184.702 Machine Learning

Exercise 3 Report

Group 35: Németh Marcell, Pethő Dominik

Topic selection

Our team picked the topic of image-to-image processing via Deep Learning, more specifically image upscaling. Our decision was based on our interest in image processing and the purpose to explore a task more difficult than image classification.

Dataset

The data we used for the image upscaling task was the div2k dataset, which contains diverse 2K resolution, high quality images which suits our task. The images come in PNG format.

Pipeline

Model architecture

We chose the Super-Resolution Convolutional Neural Network (SRCNN) listed in the task description to perform the upscaling of images. An unofficial PyTorch implementation of the model was listed in the task description and our team decided to use this to solve the task.
The model architecture is quite simple, it consists of 3 two dimensional convolutional layers followed by RELU activation.
Based on the official paper, the first layer is responsible for patch extraction and representation. The first layer extracts overlapping patches from the low-resolution image, and represents each patch as a high-dimensional vector. The second layer performs non-linear mapping, as it maps this high-dimensional vector onto another high-dimensional vector. These mapped vectors represent high-resolution patches. The third layer is responsible for reconstruction, as it aggregates the output patches of the second layer to generate the final high-resolution image.

Beyond the baseline SRCNN code, we implemented an evaluation pipeline tailored to the full DIV2K validation set. The script computes PSNR and SSIM on the Y (luminance) channel, exports per-image metrics to CSV, and produces qualitative montages of best and worst cases. This adds a reproducible, parameterized evaluation layer to the original repository.

Data preprocessing

The data preprocessing pipeline generates training pairs by first cropping high-resolution images to ensure their dimensions are divisible by the scale factor. To create the corresponding low-resolution inputs, the high-resolution images are downsampled and then upscaled back to their original size using bicubic interpolation, ensuring that the input and target images have the same spatial dimensions. Both sets of images are converted from RGB to the Y-channel of the YCbCr color space, which ensures that the model learns structural details rather than color. Finally, the script generates a dataset of overlapping sub-images by extracting 33x33 patches with a stride of 64, which are then stored in an HDF5 file for efficiency.

Training

The training is performed on the 33x33 image patches created by the data preprocessing pipeline. The data is processed in batches of configurable size, we trained the model with the batch size of 16 and set the number of epochs to 50. The Mean Squared Error (MSE) loss function and the Adam optimizer is used during training. A differential learning rate strategy is used to ensure training stability, which means that the first two layers are updated with a configurable base learning rate, and the final reconstruction layer uses a learning rate that is 10 times smaller to ensure stability. For the base learning rate, we used 1e-4.

During training, the model's performance is validated at the end of each epoch using full-resolution images. The training script tracks the average Peak Signal-to-Noise Ratio (PSNR) across this validation set and automatically saves the highest scoring model weights.

All steps are fully scripted and configurable with command-line parameters or a config file, so changing scale or training settings does not require code modifications.

Evaluation

We trained two models with the same settings which were listed above: one with scale 2x and one with scale 4x. This means that the size of high-resolution images were scaled down by 2 and 4 for the two models, respectively.

Quantitative

For quantitative evaluation, we used two metrics.

The first one is Peak Signal-to-Noise Ratio (PSNR). PSNR provides a quantitative measurement of image reconstruction quality by comparing the pixel-wise difference between the upscaled and the ground truth high-resolution image.
The second metric we used is Structural Similarity Index (SSIM). SSIM quantifies the degradation in structural information between the model output and the original high resolution-image.
It is important to mention that both metrics are calculated exclusively on the Y channel of the YCbCR color spaces. As the Y channel represents luminance, this approach focuses on structural similarity rather than color.
We also report the bicubic baseline on the same validation set, so that improvements (ΔPSNR/ΔSSIM) are directly comparable to the learned model.

