UCLouvain

École polytechnique de Louvain

Generative adversarial networks : virtual satellite images generation

Author: **Lucas Ody**
Supervisor: **Pierre Schaus**
Readers: **Pierre Schaus, Benoît Macq, Vincent François-Lavet**
Academic year 2019–2020
Master [120] in Computer Science and Engineering

Contents

1.1 Introduction

Machine learning approaches typically perform better the more training data available. A larger data set reduces risks of obtaining an over fitting model by simply learning on a more comprehensive training set. However, such data sets are hard to come by, even more so in specialized domains like medicine. Satellite imagery is no exception, and to ease machine learning projects in this domain, this work aims to use conditional image generation to enrich available data. Over the past few years, Generative Adversarial Networks (GANs) yielded impressive results in terms of generative plausible and sharp images.

1.1.1 Objectives

The objective of this work is threefold :

- **Designing and training a generative model**
- **Evaluate the quality of the produced images**
- **Make use of the generated images**

The first and main objective is to produce a model capable of generating images for practical purposes. The original objective of this work as a mean to expand a data set for another model's training, brings two limitations to the generative model: it must be practical and adaptable. Ideally, the model should be able to scale its output size as a way to compromise between training time and generated sample quality, as well as changing the conditioning of the generation, allowing the generation of different kinds of data-label pairs. The applications of satellite imagery can be diverse, be it detecting changes in landscapes, measuring the car activity of a road, or highlighting collapsed structures in archaeological sites, thus, ideally images could be conditioned with latent features like time of the day and season as well as low-level information like the layout of roads and buildings.

The second objective is inherently subjective, as even real images can be considered of different quality according to different judges, or criteria. The accuracy, and diversity of the generated virtual samples will then mostly be compared subjectively. However, a number of methods have been created over the years to try and compare generative models in a more objective fashion, like the inception-based scores[22][16]. Another criteria of quality assessment is the resolution of the generated images, as higher resolution images potentially yield more information, but are more likely to contain irregularities and being less realistic. Higher resolution images can thus be considered better than lower resolution ones if their quality

evaluation is similar.
The last objective is to produce a data augmentation experiment. A simple model will be trained on a dummy task on a data set with and without virtually generated images, in an effort to observe the effect of using generative models for data augmentation. Data augmentation represent the set of method used to increase the amount of correct information contained in data set. Classic image-related data augmentation methods consists of cropping, flipping, translating, and so on, each image in a data set. And while thoses method are fast and efficient, they are clearly limited in the amount of potential added information, hence the study of using generative models for data augmentation. Past experiments[1][23] have shown the ability of using GAN for data augmentation on relatively low dimension data (e.g. the EMNIST data set) to enhance the performance of machine learning algorithms.

1.1.2 Image generation

Data generation is a machine learning problem that could be summarised as an estimation of a data probability distribution. In the context of image generation, the data distribution potentially has very high dimension size, depending on the resolution (e.g. The ATSC digital television 480p standard uses images defined by 900k values). The goal of a generative model is to create an approximation of this distribution, and usually producing new samples of said distribution. This distribution is what dictates the relation and constraints between each pixels; For instance, one would expect a dog picture to contain little, if any, blue colored patches, as well as few geometric shapes, given the organic nature of the dog. Out of a thousands of different possibilities, only a few pixel arrangements could be considered a representation of a dog, and learning and emulating those relations is the purpose of the generative model.
Artificial neural networks provide two advantages to solve such a problem: according to the universal approximation theorem, any real-valued, continuous function on the m-dimensional unit hyper-cube can be approximated by a multi-layered perceptron model given the right parameters, and the back-propagation algorithm allows for training in linear time complexity in regards of data set size and number of training epochs.

To enable a deterministic model to generate multiple different images, the model must be provided with some sort of image seed. In the original GAN paper[12] this was a vector z sampled from a chosen random distribution, some other works[3] used higher level features to condition the generated image, and sometimes, both were used.[15]. Using high level feature as part of this seed is called conditional image generation, and the model must be trained accordingly: the generated image must respect those features.

Thus, the generative model G realises mapping from the data distribution P_z of the input seed, to the estimated data distribution of real data P_g:

$$z \sim P_z$$
$$G(z) \sim P_g = \hat{P}_{real}$$

In our context of data augmentation, a random seed is necessary to generate new samples of the real distribution, as well as conditional data to generate data adapted to another machine learning task. As such, the input used throughout this work is a semantic map indicating the layout of the satellite image, as well as a 100-dimensional random vector from a uniform distribution. The input seed is then a 103 by n by m matrix, for a n by m output image. The same random vector is copied to condition each pixel for the random vector to characterize the whole image, only the semantic map differs for each pixel at input level.

A straight-forward method to conditional image generation using neural networks is a convolutional encoder-decoder network, trained with a error gradient descent optimisation algorithm, and this method had been successfully used in the past.[3]. The model implements a learnable function $f(z)$, and each iteration computes an error, or loss function, between this output value and the real value $Loss(f(z), x)$. This loss function, usually the L1 or L2 losses, is then minimized by gradient descent to reach a local minimum of the loss function where, hopefully, $f(z)$ gets close to the real data distribution.
An encoder-decoder network architecture works by extracting features from the input image generation seed, by having the seed being progressively down-sampled, as a way to extract higher level features. The second part, the decoder, is a ensemble of upsampling layers which will lead the bottleneck layer to the target dimensions. The upsampling layers, are the so-called "deconvolution" layers, more accurately described as backwards strided convolution[13], enable for learned upsampling in a convolutional way.

However, as seen on figure 1.1, such a method yields poor results.

This is the consequence of this naive optimisation method: minimizing the average mean square error, (and to a lesser extent, Mean Absolute Error) will train the model to produce an average of images, hence it produces blurry images. This model, which took as input a map image as well as a 10-dimensional noise vector (from a uniform distribution) gained the color palette of a mix of sea blue and city grey, and the original layout of the map is recognisable, but the poor quality makes them really distinctive from real images. The added value of using such images for data augmentation is also negligible. Hence, various models have been developed to alleviate those issues like variational auto-encoders, Boltzmann machines or GANs.

