Upscaling and increasing resolution of satellite images using Single Image Super-Resolution

Final Project Report

Thomas CHABAL
Ecole des Ponts ParisTech - ENS Paris-Saclay
thomas.chabal@eleves.enpc.fr

Quentin SPINAT
Ecole des Ponts ParisTech - ENS Paris-Saclay
quentin.spinat@eleves.enpc.fr

Abstract

This paper is the project report for the Deep Learning course of the MVA 2020. The purpose of it is to study Single-Image Super-Resolution deep learning algorithms.

Improving image resolution is critical for several applications, ranging from biology to satellite imaging. When this cannot be done by physical means and development of better camera sensors, software takes over and proposes new mathematical methods. The simplest approach to generate a high-resolution image by post-processing is through linear interpolation methods such as the nearest neighbor, bilinear and bicubic interpolations. However, these conventional interpolation methods often produce over-smoothed images with artifacts such as aliasing, blur, and ringing effect (halo around the edges). Super-Resolution is the process of estimating a high-resolution image from a low-resolution input image. Single-Image Super-Resolution is a subset of Super-Resolution methods that focuses on improving the quality of a single image without any additional information about it. Recent Single-Image Super-Resolution methods are example-based methods that learn the relationship between low-resolution and high-resolution image pairs.

In this project, we focused on three recent methods of different complexity: Coupled Deep Autoencoders, Robust U-Net, and Generative Adversarial Networks. We successfully implemented them and evaluated them on a satellite imaging database. Results are visually good, especially for the Robust U-Net. However, quantitative measures do not show significant improvement compared to classical methods such as bilinear and spline interpolations. More work is still necessary to find an appropriate quantitative measure for assessing image quality.

1. Introduction

The generalization of the use of cameras and the hunt for better image quality drove camera makers and computer vision scientists to work towards the improvement of image resolution. When this cannot be done by physical means and development of better camera sensors, software takes over and proposes new mathematical methods.

Improving image resolution is critical for several applications. First, in biology, the observation of microscopic molecules faces obstacles due to physical and optical properties of the matter. Several issues appear, which must be treated with mathematical solutions. In satellite imaging, photos are taken at around 400km from the Earth. At such distances, getting a high resolution of a few dozens centimeters per pixel is very hard, and it thus justifies the use of numerical treatments. Finally, in photography, one may desire to artificially increase the resolution of an image in order to see more details. This may also be the case for image restoration and old images improvement, where the resolution was much lower than today. All these applications have different needs for precision that require the use of various methods : while biology and satellite imagery require to be certain about what is obtained and therefore to have theoretical proves of the correctness of algorithms, daily photography will not care much about some little mistakes or artifacts, as long as the image is visually plausible.

Algorithms that deal with this problem are called Super-Resolution algorithms. The subset of these algorithms that focuses on improving the quality of a single image without any additional information about it is called Single-Image Super-Resolution. We decided to focus on the application of this subset of algorithms to some satellite images.

2. Problem Definition

Given an image I^{LR} of low resolution, the problem of Single-Image Super-Resolution consists in obtaining a higher resolution image $\widehat{I}^{HR}$ upscaled by a factor s, as close

as possible to its target high resolution image I^{HR}. More formally, we are looking for the best function :

$$U : I^{LR} \in \mathbb{R}^w \times \mathbb{R}^h \to \widehat{I}^{HR} \in \mathbb{R}^{w \times s} \times \mathbb{R}^{h \times s}$$

such that $\widehat{I}^{HR}$ and I^{HR} are the closest possible with respect to a distance d. A condition of mathematical optimality :

$$\forall (i,j) \in [\![0;w]\!] \times [\![0;h]\!],\ U(I^{LR})_{s \times i, s \times j} = I^{LR}_{i,j} = I^{HR}_{s \times i, s \times j}$$

Two modeling choices arise from this problem. The first is to properly model the function U. What is its shape ? In what set of functions will we be looking for it ? How to parametrize it ? Research is very active in this field, and several propositions have already been made to answer this question. We make a brief state-of-the art in section 3.

