TECHNICAL UNIVERSITY OF LIBEREC
Faculty of Mechatronics, Informatics
and Interdisciplinary Studies

Image super sampling using deep neural networks

Master Thesis

Study programme:	N0714A150003 Mechatronics
Author:	**Ivan Gorbatenko**
Thesis Supervisors:	Ing. Karel Paleček, Ph.D. Institute of Information Technology and Electronics

Liberec 2022

TECHNICAL UNIVERSITY OF LIBEREC
Faculty of Mechatronics, Informatics
and Interdisciplinary Studies

Master Thesis Assignment Form

Image super sampling using deep neural networks

Name and surname:	**Ivan Gorbatenko**
Identification number:	M21000200
Study programme:	N0714A150003 Mechatronics
Assigning department:	Institute of Information Technology and Electronics
Academic year:	**2021/2022**

Rules for Elaboration:

1. Design and implement deep neural network models for image super sampling using PyTorch library and Python programming language.
2. Train the models using different learning criterions, e.g. pixelwise loss, content loss and/or adversarial loss.
3. Empirically evaluate and compare your models on a suitable dataset. Also, if available, compare your results with existing solutions.
4. Describe advantages and disadvantages of neural network based super sampling as compared to classical algorithms.

Scope of Graphic Work:	dle potřeby dokumentace
Scope of Report:	40-50
Thesis Form:	printed/electronic
Thesis Language:	English

List of Specialised Literature:

[1] Goodfellow, I., Bengio, Y., Courville, A. Deep learning. MIT Press, 2016

[2] Bishop, C. Pattern Recognition and Machine Learning. 2006. ISBN 13: 978-038731073

[3] Karpathy, A., Johnson, J., Li, F. Convolutional neural neworks for visual recognition. dostupné online: http://cs231n.stanford.edu

Thesis Supervisors:	Ing. Karel Paleček, Ph.D. Institute of Information Technology and Electronics
Date of Thesis Assignment:	October 12, 2021
Date of Thesis Submission:	May 16, 2022

L.S.

prof. Ing. Zdeněk Plíva, Ph.D.
Dean

prof. Ing. Ondřej Novák, CSc.
head of institute

Liberec October 19, 2021

Declaration

I hereby certify, I, myself, have written my master thesis as an original and primary work using the literature listed below and consulting it with my thesis supervisor and my thesis counsellor.

I acknowledge that my master thesis is fully governed by Act No. 121/2000 Coll., the Copyright Act, in particular Article 60 – School Work.

I acknowledge that the Technical University of Liberec does not infringe my copyrights by using my master thesis for internal purposes of the Technical University of Liberec.

I am aware of my obligation to inform the Technical University of Liberec on having used or granted license to use the results of my master thesis; in such a case the Technical University of Liberec may require reimbursement of the costs incurred for creating the result up to their actual amount.

At the same time, I honestly declare that the text of the printed version of my master thesis is identical with the text of the electronic version uploaded into the IS/STAG.

I acknowledge that the Technical University of Liberec will make my master thesis public in accordance with paragraph 47b of Act No. 111/1998 Coll., on Higher Education Institutions and on Amendment to Other Acts (the Higher Education Act), as amended.

I am aware of the consequences which may under the Higher Education Act result from a breach of this declaration.

May 23, 2022

Ivan Gorbatenko

Image super sampling using deep neural networks

Abstrakt

Tento článek poskytuje přehled současných algoritmů hlubokého učení používaných pro zvýšení rozlišení obrazu, jakož i jejich analýzu a srovnání s ostatními, stejně jako s klasickými algoritmy pro zvýšení rozlišení obrazu.

Klíčová slova: Neuronové sítě, hluboké učení, konvoluce, GAN, super sampling, EDSR, VDSR, SRGAN, ESRGAN

Image super sampling using deep neural networks

Abstract

This paper provides an overview of the current algorithms of deep learning used to increase the resolution of images, as well as their analysis and comparison with others, as well as with classical algorithms for increasing the resolution of images.

Keywords: Neural networks, deep learning, convolution, GAN, super sampling, EDSR, VDSR, SRGAN, ESRGAN

Contents

List of abbreviations

GAN	Generative adversarial network
SR	Super-resolution
MOS	Mean opinion score
SRGAN	Generative adversarial network for image super-resolution
ESRGAN	Enhanced generative adversarial network for image super-resolution
RRDB	Residual-in-residual dense block
FMEN	Fast and memory-efficient network
EISR	Efficient image super-resolution
ERB	Enhanced residual block
HAN	Holistic attention network
LAM	Layer attention module
CSAM	Channel-spatial attention module
PULSE	Photo up-sampling via latent space exploration
VDSR	Very deep convolutional network
RDN	Residual dense network
RDB	Residual dense blocks
SRCNN	Super-resolution convolutional neural network
ILR	Interpolated low resolution image
MSE	Mean square error
PSNR	Peak signal to noise ratio
SSIM	Structure similarity index
RGB	Red, green and blue
SGD	Stochastic gradient descent

Introduction

Super-resolution (SR) image reconstruction is a method for recovering an undamaged high-resolution picture from the same scene. It refers to creating a clear high-resolution image from one of the lowres degradated pictures in that same scene. With help of signal estimation theory, it solves the problem of low image resolution caused by limitations in arrangement density for sensor array due to its limitation on arranged space and makes up for the deficiency of sensor hardware. SR reconstruction has broad application prospects in industrial control, medical imaging, remote sensing, security monitoring and video signal transmission. It is also possible to effectively overcome the influence of blurring factors on image data collection by improving quality throughout the process of digitalization, as well as in other fields. The SR Reconstruction Technology is an international hot topic in the research of image processing, computer vision and applied mathematics. It has great application value and theoretical significance, and has been a global hot issue in the study of image processing, computer vision and applications. More than 30 years after the beginning of research and development, there were some good scientific results on image SR technology. In general, image SR technology is divided into three types: interpolation based method, reconstruction based method, learning based method.

Another big part of image SR algorithms, which have some common parts with learning based methods, is methods that use deep learning techniques have been fairly effective in solving the problem of image and video SR. In this work we will focus on SR technology based on deep learning techniques. In particular deep learning techniques for single-image super sampling.

Based on the below formula, low-resolution image can be modeled from high-resolution image. D is degradation function, Iy is high-resolution image, Ix is low-resolution image, and σ is the noise.

$$I_x = D(I_y; \sigma)$$

D and σ are not known; only high-resolution image and corresponding low-resolution image are known. As a result of the neural network's task, it is possible to find the inverse function of degradation by just HR and LR image data.

There are many approaches used for single-image super sampling:

- Generative Models
- Pre-Upsampling methods

- Post-Upsampling methods
- Residual Networks
- Attention-Based Networks

1 Overview of existing algorithms

This chapter will overview most modern and popular papers and algorithms of single image super sampling.

1.1 Generative Models

1.1.1 SRGAN and ESRGAN

The **SRGAN**[2] – is a generative adversarial network (GAN) for image SR. According to the authors statements, it is the first framework capable of inferring photo-realistic natural images for 4× up scaling factors. In order to achieve this, authors have proposed a perceptual loss function that consists of an adversarial and content loss. Because of the adversarial loss, this solution is in the natural image manifold by using a discriminator network that is trained to differentiate from the SR images and original photo-realistic pictures. At the same time, authors use a content loss caused by perceptual similarity alternatively to similarity in pixel space. The results of the extended mean opinion score (MOS) test show hugely significant gains in perceptual quality using SRGAN. Among the MOS scores obtained with any state-of-the art methods, those obtained with SRGAN are closer to original high resolution images than those obtained with the rest of method.

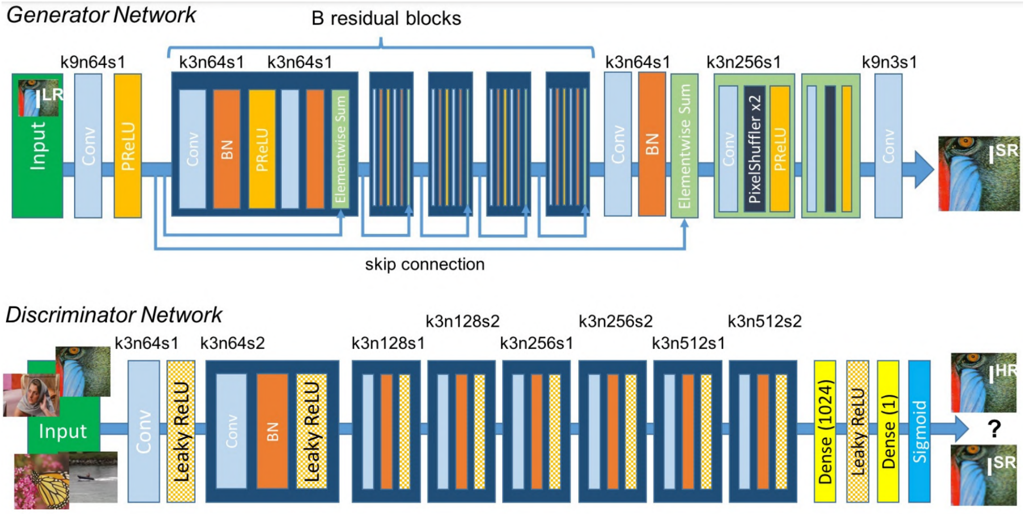


Figure 1.1: Architecture of Generator and Discriminator Network with corresponding kernel size (k), number of feature maps (n) and stride (s) indicated for each convolutional layer

The **ESRGAN**[14] – is Enhanced SRGAN. Authors introduce the Residual-in-Residual Dense Block (RRDB)[5] without batch normalization as the basic network building unit. In addition to this, they borrow an idea from relativistic GAN[7] let the discriminator predict relative realness instead of the absolute value. Finally, a perceptual loss is improved by the use of features before activation, that will provide stronger control for brightness consistency and color retention. The proposed ESRGAN achieves consistently better visual quality with more natural and real textures than SRGAN. This due to these improvements.

