Advanced Machine Learning

Specialized SR-GAN

Alessio Sampieri 1765522
Michele Meo 1599032
Data Science, "Sapienza" University of Rome

Lorenzo Ceccomancini 1667798
Paolo Mandica 1898788
Due Date: January 7, 2021

Abstract

The super-resolution image restoration problem is central in the computer vision field and it can have a lot of applications or utilities. Convolutional neural networks are widely used in the computer vision area and, recently, the SR-GAN architecture has been proposed, in the Generative Adversarial Network for image Super Resolution paper by Ledig et al. [**1**]. This NN architecture is able to produce the super resolution version of a photo-realistic image downscaled by a 4x factor.
In this work we tried to understand this architecture and use it to reconstruct the high-definition version of a specific class of images, in our case landscape images: in particular, through the transfer learning technique, we tried to specialize the pre-trained model, proposed in this github repository from the official paper [**2**], in order to be able to generate high-resolution photos of a specific landscape class. We used the DIV2K dataset for the training of a generic SR-GAN, which is also the same dataset used for the pre-training phase, and a Landscape dataset for the model specialization.
The choice of a loss that captures differences among texture details is fundamental, the standard choice falls on the mse-loss which, in the model used, is incorporated into a perceptual loss function [**7**].
Our Specialized-SR-GAN [**3**] is able to obtain better results, so lower loss discrepancies from the original high resolution set of images, on almost all the landscape classes we have worked on and we will show this in a matrix specialized models-classes comparison.

Keywords: Super-Resolution, Image Restoration, GAN, DIV2K

Data and Model

Datasets

To train the models, described in the next section, we used two different datasets:

- **DIV2K dataset**: 900 RGB high resolution images, belonging to different classes and scenarios, subdivided in TRAIN (800 images) and TEST (100 images) [**4**].
- **Ladscapes dataset**: 6 landscapes classes containing at least 100 RGB images each. A sample of 50 images has been taken from the generic class of images to obtain a TEST set [**5**].

Model

The model selected for the resolution of our task is a Super Resolution Generative Adversarial Network (SRGAN). As described in detail in the paper by Ian Goodfellow et al. [**6**], a Generative Adversarial Network is a framework which allows to train, simultaneously, two models: a generator G and a discriminator D. G's main goal is to learn the data distribution, while D's goal is to estimate the probability that a sample came from the training data rather than G. The final goal is to obtain a generator whose output samples cannot be correctly identified by the discriminator.
In the specific case of a SRGAN, the generator G takes in input low-resolution images and learns to produce in output up-scaled (4x factor) high resolution images. During the adversarial training, the discriminator D estimates the probability of each sample to come from the training data rather than G.
To achieve the goal, different types of loss functions are used in training, which will be discussed in detail in the next section.

The SRGAN generator is composed by:

- B residual blocks, each made up of two convolutional layers with 64 feature maps and $k \times k$ filters. Each convolutional layer is followed by batch-normalization and ReLU activation function. The resolution of the image is increased in the block using sub-pixel convolutional layers.
- A skip connection between each consecutive residual block and one between the first convolutional layer of the network and the last residual block.
- A variable number of convolutional layers, follower by ReLU activation, after the residual blocks.
- A convolutional output layer with *tanh* activation function.

The SRGAN discriminator is composed by:

- Eight convolutional layers with an increasing number of 3×3 filter kernels and an increasing number of kernels, from 64 to 512. The activation function used is a LeakyReLU, which follows a batch-normalization layer for each convolutional layer.
- Three strided convolution layers with batch-normalization and doubling number of kernels are used to reduce the image resolution.
- A final dense layer to obtain a probability for the image classification.