Then comes the question of the metric d to use: How to define that two images are close to each other ? Basic metrics such as the $L1$ and $L2$ distances, may be considered as a first try, but do not necessary take into account the visual impression of the reconstruction. Evaluation metrics are described in section 5.2.

3. Related Work

The question of super-resolution has first been studied by looking for exact methods from a mathematical perspective. In this way, exact interpolations were found out such as bilinear interpolation and spline interpolations, of which the bicubic interpolation is an example. These interpolations satisfy the property of exactness presented above. Yet, they remain approximations of unequal quality of high resolution images on the rest of the pictures, and the higher the order of these interpolations the longer the computation time is. This drove researchers towards the development of other methods, which took advantage of the success of Deep Learning.

With the democratization of neural networks and deep learning, plenty of researchers studied the problem of super-resolution under the prism of specific neural network architectures. Among all solutions that have been proposed by the scientific community, we studied three of them. One solution is to apply autoencoder architectures to solve the problem [1], which finds a representation of the input image in a low dimension and then maps it to a low dimensional representation of the high resolution image. Another solution is to use robust U-Net architecture with degrades training images in order to bring robustness to the network [2]. This structure enables to get refinement of the prediction by looking at features at various scales and receptive fields. Finally, Generative Adversarial Networks [3][4] were also considered to generate details and information where they lack, and therefore create more realistic images. These methods are presented in details in section 4.

Yet, it is important to recall that all these methods are learning-based and are thus biased by the training data, meaning that the pixels they create are only statistical inferences they make with respect to the training dataset. In the framework of our work, our purpose is to generate realistic images. As a consequence, inventing data is not an issue here, even though it may be in more constrained fields such as medical applications.

4. Methodology

4.1. Data Processing

Before going into the details about the models we chose, it is important to explain how we processed the data beforehand. The goal of each model is to produce an upsampled version of its input, with additional details. No matter the model, the processing of the data was the same. We first took the images from our dataset and downsampled them by a factor s to create the low resolution images. Once this was done, we upsampled the image by the same factor s using a bilinear interpolation, which is one of the coarsest smooth upscaling method. This new image was then fed to our models. Additionally to that simple process, we used data augmentation by cropping random portions of original high quality images before downsampling them.

4.2. Exact interpolations

We first evaluated the performances of exact interpolations on the problem of super-resolution. These are spline and bilinear interpolations, which use the values of the low resolution image and weigh them in order to obtain values of other pixels. More precisely :

$$\widehat{I}^{HR}_{k,l} = \sum_{i=1}^{w} \sum_{j=1}^{h} \varphi(k - \frac{i}{s}, l - \frac{j}{s}) I^{LR}_{i,j}$$

with :

$$\varphi_{n-spline} = \bigstar_{i=0}^{n} \mathbf{1}_{[-\frac{1}{2};\frac{1}{2}[^2}$$

and:

$$\varphi_{bilinear} = \mathbf{1}_{[-\frac{1}{2};\frac{1}{2}[^2} \star \mathbf{1}_{[-\frac{1}{2};\frac{1}{2}[^2}$$

Where $\bigstar$ is the convolution product. Yet, the visual results are rather bad. This is explained theoretically: spline interpolations act on the Fourier domain but don't change its size. However, getting higher resolution in images means obtaining details that are associated to very high 2D frequencies. This would require to augment the Fourier domain and fill it with non-zero values, but these values are impossible to get back and extending a Fourier domain is not a simple operation as this may trigger spectral overlap. Thus, we studied other methods to significantly improve the results.

4.3. Autoencoder

4.3.1 General idea

Based on the paper [1], we implemented from scratch a Couple Deep Autoencoder (CDA). The idea of this model is based on a very simple principle : images can be encoded into a low dimension code that contains all its information. As a consequence, no matter the size of the image or its quality, what is important is its low dimensional representation. The CDA consists in (1) finding the representation of low resolution images in one of their low dimensional space, (2) finding the representation of high resolution images in one of their low dimensional space, and (3) mapping low and high resolution representations together. If this is done with an appropriate method, Single Image Super Resolution is done by (4) encoding the low resolution image into its low dimensional space, matching its equivalent code in the high resolution images low dimensional space, and decoding this representation into a high resolution image.