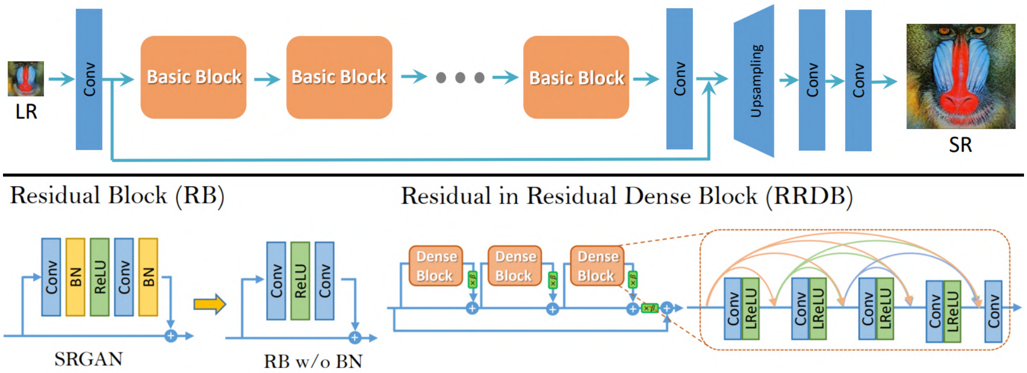


Figure 1.2: Was selected basic architecture of SRResNet [2], where most computation is done in the LR feature space and as basic block was taken RRDB

1.1.2 PULSE

The **PULSE**[10] – is Photo up-sampling via latent space exploration. The author offers other formulas for the super-resolution problem, based on creating realistic SR images that downscale correctly. For the reason that usual approaches lead to blurring, especially in detailed regions. Their algorithm creates high-resolution, realistic images at resolutions previously unseen in the literature up to x64. Other benefit is that the algorithm can be trained without high-resolution images. Instead of starting from the LR image, and slowly adding detail, algorithm traverses the high-resolution natural image manifold, searching for pictures that downscale to an initial LR picture.

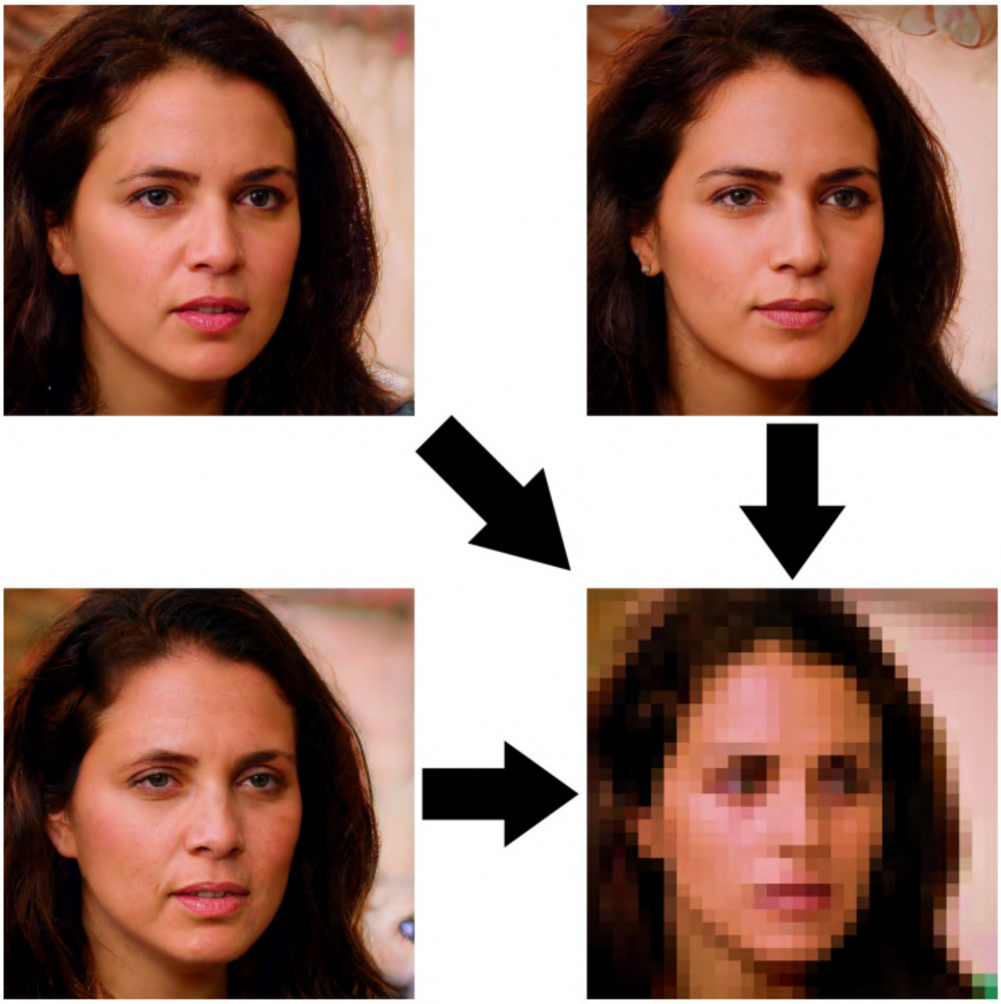


Figure 1.3: Shown how image generated by PULSE corresponds with original LR image

1.2 Convolutional / Residual Networks

1.2.1 EDSR and MDSR

The **EDSR**[9] – is enhanced deep SR network. According to the authors statements, performance of this model exceeded those of current state-of-the-art SR methods on moment of release. This model has a great performance improvement, thanks to the removing of unnecessary modules in conventional residual networks. A further performance is further improved by expanding the model size, as well as stabilizing the training procedure. In addition, Authors propose to create and train a new multi-scale deep SR system (**MDSR**) and training method, which can reconstruct high-resolution images of different up scaling factors in one model.

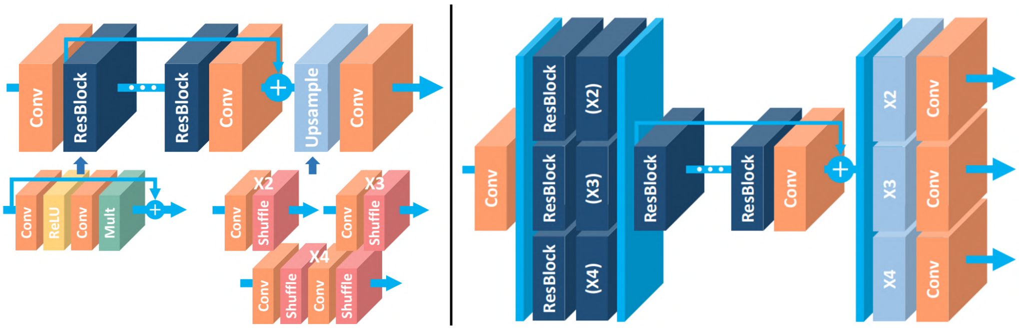


Figure 1.4: architecture of EDSR on the left side and architecture of MDSR on the right

1.2.2 FMEN

The **FMEN**[16] – is fast and memory-efficient network. Authors have created lightweight network backbone, based on the idea that sequential network operations are not frequently accessing preceding states and extra nodes, and consequently it is beneficial to reduce memory consumption and run-time overhead. As well, they stack multiple highly optimized convolutions and activation layers and reduce the use of feature fusion. The authors also offer the new sequential attention branch, where every pixel is assigned an important factor according to local and global contexts. In addition, they tailor the residual block for EISR and propose an enhanced residual block (ERB) to accelerate network inference. In addition, all this advantages allow the author to design a FMEN and its small version FMEN-S, that runs 33% faster and reduces 74% memory consumption compared with the state-of-the-art Efficient image SR (EISR) model.

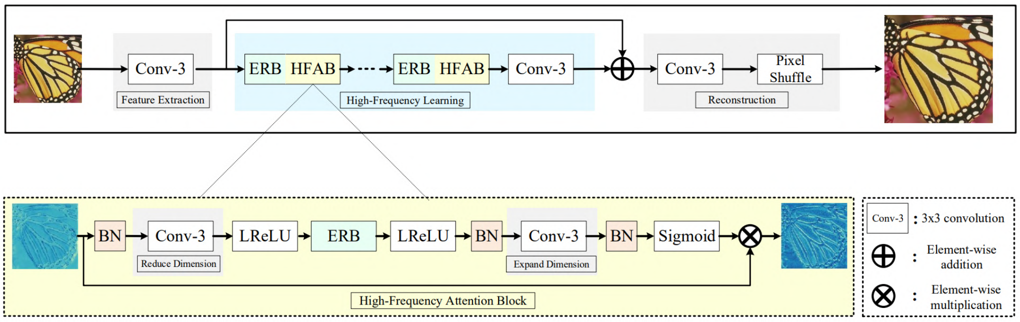


Figure 1.5: The architecture of FMEN

1.2.3 VDSR and RDN

The **VDSR**[8] and **RDN**[15] – are very deep convolutional network and residual dense network. Authors of VDSR algorithm propose model structure with cascade a pair of convolutional and nonlinear layers repeatedly, which gave good and accurate results on publication moment. RDN it is an improvement of VDSR which change pairs of convolutional and nonlinear layer to residual dense blocks (RDB). When local feature fusion in RDB is then used to adaptively learn more effective features from previous and current local features and stabilizes the training of wider network.