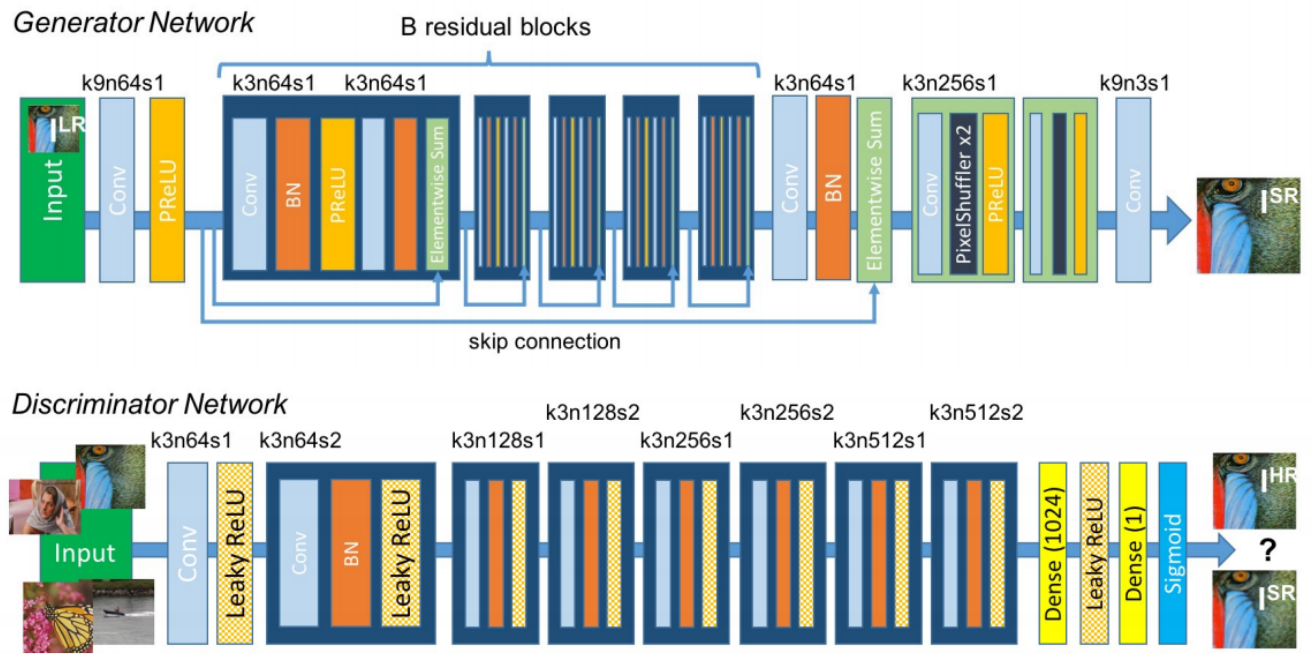

Figure 1: SRGAN generator and discriminator architectures.

It's important to specify that the discriminator model is not initialized with random weights, but it inherits them from the VGG16 pre-trained model. This is possible because the discriminator D has the same structure of the VGG16.

Loss functions

The loss function used in the model is a *perceptual loss function*, that is defined as a weighted sum of a content loss and an adversarial loss components:

$$l^{SR} = {l^{SR}}_{content} + {l^{SR}}_{adversarial} \tag{1}$$

where the content loss ${l^{SR}}_{content}$ is a sum of the MSE loss and the VGG loss ${l^{SR}}_{content} = l^{SR}_{MSE} + 2 \cdot 10^{-6} \cdot l^{SR}_{VGG}$; while the adversarial component is the generative loss based on the loss of the discriminator ${l^{SR}}_{adversarial} = 10^{-3} \cdot {l^{SR}}_{Gen}$, where it is defined on the probability that the reconstructed image is a natural high-resolution image. The factors in front of the components are needed in order to make the various types of losses comparable with the MSE loss [**1**].

Data Augmentation

In order to have an higher variety of images, data augmentation has been performed on the training data. In particular, the following augmentations have been used:

- Random flip;
- Random brightness variation (max 0.4);
- Random contrast variation (from 0.1 to 0.4)

However, the method that resulted in the best improvement for details prediction was the *random crop* applied to the original image to obtain a smaller portion of pixels (384x384). The random crops obtained from all the images in the dataset have been used to train the models.

