode collapse and poor convergence. Additionally, GANs are known to be sensitive to the choice of hyper-parameters and the quality of the training data. Another limitation of GANs is the difficulty of interpretability and the lack of control over the generated samples.

Finally, it's important to consider the ethical aspects of GANs, especially when applied to sensitive tasks such as image synthesis or generation. GANs can be used to generate fake images or videos that can be used to deceive or manipulate people. It's important to ensure that GANs are used responsibly and with transparency to avoid potential misuse and harm. Additionally, GANs have the potential to perpetuate biases present in the training data. It is important to be aware of these biases and take steps to mitigate them, such as using diverse and representative training data.

6 Conclusion

In conclusion, the Pix2Pix model is a powerful tool for image-to-image translation that can generate high-quality and realistic images. However, it is important to keep in mind its limitations and its generalization capabilities to other tasks and data sets, as well as its ethical implications. Further research is necessary to continue improving the stability and interpretability of GANs and to ensure they are used responsibly. Despite these limitations, the progress and utility of these generative models in various application areas is highly fascinating and is one of the most significant advancements in recent years. We can cite for example some applications of the cGAN model in the medical field such as in [4] and [3].

References

[1] Ian J. Goodfellow et al. *Generative Adversarial Networks*. 2014. DOI: 10.48550/ARXIV.1406.2661. URL: https://arxiv.org/abs/1406.2661.

[2] Phillip Isola et al. *Image-to-Image Translation with Conditional Adversarial Networks*. 2016. DOI: 10.48550/ARXIV.1611.07004. URL: https://arxiv.org/abs/1611.07004.

[3] Christof Kauba et al. "Inverse Biometrics: Generating Vascular Images From Binary Templates". In: *IEEE Transactions on Biometrics, Behavior, and Identity Science* PP (Apr. 2021), pp. 1–1. DOI: 10.1109/TBIOM.2021.3073666.

[4] Anita Rau et al. "Implicit domain adaptation with conditional generative adversarial networks for depth prediction in endoscopy". In: *International Journal of Computer Assisted Radiology and Surgery* 14 (Apr. 2019). DOI: 10.1007/s11548-019-01962-w.

A Testing and experimentation's reproduction

As required by the project, once I understood the model proposed in the article, I wanted to test it and reproduce their results, and also test it on a new data-set.

A.1 Implementation

The original implementation cited in the article itself posed some dependency issues and required older versions of some libraries. So I decided to redo the implementation from scratch, taking inspiration from some source codes that do the same thing (implementation of the pix2pix model from scratch). I paid close attention to exactly re-code the same architecture of the generator and discriminator using the information provided in the original article. Implementation code is available **here**.

A.2 Data-sets

Once I managed to have a functional implementation, I became interested in two data-sets:

1. **Aerial views and Google Maps images** : the same data set used in the original article. Direct download **link**.
2. **Face to Comic** : containing synthetic person's faces and their correspondences in comic form, containing 10000 image pairs, the size of each image is equal to 1024x1024. Direct download **link**.

Figure 6: Aerial views and Google Maps images overview picked from the original article