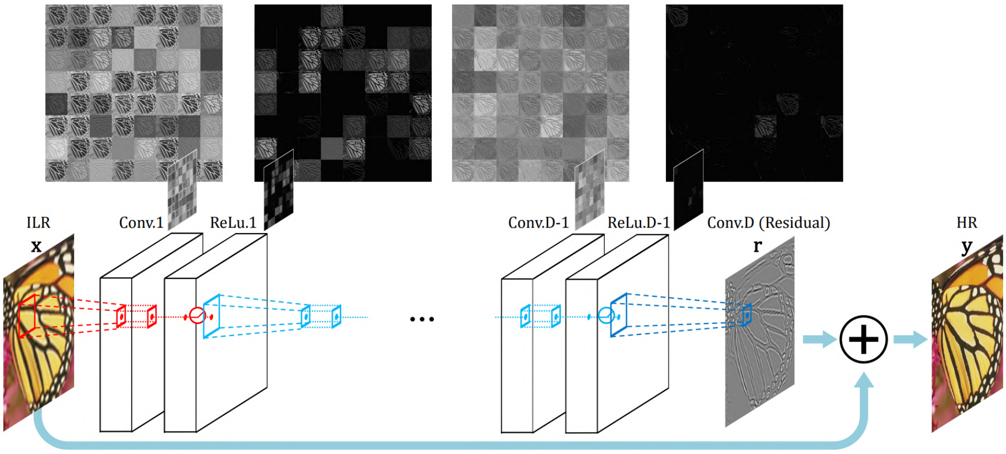


Figure 1.6: The structure of VDSR

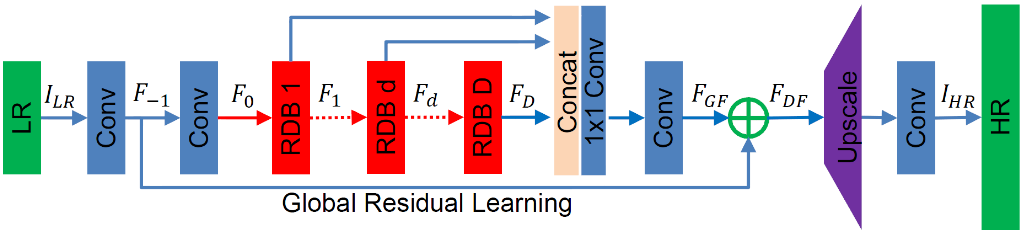


Figure 1.7: The structure of RDN

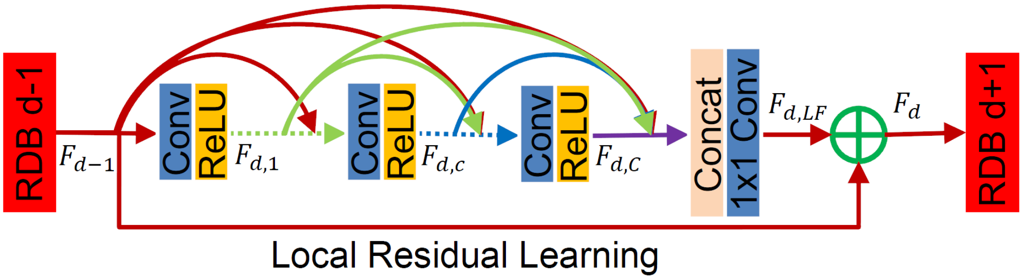


Figure 1.8: The structure of RDB

1.3 Attention-Based Networks

The **HAN**[11] – is holistic attention network. Authors propose this network by solving the question of what channel attention treats convolution layers as an isolated process that misses the correlation among different layers, despite Channel attention has been demonstrated to be effective for preserving information-rich features in each layer. Their network consists of a layer attention module (LAM) and an channel-spatial attention module (CSAM), to model the holistic inter-dependencies among all elements, such as: layers, channels, and positions. In addition to this, the proposed LAM adaptively emphasizes hierarchical features by considering correlations in layer-layer relations. In addition to this, CSAM learns confidence in all the positions of every channel and is capable of selectively capturing more informative features.

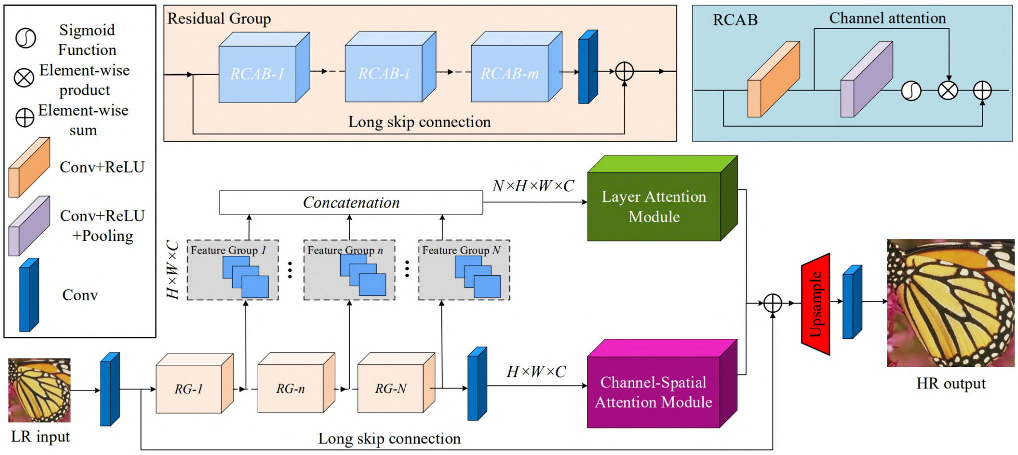


Figure 1.9: The architecture of HAN

2 Tested models

Were they chosen by me for comparison ???? models that demonstrate the current state and progress in the field of single image SR.

2.1 VDSR

This network model use global residual learning techniques and shows its advantages above non-residual networks in particular SRCNN[3].

Stricture of this networks it is up to 20 same stricture layers except of first and last, it is 64 filter of the size 3 by 3 by 64, it is 3 by 3 convolution filter with 64 channels, first layer compute input images, last layer reconstruct image. All structure shown on Figure 1.6. As input used interpolated low resolution (ILR) image. Convolutional operations reduce size of image. For deciding this problem all 64 spatial region padding with zeros before convolutional operations to keep same size. All layers predict only details which are added to input ILR image for achieve high resolution image.

For training this model used residual learning approach. Usual learning approach is concluded in minimization of mean square error (MSE):

$$MSE = \frac{1}{2}||y\text{-}f(x)||^2$$

Where x – is an interpolated low resolution image y – is a high resolution image f – is the desired model which predict $\hat{y} = f(x)$, where $\hat{y}$ – is a super resolution image.

In residual learning for avoid keeping a lot of image parts which newer change a long of all training process and save the memory used another equation for MSE:

$$MSE = \frac{1}{2}||r\text{-}f(x)||^2$$

Where $r = y\text{-}x$ and r – is residual image, only that parts of image which will change through training process.

Due to the above described loss layer have three inputs: residual estimate, ILR image and ground truth HR image. And loss is calculating as the Euclidean distance between the sum of network input and output and ground truth.

Training is carried out by optimizing the regression objective using mini-batch gradient descent based on back propagation. Momentum parameter set to 0.9 and for training regularization used L2 penalty multiplied by 0.0001

2.2 SRGAN

The aim of this model is generate High resolution images from low resolution input with attention to high frequency texture details. For achieve this aim method used perceptual loss which is weighted combination of content loss and adversarial loss with coefficient 10^{-3} this justified by the fact that MSE loss minimization based solutions take pixel-wise average of possible solutions that gives smooth result without high frequency details.

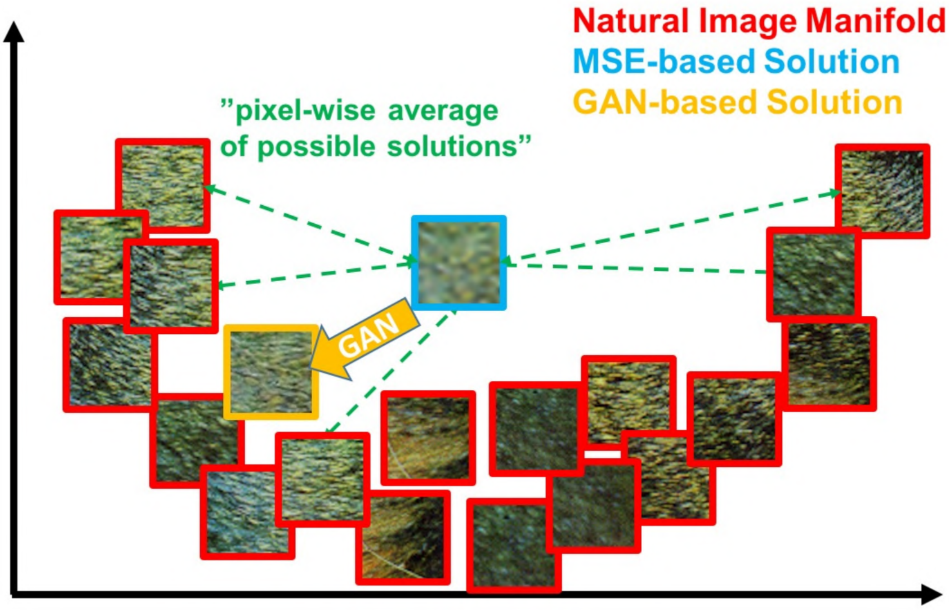


Figure 2.1: illustration of meaning result in approaches used MSE loss[2]

Therefore this method realize generator G trained for generate images which could fool the discriminator D trained to distinguish super resolution images from natural high resolution images.

In this method low resolution image input made from high resolution image using downscale operation with downscale factor 4 and subsequent Gaussian filtering.

As mentioned above method realize combination of two losses.