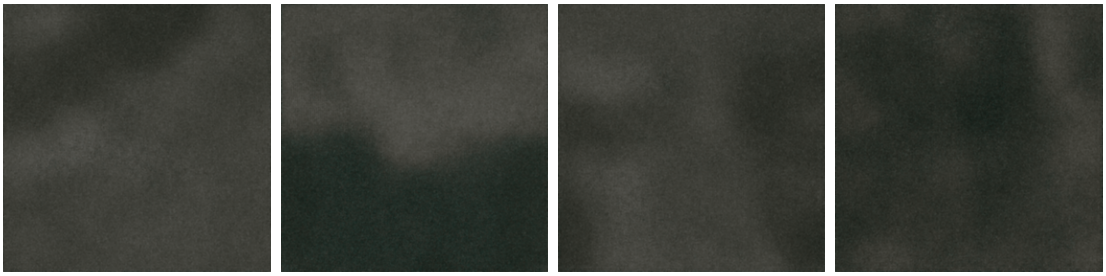

Figure 1.1: **samples generated from simple encoder-decoder**

1.2 Generative Adversarial networks

1.2.1 Principle

The advantages of GAN models is their ability to subjectively generate more realistic samples than other models[12], as well as providing a larger range of unconstrained generator models to be used, when compared to other generators models.

A generative adversarial network (GAN) actually consists of 2 artificial neural networks: the generator itself, and a discriminator. This discriminator network's job is to classify **real** from **fake** samples (i.e. discriminate a sample taken from the actual data set from a artificially generated sample from the generator network). And instead of training the generator to create data in the likeness of the training set, it is trained to fool this discriminator. As described in the original GAN paper [12], this allows a definition of the GAN Loss, based on the following two-player optimization problem:

$$min_G max_D V(D,G) = \mathrm{E}_{x\sim p_{data}}[Log(D(x))] + \mathrm{E}_{z\sim p_z}[1 - Log(D(G(z))]$$

where G is the generator model, D is the discriminator model, p_{data} is the original data distribution, and p_z is the distribution used as a seed for generation.

To maximize this value, the discriminator model should reward real data sample x with high values, and give low values to the fake counterpart $G(z)$. On the other hand, the generator must produce values maximizing $D(G(z))$, hence tricking the discriminator with counterfeits.
Ian Goodfellow and his colleagues proved in the original GAN paper[12] that optimizing this problem will yield a Generator emulating a distribution p_g approaching the data distribution, on the hypothesis of a perfect generator, capable of emulating any distribution. While artificial neural networks cannot replace such a perfect generator, they also show empirically how ANN are suitable models to use instead.

One step of the stochastic gradient descent process for GAN models is as follows:

- sample a batch of m examples $\mathbf{z}$ from noise distribution p_z
- sample a batch of m examples $\mathbf{x}$ from real distribution p_data
- update the discriminator by ascending its gradient:

$$\nabla_D \sum_{i=1}^{m} [log(D(x_i)) + log(1 - D(G(z_i)))]$$

- update the generator by descending its gradient:

$$\nabla_G \sum_{i=1}^{m} [log(1 - D(G(z_i)))]$$

1.2.2 Results

The generator implementation consists of first 5 consecutive convolutional layers, using rectified linear unit (ReLU) activation functions and batch normalization[20]. Then, 5 layers of backwards strided convolution, also with ReLU activation function and batch normalisation, as well as dropout layers[9].

As one can see on figure 1.2, the GAN model provide a generator able to create details. However, some defaults are present in most images, especially on areas where the semantic map has large uniform areas, the prime example being the sea.

1.2.3 Limitations

Some problems arise when trying the model with a higher resolution, as shown in figure 1.3.
The model used for producing 512 pixels wide images is deepened in comparison to its lower resolution counterpart to enable finer detail generation. The results shown here were produced by a generator with 7 downsampling, and 7 upsampling layers, trained with a 8-layered discriminator.

The grid effect shown here is a result of *training instability*. Looking at a training sample from the 4 preceding training epochs, as shown on figure 1.4, the quality of the samples is shown deteriorating, while the unwanted grid pattern appears. Training instability is one of the two most well known flaws of the GAN model,

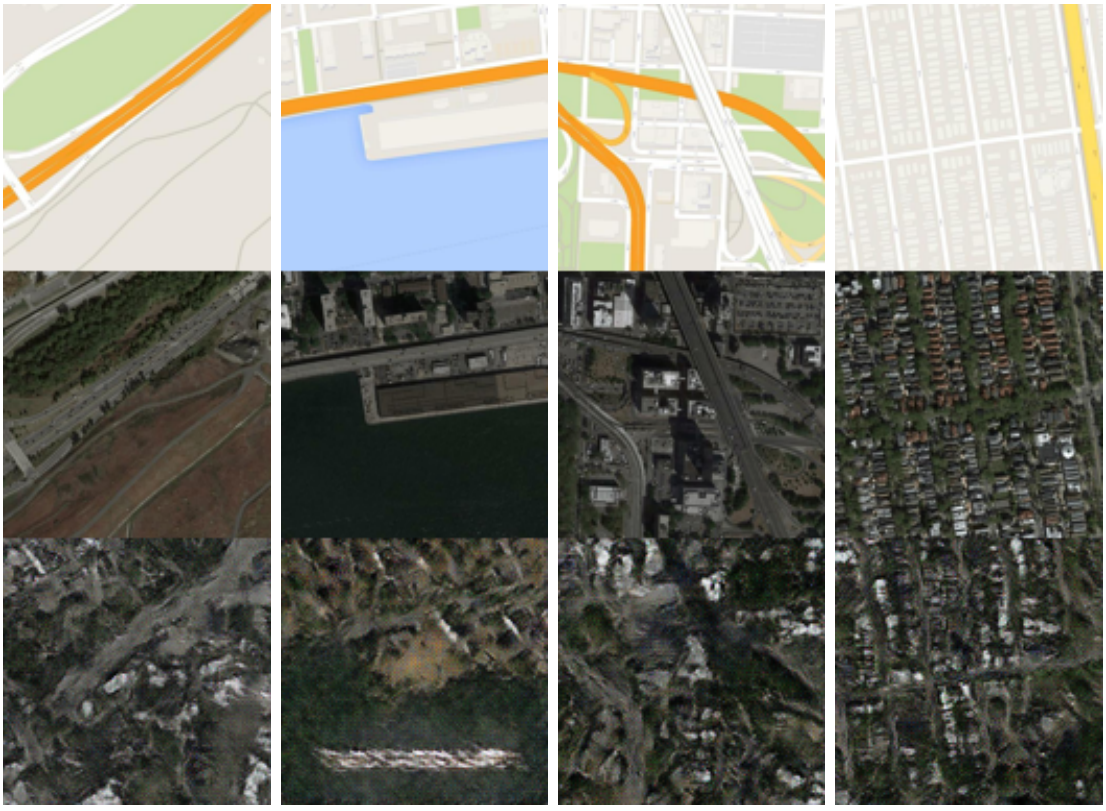

Figure 1.2: **samples generated from the gan model**. From top to bottom: input image, real satellite image, generated image, respectively. (128 by 128 pixels)

and this behavior is caused by the gradient used to train the generator.