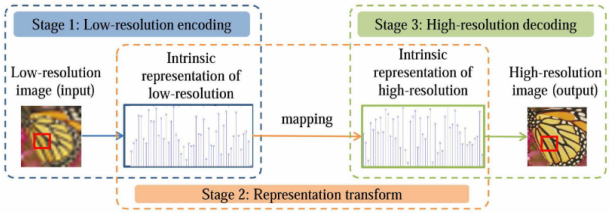

Figure 1. CDA general idea : compute low-dimensional representations of both low and high-resolution images and map one space to the other. Then, perform inference through this created pipeline.

4.3.2 Method

For step (1) and (2), two autoencoders are trained to code and decode image patches of respectively low resolution images and high resolution images. The pixel by pixel size of the patches is fixed at 3 times the scale factor. Then step (3) is simply done by learning a linear mapping between the two encodings previously learnt. Finally, step (4) is done with a network composed of the encoding part of the first autoencoder, the mapping, and the decoding part of the second autoencoder. This network is fine-tuned by being trained on patches of the dataset.

This final network is only able to improve the resolution of image patches of a fixed size. As a consequence, to improve the quality of an entire image, we need to decompose it in patches, improve the quality of all the patches separately, and finally recompose the image thanks to these new patches. Pixels where patches overlap take the mean value of the patches as final value.

The two autoencoders used here are fully connected neural networks with one hidden layer, sigmoid activation, and 2.5 times the dimension of the input (and output) layer in the hidden layer.

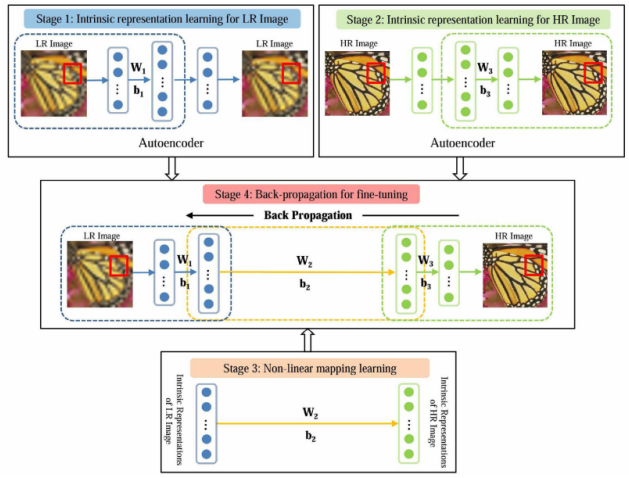

Figure 2. CDA training method : extract patches, compute their higher-dimension equivalents and recompose them together.

4.4. Robust UNet

4.4.1 General Idea

The U-net [5] architecture is well-known for its performances in image segmentation. Like a deep autoencoder, it consists in an encoding path, where the image dimension decreases, a transformation of the low-dimensional latent space and a decoding path where the image dimension is increased to come back to its original shape. However what makes the difference is the presence of skip connections between the decoding and encoding paths, so that the network can use much more prior information from its input when refining its output. Hu et al.[2] proposed a modified version of the U-Net for Single-Image Super-Resolution that they called Robust U-Net. In the Robust U-Net encoding path, classical convolutional blocks are replaced by residual convolutional blocks. This allows the network to learn more complex structures while preventing the gradients from vanishing. As skip connections bring prior information to the result while decoding the intermediate embedding, it is a way to learn slight modifications of the initial input, which justifies this application of U-Net to Single-Image Super-Resolution.