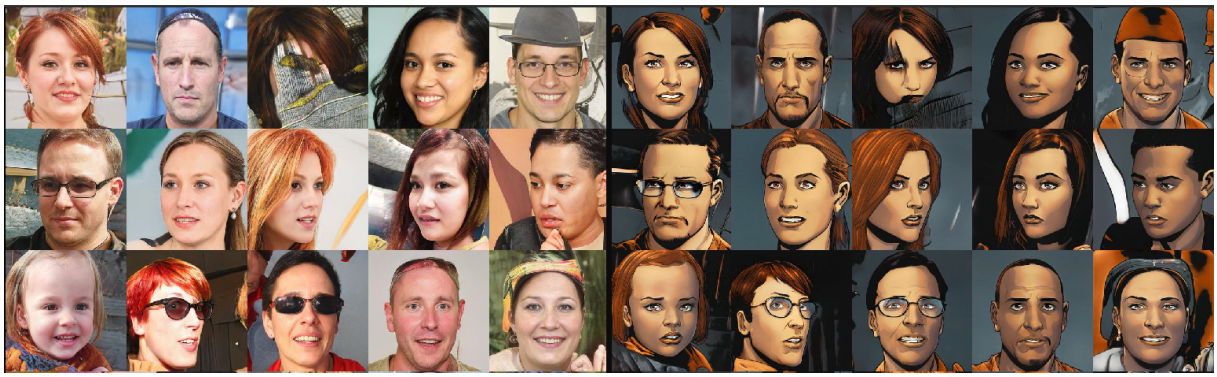

Figure 7: Face to Comic samples.

A.3 Experimentation's

I carried out a total of 3 training's on a GeForce RTX 3060 laptop graphics card from Nvidia. For each of the experiments, I reused the same parameters as those used in the original article, except for the batch size, which I set to 20 instead of 1 after the first two exploration experiments of the batch size effect. As it is not pleasant to hear the noise of the computer fan at full speed for hours, I made 200 epochs for some trainings at maximum, and for others I only made 100. All these details will be reminded in the discussion section of each experiment.

Below is a summary of the fixed parameters taken from the original article:

- Input and Output image size: 256 x 256
- Learning rate: 0.0002
- Momentum: $[\beta_1, \beta_1] = [0.5, 0.999]$
- $\lambda_{\mathcal{L}_1} = 100$

A.3.1 Face to comic

In this experiment, I trained the model for only 100 iterations with a batch size of 20. The training data set contains 9500 training pairs of images, and for validation, I generated images on the validation data-set containing 500 pairs of images. Then, I reported the 4 best and worst generations according to the L1 loss between the generated image and the label. See Figure 8 and Figure 9.

By observing the results, we can see that the model gives results that are not so bad only for a small number of iterations. But looking more closely at the best and worst generations, one might say that if we had trained it a bit more, we could achieve better results.

It should be noted that on the visualizations, the first column on the right contains the input image, the second contains the result of the generation, the third contains the label, and the last contains the L1 norm between the generated image and the pixel-by-pixel label. The last column can show us places where the model is not able to generate the image well. We can see this last column as a map of the model's errors, and as our image

loader includes random horizontal rotation, the results of this last column are no longer significant and we can rely on visual evaluation.

If we try to evaluate our results in the same way as in the article, that is, with the FCN-8 score. We quickly realize that the nature of the problem does not allow us to rely on classification results of the original and generated images. And we realize more of the difficulty of choosing appropriate metrics for evaluating models.

Figure 8: Best 4 generation based on the L1 loss.

