SN-CycleGAN

Matthieu TOULEMONT, Jean-Baptiste SEVESTRE
M.Sc. MVA, ENS Paris-Saclay
27/01/2019
matthieu.toulemont@ens-paris-saclay.fr
jean-baptiste.sevestre@ens-paris-saclay.fr

Abstract

Recently methods have been introduced to stabilized the training of GANs. Some use carefully designed loss functions, others have introduced normalization schemes to improve the training of the discriminator. In this piece of work, we try to evaluate the generalization of those methods to the CycleGAN framework. In particular, we study the use of the Wasserstein distance associated with a gradient penalty, and the recently proposed spectral normalization.

1. Introduction : CycleGAN Framework

CycleGAN [2] was introduced in 2017 by Zhu and al. ; it allows image-to-image translation with unpaired images, by learning iamge mapping functions from a source domain A to a target domain B, and vice versa.

For the mapping function $G_A : A \to B$ and its discriminator D_A, we have the objective :

$$\begin{aligned}\mathcal{L}_{GAN}(G_A, D_A) = \mathbb{E}[log(D_A(B))] \\ +\mathbb{E}[log(1 - D_A(G_A(A)))]\end{aligned}$$

and likewise for G_B and D_B.

Authors also introduced a cycle consistency loss :

$$\begin{aligned}\mathcal{L}_{cyc}(G_A, G_B) = \lambda_A \mathbb{E}[||G_B(G_A(A)) - A||_1] \\ +\lambda_B \mathbb{E}[||G_A(G_B(B)) - B||_1]\end{aligned}$$

That gives the following full objective :

$$\mathcal{L} = \mathcal{L}_{GAN}(G_A, D_A) + \mathcal{L}_{GAN}(G_B, D_B) + \mathcal{L}_{cyc}(G_A, G_B)$$

This full objective will be modified in our experiments. In our experiments we have used the author's code from [2].

2. Method

2.1. Objective

2.1.1 CycleGAN and SN-CycleGAN (with LS-Loss)

The major contribution of the SN-GAN paper is its spectral normalization scheme. To evaluate its impact we have kept the best CycleGAN loss (LS-loss) and only introduced the SN for the discriminator. We have the following gradients :

$$\nabla_{\theta_{G_A}} \mathcal{L} = \nabla_{\theta_{G_A}} (D_A(G_A(A)) - 1)^2 + \nabla_{\theta_{G_A}} \mathcal{L}_{cyc}(G_A, G_B)$$
$$\nabla_{\theta_{G_B}} \mathcal{L} = \nabla_{\theta_{G_B}} (D_B(G_B(B)) - 1)^2 + \nabla_{\theta_{G_B}} \mathcal{L}_{cyc}(G_A, G_B)$$
$$\nabla_{\theta_{D_A}} \mathcal{L} = \nabla_{\theta_{D_A}} (D_A(B) - 1)^2 + \nabla_{\theta_{D_A}} (D_A(G_A(A)))^2$$
$$\nabla_{\theta_{D_B}} \mathcal{L} = \nabla_{\theta_{D_B}} (D_B(A) - 1)^2 + \nabla_{\theta_{D_B}} (D_B(G_B(B)))^2$$

The Spectral Normalization being already implemented as a pytorch wrapper, applying it amounted to wrap it around each convolutional layer in the discriminator's code.

2.1.2 CycleWGAN-GP

As Martin Arjovski showed in 2016 [3], using the Wasserstein distance for GANs makes the discriminator loss function continuous w.r.t. to the generator's parameters, which supposedly leads to better convergence properties. However, this technique needed the discriminator to be K-Lipschitz which meant that its weights had to be clipped. Gradient Penalty was later introduced to get rid of the weight clipping while keeping the Lipschitz condition. We've implemented the Wasserstein distance with Gradient penalty ourselves. We have the following new gradients :