4.4.2 Structure and training

As shown on Figure 3, the specificity of the Robust U-Net is the use of Residual Blocks on the left encoding part of the network. The training is also specific to this method: in order to make up a training dataset, and starting from a given set of images, we compute for each batch a downscale version of it, then blur each image with a random Gaussian

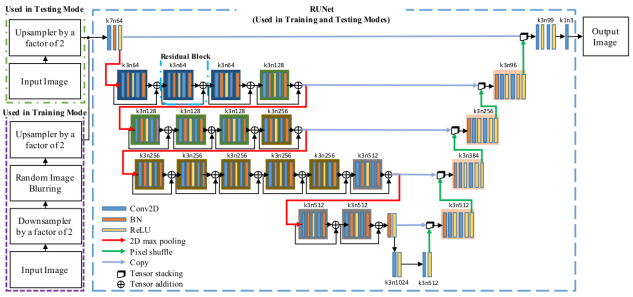

Figure 3. Robust U-Net Structure: Residual blocks replaced traditional convolutional blocks of UNet.

blur, and eventually upscale the whole batch so as to get the same size between the initial images and the transformed ones. During the testing phase, only a usual bilinear upscale is applied to the images, and there is no blurring. We implemented this network and the training procedure from scratch and tested them for Super-Resolution.

4.5. ESRGAN

Some methods explore the use of Generative Adversarial Networks to refine images and improve their resolution, artificially generating some details. This is the case of ESRGAN [4]. This model's generator mainly acts on low-dimensional representations of the input image, applying a succession of convolutions and LeakyReLU, with both dense and residual connections, before upsampling this low-dimension representation and applying two more convolutions. Its architecture is represented on Figure 4.

It then discriminates the generated images with what the authors call a relativistic discriminator, which substracts the mean of a batch of images to each image of the batch before computing the discrimination. The discriminator then learns to differentiate fake and real images comparing them with mean real or fake examples, which may highlight some specificity of either fake or real images.

The model learns from a broader loss than simply perceptual loss : the loss used here is the sum of a VGG-19 perceptual loss like in section 5.2, a generative relativistic loss presented above and a L1 regularization term.

The networ eventually mixes up two images computed with a GAN-based and a PSNR-oriented methods, which has the advantage of reducing or removing noise and artifacts that the GAN frequently produce.

5. Evaluation

5.1. Massachussetts Road Dataset

We downloaded the Massachussetts Road Dataset [6] which is made of aerial views of the Earth, and was created for image detection. The labels for all the data were road detection maps, so we got rid of them and simply used the

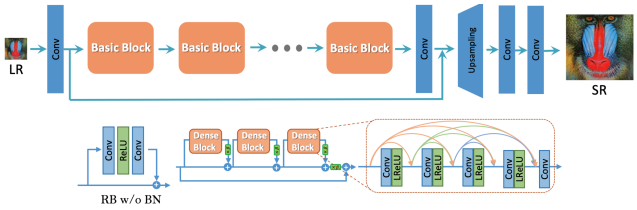

Figure 4. ESRGAN architecture: a first processing of the image is performed with low-dimension representations, then it is upsampled and refined with a few convolutions. The pipeline only uses convolutions and LeakyReLU.

RGB images. Some images contained large white parts corresponding to a lack of data acquired by the satellite camera, and we rejected some of them when the white portion was too important. After selecting good images, we had a training dataset of 807 images, and a test set of 13 images.

5.2. Distance between two images

Basic metrics such as the $L1$ and $L2$ distances, may be considered as a first try. But there exists other classical metrics for evaluating distances between images and assessing image quality. For example, the Peak Signal-to-Noise Ratio (PSNR) links the noise and defects of the image to the details recovered. The greater it is, the more similar images are.

$$PSNR(I_1, I_2) = 10.\log_{10}\left(\frac{Max(I)^2}{MSE(I_1, I_2)}\right)$$

Then, the Structural Similarity Index Measure (SSIM) is also frequently used for such problems : it has a more global overview of the quality of an image while the PSNR tends to consider each pixel independently from the rest. Its value is between -1 and 1, 1 corresponding to similar images.

$$SSIM(I_1, I_2) = \frac{(2\mu_{I_1}\mu_{I_2} + 0.01L)(2Cov(I_1, I_2) + 0.03L)}{(\mu_{I_1}^2 + \mu_{I_2}^2 + 0.01L)(Var(I_1) + Var(I_2) + 0.03L)}$$

Where L is the dynamical range of the pixel values.