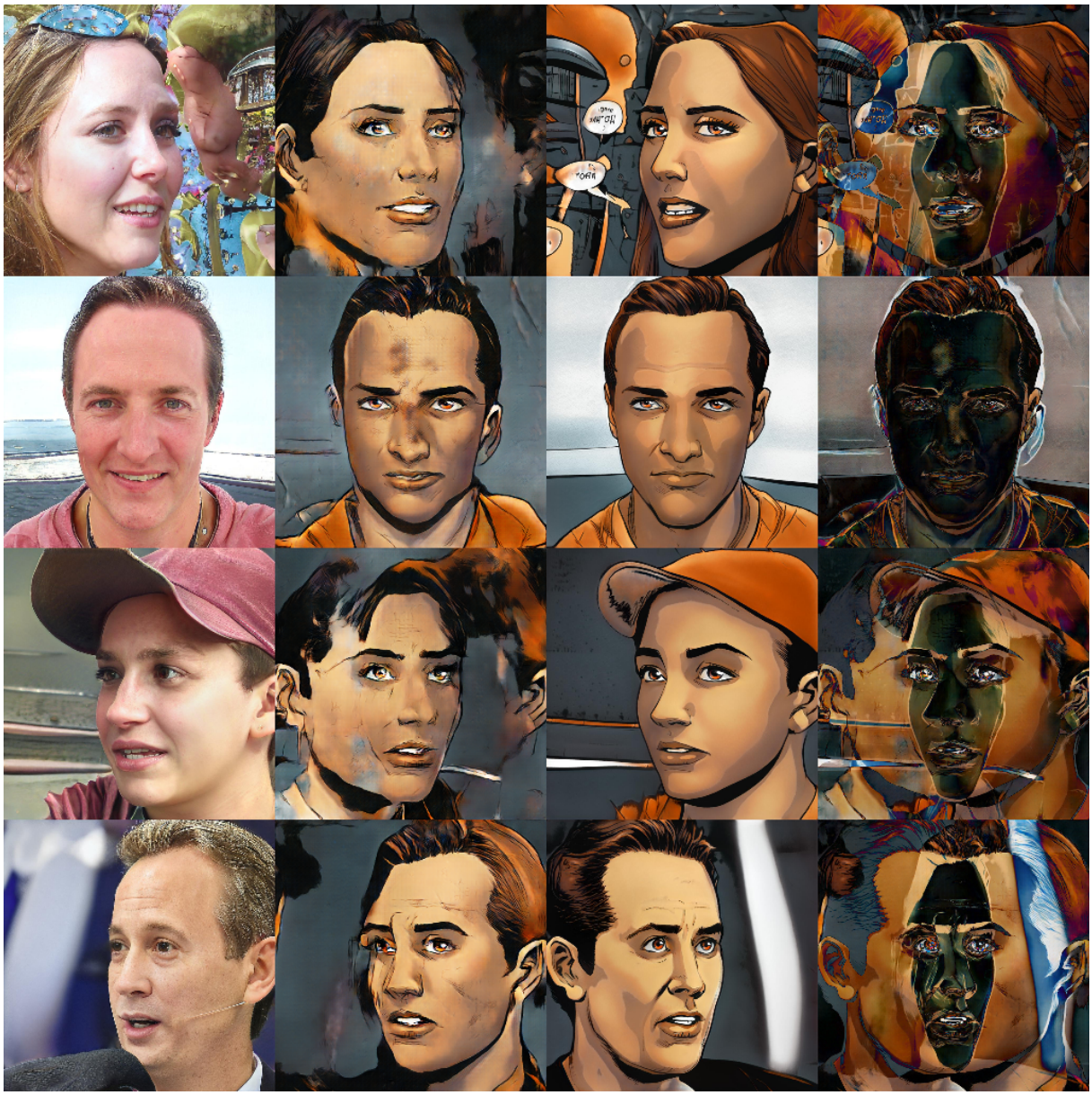

Figure 9: Worst 4 generation based on the L1 loss, we can figure the the horizontal rotation affect significantly the L1 loss and the error map in the last column.

A.3.2 Aerial view to map

In this experiment, I trained for 200 epochs with a batch size of 1 as proposed in the original article. Upon observing the results and comparing them to those of the original authors, I found that my model did not achieve the same performance after 200 iterations. See Figures 10 11

Additionally, I attempted to further analyze the results of the original authors and selected some generations that I found relevant to evaluate the model's capabilities, which can be found in the figures with explanatory captions.