$$\nabla_{\theta_{G_A}} \mathcal{L} = -\nabla_{\theta_{G_A}} D_A(G_A(A)) + \nabla_{\theta_{G_A}} \mathcal{L}_{cyc}(G_A, G_B)$$
$$\nabla_{\theta_{G_B}} \mathcal{L} = -\nabla_{\theta_{G_B}} D_B(G_B(B)) + \nabla_{\theta_{G_B}} \mathcal{L}_{cyc}(G_A, G_B)$$
$$\begin{aligned}\nabla_{\theta_{D_A}} \mathcal{L} = -\nabla_{\theta_{D_A}} D_A(B) + \nabla_{\theta_{D_A}} D_A(G_A(A)) \\ +\nabla_{\theta_{D_A}} \lambda(||\nabla_t D_A(t)||_2 - 1)^2\end{aligned}$$
$$\begin{aligned}\nabla_{\theta_{D_B}} \mathcal{L} = -\nabla_{\theta_{D_B}} D_B(A) + \nabla_{\theta_{D_B}} D_B(G_B(B)) \\ +\nabla_{\theta_{D_B}} \lambda(||\nabla_t D_B(t)||_2 - 1)^2\end{aligned}$$

and t is a convex combination between real and fake images.

2.2. Architectures and Hyperparameters

For fair comparison we will use the same architectures as those used in CycleGAN [2]. We use ADAM optimizer with $\alpha = 0.0002$, $\beta_1 = 0.5$ and $\beta_1 = 0.999$, Furthermore for the Gradient Penalty we take $\lambda = 10$, for the Cycle Consistency Loss $\lambda_A = \lambda_B = 10$, and we use 200 iterations, with decay from the first 100 ones, with batch-size of 1.

3. Assessment of image quality

3.1. Semantic segmentation metrics

We use the following metrics : Per-pixel accuracy :

$$\frac{\sum_i n_{ii}}{\sum_i t_i}$$

Per-class accuracy :

$$\frac{1}{n_{cl}} \sum_i \frac{n_{ii}}{t_i}$$

Mean Class Intersection-Over-Union :

$$\frac{1}{n_{cl}} \sum_i \frac{n_{ii}}{t_i + \sum_j n_{ji} - n_{ii}}$$

where n_{ij} is the number of pixels of class i predicted to belong to class j, n_{cl} is the number of different classes, and $t_i = \sum_j n_{ij}$ is the total number of pixels of class i

3.2. FCN-Score

FCN-Score was introduced by Isola and al. "If the generated images are realistic, classifiers trained on real images will be able to classify the synthesized image correctly as well." [6]. But FCN-Score only evaluates the quality of one of the two generators. Futhermore, FCN-Score need paired images to be trained.

3.3. Semantic-Segmentation-Score

In this paper "Semantic-Segmentation-Score" refers to "Semantic segmentation metrics" in Isola and al. [6]. Compared to the FCN-Score we find the Semantic-Segmentation-Score more relevant as it does not rely on a previously trained network. However we still need to evaluate the both generators in the CycleGAN architecture. As presented in the next section we have also used a mechanical turk.

3.4. Mechanical Turk

We have asked five persons to evaluate the quality of the images generated through the task : "labels→photo" on the Cityscapes dataset.

We have used the following protocol :

- 5 persons (different from us) evaluate qualitatively the images.
- For each image, they can choose among 5 categories ; "very bad", "bad", "soso", "good", "very good".
- Each person evaluate 150 photos (50 photos for each different methods, that appears randomly) from Cityscapes dataset generate by the generator "labels → photo".
- After this, we take an average on all the categories, for each model. The results are in Figure 1

4. Experiments

To Evaluate the different methods we chose to use the three datasets :

- Cityscapes Dataset, in which we must map pictures taken from a car to semantic labels.
- Aerial photo to Google maps, which contains satellite images and their plan counterpart.
- CMP Facades dataset which countains semantic labels for buildings.

We trained the FCN ourselves. So due to time consideration, the FCN score could only be used for the Cityscapes dataset. CycleGAN* and Ground Truth* scores are obtained thanks to pre-trained model and CycleGAN** and pix2pix** scores are taken directly from the papers [2] and [6]. In the tables below we show the mean performances and their standard deviations on each dataset.

Model	Per-pixel acc.	Per-class acc.	Class IOU
CycleGAN	0.59 ±0.06	0.26 ±0.03	0.18 ±0.03
WCycleGAN-GP	**0.67 ±0.08**	**0.29 ±0.04**	0.19 ±0.03
SN-CycleGAN	0.64 ±0.10	0.27 ±0.04	**0.20 ±0.04**
CycleGAN*	0.52 ±0.07	0.28 ±0.04	0.14 ±0.02
Ground Truth*	**0.85 ±0.08**	**0.37 ±0.05**	**0.30 ±0.05**
CycleGAN**	0.52	0.17	0.11
pix2pix**	**0.66**	**0.23**	**0.17**

Table 1. FCN-scores for different methods, evaluated on Cityscapes labels→photo.