The original gradient used with the generator, $\Delta G = \nabla_G \mathrm{E}_{z\sim pz}[1-log(D(G(z))]]$, was suspected to cause a vanishing gradient problem when the discriminator performs well since the original GAN publication[12], so $\Delta G = \nabla_G \mathrm{E}_{z\sim pz}[-log(D(G(z))]]$ is usually used instead. Using this gradient to train the generator still holds the convergence property, while the gradients generated have higher norm.
However, If the generator fails to catch up to the discriminator, the inverse effect will occur. As it was shown[5], as the discriminator gets closer to the optimal discriminator, the gradient instead grows drastically, inducing instability in the training of the generator: poorer results will further increase the accuracy of the discriminator, which will in turn result in a increased generator gradient value.

It is important to note that such behavior is not guaranteed to occur. Some satisfactory results can be seen in the samples generated at training time. But due

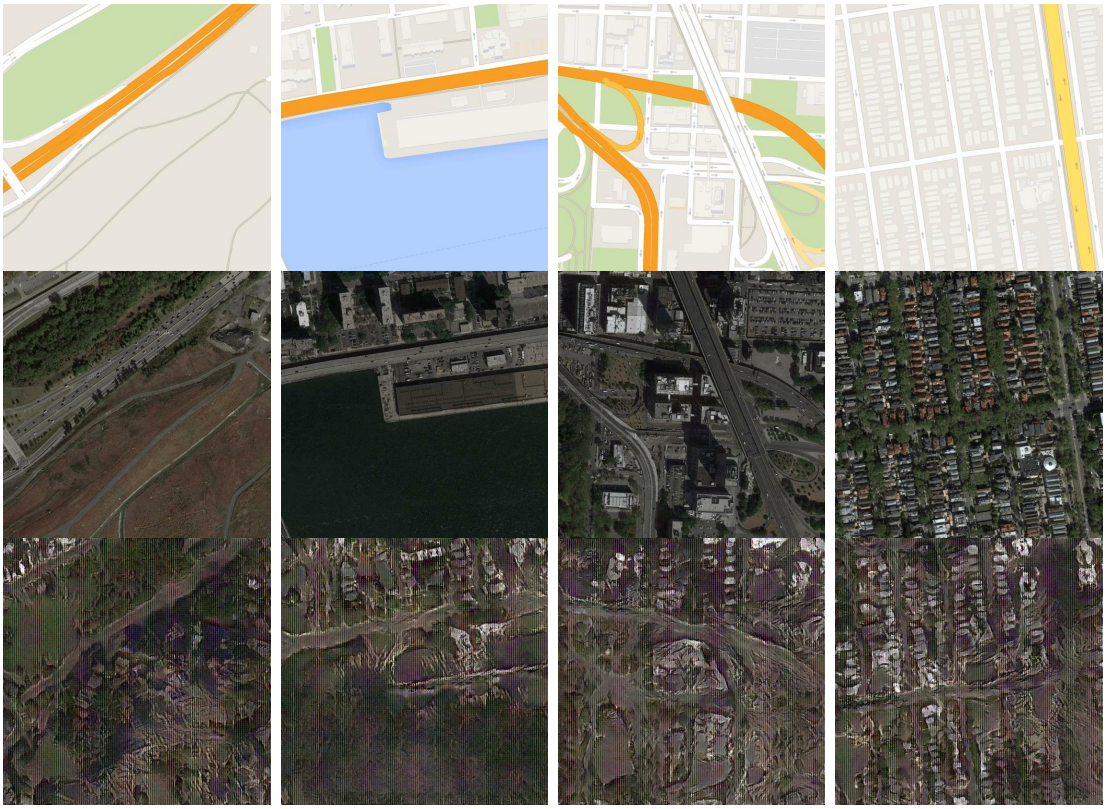

Figure 1.3: **samples generated from the gan model**. From top to bottom: input image, real satellite image, generated image, respectively. (512 by 512 pixels)

to the randomness of the training stability, the simple GAN model is insufficient to be used at higher resolutions. Adding more layers for deeper networks, or more filters for wider ones result on even more instability, as the increased amount of parameters makes them more difficult to train, while smaller networks in the like of those used for 256 by 256 pixel resolution fail to produce reasonable results.

The other well known flaw of GAN model, is called mode collapse. Mode collapse is a phenomenon impacting the diversity of the generated data. This comes from the fact that the generator is not trained to match the true distribution *per se*, it learns to fool the discriminator instead. The mode collapse problem occurs when the generator learns to generate a single realistic sample from any input seed. However, in a conditional generation context, the discriminators is given the conditional data as well, forcing the generator to be at least as diverse as the condition data.

Figure 1.4: **training samples generated from the gan model** left to right : training epoch 50, 60, 70 and 80. (512 by 512 pixels)

1.3 Image to image translation baseline

The so-called "pix2pix" model[19] is a image-to-image translation model based on conditional GANs networks, proposing many improvements to the standard GAN model to produce better images. This improvement also stabilises the training of bigger generator models

1.3.1 Specificity

The pix2pix model is based on 3 different main points:

U-net Generator

The U-net is at first a symmetric encoder-decoder network, the input image seed is progressively down-sampled to a bottleneck, and then a series of upsampling layers transform back the data to its original dimensions. Then, skip connections are added each pair of same-sized layers, to form the U-net as it was first described[18] . Theses skip connections aim to preserve low-level information through the network, as well as reducing risks of gradient vanishing by virtually shortening the network. In the case of satellite image generation, the locations of buildings, roads and fields is the same in the source map image to the destination satellite image, hence providing this source image helps at building the latter.
Assuming a neural network part would approximate function $H(x)$ by implementing function $F(x)$, adding a skip connection will result in the network implementing function $F(x)+x$, then the network will learn to approximate function $H(x)-x$[10]. Moreover, the skip connection adds no computational complexity or trainable parameter to the network, thus not impacting training speed.
The U-net architecture is then easier to train, and usually performs better than a simple encoder-decoder. This better generator stabilises GAN training by avoiding the appearance of a too accurate discriminator.

