УДК 681.3.01+681.325

DOI: 10.15407/fmmit2025.40.075

Image upscaling using generative adversarial networks

Yaroslav Lys[1], Adrian Nakonechnyi[2]
[1]Student at Lviv Polytechnic National University, 12, Bandera Str, Lviv, 79013, email: yaroslav.lys.mknuo.2024@lpnu.ua
[2]SC. D, Prof at Computer Science Department of Lviv Polytechnic National University, 12, Bandera Str, Lviv, 79013, email: adrian.y.nakonechnyi@lpnu.ua

The purpose of this article is to develop and improve image scaling algorithms aimed at preserving details and visual appearance when changing image size. The subject of this article is image scaling algorithms, particularly their enhancement to ensure high-quality images. An analysis of artifacts arising during image scaling are conducted. Additionally, various image scaling algorithms are analyzed. Special attention is given to algorithms and methods of image scaling that use machine learning and neural networks. In this article a software implementation of the image scaling method using Generative Adversarial Networks is proposed. The architecture of GAN is described in detail and the loss functions for generator and discriminator are calculated. The results of this article demonstrate that improved algorithms based on GANs allow achieving high-quality image scaling with minimal detail loss and artifacts. The software implementation of the algorithms showed efficiency and the possibility of real-time image scaling, which is crucial for various practical applications such as medical imaging, video processing, and other fields where image quality is critical.

Keywords: image processing, image scaling, neural networks, generative adversarial networks, generator, discriminator.

Introduction. Image scaling plays a critical role in many areas of activity such as computer graphics, web development, medicine, and astronomy, ensuring accurate reproduction, optimization, and analysis of visual data.

Image scaling is an indispensable tool in computer graphics, where the accuracy of scaling determines the quality of visual effects in games, movies, and other multimedia products. In web development, scaling is used to adapt images to different screen sizes, providing a better user experience. In medicine, scaling algorithms improve the analysis of medical images, aiding in more accurate diagnosis and treatment of patients. In astronomy, scaling assists in processing astronomical images, allowing for more detailed studies of celestial objects.

The need to develop and implement new image scaling algorithms is driven by the demand for improved quality and speed of image processing. Existing methods, although quite effective, have certain limitations that affect the final quality of the result. For instance, some algorithms may lead to the loss of details or image distortion when increasing or decreasing image size. Therefore, the continuous improvement of scaling methods is essential to meet the growing demands of various fields.

The advantages of image scaling include the ability to precisely fit image sizes to different needs, preserve important details and structure of the image, and enhance the efficiency of visual information processing. However, drawbacks such as potential artifacts, blurring, or loss of quality during improper scaling highlight the necessity of carefully selecting and configuring algorithms for specific tasks.

The prospects for further development of image scaling technologies include the use of artificial intelligence and machine learning to create more adaptive and accurate algorithms. These methods can significantly improve the quality of scaling, reduce the number of artifacts, and ensure better performance. Furthermore, the integration of new technologies into various fields promises to enhance the efficiency and accuracy of working with visual data.

Thus, the problem of image scaling is multifaceted and requires a comprehensive approach to its solution. Understanding the specific requirements and limitations of different fields allows for the development of more effective and adaptive algorithms. Image scaling not only improves the quality of visualization but also opens new possibilities for analysis and use of images in various scientific and applied areas. Continued research in this field holds great potential for significantly impacting the development of modern technologies.

1. Nearest neighbor interpolation

Nearest Neighbor Interpolation [1] is a technique used to estimate the value of a function at a new, unknown point by looking at the values of the function at surrounding known points. Essentially, it is a way to fill in gaps between existing data points. In the context of image scaling, the data points are pixels.

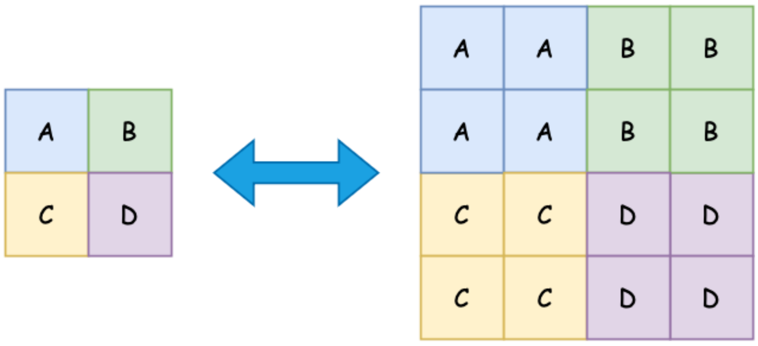


Fig.1 Nearest neighbor interpolation

The key idea of nearest neighbor interpolation is that it simply replicates the value of the nearest pixel when enlarging or removes it when shrinking an image. This method is quick and easy to apply, but it can lead to a loss of image quality, especially when significantly enlarging images, as it can result in blocky or stair-step edges. Therefore, it is often not used for resizing high-quality images. However, it can be useful in certain situations where speed is more important than image quality.

To summarize the advantages, the main advantage of nearest neighbor interpolation is its simplicity and understandability. It also has low algorithmic complexity, making it ideal for use in real-time systems.

The disadvantages include the frequent occurrence of the stair-step effect mentioned earlier. Additionally, since it only considers the nearest neighbor, it ignores potentially valuable data from other nearby points, leading to information loss.