Table 1 shows that, even if Ground Truth FCN-score is very correlated with our perception, WCycleGAN-GP has the best FCN-score in general. This is a very surprisingly result compared to the images provided in Appendix.

In the litterature [2], and [6], results are only evaluated on Cityscapes Dataset. But data scraped from Google Maps and CMP Facades dataset can be seen as "photo→labels" too. So in this paper we evaluated our results on this datasets, in table 3 and 4.

Model	Per-pixel acc.	Per-class acc.	Class IOU
CycleGAN	**0.55 ±0.09**	**0.19 ±0.03**	**0.13 ±0.02**
WCycleGAN-GP	0.49 ±0.08	0.13 ±0.02	0.09 ±0.01
SN-CycleGAN	0.53 ±0.09	0.18 ±0.03	0.12 ±0.03
CycleGAN**	0.58	0.22	0.16
pix2pix**	**0.83**	**0.36**	**0.29**

Table 2. Semantic-Segmentation-Scores of photo→labels for different methods on Cityscapes dataset.

Model	Per-pixel acc.	Per-class acc.	Class IOU
CycleGAN	**0.66 ±0.13**	**0.37 ±0.05**	**0.27 ±0.06**
WCycleGAN-GP	0.48 ±0.09	0.20 ±0.02	0.12 ±0.02
SN-CycleGAN	0.64 ±0.14	0.34 ±0.05	0.24 ±0.06

Table 3. Semantic-Segmentation-Scores of aerial photo→maps for different methods on data scraped from Google Maps.

Model	Per-pixel acc.	Per-class acc.	Class IOU
CycleGAN	**0.29 ±0.12**	**0.15 ±0.05**	**0.09 ±0.03**
WCycleGAN-GP	0.24 ±0.08	0.14 ±0.03	0.07 ±0.02
SN-CycleGAN	0.21 ±0.09	0.11 ±0.03	0.06 ±0.02

Table 4. Semantic-Segmentation-Scores of photo→labels for different methods on CMP Facades.

The results show in Table 2, 3 and 4 show different results than those from Table 1. Indeed, CycleGAN provides the best Semantic-Segmentation-Scores followed by SN-CycleGAN and WCycleGAN-GP.

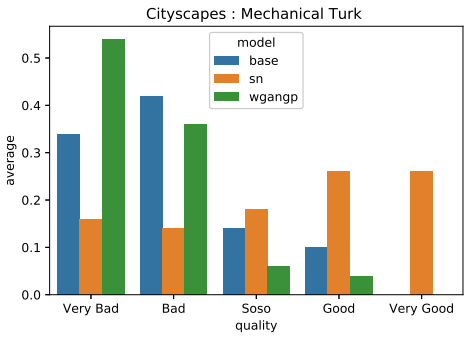

Figure 1. Qualitive results for different methods, evaluated on Cityscapes labels → photo.

Figure 1 provides qualitative results from the "labels → photo" generator.

It shows that SN-CycleGAN seems to generate better images than the other methods, whereas WCycleGAN-GP generated the worst images according to our mechanical turks.

Interestingly, these results differ from the FCN-scores computed on the same images. Ultimately, we wonder if the FCN-score is good metric for assessment of image quality. However, one needs to be careful when interpreting such results as only a small set of people where asked to participate in this evaluation. We invite you to check the images produced in the Appendix for the Cityscapes Dataset. For each labels→photo sample, the WCycleGAN had the best FCN score and the lowest quality according to our jury.

Those results are only for the labels→photo task. When looking at the quality of the images generated for the photo→labels task, it seems that the CycleGAN's have the best quality, followed by SN-CycleGAN and WCycleGAN-GP. The image quality coincides with the rankings of tables 2, 3.

5. Conclusion

As seen previously, WCycleGAN-GP does not reach the CycleGAN's quality for both photo generation and label generation. On the other hand, SN-CycleGAN seems to generate better photos than all the other methods, but fails to do the same for the photo→labels task.

As we could not try different hyperparameter settings, the difference in performance might be due to a specific setting working better for SN-CycleGAN than for WCycleGAN-GP.[5] shows SN-GAN is stable with respect to hyperparameter choice.

Our experiments have shown that the FCN-score is not always correlated to image quality, at least on the Cityscapes and according to our jury. Semantic-Segmentation-Scores proved otherwise for the labels→photo task.