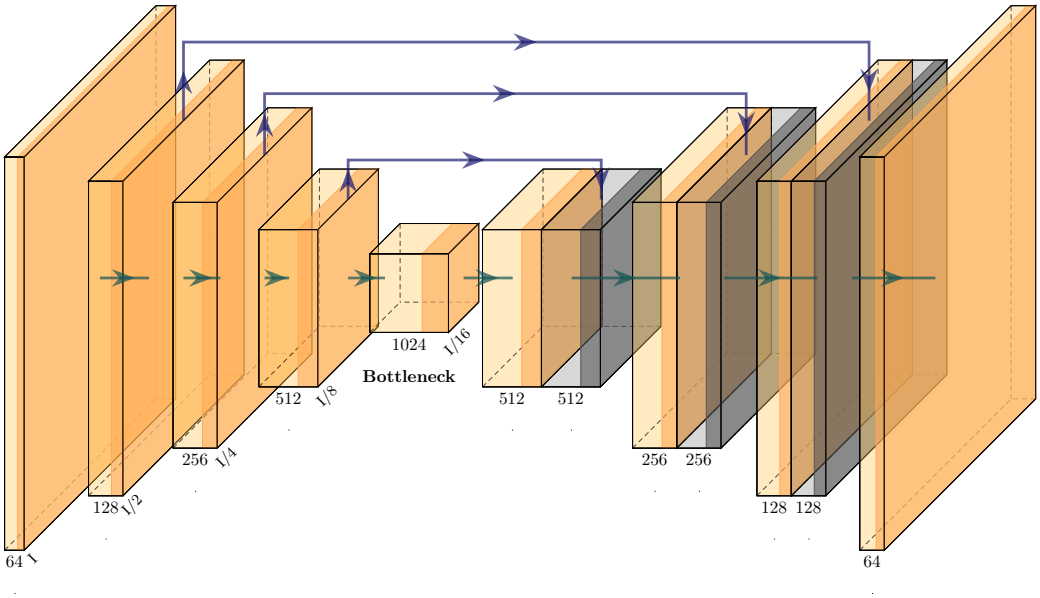

Figure 1.5: unet architecture

Markovian Discriminator (Patch GAN)

Assuming that two pixels of an image are independent if they are at least at a distance d from each other, the image can be modeled as a Markov random field. Using this, we define a smaller discriminator, spanning a n by m pixel area, with n and m being smaller than the width and length of the image, then this discriminator is run in a convolutional fashion and the results are averaged. The model will then learn to generate images such that each of the n by m part of the generated image is a plausible part of image sample.
This patch discriminator is smaller than a full-fledged discriminator, thus trains faster, but more importantly, it helps the GAN model to generate adequate detailing, since it looks as smaller parts of the image, much like one would inspect a photography using a magnifying glass. However, it looses the ability to generate a plausible sample as a whole, and that problem is alleviated by the Hybrid objective function.

Hybrid objective function

It was shown previously [8] that mixing the GAN loss with more traditional losses, such as the L2 or L1 Loss could improve the generation. Although these losses provide poor results on their own, their influence is minor when associated to the

GAN loss while reducing the frequency of artifact occurrence in the generated images.

With target y and input c, and noise z:

$$L_{L1}(G) = \mathrm{E}_{c,y,z}[||G(c,z) - y||] \qquad L_{L2}(G) = \mathrm{E}_{c,y,z}[(G(c,z) - y)^2]$$

We can express the original minmax game using a Loss function. Moreover, the discriminator has an additional input c in our conditional generation context.

$$L_{CGAN}(G,D) = \mathrm{E}_{x\sim P_{data}}[Log(D(c,x))] + \mathrm{E}_{z\sim p_z}[1 - Log(D(c,G(z))]$$

We can now define a hybrid objective function to define the optimal generator $G*$:

$$G* = argmax_G min_G max_D L_{CGAN}(G,D) + \lambda * L_{L1}(G)$$

This introduces a new meta-parameter for the learning process: λ which weights the L1 loss. The L1 loss is produces more realistic images over the L2 Loss because the norm of the error produces less blurry results, whereas minimizing a square function will tend to smooth the produced images.

1.3.2 Results

Using a unet model with 9 downsampling layers, a patch discriminator with a 70 by 70 pixel field of view, and a weight $\lambda = 50$ for the L1 loss, the results shown on figure 1.7 can be produced.

1.3.3 Limitations

There's at least two shortcomings of the pix2pix model: the quality of the produced images are still lacking compared to state-of-the-art image generation, and the model architecture is not scalable: either the same U-net can be used for higher dimension data, which could be insufficient, or larger U-net are used, risking the unstable training to occur.

1.4 Higher resolution generation: a scalable model

1.4.1 Embedded residual networks

To provide a stable training for any image resolution, the idea is to develop a coarse-to-fine generator[21]. Multiple generators are defined to generate data at

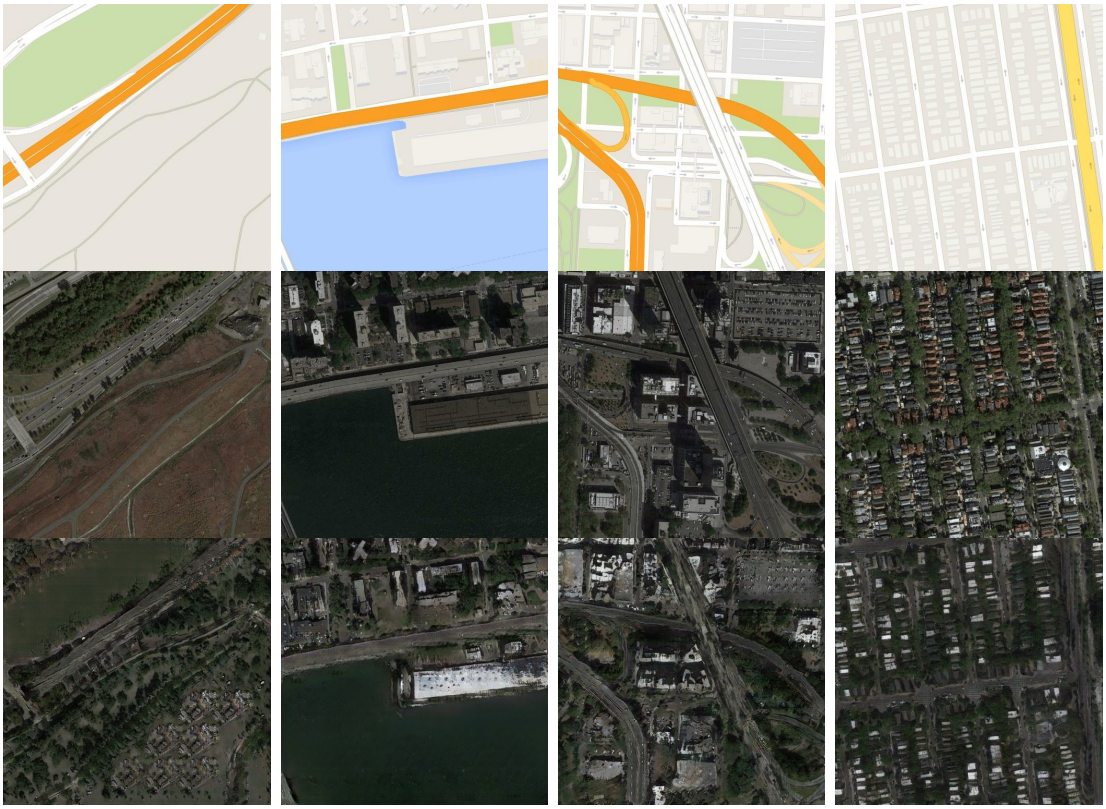

Figure 1.7: **samples generated from the pix2pix model**. From top to bottom: input image, real satellite image, generated image, respectively. (512 by 512 pixels)

multiple scale. The first generator, $G_1(x)$, the coarse generator, will be trained to generate images at a low resolution. Then, a second generator, $G_2(x)$, the enhancer network, will be trained to generate higher resolution images using the features generated by G_1 from the downsampled semantic map. The last step of the training is to train both networks together, forming the full-fledged $G(x)$, fine-tuning them to produce finer details. This architecture can be further expanded by adding additional networks, using $G(x)$ as a coarse generator and the new $G_3(x)$ as the enhancer, and so on.