First is content loss. It is mean of absolute differences between extracted features of generated super resolution image $f_{vgg}(SR)$ and natural high resolution image $f_{vgg}(HR)$ achieved with VGG19 pretrained[12]

$$ContentLoss = mean\left(|f_{vgg}(SR) - f_{vgg}(HR)|\right)$$

Second is adversarial loss. It is mean squared differences between discriminator estimation of generated super resolution image $D(SR)$ and validation ground truth I in another words mean squared error of generated image.

$$AdversarialLoss = mean\left(D(SR) - I\right)^2$$

Also for train the discriminator used loss which is calculated as mean value of two MSE. Firs MSE between discriminator estimation of natural high resolution image$D(HR)$ and validation ground truth I. Second MSE between discriminator estimation of super resolution image $D(SR)$ and fake ground truth Z.

$$DiscriminatorLoss = \frac{mean\left(D(HR) - I\right) + mean\left(D(SR) - Z\right)}{2}$$

Structurally, the generator consists of an input layer containing a convolution filter and a nonlinear Rally activation, followed by a sequence of 16 residual blocks, then another convolution layer and two consecutive up-scaling blocks with a factor of x2, the output convolution layer completes the structure. For all convolutions , the size of features maps x64. This generator structure is called a super resolution residual network (SRResNet)[2]

2.3 EDSR

Advantages of this method is concluded in better version of residual network with optimized structure.

How shown on Figure 2.2 simple modification in compared with SRResNet[2] is removing Batch Normalization blocks. This simple modification gives two advantages improves the performance of the algorithm and save up to 40% memory during training process.

Advantages described above allow increase number of blocks from 16 in SRResNet[2] to 32 and number of feature channels from 64 to 256 and for solve the problem of numerical instability arising from an increase in the number of feature channels, described in paper wrote by Szegedy et al.[13], was introduced residual scaling with factor 0.1 it means was added scaling block after every convolutional block in each residual block. This increase the performance of the model.

The last thing which improved performance was start of the training process of model for x4 scale factor from pretrained model parameters of x2 scale factor.

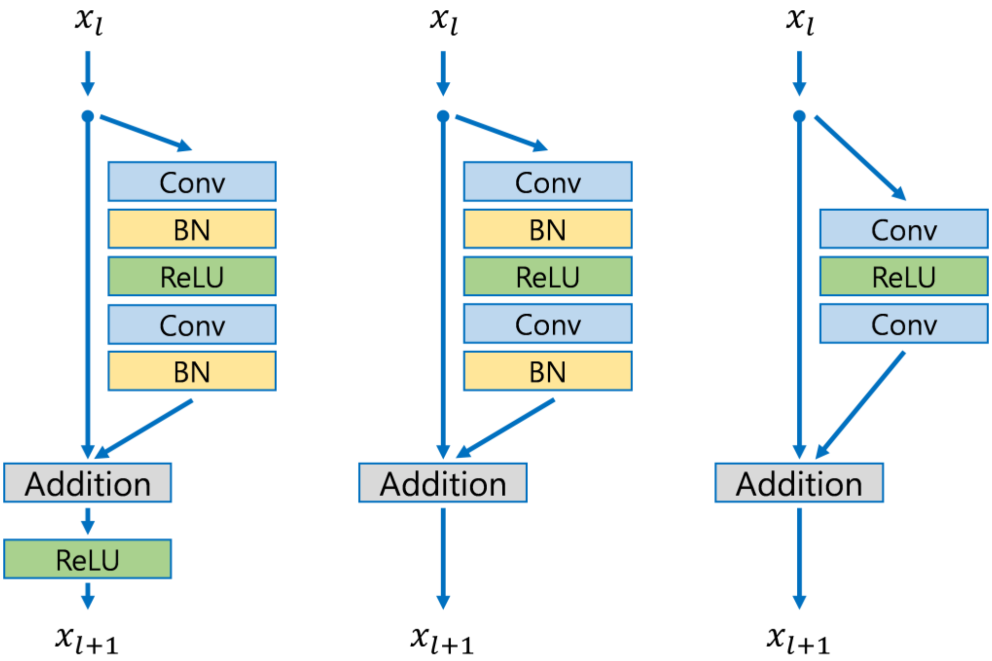


Figure 2.2: On the left shown residual block of original ResNet[4], in a middle shown residual block of SRResNet[2] and on the right shown building block of EDSR[9]

2.4 ESRGAN

ESRGAN[14] proposed improvements of network achieved by SRGAN[2] approach. From SRResNet structure was removed all batch normalization blocks. The features are normalized by mean and variance in the training-batch, use calculated meant of varying the value of the whole train data at the time of learning, and use predicted mean and variance of the whole training database for testing with an estimated mean and variance of all training data before study.Unpleasant artifacts and limit the generalization ability are introduced by batch normalization layers, as the data of training and testing is different in many forms. Also removing batch normalization layers increase generalization ability and decrease computational complexity and safe memory.

Second thing is replacing all basic blocks with RRDB blocks[5] it is shown on Figure 1.2. This improvement was made guided the statement what more layers and connections always increase the performance of the network.

Two techniques also was used for helping to increase depth of model it was residual scaling [13] and initialization of training with low variance parameters. This two techniques also used in EDSR approach[9].

Generator of ESRGAN algorithm realize structure which calls RRDB network, because it used residual in residual dense blocks as main structure element instead of common residual blocks. The implemented structure is as follows: The same as in SRGAN input layer, then 23 RRD blocks, the second convolution layer, after two consecutive up-scaling blocks and an output layer containing two convolution filters. The whole architecture does not contain BN layers.

3 Data sets

For avoid from comparison influence of different training data all chosen algorithms was trained on the same data set. For this aim was chosen DIV2K[1] data set which contain 1000 different images in 2k resolution and suit for training SR models with x4 up scaling factor.

For calculation numerical values which allows to compare of models performance numerically, was chosen three data-sets:

- Set5
- Test part of DIV2K
- Urban100[6]

3.1 DIV2K

DIV2K is an extremely popular single-image super resolution data-set that has 1,000 various image scenes and is splitted to 80 percent for training, 10 percent for validation and 10 percent test. At the NTIRE 2017 and NTIRE 2018 Super-Resolution Challenges, it was collected for super resolution tasks and in order to encourage research on image super-resolution with more realistic degradation. A low resolution image with different types of degradation is included in this data-set. After the standard bicubic down-sampling, several types of degradation are considered in synthesizing low resolution images. Low resolution images under the most comfortable setting of x4 are suffering from motion blur, Poisson noise and pixel shifting. The extended range of degradation under the wild x4 setting is further expanded to be of different levels from image to image.

DIV2K data-set propose a lot types of distorted images and high variance types of down-scaled images, but for comparison models in single image super resolution was taken only high resolution images, because every model have own method for preparing training data.

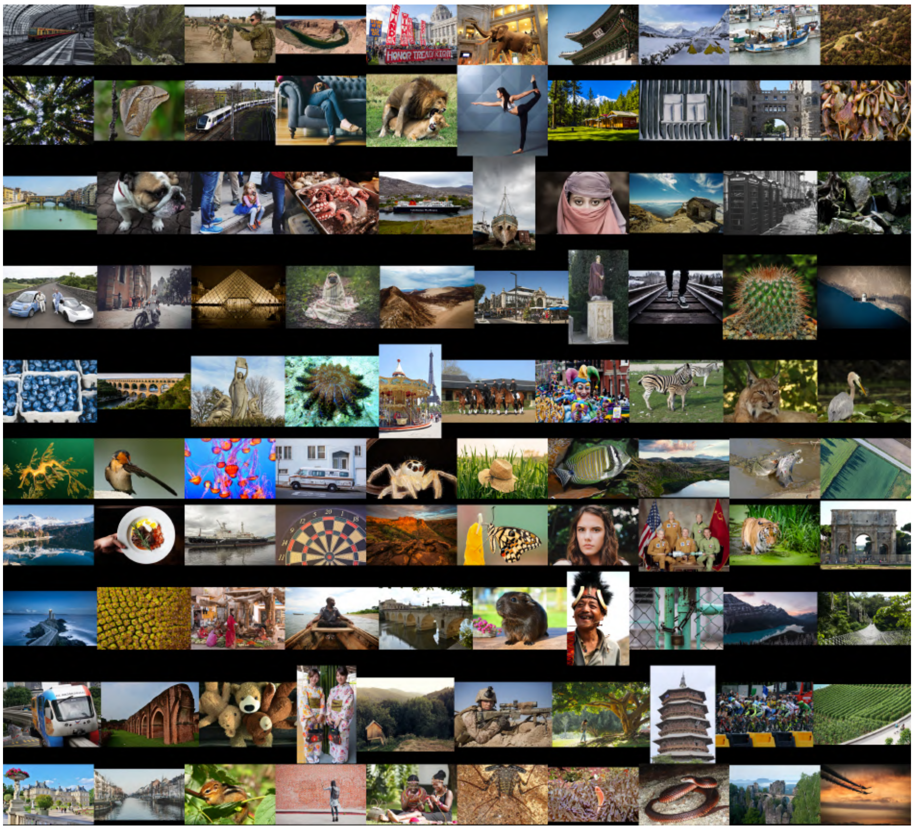


Figure 3.1: Examples of training images. DIV2K data-set

3.2 Set5

The Set5 data-set – is small data-set which contain only 5 images it is image of baby, bird, butterfly, head and woman. This test data set widely used for tasting in many papers describing super resolution algorithms.

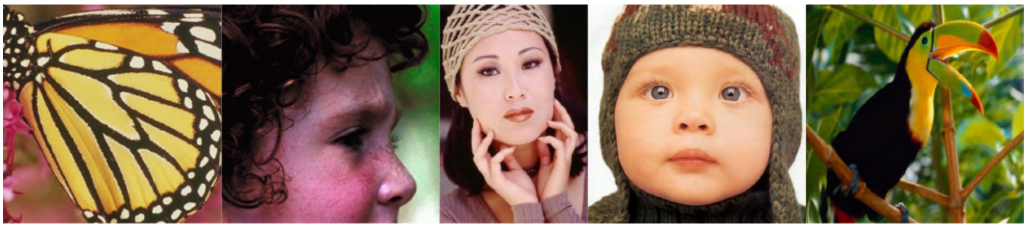


Figure 3.2: Set5 data-set

3.3 Urban100

The Urban100[6] data-set – is popular data-set which contains 100 images of urban environment. It means a lot of repeatable strict patterns and small details which could be lost after down sampling and it will be hard to predict or reconstruct it using algorithms. This data-set could be hard task for compared algorithms.