Ultimately we would have liked to tune hyperparameters, perform ablation studies or trying other losses. For instance, we could try using a Hinge Loss instead of a LS-Loss as done in [5].

References

[1] Generative Adversarial Networks: I. Goodfellow et. al, NIPS 2014

[2] Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks, JY Zhu et al., ICCV 2017.

[3] Wasserstein GAN, 2017

[4] Improved training of Wasserstein GAN, I. Gulrajani and al.

[5] Spectral Normalization for Generative Adversarial Networks, T. Miyato and al.

[6] Image-to-Image Translation with Conditional Adversarial Nets. Isola et al. CVPR 2017

6. Appendix : Images

Figure 2. Images come from the ”labels→photo” generator. Left to right : Image, CycleGAN, WCycleGAN-GP, SN-CycleGAN, Ground Truth. WCycleGAN-GP has the best FCN-score on all those examples.

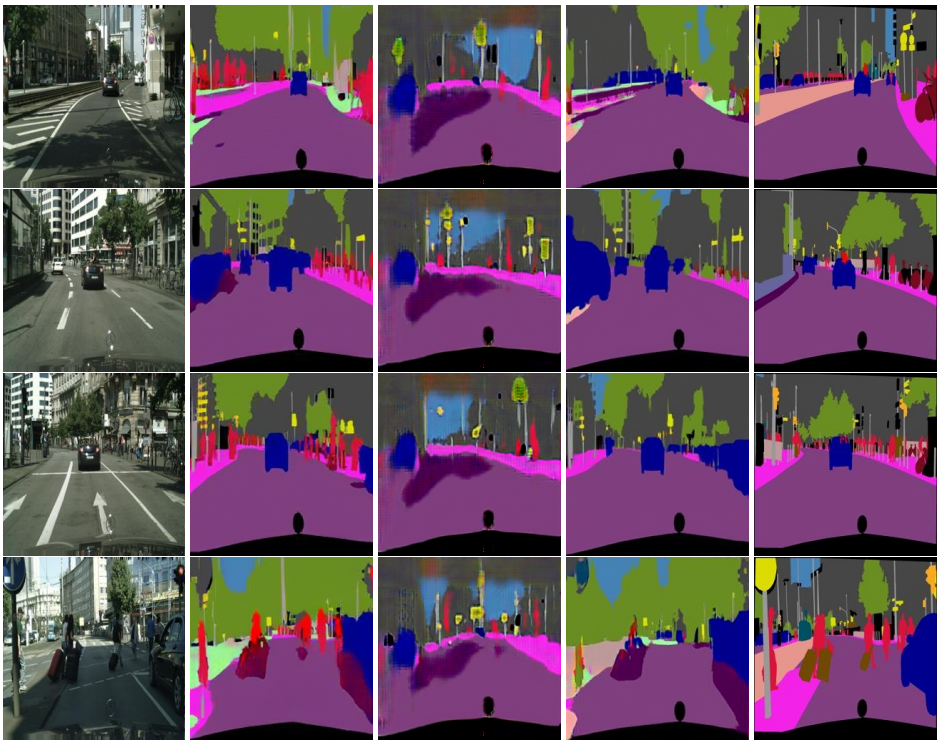

Figure 3. Images come from the ”photo→labels” generator. Left to right : Photo, CycleGAN, WCycleGAN-GP, SN-CycleGAN, Ground Truth.