Yet, these metrics may not always be adapted to image restoration problems : if they quantify values, they are not able to quantify visual impressions and are often wrong on discriminating the best images. This encourages us to look for other more appropriate metrics like perceptual losses [7] that rely on VGG-16 backbones and extract visual features of various images to compare them with a Mean Square Error loss. More precisely, we consider the MSE score of the VGG-16 feature maps ϕ_i just before its Maxpool operations (hence 5 feature maps). If the feature map i is of size $C_i \times W_i \times H_i$, then :

$$Loss(I_1, I_2) = \sum_{i=1}^{5} \frac{1}{C_i W_i H_i} ||\phi_i(I_1) - \phi(I_2)||_2^2$$

5.3. Training Parameters

5.3.1 CDA

Step one and two of the CDA where conducted with the Adam optimizer with a learning rate of 0.01, a batch size of 256, and 10 epochs over 800 000 patches randomly taken in the train dataset images.

Step three was conducted with the Adam optimizer with a learning rate of 0.001, a batch size of 256, and 10 epochs over the coding of the 800 000 patches by the two previously trained autoencoder.

The criterion used for all the training was the Mean Square Error loss.

5.3.2 RUNET

The RUNet training was conducted with the Adam optimizer with a learning rate of 0.001, a batch size of 32, and 300 epochs over random 128×128 crops the training set images. Random crops were different at each epoch. We used a VGG-16 perceptual loss as the training criterion and a ReduceLROnPlateau learning rate scheduler to better tune the network.

6. Results

In table 1 are quantitative results of our models and classical computer vision interpolations. As you can see, there is no significant quantitative improvement by using Deep Learning methods.

	MSE ($\times 10^{-3}$)	SSIM ($\times 10^{-1}$)	PSNR	Perceptual
Bilinear	3.025	8.621	22.56	6.388
Spline-5	2.925	9.126	24.46	5.547
CDA	2.522	**9.282**	**25.17**	5.152
RUNET	3.486	8.903	22.52	5.931
GAN	**2.033**	9.125	25.03	**4.858**

Table 1. Quantitative results of all the studied methods over the test set. The Coupled Deep Autoencoder performs best with respect to SSIM and PSNR metrics, while the ESRGAN is the best for MSE and Perceptual loss.

However, there is qualitative improvement, as you can see in Figure 5. That means that the quantitative measures we used are not good enough to take into account the visual impression of images, even though we used a perceptual loss. That raises again the problem of finding an appropriate descriptor of images for Super-Resolution.

Despite a small problem with the colors (that can easily be solved with color transfer methods, and explains its bad quantitative scores), the robust U-Net makes visually the best Super-Resolution Image.

7. Conclusion

In this project, we have compared three Deep-Learning methods for Single-Image Super-Resolution. Every one of them had comparable quantitative results, and did not seem to have significantly better results than the classical computer vision algorithms such as bilinear and spline interpolations (see Table 1). However, we can see a great improvement of the visual results (see Figure 5). That raises again the question of finding a relevant measure to evaluate the quality of an image.

Additional work that could be done in the future would be: try a training of the CDA with the perceptual loss, try super-resolution with different upscaling factors (x3 or x4), fine-tune the ESRGAN specifically on our dataset to get better results on satellite imaging and do more research in order to find an appropriate measure to assess image quality.

References

[1] K. Zeng, J. Yu, R. Wang, C. Li, and D. Tao. Coupled deep autoencoder for single image super-resolution. *IEEE Transactions on Cybernetics*, 47(1):27–37, 2017. 2, 3

[2] Xiaodan Hu, Mohamed A. Naiel, Alexander Wong, Mark Lamm, and Paul Fieguth. Runet: A robust unet architecture for image super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2019. 2, 3

[3] Francesco Cardinale et al. Isr. https://github.com/idealo/image-super-resolution, 2018. 2

[4] Xintao Wang, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Yu Qiao, and Chen Change Loy. Esrgan: Enhanced super-resolution generative adversarial networks. In *The European Conference on Computer Vision Workshops (ECCVW)*, September 2018. 2, 4

[5] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *CoRR*, abs/1505.04597, 2015. 3

[6] Volodymyr Mnih. *Machine Learning for Aerial Image Labeling*. PhD thesis, University of Toronto, 2013. 4