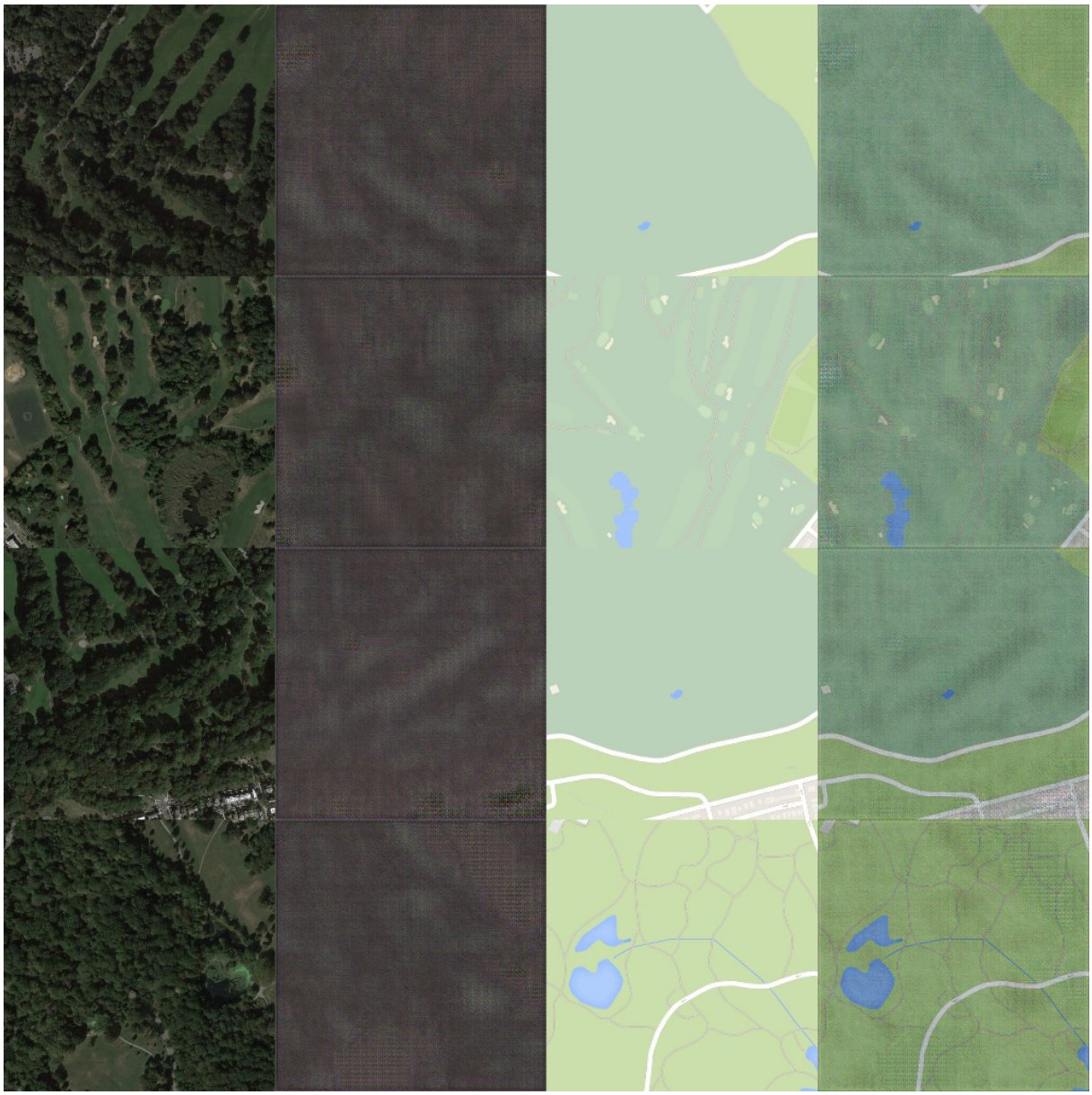

Figure 10: Best 4 generation based on the L1 loss, for the aerial to map task.

A.3.3 Map to aerial view

In the last experiment, I did the opposite of the aerial view to map experiment and trained for only 100 iterations with a batch size of 20. Once again, my results did not reach the same quality as the authors' results, likely due to the insufficient number of iterations. Despite this, I will display my best and worst 4 results as in the first two experiments see Figures 13 14, and I will also show and comment on some results from the original authors as I did in the previous experiments**??**.