It is important to respect the 3 stages of training. Obviously, skipping the training of the coarse network alone is equivalent to directly training a network for high resolution images, inducing instability risk in the training. Skipping the training of the enhancer network with a fixed coarse generator will degrade said coarse generator, because the untrained enhancer will first generate noise-like images whatever the quality achieved by the coarse generator. And finally, the fine-tuning step is the one enabling the network to generate details at its full capacity.

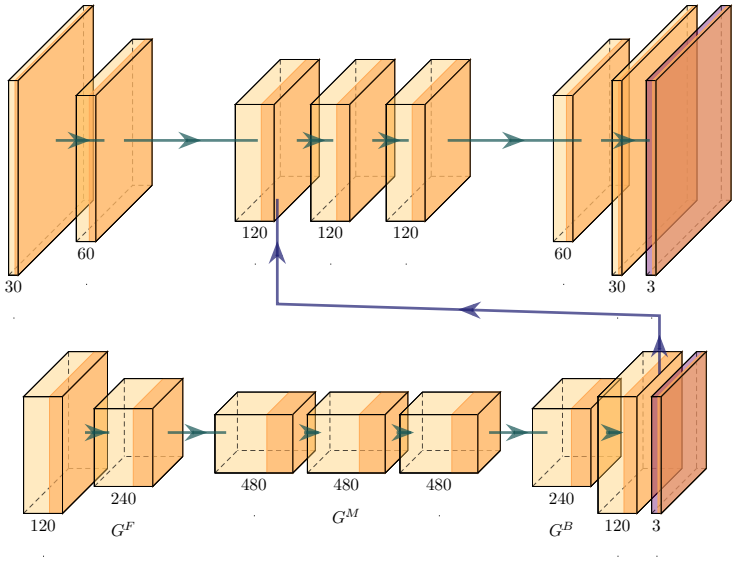

Figure 1.8: Embedded generators architecture

The scalable generators are divided into 4 parts : a downsampling front-end $G_1^F(x)$, the middle part made from convolutional residual blocks[10] $G_1^M(x)$, the upsampling backward strided convolutional back-end $G_1^B(x)$ and finally an output convolutional layer $G_1^O(x)$ which produce the 3-channel image. The residuals blocks consists of 2 consecutive convolutional layers with a skip connection.
With this architecture, the generator embedding is done by adding the output of the back-end of the coarse generator to the input of the middle part of the enhancer network: $G_2^M(G_2^F(x) + G_1^B(downsample(x)))$.
The U-net architecture model is less suitable for this scaled generation. One possibility would be to divide G_1 and G_2 such that G forms a fully fledged U-net, but then G_2 would be much smaller than G_1, reducing the stability gained by embedded generators. On the other hand, using 2 full U-nets is impractical from a computational time point of view, with the 3-stage training making it all the more time consuming to train.

1.4.2 Multi-scale discriminators

For the GAN model to produce detailed yet coherent images, multiple discriminators will be used. the discriminators are identical, but work at different scale. The real and the generated image will be downsampled by a factor 2 to train the second discriminator. Using multiple scale discriminators share some motivation with the choice of patch-discriminator of the pix2pix model: by avoiding a deeper or larger

network, the training remains fast and saves memory. Simpler models also avoid overfitting in comparison to larger models, since they lack the capacity to memorise much.
Using multiple discriminators also guarantees scalability, since a third or forth discriminator can be added with further downsampling for even higher resolution images generation.

1.4.3 Improved GAN loss

The standard loss of the generator network consists of fooling a discriminator's judgment between fake and real samples. This principle can be improved used a feature matching loss. Feature matching is an extension of the GAN loss: instead of training the generator to match the output of the discriminator between real and fake samples, it is trained to match the value at each layer of the discriminator.
With D_k^i being the ith layer of discriminator k, the feature matching loss over a discriminator can be defined as follows:

$$L_{FM}(G, D_k) = \mathrm{E}_{c,x} \sum_i ||D_k^i(c, x) - D_k^i(c, G(c))||$$

Feature matching loss stabilises generator training by providing a more accurate metric to mimic real data. An additional benefit of such a loss is in leveraging the mode collapse of the GAN model. Using standard GAN loss, a generator could always transform green patches from the semantic map to its approximation of a forest, for instance, and which could fool the discriminator. Using feature matching loss, however, if the discriminator recognises different features for a field, a garden or pasture from the real sample, then the generator will be trained to match those features and offer more variation.

1.4.4 Results

The results of the scalable model can be seen on figure 1.9. Compared to the pix2pix model, the images are of equivalent quality: details are more visible, like small houses or roads, on the other hand, There's less coherence as a whole in the image, with roads displaying multiple colouration, and the average-looking details leaving the samples short of being photo-realistics.

1.5 Evaluation method

Estimating the performance of a GAN model in a objective, quantitative way, is an ongoing problem which challenges the usual machine learning evaluation