[7] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision*, 2016. 4

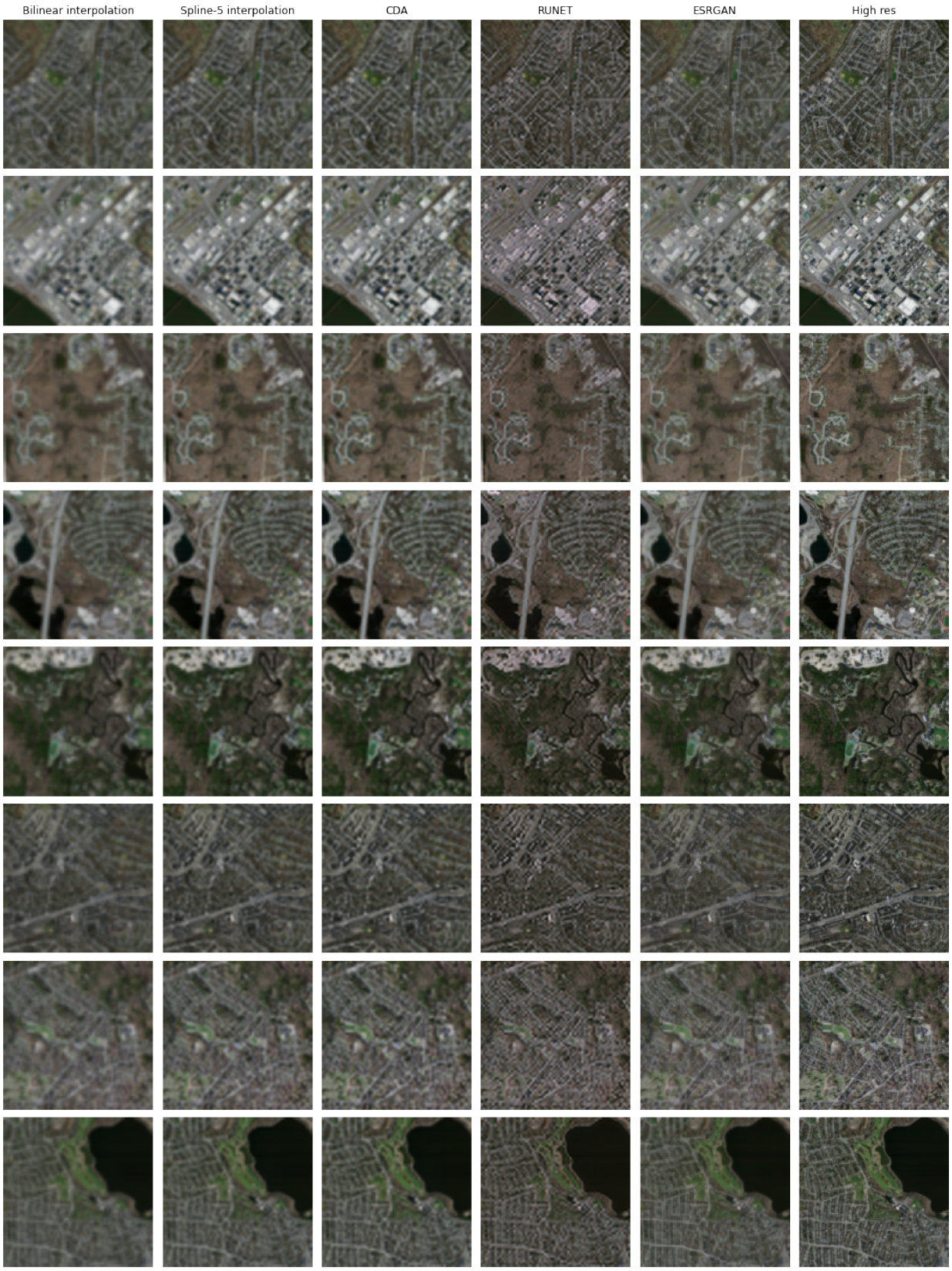

Figure 5. Visual results on the test set for all the methods studied. From left to right: Bilinear interpolation, spline interpolation of order 5, Coupled Deep Autoencoder, Robust UNet, ESRGAN and the high-resolution target.