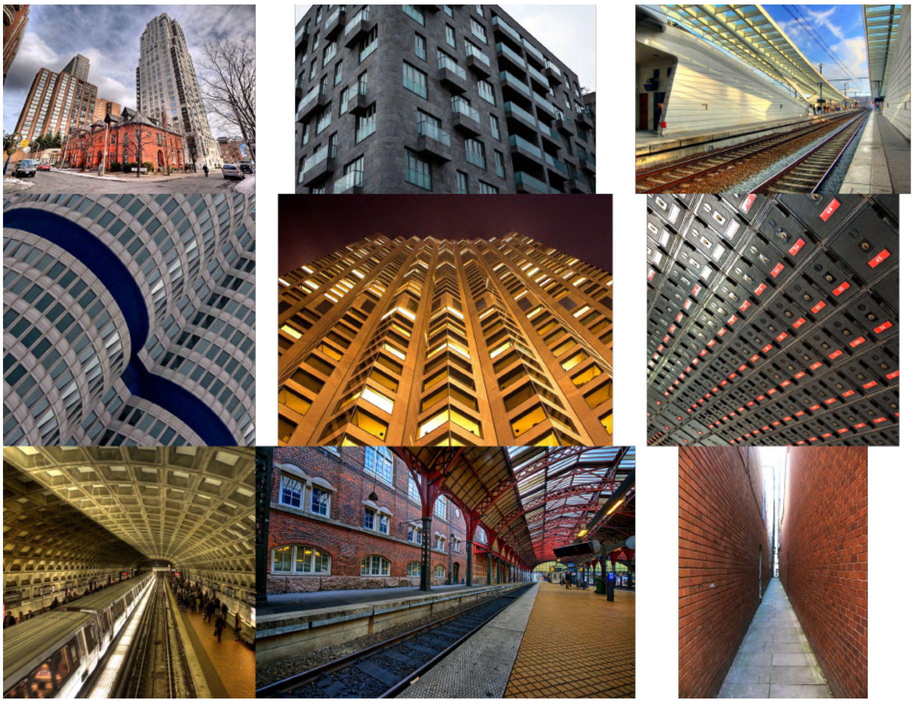


Figure 3.3: Examples of images. Urban100 data-set

4 Experiments

Conducted experiments concluded in training chosen algorithms on chosen train data-set for single image super sampling with upscale factor 4 and calculating numerical values such as peak signal to noise ratio (PSNR) and structure similarity index (SSIM) using chosen test data sets which represent a lot variance of data.

4.1 PSNR and SSIM

PSNR – is the peak signal-to-noise ratio that defines the ratio between the maximum possible signal value and the noise power that distorts the signal values. PSNR is measured on logarithmic scale in decibels, since many signals have wide dynamic range. In case of image super sampling PSNR great for evaluation the distortion of achieved images using algorithms.

Calculation of PSNR have following formula:

$$PSNR = 10 \log_{10} \left(\frac{MAX_I^2}{MSE} \right) = 20 \log_{10} \left(\frac{MAX_I}{\sqrt{MSE}} \right)$$

MAX_I – this is the maximum value taken by an image pixel, in our case all images in 8 bit RGB format it means that $MAX_I = 255$. MSE there is mean square error mentioned in section 2.2 and section 2.1. It computes as:

$$MSE = \frac{1}{mn} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} |I(i,j) - K(i,j)|^2$$

Where mxn is size of inputs arrays and $I(i,j)$, $K(i,j)$ elements of inputs arrays, in case of images mxn is resolution of image and $I(i,j)$, $K(i,j)$ is a pixels.

SSIM – Structure similarity index is perception-based model that recognizes image degradation as an inherent change of information, but also incorporates important perceptual phenomena such as luminance masking and contrast masking terms. With other techniques such as MSE or PSNR, the difference with other methods is that this approaches are calculating an absolute error. The structural information is the statement that the pixels the stronger the relationships, the closer they are located. In addition, this relationships bring important information about the object structures in the visual meaning. In our case, luminance masking is a situation of image distortions may to be less visible or not visible in regions with high brightens, and contrast masking is the situation of image disruptions may to be less visible or not visible in regions with high frequency information or complicated textures.

The general formula has the form:

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{\left(\mu_x^2 + \mu_y^2 + c_1\right)\left(\sigma_x^2 + \sigma_y^2 + c_2\right)}$$

There μ_x, μ_y are of mean input arrays in our case images. Next σ_x, σ_y are variance of input images and c_1, c_2 are correction coefficients that we need due to the smallness of the denominator which are calculated as $c_1 = (k_1 L)^2$, $c_2 = (k_2 L)^2$, k_1, k_2 always equal 0.01 and 0.03 accordingly and L is the dynamic range in our case is 255 for 8-bit RGB images.

4.2 Training procedure

In order to train the selected models, several platforms were used that provide computing power and allow executing program code written in python on remote machines. The run-time environment on such platforms is accessed via a browser. Google colaboratory with a pro subscription and Kaggle, which provides sufficient capacity in free mode, were used as such platforms. Also, to train the algorithms, the author's personal laptop was used to execute the program code on which the Jupyter Lab IDE was used.

Google colaboratory is a product from Google Research. It is allows anybody to write and execute arbitrary python code through the browser, and is especially well suited to machine learning, data analysis and education. With subscription Pro Google colaboratory offers powers of graphic card such as Tesla p100 or Tesla t4, 16 gigabytes of GPU memory, 4 cores of CPU, and extended RAM space. It also offers near to 150 gigabytes of HDD space and 12 hours of single environment run, also possible connect your environment to your Google drive for data management. The disadvantages of this platform are the inability to manually configure the hardware configuration for the environment, the limitation on the one-time execution time of the environment, as well as if the execution of the running algorithm ends while you are not near the computer to save the result after a certain time, the environment will be thrown automatically and the received data will be deleted. Another disadvantage is that each time the environment is started, the algorithm files need to be uploaded from scratch, since the environment is formatted every time it is restarted, but this disadvantage is solved by connecting a Google drive and storing information on it.

Kaggle allows users to find and publish data sets, explore and build models in a web based data science environment, work with other data scientists and machine learning engineers, and enter competitions to solve data science challenges. Kaggle provide Tesla p100 GPU with 16 gigabytes of memory, 2 cores of CUP, up to 13 gigabytes of RAM and 74 gigabytes of Hard drive. It have restriction for GPU usage 30 hours per week and 12 hours for single run. Also Kaggle allows save data-sets and executable files to your profile and it will be automated uploaded to the environment when it start. Kaggle have same disadvantage as Google colaboratory with loosing your results, but it is free as opposed to colaboratory which cost 10 euro per month.

The characteristic of authors computer are as follows:

- GPU: NVIDIA GeForce RTX 3060 (laptop) 6 gigabytes memory
- CPU: AMD Ryzen 5 5600H, 6 cores, 12 threads, 4,2 GHz boost frequency
- RAM: 16 gigabytes

Running algorithms on own computer does not have the disadvantages of the platforms described above, but due to the limited amount of GPU and RAM, most of the selected algorithms cannot be run.

Implementations of the selected models on the PyTorch framework were selected to perform the training. Also chosen algorithms used following packages for tasks as creation data pipeline, preparation data-set, before training, generating statistic and logs, demonstrating information while training procedure etc.

- numpy
- scikit-image
- imageio
- matplotlib
- tqdm
- opencv-python
- setuptools
- torchvision
- natsort
- scipy
- pillow
- urllib3

4.2.1 VDSR training

For training VDSR[8] algorithm from 800 training images of DIV2K[1] data-set was prepared 13900 cropped high resolution images with 256x256 resolution and same amount corresponding low resolution images with scale factor 4. Also from 100 validation images of same data-set was prepared 1387 images in same high resolution and corresponding low resolution images.

Training parameters was setting up as follows:

- Scale factor: 4
- Image size: 256
- Batch size: 16
- Optimizer: SGD
- Learning rate: 0.1
- Momentum: 0.9
- Model weight decay = 10^{-4}
- Total training epochs: 80

More about model stricture in section2.1.

For training was used Kaggle, because for training on laptop was not enough GPU memory.

4.2.2 SRGAN and ESRGAN training

To train the SRGAN[2] algorithm, 800 images from the DIV2K[1] data-set were compressed using bicubic interpolation to a resolution of 256x256 and used as high-resolution images, the same images were compressed by the same method to 4 times lower resolution and used as low-resolution images. For validation was used Set5 data-set with same preparing operations. Also , the following generator and discriminator parameters were set for training:

- Scale factor: 4
- Image size: 256
- Batch size: 4
- Optimizer: ADAM
- Learning rate: 0.0002
- Betas: b_1 = 0.5, b_2 = 0.999
- Total training epochs: 200

This algorithm was trained using author's laptop, because for this algorithm was enough GPU memory and the training process was faster when on cloud resources.

For ESRGAN[14] algorithm training procedure image preparation procedure was the same. For validation during training process also used Set5 dataset. More of model and training parameters was similar to SRGAN training instead of number of residual blocks for ESRGAN it was 23, for SRGAN it was 16.

ESRGAN requires more GPU memory for training, therefore for training was used Kaggle.