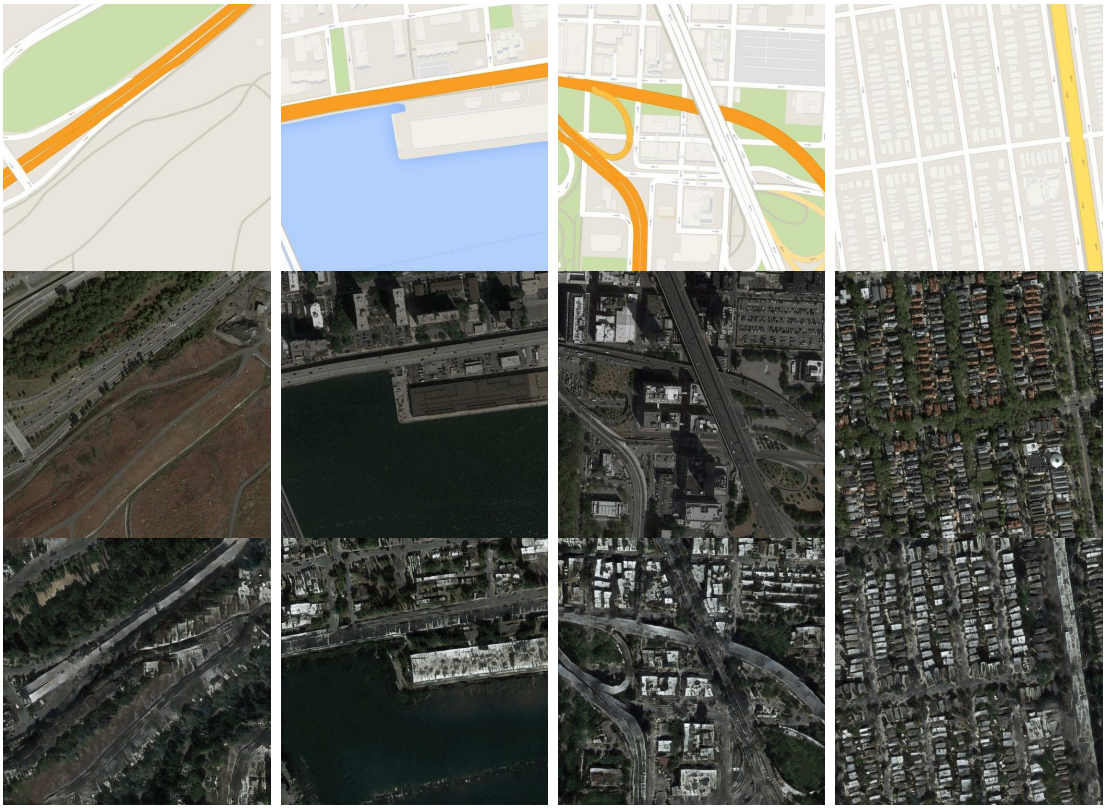

Figure 1.9: **samples generated from the scalable model**. From top to bottom: input image, real satellite image, generated image, respectively. (512 by 512 pixels)

methods. The likelihood estimation used for generative models evaluation becomes a highly challenging approximation to comute with high-dimension data, and does not always reflect the quality of the model, as it has been found clashing with qualitative evaluation[16][11]. Alternative evaluation methods have then been developed as GAN models kept being used and improved for a variety of problems.

1.5.1 Scoring metric

The performance trade off of generative models is between the *quality* and **diversity** of the generated samples. The ideal generative model should provide samples that are indistinguishable from the real data set, while being different enough from it to actually provide value.

The inception score (IS)[22] takes both quality and diversity into account by considering two ideas :

- *The conditional probability distribution of labels on samples should have low entropy.* This part measure the quality of the sample, since the low entropy of the label distribution means that the sample is easily recognisable as a part of the initial data distribution.
- *The marginal distribution of label should have high entropy.*

The IS can then be expressed as:

$$IS(G) = exp(\mathrm{E}_{\mathrm{x}\sim\mathrm{G}}[d_{KL}(p(y|x), p(y)])$$

with d_{KL} being the Kullback–Leibler divergence, measuring a information difference between two probability distribution. The probabilities themselves are estimated from a pre-trained discriminator on the ImageNet data set.
This score, however, does not allow to measure performance of generators on the satellite data set. An another shortcoming is the limitation of diversity guarantees: a generator always producing the same sample for each label will still produce high inception score.

The Fréchet Inception Distance (FID)[17], improve some shortcomings of the IS. Instead of relying on some predicted label, the last few prediction layers are removed, resulting in a model embedding the sample into a feature vector. Under the hypothesis that this embedded layer is a multivariate Gaussian distribution, a Fréchet distance is computed between the Gaussian estimated from the real samples and the generated samples following

$$FID(g, x) = ||\mu_x - \mu_g||^2 + Tr(\Sigma_g + \Sigma_x - 2(\Sigma_x\Sigma_g))$$

Where g, x represent the generator, and real estimated distributions, respectively, and $Tr(A)$ is the trace of the matrix A.

To use this metric as a performance evaluation in satellite image generation, a discriminator must be trained. The lack of images classes can be overcome by training a model to recognise satellite images from others, just like GAN discriminators. But the GAN discriminators can't be used, as the FID must be computed impartially for all models, an exterior discriminator must be used. In addition, to ensure a better diversity evaluation, the feature embedding layer used should be quite large. For instance, the original FID on the ImageNet data set used the last pooling layer of the inception-v3[7] network, thus using a 2048-dimensional encoding. On the other hand, to distinguish between only two classes, a smaller network can be used for computation efficiency.
Thus, a discriminator model is trained to classify between satellite images and

scenery images taken from the ADE20K data set[6], and the model embedding is made with 512 dimensions. This data set has the advantage of providing high-resolution images of outdoor scenery, which share similarities with the satellite data set. The aim is to provide a rather challenging discrimination challenge to force the network to learn distinctive features of satellite images. In the end though, this model will serve only to score between different fake satellite images, and the inherent performance of the pre-trained model will primarily affect the absolute value of the metric instead of the difference between models.

The FID score used to assess the model performance requires a pre-trained discriminator model, this model itself also needs performance assessment to assure the meaningfulness of the FID score. The FID score itself needs a real data set from which to compute the distance. Finally, the generated samples must use semantic maps unseen from the train set. To avoid a bias in the scoring metric, the original test set as been split in three parts : 160 samples for the discriminator training, 160 for its testing, leaving 618 samples, with 309 semantic maps from which to actually generate fake samples, and 309 real images to compute de FID.
The discriminator train and test set also contain 160 sample each taken from the ADE20K dataset[6] to form the other class, to discriminate satellite images from. Such arrangement gives datasets of more than 300 samples for each process of the performance assessment.

The pre-trained model achieve a F1 score of 0.967742 on its test set, this score shows it is sufficiently accurate at recognizing real satellite images. Now, to comprehensively assess the quality FID computation using this model, known data distributions can be used. With *test* being a reserved part of the test set for the distance computaion, *train* being the train set of the generative models, and ADE being the 160 test images from the ADE20K data set:

$FID(x_{test}, x_{test})$	0.05275957620686425
$FID(x_{test}, x_{train})$	3.8890034692103193
$FID(x_{test}, x_{ADE})$	1106.850290455613

The distance between the satellite test set and itself should be exactly zero. However, the estimation of the covariances of large amount of data suffer from numerical error. In theory, the FID between test and train set should also approach zero, but the sample estimations of means and covariances means the estimated probability distributions are different. Finally, the distance computed between the satellite test set and the ADE20K test set is greater by three order of magnitude,

setting a FID scale for this pre-trained model.