Qualitative

The output of the original unofficial implementation is two images with five rows, one of them visualizing the five best, the other one the five worst results based on PSNR scores. Originally, each row contains three images: the ground truth, the image after applying (downscaling and) bicubic upscaling, and the output of the super-resolution model. We also decided to add the downscaled image (upscaled with NEAREST method to match the resolutions). As the differences between the images are barely visible by human eye, a difference map between the SRCNN output and the Bicubic baseline was added too, to show exactly where the model changed the pixels. These pixel differences are amplified by 15 times, so that the fine-grained error patterns are visible to the human eye.To make the visual differences more apparent, we additionally include the downscaled input (upsampled with nearest-neighbor) in the montages.

Results

The model performance reported in this section represents the peak validation accuracy, not the performance on a hold-out test set. The validation set was utilized for both monitoring training convergence and for the final quantitative evaluation. Therefore, the results are slightly biased and optimistic.
The table below displays the quantitative results of our experiment:

Scale	Metric	Bicubic	SRCNN	Improvement
x2	PSNR	32.65 dB	34.98 dB	+2.34 dB

x2	SSIM	0.9135	0.9407	+0.0272
x4	PSNR	28.21 dB	29.44 dB	+1.23 dB
x4	SSIM	0.7882	0.8252	0.0370

***Table 1**: Quantitative comparison of Bicubic vs. SRCNN on DIV2K validation (PSNR/SSIM on Y channel), including improvement*

PSNR is measured in decibels, where higher values indicate lower reconstruction error. Our model achieved an improvement of +2.34 dB at scale x2, and +1.23 dB at scale x4. Using the PSNR's formula, the +2.34 dB improvement corresponds to a ~42%, the +1.23 dB improvement corresponds to a ~25% reduction in MSE compared to the bicubic baseline.
Meanwhile at scale 2x, the SSIM score of 0.9135 for the bicubic baseline is already high (1 is the maximum), the SRCNN model achieved the score of 0.9407, which means that its output preserves the vast majority of the original image's high-frequency details. The absolute improvement at scale x4 is larger than at scale x2, as the superscaling model improved the bicubic SSIM score of 0.7882 to 0.8252. It is visible from the results that the bicubic baseline and SRCNN output quality is significantly lower at scale x4 than at scale x2, which was expected and was the main reason to try out the model at two different scales.
The image below displays the five best results of the SRCNN model at scale x2:

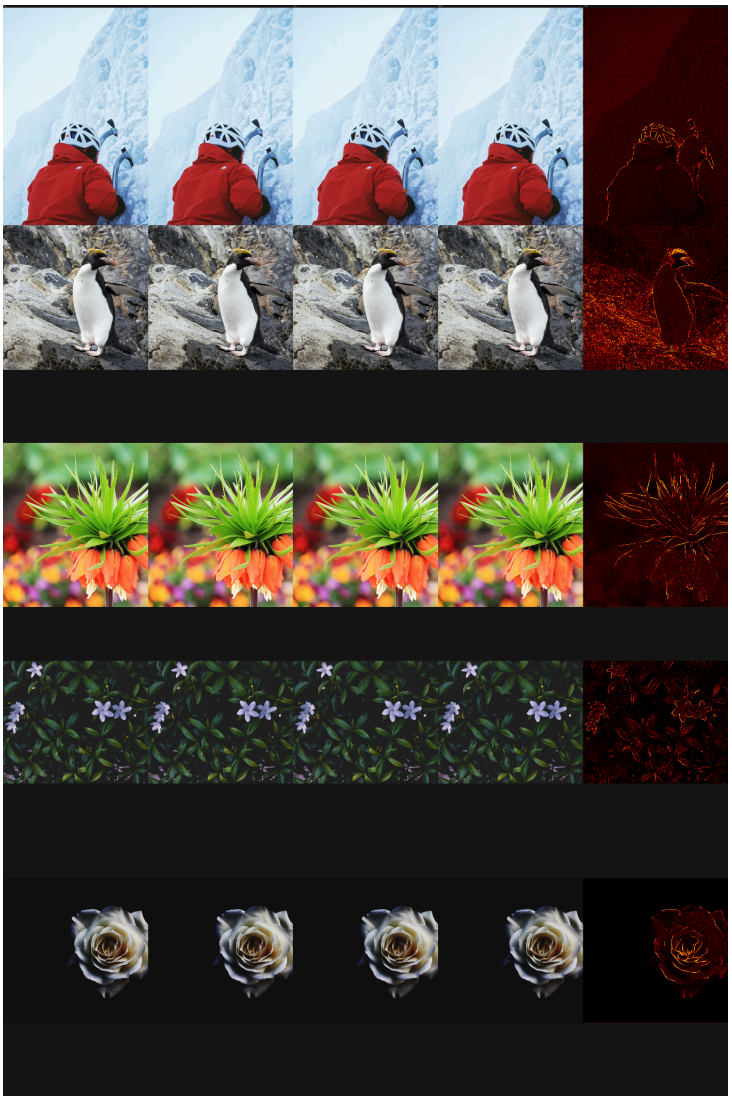

Figure 1: Best SRCNN results on DIV2K val (scale x2). From left to right: ***HR ground truth****,* ***LR*** *(downscaled, nearest-neighbor upsampled for visualization),* ***Bicubic*** *(input),* ***SRCNN output***

The differences on the images are barely visible by human eye, but the difference map reveals the differences between the bicubic images and the model outputs. The black areas represent no change in pixels, the red ones minor and the yellow ones the significant changes. It is visible that the yellow color appears by edges, and as these images don't contain much high-frequency detail, the model was able to perform upscaling well.
In comparison, these are the five worst results of the SRCNN model based on PSNR:

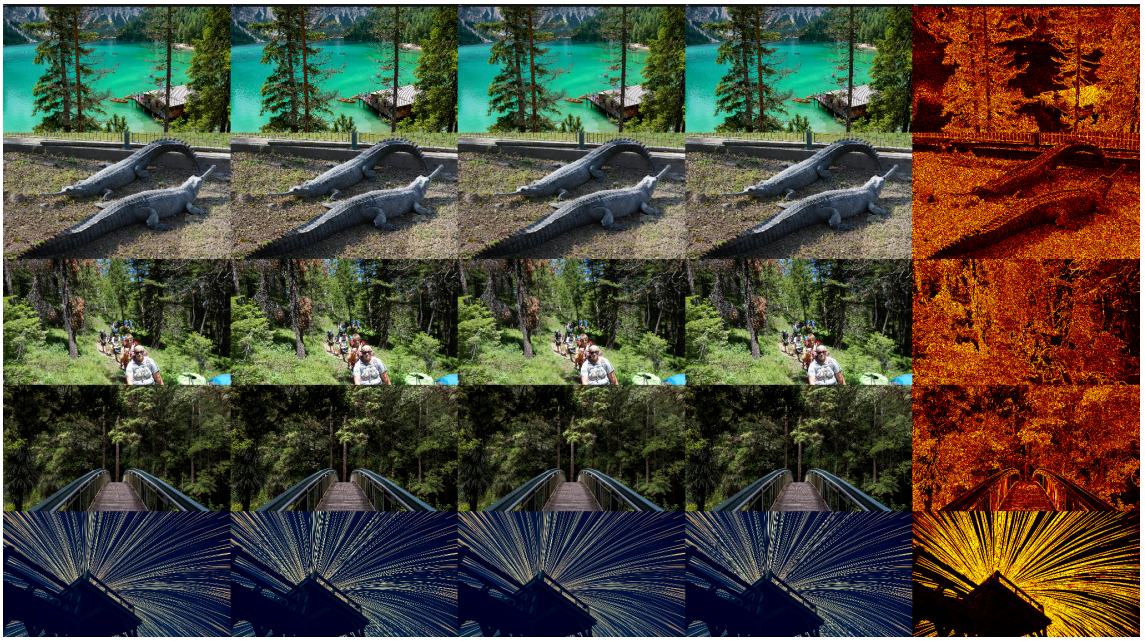

Figure 2: Worst SRCNN results on DIV2K val (scale x2). From left to right: ***HR ground truth****,* ***LR*** *(downscaled, nearest-neighbor upsampled for visualization),* ***Bicubic*** *(input),* ***SRCNN output***