4.2.3 EDSR training

To obtain the results of the EDSR[9] algorithm, already trained models prepared by the authors of the algorithm were used. The models used were trained on the DIV2K[1] data-set. Two trained models with different parameters were used, which will be given below. The differences are in the depth of the models, namely in the number of layers and channels in the convolution filters, as well as the use of scaling to stabilize the training of the larger model. When training the models , the following parameters were set:

EDSR baseline:

- Scale factor: 4
- Residual blocks: 16
- Feature maps: 64
- Residual scaling: 1
- Batch size: 16
- Optimizer: ADAM
- Learning rate: 10^{-4}
- Betas: $b_1 = 0.9$, $b_2 = 0.999$
- Total training epochs: 300

EDSR:

- Scale factor: 4
- Residual blocks: 32
- Feature maps: 256
- Residual scaling: 0.1

- Batch size: 16
- Optimizer: ADAM
- Learning rate: 10^{-4}
- Betas: $b_1 = 0.9$, $b_2 = 0.999$
- Total training epochs: 300

4.3 Results and comparison

This chapter will present the results of the conducted experiments and tests. The selected models were tested on three testing data-sets validation part of DIV2K[1] one hundred images of different subjects in 2k resolution, Urban 100[6] data-set one hundred images of the urban environment and Set5 five images in a small resolution of various subjects. There are also examples of upscale of specific images from the presented data-sets, as well as one image in high resolution.

Also in this chapter are the results and specific examples of work for three classical methods of increasing image resolution such as nearest-neighbor interpolation, bilinear interpolation and bicubic interpolation.

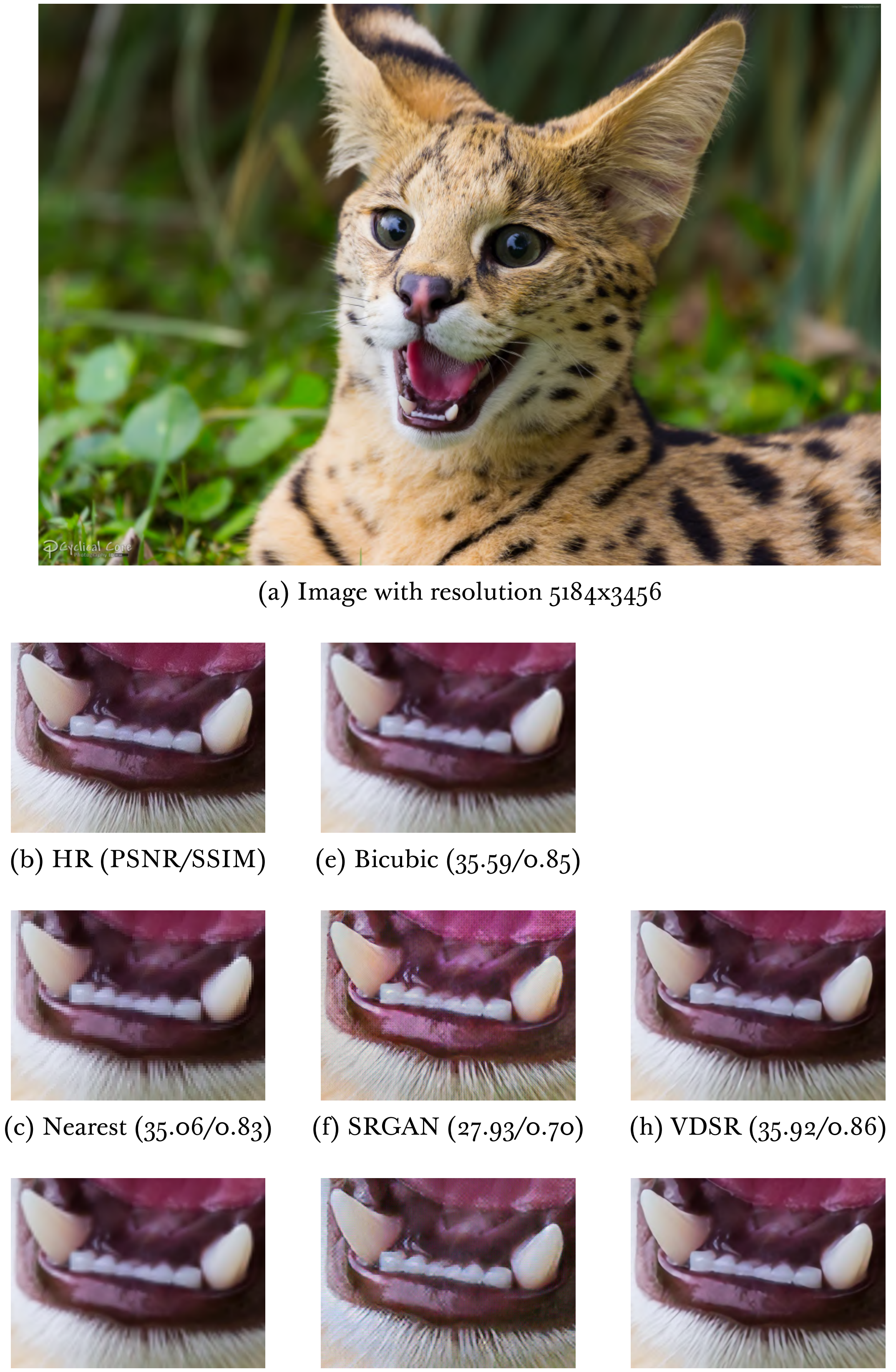
Table 4.1: PSNR and SSIM values of selected algorithms

	DIV2K (PSNR/SSIM)	Urban 100 (PSNR/SSIM)	Set5 (PSNR/SSIM)
Bicubic	32.92/0.77	30.93/0.66	32.00/0.81
Bilinear	32.64/0.75	30.78/0.63	31.67/0.78
Nearest	32.50/0.73	30.90/0.62	31.55/0.75
EDSR	34.01/0.84	32.12/0.81	33.70/0.90
EDSR-baseline	33.86/0.84	31.91/0.79	33.50/0.89
ESRGAN	28.77/0.66	28.61/0.60	28.71/0.68
SRGAN	27.97/0.61	28.08/0.59	28.29/0.69
VDSR	33.65/0.82	31.72/0.76	33.48/0.88

The table 4.1 shows the average values of PSNR and SSIM for each selected algorithm, as well as for several classical methods of increasing the image resolution. The best result is highlighted in red, and the second best result is highlighted in blue.

According to the data from the table 4.1 The best result in terms of PSNR and SSIM values is shown by the EDSR[9] algorithms, its less deep version and VDSR[8]. These algorithms are the development of the same ideas and approaches to the design of a neural network and show a systematic growth of results in accordance with the implemented improvements of algorithms. The following three algorithms are classical methods of increasing resolution.

Classical algorithms give high PSNR and SSIM values, but from a visual point of view, the results relative to the rest of the algorithms are the least impressive, as will be discussed later. The last in terms of these indicators are the SRGAN[2] Algorithms and its improved version ESRGAN[14]. Although these algorithms demonstrate low indicators, the visually obtained images are more satisfying than those of classical algorithms, which will also be discussed later. According to the authors, these two approaches are not aimed at high PSNR and SSIM values, but rather focus more on the best representation of images from a visual point of view. Also, the data obtained may be worse than the potentially possible values for these algorithms due to the choice of not the best implementation.



(a) Image with resolution 5184x3456

(b) HR (PSNR/SSIM)

(e) Bicubic (35.59/0.85)

(c) Nearest (35.06/0.83)

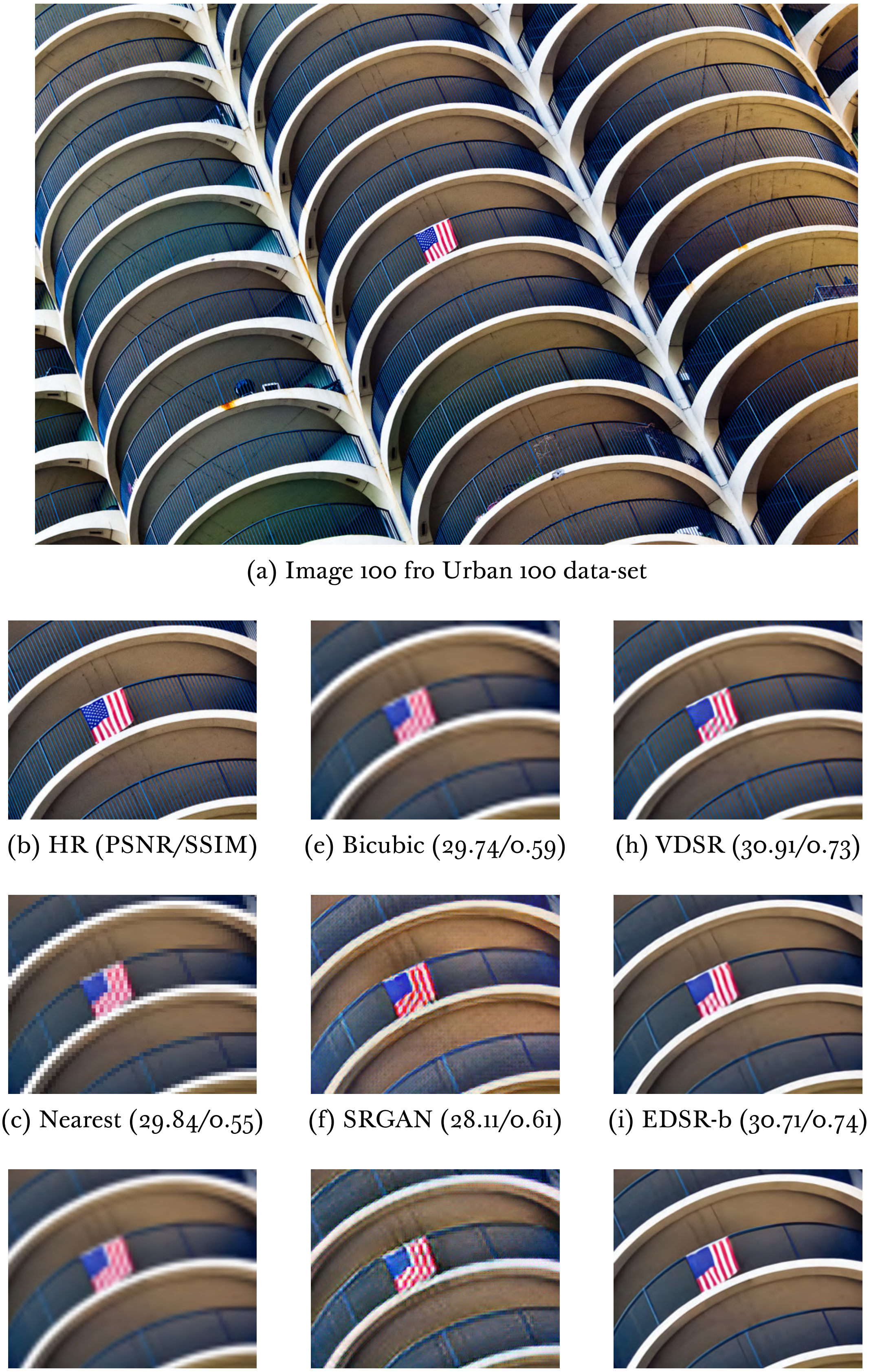
(f) SRGAN (27.93/0.70)