To evaluate the performance of different models, some common modus operandi must be defined to allow fair comparison. Each model will be trained for a fixed amount of iterations: 200 iterations of the whole training set. The models are trained using the Google colaboratory shared GPUs system, whose performance varies with GPU availability, hence the comparison can't be done based on computation time. Smaller models should be able to learn faster than larger models, eventually showing better performance than larger models, but without considering the computation time being the limiting factor of learning, the amount of learning steps is the fairest comparison method. The pix2pix model is thus trained over 200 epochs, while the scalable model is trained on 100, 50 and 50 epoch for each of the three training steps.
Then, from each mask of the generator's test set, two satellites images are generated for the FID estimation to better capture diversity.

Using $\lambda * L_{L1} = 50$ for the pix2pix model, and $\lambda * L_{FM} = 50$ for the scalable model:

model	FID	#parameters
pix2pix	87.31940464333985	67.101 M
scalable-model	89.26761868832400	52.959 M

The FID score shows both the pix2pix model and the higher-resolution scalable model produces results of equivalent qualities. However, the scalable model based on residual block is not only scalable, but is also 15 million parameters short of the former. It is also worth noting that each parameter of the scalable model has less gradient descent steps compared to the full model, due to the multi-step training.

Using a fixed U-net generator architecture and patch-discriminators taken from the pix2pix model, The influence of the different training parameters can be evaluated.

multi-scale discriminators	λ_{L1}	λ_{FMLoss}	FID
No	0	0	71.42698421275169
No	20	0	79.42698421275169
Yes	0	0	94.46347164288372
Yes	0	20	51.15582242137991

The first observation can be seen from the two first row: using the $L1$ loss

seems reduces the quality of the generator. From the complete pix2pix models with a $\lambda * L_{L1}$ of 50, to the absence of L1 loss usage, the FID score evaluation goes down. The FID discriminator was trained to recognise satellite from scenery, and one very distinct feature between the two is the presence of sharp edges and straight lines in the satellite images, due to buildings and roads. As the L1 loss as a tendency of smoothing and blurring images, it can be imagined that such a Loss would under-perform with this peculiar FID discriminator.
Multi-scale discriminators alone seems to worsen performance as well, presumably for similar reasons: the added discriminator works with a downsampled version of the samples, as such, a generator bypassing sharper edges can more easily be recognised as real. The semantic map conditioning also helps at generating realistic-looking samples from a larger perspective, making the additional discriminator redundant and mitigating its beneficial effects.
The feature matching loss, however, helps the models outperform models with classical GAN and/or L1 Losses. This is not surprising considering the fact that the FID score is computed based on learned feature recognition. A model trained to match the features of a discriminator, albeit a smaller one, is bound to produce samples which are in turn more rich in recognized features.

Samples produced by the best tested model, that is, the U-net with multi-scale discriminators and $\lambda_{FMLoss} = 20$, can be seen with figure 1.10. At a qualitative level, these results seems of equivalent quality than the residual block-based scalable model. When compared to the samples produced by the worst FID model (the U-net only with the multi-scale discriminators) shown at figure 1.11, the fairness of FID-based assessment is confirmed : the model with FID = 51 produces slightly better results than the model with FID = 94 (sharper edges, more distinct road lanes, even some "cars" on the roads) while both fail at being photo-realistic and blend with the reals image distributions, which hold near zero FID values.

Figure 1.10: **samples generated from the model with the best FID**. (512 by 512 pixels)

Figure 1.11: **samples generated from the model with the worst FID**. (512 by 512 pixels)

1.5.2 Data augmentation

Another way to assess the performance of the different models is to use them for data augmentation. Firstly, larger data set prevent overfitting, since it contains more information, and a fixed-size network won't be able to memorise more data. Also, a larger data set allows for a better estimation of the true data distribution it supposed to represent. Hence, a generator model which successfully captured said data distribution should generate samples improving the performance of other machine learning problems,.

To test this data augmentation method, the Highway detection problem will be used. On maps, highways are easily recognisable by the bright orange color code. However, recognising highways on the satellite image itself is challenging, since there's no real visual difference between a wide road, and a narrow highway. The training data is also imbalanced: out of the 1096 training samples, only 301 contain a highway. To test the effect of the imbalance, the F_1 score and confusion matrices will be used on a test set of 371 samples, to show the impact of false positive (FP) rates and false negative (FN) rates.

Once again, performance comparison calls for equivalent amount of training; Since the augmented data sets are larger, the training limitation comes from the number of samples used for training. In the non-augmented context, if the model is trained over 40 epoch of 602 samples, then the augmented data set model should be trained over 40*602 samples.
The discrimination model used is a simple fully convolutional network. Different methods can be used to train the network to mitigate the imbalanced data set.

- oversampling the highway class gives a F_1 score of 0.208669
- down sampling the non-highway class gives a F_1 of 0.241546 (this is equivalent to using an original balanced data set)

- train 3 networks on random partitions and average the predictions, gives a F_1 of 0.293839. Denoted as the bagging models (**b**ootstrap **agg**regat**ing**) from now on

(a) Confusion matrix of the bagging models

	true	false
predicted true	31	10
predicted false	139	191

The Confusion matrix of the bagging method to balance classes**??** shows the difficulty of highway recognition task: despite a accuracy of 60%, it shows a FN rate of 82%. Some applications may consider such a result to be satisfactory, since false positive and false negative may not imply the same cost, but in this balanced setting, it's a poor performance. The data augmentation experiment consists of seeing if improving these error rates is possible using the different GAN models.

The GAN based data augmentation will be tested with the generative models trained on the full highway detection train set, and generating new samples from semantic maps taken from the same train set. The confusion matrices are shown on figure **??**. The performance of each model trained with different augmented

(a) Confusion matrix with scalable model based data augmentation

	true	false
predicted true	151	179
predicted false	19	22

(b) Confusion matrix with pix2pix model based data augmentation

	true	false
predicted true	131	161
predicted false	39	40

(c) Confusion matrix with best FID model based data augmentation

	true	false
predicted true	41	20
predicted false	129	181

data does not differ much from the model trained on the original data. Depending on the cost of false positive/negative, the scalable and pix2pix based models could even be considered to perform worse. However, the model trained from the model evaluated with the best FID score (see figure **??**), has a similar accuracy of 60%, while also having a more balanced FN/FP ratio. Once again, this shows how this model is the closest model at reproducing real satellite data .

Performing the data augmentation experiment on a lower resolution with the best FID model and discriminator scaled down gives better results, as shown on figure **??**. While the original models with bagging give out similar performance in the lower and higher resolution context with its 61% accuracy, the data augmented model shows a better total accuracy of 64%, while also providing a more balanced rate of false positives and false negatives.