The reason why the model's performance was the poorest on these inputs is clear: the images contain a lot of high-frequency information, for example the leaves of the trees and grass. In these dense textures, a tiny misalignment between the reconstructed edge and the ground trough result in higher error.

The image below displays the best results of the model trained at a x4 scale:

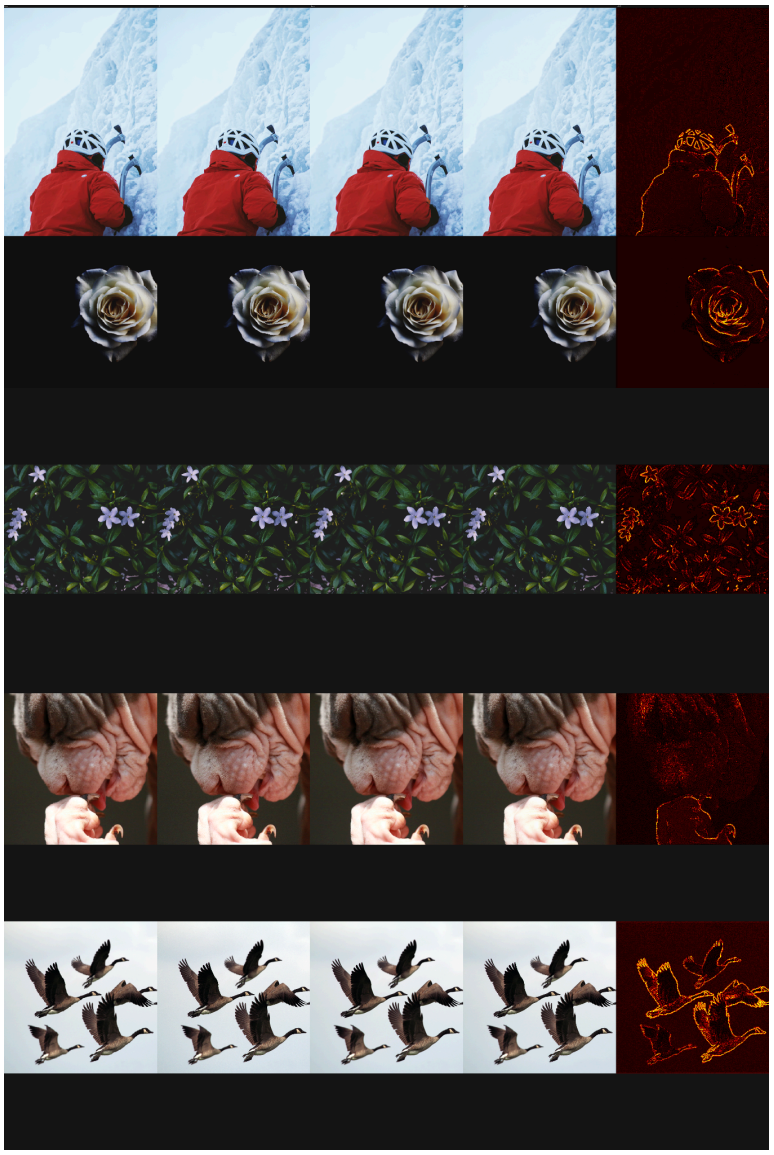

Figure 3: Best SRCNN results on DIV2K val (scale x4). From left to right: ***HR ground truth****,* ***LR*** *(downscaled, nearest-neighbor upsampled for visualization),* ***Bicubic*** *(input),* ***SRCNN output***