DIV2K Training

After having adequately built a model suitable for the up-scaling of low resolution images, we moved on to the training of the model on the chosen dataset, in this case the DIV2K.
Precisely, the goal is to minimize the loss of the generator and maximize the one of the discriminator. First of all, we observed that a decent quality for the up-scaled image could be reached after 20 epochs, as the losses were not high and the colors started to be realistic.
Then, the tuning of the model parameters was performed. The parameters tweaked in the Discriminator were filter size and the alpha of the activation function, that is the LeakyReLU. We also tried different numbers of residual blocks in order to obtain the best performance.
Moreover, it was necessary to identify the best layer from which to perform transfer learning of the VGG16 weights to our model; the Pool4 layer resulted as the optimal one.
The best results obtained with our model on the DIV2K dataset are showed in the following plots.

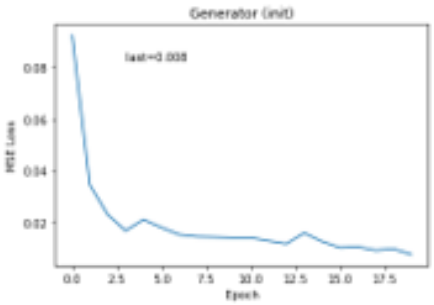

Figure 2: Generator initialization loss history.

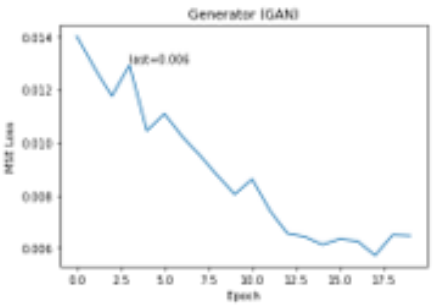

Figure 3: Generator adversarial training loss history.

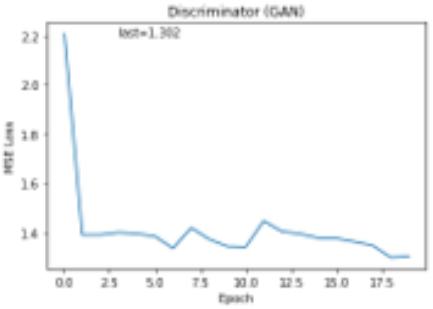

Figure 4: Discriminator adversarial training loss history.

From these plots it is possible to observe the trend of the losses both for the Generator (Fig.3) and for the Discriminator (Fig.4). We can notice that, after 20 epochs, the loss of the discriminator has started to get larger, yet; this means that the generator still has to learn. However, the generated image (Fig.5, left image) does not seem to differ much from the original one (Fig.5, right image).

Looking at the images zoomed-in on a specific point, it is possible to notice how the details of the generated one are not so defined. This result is given by the fact that the model has not been trained for the ages necessary for it to reach an adequate level of detail, and we are unable to do it due to the lack of computational power at our disposal. Therefore, for our specialized model we will use our own structure but a pre-trained model on the same dataset.

Specialized SR-GAN

As highlighted in the previous section, the computational power supplied by Google Colab, or by our machines, is not enough to train the model for a sufficient number of epochs; so, we used a the pre-trained model from the official github

Figure 5: Generated image v. HR Original image

repository [**2**], on which we performed transfer learning. At this point, out final goal was to verify if the SR-GAN model could be specialized on a certain class of images, in order to be more precise in predicting images of the same type. To specialize the mode we used the Landscape dataset, consisting of variegated images but divided into specific classes. Those considered for the above study include:

- Mountain landscapes;
- Desert landscapes;
- Sea landscapes;
- Beach landscapes;
- Island landscapes;

A model was therefore trained on each aforementioned class, plus a general one where the images taken into consideration were a sample from the whole landscapes dataset. Each model was trained on 100 images.

Transfer learning was performed by keeping all the weights of the pre-trained model layers with the exception of the weights of the last one, which have been initialized from a random normal and trained on the chosen dataset. Before coming to the conclusion of keeping just the last layer, we performed multiple tests where we applied transfer learning respectively on the last 1, 3 and 5 layers. The first configuration gave the best results in terms of loss and image quality. This was probably due to the low quantity of images at our disposal and to the low number of epochs of training.