(a) Confusion matrix of the bagging models(256 by 256 pixels)

	true	false
predicted true	33	7
predicted false	137	194

(b) Confusion matrix with best FID model based data augmentation(256 by 256 pixels)

	true	false
predicted true	92	51
predicted false	78	148

1.6 Dataset

The satellite data set used is a collection of Google maps and google earth paired together to be aligned. It can be found at `http://efrosgans.eecs.berkeley.edu/pix2pix/datasets/maps.tar.gz`.

The images from the data set are of 600 by 600 pixel resolution. Throughout this work, these images were reduced to 512 by 512 images using random cropping. In addition, random horizontal flipping was applied, as a classic data augmentation method. Those affine transformation have been used for efficient data augmentation for a large amount of application[4][23]. However, those are limited in data diversity, and are not always applicable. For instance, since the google maps data set contains a 3D effect for some buildings, a vertical flipping would result in some reversed buildings. The choice of cropping to 512-dimensional data and any power of 2 based resolution allows for simpler model architectures using 2-strided convolution layers.

1.7 Implementation

The implementation of this project was done in python, using the pytorch open-source machine learning library. The implementation of this work can be found in its github repository at `https://github.com/lodykas/SatelliteImageGeneration`

under the BSD-3 open source license, and it was build upon an implementation of the basic pix2pix model, which can be found here `https://github.com/junyanz/pytorch-CycleGAN-and-pix2pix`.

At the root of the pix2pix folder are the main scripts of the project:

- **train_discriminator** and **test_discriminator** are the two scripts used for training and evaluating the discriminators used in for the evaluation of the generative models
- **inception_score** is the script computing the Fréchet Inception Distance over 2 datasets given a discriminator.
- **train** and **test** are the scripts used for training and testing the generative models.

The remaining folders contains the modules used by the main scripts:

- **datasets** contains the actual data set used throughout the project
- **data** defines Data sets data structures used by both generator and discriminator models
- **models** defines the generative models; Specifically the networks file defines all networks architectures used, as well as the novel losses like the GAN loss and FM loss, and the pix2pix_model file uses this to build the GAN model (i.e. defines the GAN training sequence, or loads the embedded generators).
- **options** defines command-line argument parsers.

1.8 Conclusion

This work builds upon previous approaches to improve image generation in the context of satellite image generation. A high-resolution model was successfully been developed and produces satisfactory results. However, the model still has room for improvement, as the original objective of data augmentation only performs well at lower resolutions. Still, using generative models have been shown to be able to help improve satellite imagery-based models, albeit at lower resolution.
Further work could improve on the generator architectures to help achieve even more realistic virtual images, for instance with full U-net embedded generators. Other improvements could come from the data set, to try and generate high definition samples, or using another learning problem for data augmentation, to evaluate the model's ability to be conditioned with different labels.

Bibliography

[1] A. Storkey A. Antoniou and H. Edwards. Data augmentation generative adversarial networks, 2017.

[2] H. Larochelle A. B. L. Larsen, S. K. Sønderby and O. Winther. Autoencoding beyond pixels using a learned similarity metric. 2015.

[3] T. Brox A. Dosovitskiy, J. T. Springenberg. Learning to generate chairs with convolutional neural networks. 2014.

[4] I. Sutskever A. Krizhevsky and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *In Advances in neural information processing systems*, 2012.

[5] M. Arjovsky and L. Bottou. Towards principled methods for training generative adversarial networks. *In ICLR*, 2017.

[6] X. Puig S. Fidler A. Barriuso B. Zhou, H. Zhao and A. Torralba. Scene parsing through ade20k dataset. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[7] S. Ioffe J. Shlens C. Szegedy, V. Vanhoucke and Z. Wojna. Rethinking the inception architecture for computer vision. *in CVPR*, 2016.

[8] J. Donahue T. Darrell D. Pathak, P. Krahenbuhl and A. A. Efros. Context encoders: Feature learning by inpainting. *In CVPR*, 2016.

[9] A. Krizhevsky I. Sutskever G. E. Hinton, N. Srivastava and R. R. Salakhutdinov. Improving neural networks by preventingco-adaptation of feature detectors. *arXiv:1207.05*, 2012b.

[10] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *In CVPR*, 2016.

[11] F. Huszár. How (not) to train your generative model: Scheduled sampling, likelihood, adversary? 2015.

[12] M. Mirza B. Xu D. Warde-Farley S. Ozair A. Courville I. Goodfellow, J. Pouget-Abadie and Y. Bengio. Generative Adversarial Nets. *NIPS*, 2014.

[13] T. Darrell J. Long, E. Shelhamer. Fully convolutional networks for semantic segmentation. *In CVPR*, 2015.

[14] P. Isola J.-Y. Zhu, T. Park and A.A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, 2017.

[15] A. Erdem L. Karacan, Z. Akata and E. Erdem. Learning to Generate Images of Outdoor Scenes from Attributes and Semantic Layouts. 2016.

[16] A. van den Oord L. Theis and M. Bethge. A note on the evaluation of generative models. 2015.

[17] T. Unterthiner B. Nessler S. Hochreiter M. Heusel, H. Ramsauer. Gans trained by a two time-scale update rule converge to a local nash equilibrium. 2017.

[18] T. Brox O. Ronneberger, P. Fischer. U-net: Convolutional networks for biomedical image segmentation. *InMIC-CAI*, 2015.

[19] T. Zhou A. A. Efros P. Isola, J.-Y. Zhu. Image-to-image translation with conditional adversarial networks. *CVPR*, 2017.

[20] C. Szegedy S. Ioffe. Batch normalization: Accelerating deep network training by reducing internal covariate shift. 2015.

[21] J.-Y. Zhu A. Tao-J. Kautz B. Catanzaro T.-C. Wang1, M.-Y. Liu. High-resolution image synthesis and semantic manipulation with conditional gans. *in CVPR*, 2017.

[22] W. Zaremba V. Cheung A. Radford T. Salimans, I. Goodfellow and X. Chen. Improved techniques for training gans. *In CVPR*, 2016.

[23] F. H. K. S. Tanaka and C. Aranha. Synthetic data augmentation using gan for improved liver lesion classification. *ACML*, 2018.

[24] S.K. Zhou B. Georgescu X. Liu Z. Shen, Y. Chen and T.S. Huang. Towards Learning a Self-inverse Network for Bidirectional Image-to-image Translation. *arXiv preprint arXiv:1909.94194v2*, September 2019.

UNIVERSITÉ CATHOLIQUE DE LOUVAIN
École polytechnique de Louvain
Rue Archimède, 1 bte L6.11.01, 1348 Louvain-la-Neuve, Belgique | **www.uclouvain.be/epl**