3 out of 5 images are the same as for the model trained at scale x4. It is visible that compared to the best results of the other model, the yellow color at the edges is more dominant, which shows that the SRCNN had to restore the blurred edges more aggressively.
These are the worst results of the upscaling model at scale x4:

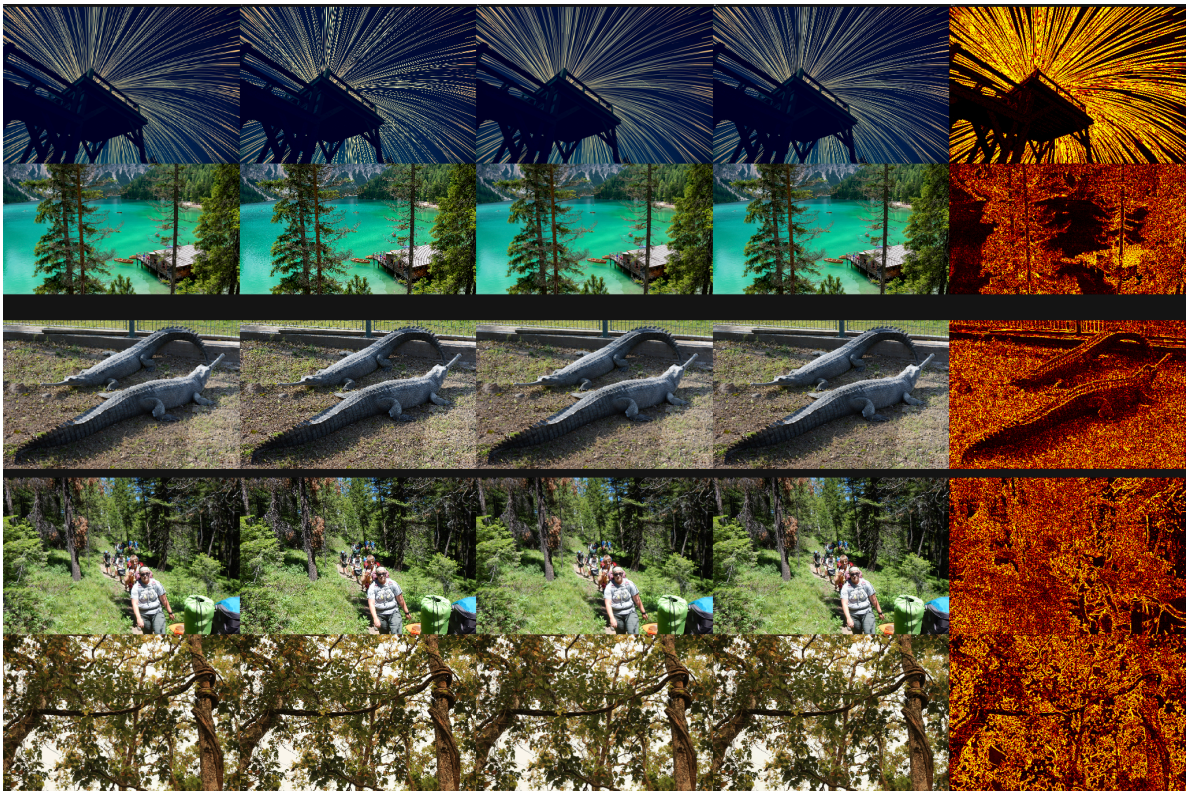

Figure 4: Worst SRCNN results on DIV2K val (scale x4). From left to right: ***HR ground truth****,* ***LR*** *(downscaled, nearest-neighbor upsampled for visualization),* ***Bicubic*** *(input),* ***SRCNN output***

On this output, only one row differs from the worst results of the other model. The same conclusion can be drawn about the high-frequency details as before, and additionally, the yellow areas are also more dominant, than one the previous worst results.