A.4 Learning curves

In this section you will find learning curves of the Discriminator, Generator, $\mathcal{L}_1$ and the $\mathcal{L}_cGAN$ of each experience.

Figure 11: Worst 4 generation based on the L1 loss, for the aerial to map task.

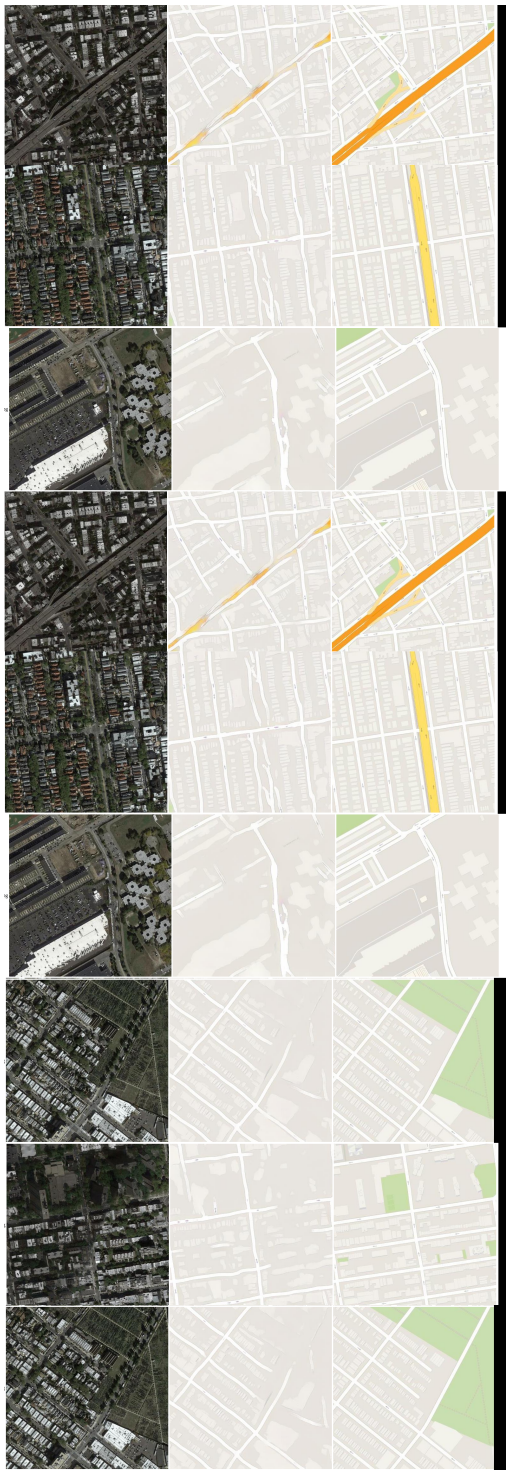

Figure 12: Worst 4 generation of the authors published model, we can see in the second column that the model have difficulties to learn highways in orange for example and miss some other details, this results will be discussed in detail in the presentation day.

Figure 13: Best 4 our generations based on the L1 loss, for the map to aerial task.

Figure 14: Worst 4 generations based on the L1 loss, for the map to aerial task.