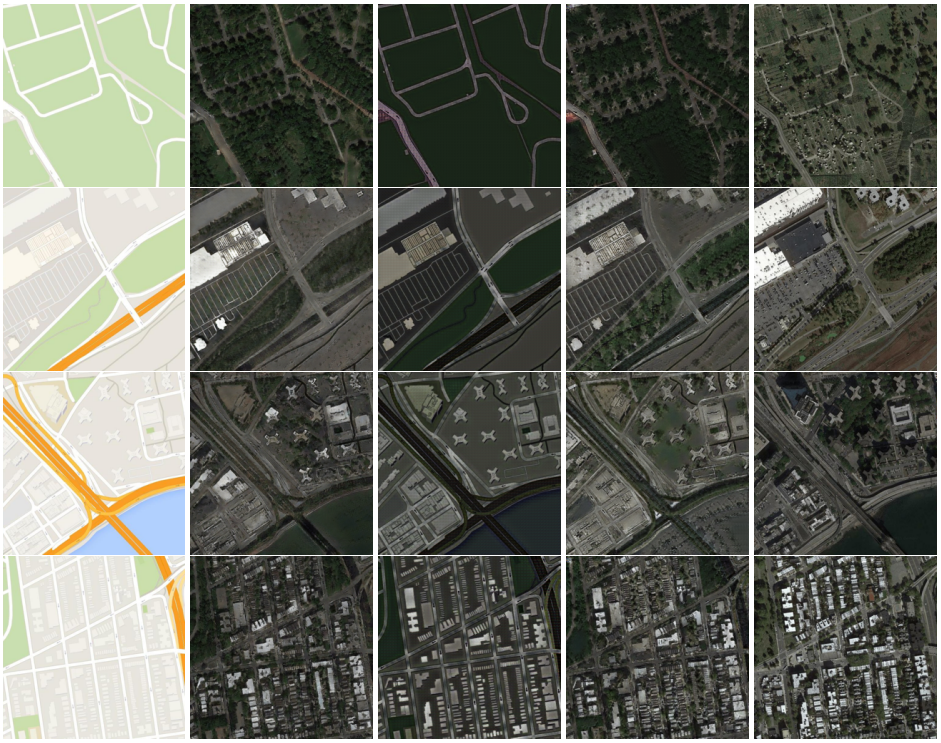

Figure 4. Images come from the ”labels→photo” generator. Left to right : Image, CycleGAN, WCycleGAN-GP, SN-CycleGAN, Ground Truth.

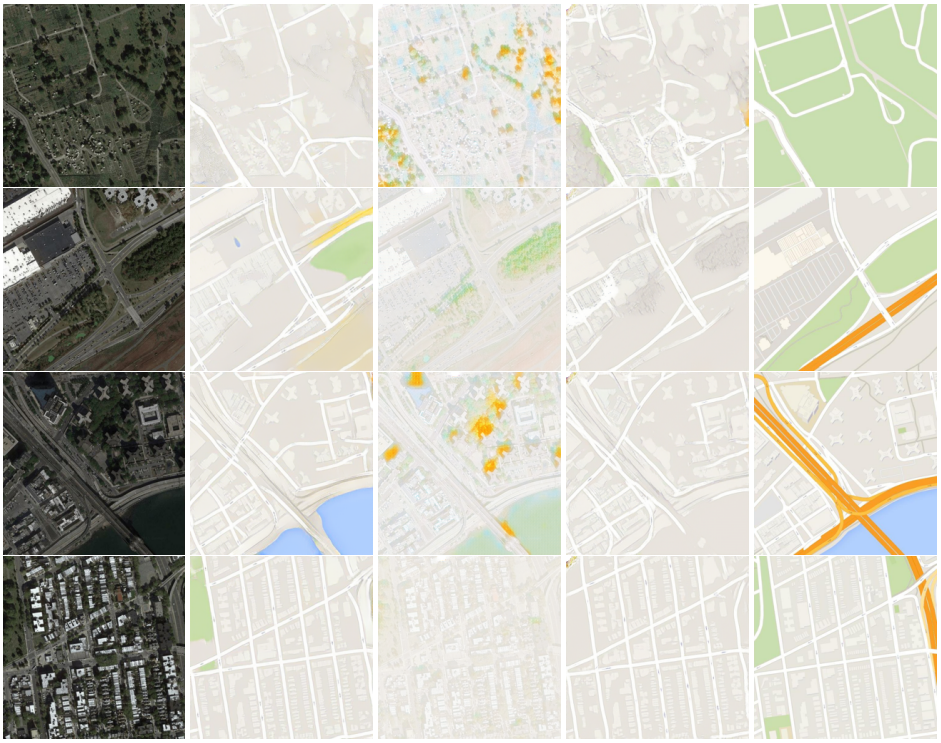

Figure 5. Images come from the ”photo→labels” generator. Left to right : Photo, CycleGAN, WCycleGAN-GP, SN-CycleGAN, Ground Truth.