(h) VDSR (35.92/0.86)

(d) Bilinear (35.38/0.84)

(g) ESRGAN (29.15/0.74)

(i) EDSR-b (36.06/0.87)



(a) Image 100 fro Urban 100 data-set

(b) HR (PSNR/SSIM)

(e) Bicubic (29.74/0.59)

(h) VDSR (30.91/0.73)

(c) Nearest (29.84/0.55)

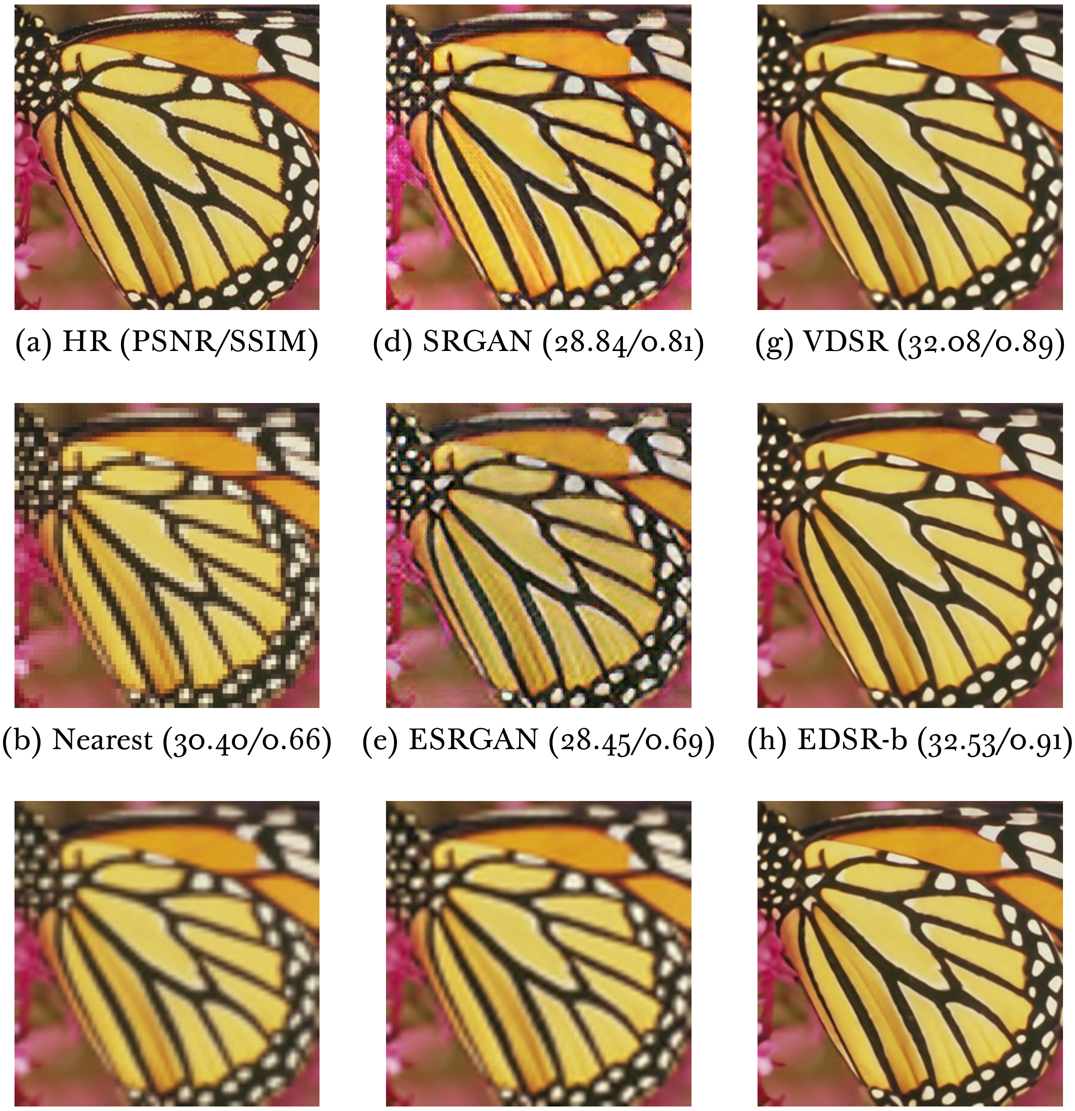
(f) SRGAN (28.11/0.61)

(i) EDSR-b (30.71/0.74)

(d) Bilinear (29.61/0.56)

(g) ESRGAN (28.58/0.57)

(j) EDSR (31.02/0.76)



(a) HR (PSNR/SSIM)

(d) SRGAN (28.84/0.81)

(g) VDSR (32.08/0.89)

(b) Nearest (30.40/0.66)

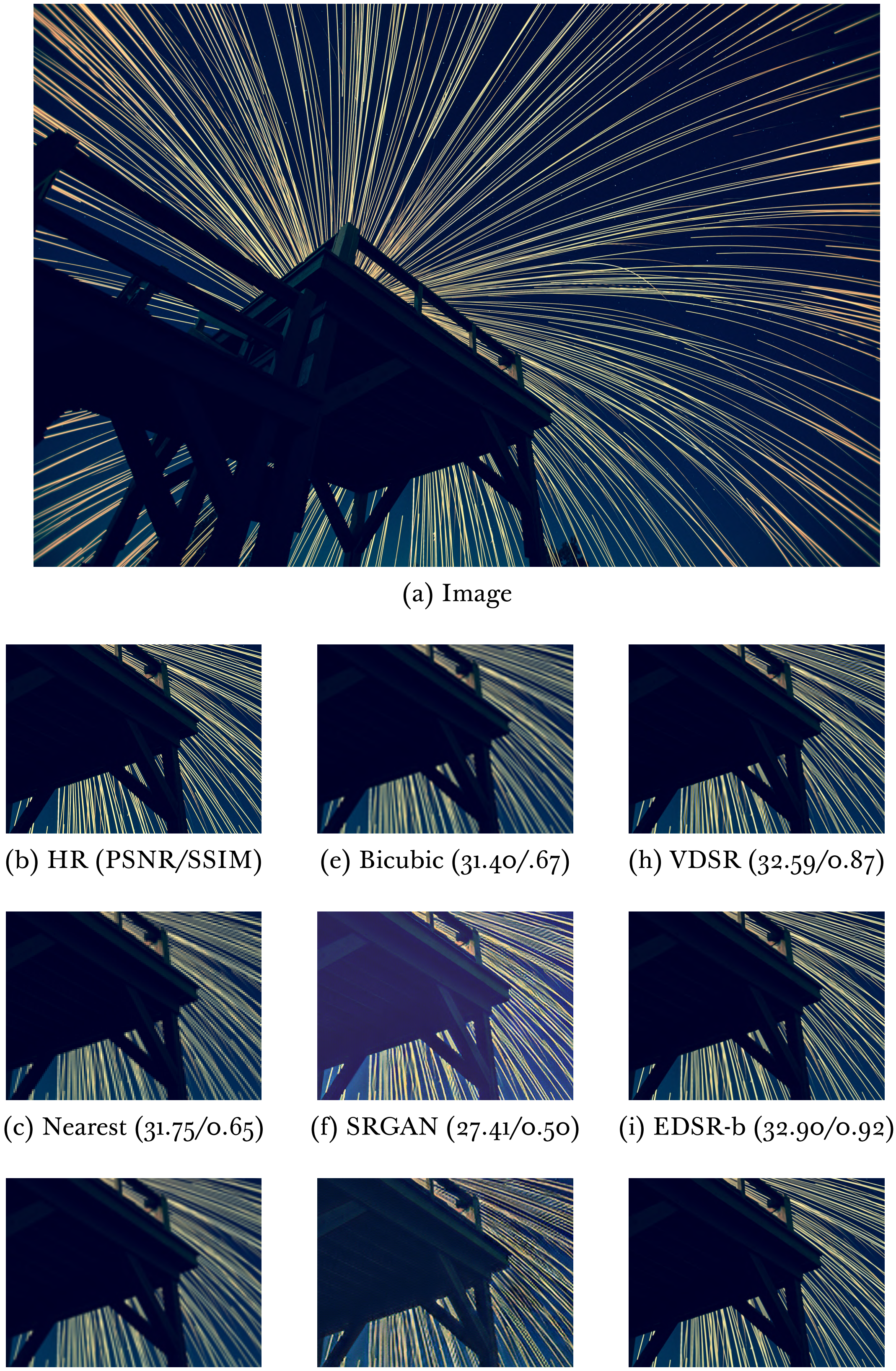
(e) ESRGAN (28.45/0.69)

(h) EDSR-b (32.53/0.91)

(c) Bilinear (30.10/0.70)

(f) Bicubic (30.20/0.74)

(i) EDSR (32.90/0.92)



(a) Image

(b) HR (PSNR/SSIM)

(e) Bicubic (31.40/.67)

(h) VDSR (32.59/0.87)

(c) Nearest (31.75/0.65)

(f) SRGAN (27.41/0.50)

(i) EDSR-b (32.90/0.92)

(d) Bilinear (31.21/0.64)

(g) ESRGAN (28.51/0.50)

(j) EDSR (33.54/0.94)

Four images with different characteristics and scenes were selected for visual comparison of the result. The first image of an animal in a very large original resolution, specifically 5184x3456. The image of the animal is interesting due to the complex texture of the fur of a large volume with small details. A high-resolution image was chosen to demonstrate the operation of algorithms in the condition of upscaling images in a resolution at which a sufficient number of small details are preserved and they are not lost during compression. The second image from the Urban 100 dataset demonstrates working with repeating small patterns. The third image is a popular butterfly from the Set5 dataset that demonstrates the work of algorithms on the whole image in an initially small resolution, as well as this image makes it possible to visually compare the results with other works, since it is often used to demonstrate the results. The fourth image from the DIV2K dataset demonstrates working in dark lighting, as well as contains many thin lines on which you can well observe the mistakes of the algorithms.