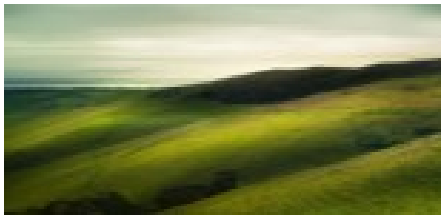

Figure 6: Mountain Landscape

Figure 7: Desert Landscape

Figure 8: Sea Landscape

To evaluate the goodness of the trained models, the average losses of twenty images for each class considered were compared. Specifically, a set of images of each class has been evaluated on each trained model; in this way it was possible to see the behavior of the models on each type of landscape.

Figure 9 shows how the minimum values for each class (highlighted in green) are arranged most of the time on the diagonal. This means that the model corresponding to the specific class gives us the best results for that set of images. There are two exceptions, one concerning the 'Sea' class, for which the General model shows a better performance, and one for the Island class, where the General model and the specialized model have the same loss value.
Looking at the values for each class, it can be seen that models that may have references to the 'mother' class perform better than the others. For example the Beach model and the Sea model have similar values for the Beach class. Same thing for the Desert and Beach models for the Desert class, in fact we can think that the peculiarity of both sets is the sand.

		Models					
		Beach	Mountain	Desert	Sea	Island	General
Classes	Beach	0.015300	0.019600	0.027000	0.016500	0.026300	0.016700
	Mountain	0.018700	0.017500	0.024700	0.020500	0.027300	0.017600
	Desert	0.019500	0.020800	0.018700	0.021300	0.025300	0.019300
	Sea	0.013400	0.023200	0.024600	0.013300	0.026000	0.012200
	Island	0.018200	0.023200	0.028700	0.019600	0.016700	0.016700
	General	0.024200	0.022800	0.026300	0.017500	0.023700	0.014100

Figure 9: Class/Model Losses Average

It is also interesting to make a comparison of the images generated by comparing the models that performs the worst and the best. In particular, looking at the Figure 9, we can say that the best model is the General model on the Sea class and the worst one is the Desert model predicting images from the Island class (Fig. 10).

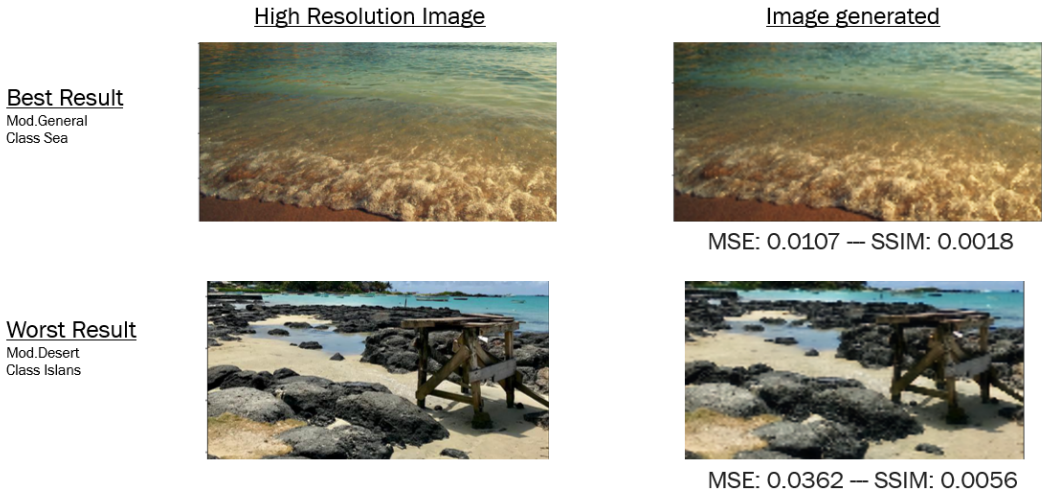

Figure 10: Best/Worst comparison

Finally, it is possible to look at the general behavior of the loss during training, both for the Generator and for the Discriminator. The plots (Fig. 11) show that the best model in terms of loss is the one specialized on Beach class images, while the worst is the one specialized on images of Sea (in fact the best model to increase their resolution is the General one, as per Fig. 9). Looking again at the loss table, it is good to underline how on average the General model performs better on each class of images, this is reflected in its behavior during training.