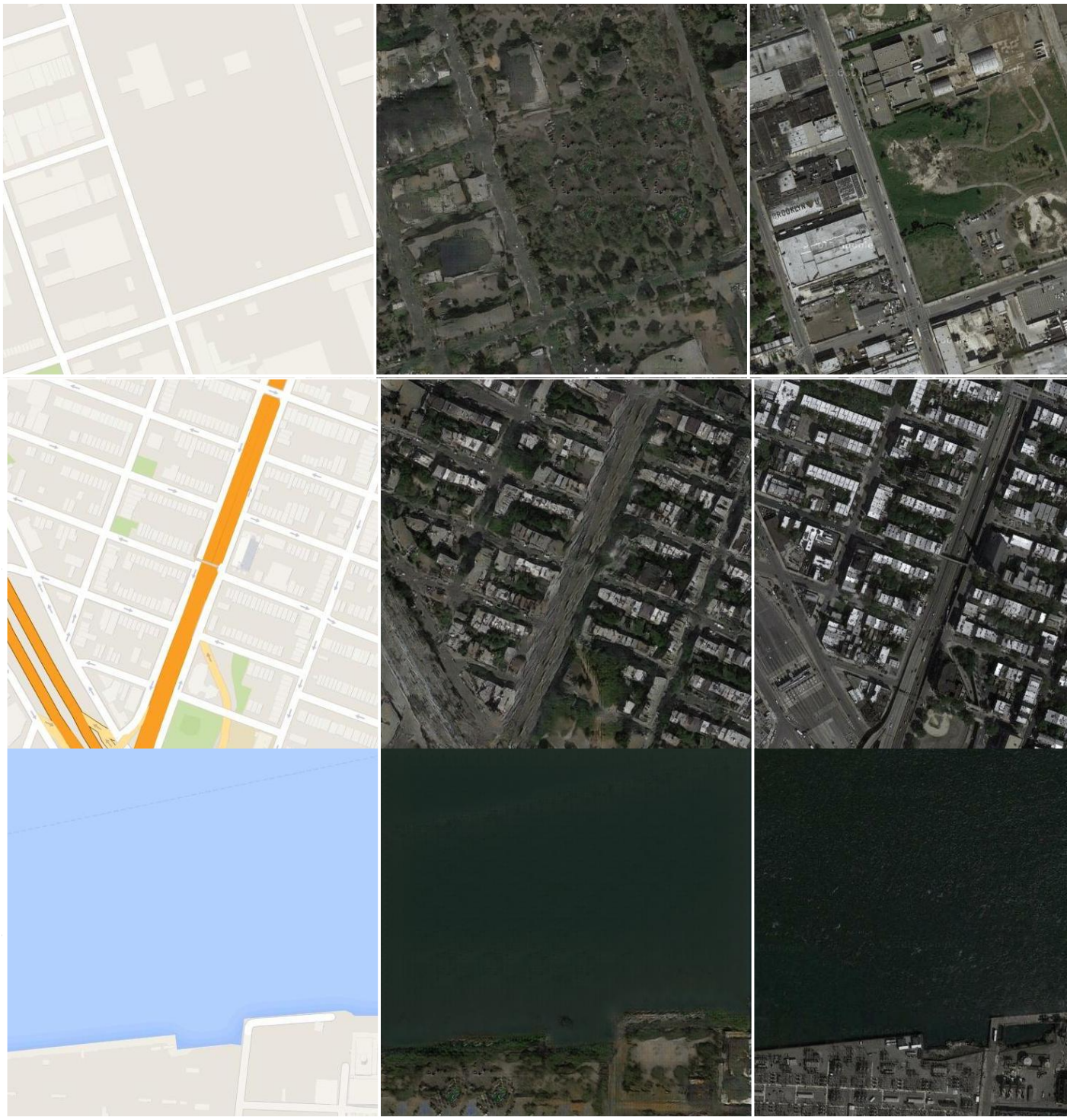

Figure 15: Worst 4 generation of the authors published model, we can see in the second column that the model have difficulties to learn highways view details, and surfaces near to water for example and miss some other details, this results will be discussed in detail in the presentation day.