Considering the results of the algorithms for each selected image separately, as with the general values from the table, one can observe the dominance of algorithms such as EDSR and VDSR. In the case of the first images in very high resolution all algorithms show a high PSNR value, with the exception of the SRGAN and ESRGAN algorithms, they are prevented from getting a high PSNR value by the presence of artifacts that they contribute to the image and the distortion of the image gamut that they contribute when working. Artifacts introduced by BN layers of the SRGAN algorithm are shown in Figure.

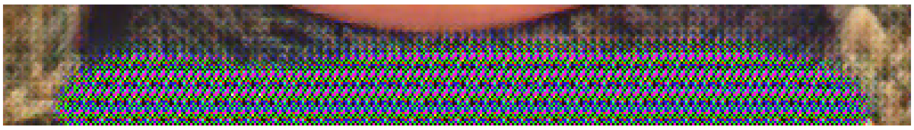


Figure 4.5: Examples of BN layers artifacts of SRGAN algorithm

Despite the low PSNR values, the images obtained by these two algorithms look much clearer than the images obtained by classical methods of increasing resolution, and the introduced artifacts at a higher resolution are invisible. It is impossible to observe the operation of a deeper version of EDSR in the first image due to the insufficient amount of GPU memory at the author's disposal, to obtain an image in this resolution with the help of a deeper version of EDSR, more than 16 gigabytes of GPU memory are needed. In the image from the Urban 100 data-set, you can see that when the stars on the flag are compressed, they are not saved and no algorithm can predict them. In a dark image, color distortions are strongly visible as a result of the SRGAN and ESRGAN algorithms.

Also in the last figure, you can clearly demonstrate the improvements achieved by increasing the depth of the network and the number of feature maps in the EDSR algorithm. The figures show that with an increase in the number of features maps, even smaller details cease to merge with each other and become clearly distinguishable. It can be noticed that in the image obtained using a deeper version of EDSR, smaller sparks are discernible. Also, in the right part of the image on the wooden beam, it can be seen that the texture of the tree is completely smoothed in the images obtained using algorithms.

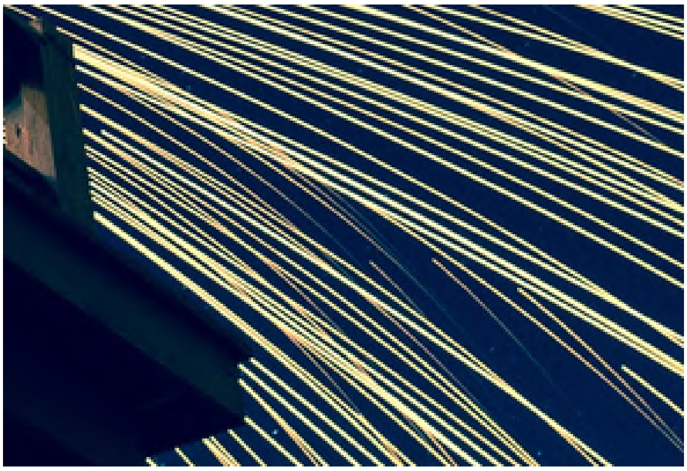


Figure 4.6: HR image

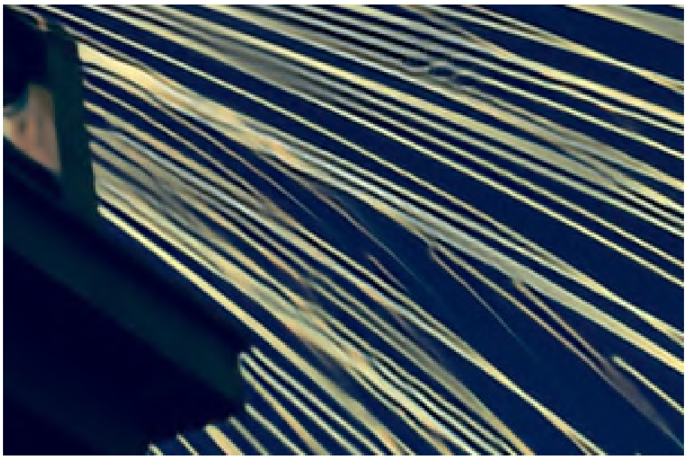


Figure 4.7: EDSR-baseline image)

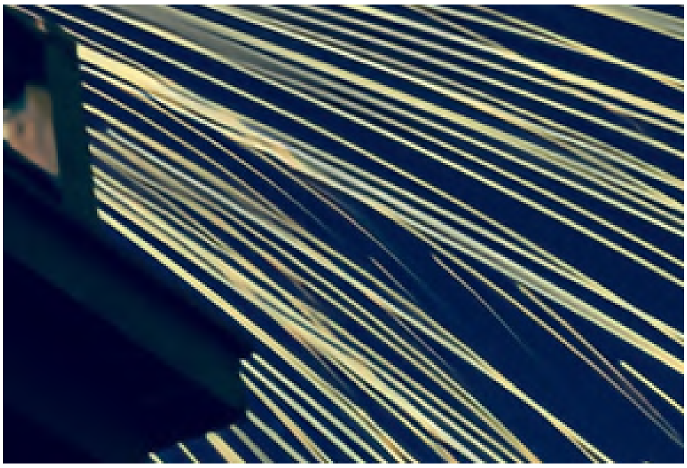


Figure 4.8: EDSR image

Conclusion

Summing up the comparison, we can say that according to numerical estimates, the EDSR and VDSR algorithms have a clear advantage. The values obtained by these methods generally exceed the values of the classical methods of increasing the image resolution. The results of the ESRGAN[14] and SRGAN[2] algorithms are the weakest due to the introduced artifacts and changes in the gamma of the images. From a visual point of view, the images obtained by the SRGAN and ESRGAN algorithms have greater sharpness in comparison with classical algorithms. In most cases, images obtained with a deeper EDSR[9] model look the most correct, but in cases of images with large volumes of high-frequency areas such as animal hair, images obtained with SRGAN and ESRGAN look sharper. The SRGAN algorithm has a problem with artifacts introduced by BN layers and strong gamma distortion, these problems are improved in the ESRGAN algorithm. Classical algorithms have the advantage that they do not require a pre-trained model for their work, but they require large computational costs to obtain a specific image, as well as the resulting images are always less clear than the images obtained using algorithms based on neural networks. The reviewed algorithms allow us to get very good results both from the point of view of numerical estimates and from a visual point of view, but the EDSR algorithm that has demonstrated the best results strongly smoothes surface textures and does not restore the image details lost during compression when super-sampling low-resolution images.

Bibliography

[1] Eirikur Agustsson and Radu Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, July 2017.

[2] Ledig Christian, Theis Lucas, Huszar Ferenc, Caballero Jose, Acosta Andrew, Cunningham and´Alejandro, Aitken Andrew, Tejani Alykhan, Totz Johannes, Wang Zehan, and Shi Twitter Wenzhe. Photo-realistic single image super-resolution using a generative adversarial network. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, May 2017.

[3] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Image super-resolution using deep convolutional networks. *eprint arXiv:1501.00092*, page 14, December 2014.

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[5] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[6] Jia-Bin Huang, Abhishek Singh, and Narendra Ahuja. Single image super-resolution from transformed self-exemplars. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5197–5206, 2015.

[7] Alexia Jolicoeur-Martineau. The relativistic discriminator: a key element missing from standard gan. *arXiv preprint arXiv:1807.00734*, 2018.

[8] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. Accurate image super-resolution using very deep convolutional networks. In *Computer Vision and Pattern Recognition (CVPR)*, 2016.

[9] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, July 2017.

[10] Sachit Menon, Alexandru Damian, Shijia Hu, Nikhil Ravi, and Cynthia Rudin. Pulse: Self-supervised photo upsampling via latent space exploration of generative models. In *Computer Vision and Pattern Recognition (CVPR)*, 2020.

[11] Ben Niu, Weilei Wen, Wenqi Ren, Xiangde Zhang, Lianping Yang, Shuzhen Wang, Kaihao Zhang, Xiaochun Cao, and Haifeng Shen. Single image super-resolution via a holistic attention network. *IEEE International Conference on Computer Vision*, page 16, August 2020.

[12] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *eprint arXiv:1409.1556*, September 2014.

[13] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alex Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *eprint arXiv:1602.07261*, February 2016.

[14] 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.

[15] Yulun Zhang, Yapeng Tian, Yu Kong, Bineng Zhong, and Yun Fu. Residual dense network for image super-resolution. In *CVPR*, 2018.

[16] Du Zongcai, Liu Ding, Liu Jie, Tang Jie, Wu Gangshan, and Fu Lean. Fast and memory-efficient network towards efficient image super-resolution. In *NTIRE (CVPR Workshop)*, 2022.