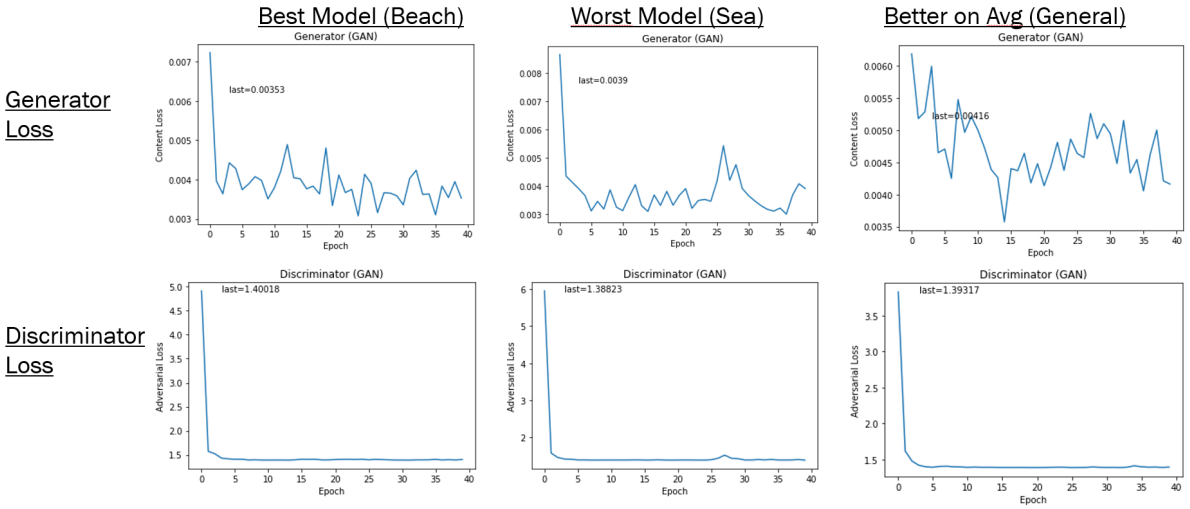

Figure 11: Losses comparison

From figure 11 can be seen how the Generator of the Beach specialized model has a decreasing loss trend and how the Discriminator loss quickly settles around a value, which is the highest among all the models. The Sea model loss stops decreasing after a few epochs and starts to fluctuate from that moment going forward. Regarding the General model, it is evident how other epochs would have been needed as the Generator seemed to be able to learn more with more training.

Conclusions

As we can see from the specialized-models comparison shown in Fig.(9), we managed to obtain better results with respect to the general model on almost all the landscape classes, except for the class *Sea*, that is reasonably confused with the class *Beach*. This could be improved through more advanced pre-processing on the landscapes images collection, for example using an image segmentation technique in order to improve the classification of ambiguous images that could belong to more than one landscape class or just ignoring them.

Another consideration regarding the matrix comparison in Fig.(9) is that the general model is almost always better with respect to the average loss value obtained by a model trained on a specific class and used to rebuild the high-resolution validation set of images from another specific class.

References

[1] Ledig et al. (2017) *Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network* `https://arxiv.org/pdf/1609.04802.pdf`

[2] Official github repository: `https://github.com/tensorlayer/srgan`

[3] Project repository: `https://github.com/paolomandica/AML/tree/master/Final_project`

[4] DIV2K dataset: `https://data.vision.ee.ethz.ch/cvl/DIV2K/`

[5] Landscapes dataset: `https://www.kaggle.com/arnaud58/landscape-pictures`

[6] Ian J. Goodfellow et al. (2014) *Generative Adversarial Networks* `https://arxiv.org/abs/1406.2661`

[7] Zhao et al. (2018) *Loss Functions for Image Restoration with Neural Networks* `https://arxiv.org/pdf/1511.08861.pdf`