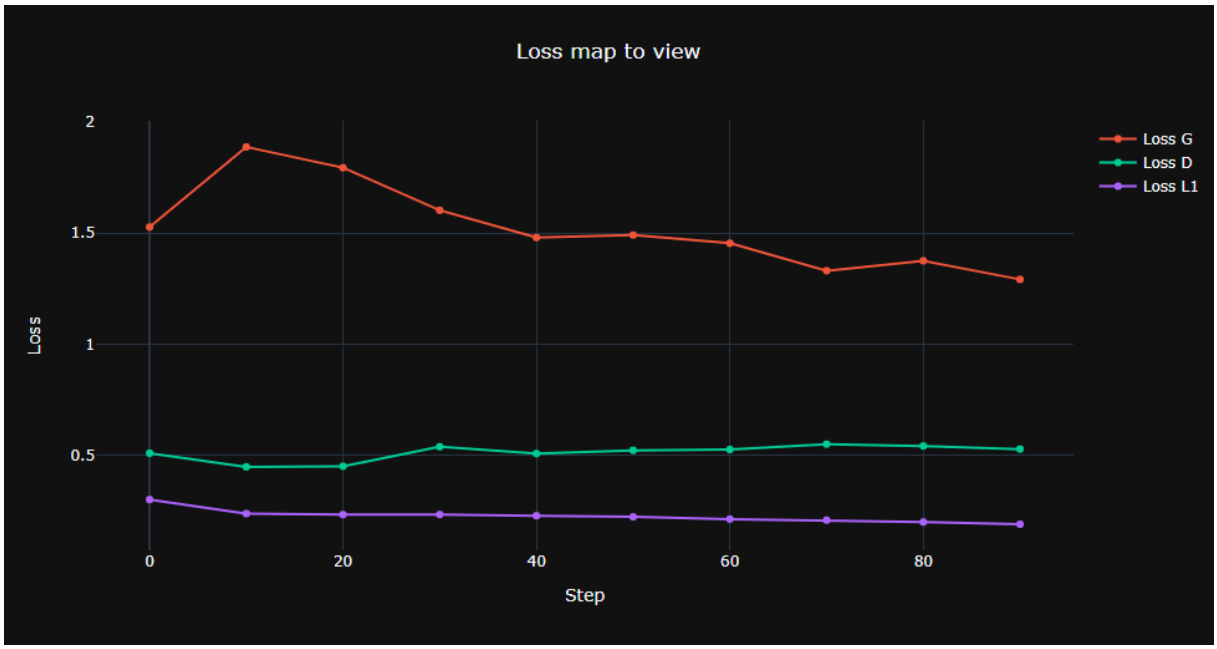

Figure 16: Learning curve of map to view task.