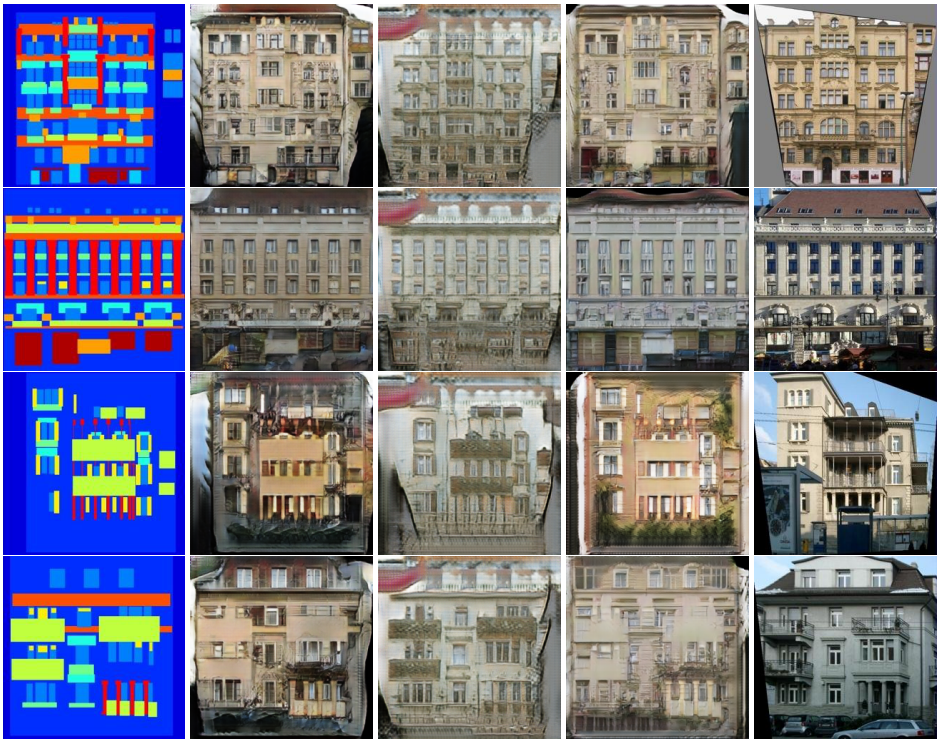

Figure 6. Images come from the ”labels→photo” generator. Left to right : Image, CycleGAN, WCycleGAN-GP, SN-CycleGAN, Ground Truth.

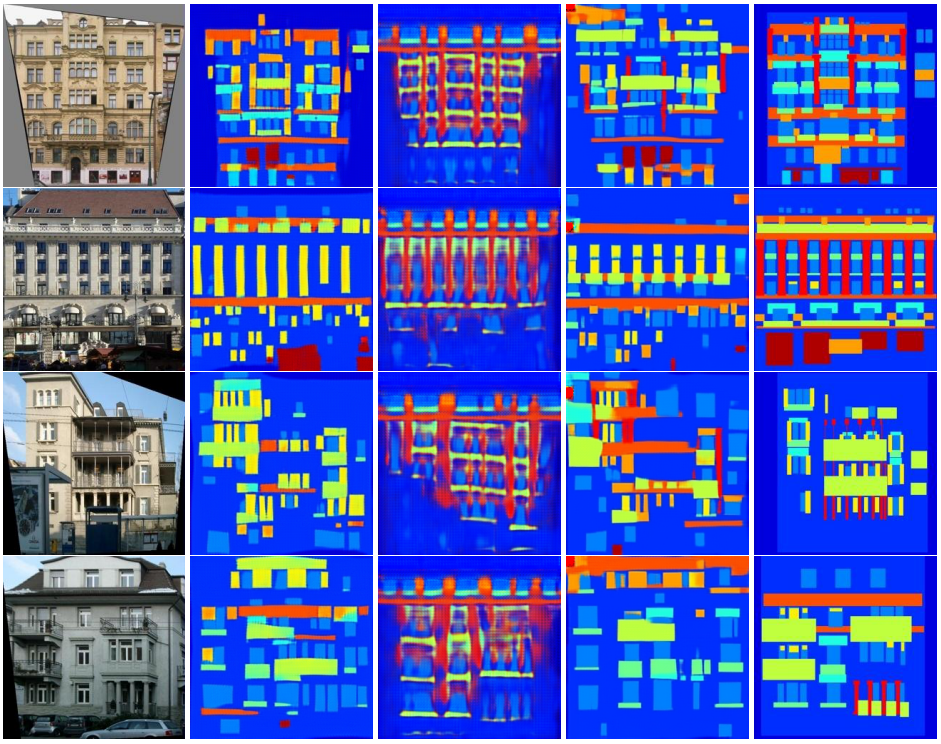

Figure 7. Images come from the ”photo→labels” generator. Left to right : Photo, CycleGAN, WCycleGAN-GP, SN-CycleGAN, Ground Truth.