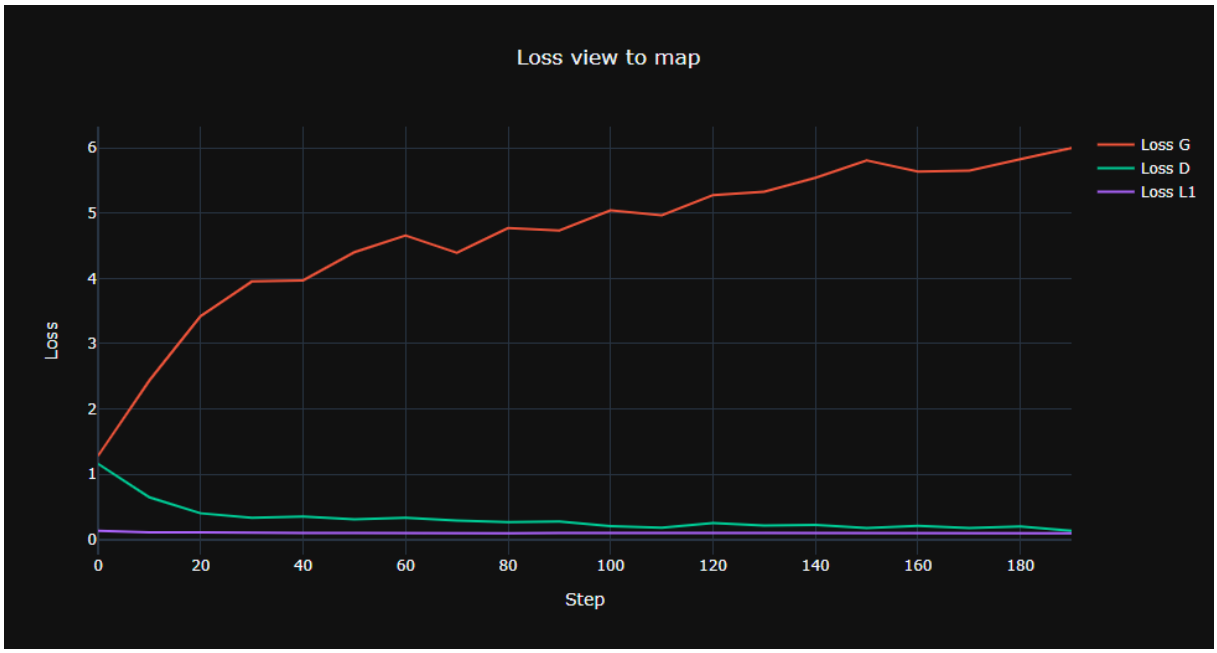

Figure 17: Learning curve of the aerial view to map task.

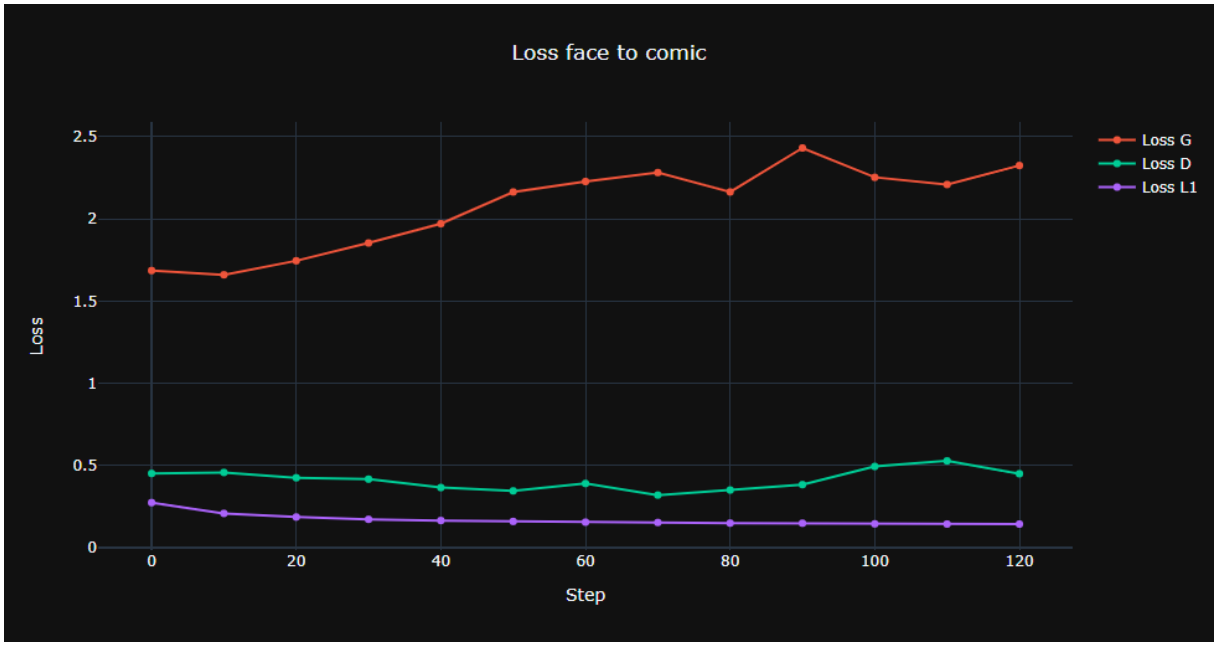

Figure 18: Learning curve of the face to comic task.

Figure 19: cGAN loss of the three tasks.