Overall, nearest neighbor interpolation is a fast and simple method, but it may be less accurate compared to other interpolation methods.

2. Bilinear interpolation

Bilinear Interpolation [1] is another method for estimating the value of a function at a specific point based on its known values at surrounding points. In image scaling, the known data points are the pixel values of the original image, and the function $f(x, y)$ represents the intensity or color value of a pixel at coordinates (x, y).

Let's consider how bilinear interpolation works:

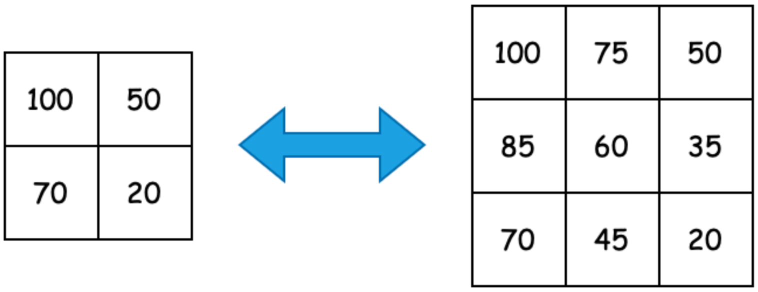
1. Consider a two-dimensional plane of known data points, where (x_1, y_1) and (x_2, y_2) are the coordinates of two adjacent points along the x-axis, and (y_1, y_2) are the coordinates of two adjacent points along the y-axis.
2. Suppose we want to estimate the value of the function $f(x, y)$ at an arbitrary point (x, y) inside the rectangle formed by the points (x_1, y_1), (x_2, y_1), (x_1, y_2), and (x_2, y_2)
3. Next, perform linear interpolation in the x direction for both pairs of points along the y-axis. For the lower pair of points (x_1, y_1) and (x_2, y_1), calculate the interpolated value R_1:

$$R_1 = f(x_1, y_1)\frac{x_2 - x}{x_2 - x_1} + f(x_2, y_1)\frac{x - x_1}{x_2 - x_1}$$

$$R_2 = f(x_1, y_2)\frac{x_2 - x}{x_2 - x_1} + f(x_2, y_2)\frac{x - x_1}{x_2 - x_1}$$

4. Now perform linear interpolation in the y direction using the interpolated values R_1 and R_2:

$$f(x, y) = R_1\frac{y_2 - y}{y_2 - y_1} + R_2\frac{y - y_1}{y_2 - y_1}$$



100	50
70	20

100	75	50
85	60	35
70	45	20

Fig. 2 Bilinear interpolation

The result, $f(x, y)$, is the bilinearly interpolated value of the function at point (x, y).

To summarize, bilinear interpolation, while providing smoother results than nearest neighbor interpolation, can often result in overly blurry outcomes, leading to a loss of detail in the image.

3. Bicubic interpolation

Bicubic interpolation [1] is a method that extends cubic spline interpolation to interpolate data points on a plane (two-dimensional space). Bicubic interpolation can be performed using Lagrange polynomials, cubic splines, or the cubic convolution algorithm.

This method uses a more complex algorithm to calculate the new pixel value based on the 16 nearest pixels. It produces sharper images than bilinear interpolation but can also lead to unwanted distortions, particularly the halo effect around objects in the image.

4. Lanczos interpolation

Lanczos interpolation [1] is a method based on the Lanczos filter. The idea of this filter is based on using the normalized $sinc$ function $sinc(x) = \frac{\sin \pi x}{\pi x}$, where the $sinc$ function is the cardinal sine function.

Lanczos interpolation involves using the Lanczos window function:

$$L_{\omega}(x) = sinc(\frac{x}{a})$$

The filtered function $S(x)$ is a discrete convolution of the original function, given by an ordered array of samples s_i, and a function called the Lanczos kernel:

$$S(x) = \sum_{i=[x]-a+1}^{[x]+a} s_i L(x-i)$$

where $[x]$ denotes the integer part of x.

The Lanczos kernel represents the derivatives of the $sinc$ function on the Lanczos window function, defined to be zero outside the given support parameter a.

In image processing, since images are two-dimensional functions, the convolution uses a two-dimensional Lanczos kernel:

$$L(x,y) = L(x)L(y)$$

The Lanczos interpolation method provides better results compared to previous methods, but it can be computationally expensive. Additionally, using Lanczos interpolation can lead to the appearance of the "ringing" effect, where an unwanted "halo" appears around objects. This effect is due to the fact that for values of the parameter $a > 1$, the Lanczos kernel takes on negative values for certain argument values. These negative values can appear even if all sample values are positive.

5. Interpolation using Fourier Transform

This interpolation method is based on the use of the Fourier transform. In this method of image scaling, the Fourier transform is used to decompose the image into sine and cosine components. The data obtained from the Fourier transform represent the image in the frequency domain, while the input image is the equivalent in the spatial domain. Generally, this method transforms the image into the frequency domain and then back into the spatial domain, resulting in an image with higher resampling and thus higher resolution.

Since we are only interested in digital images, we can limit ourselves to the Discrete Fourier Transform (DFT). The DFT is a discretized Fourier transform, so it contains not all frequencies that form the image, but only a set of samples that is large enough to fully describe the image in the spatial domain. The number of frequencies corresponds to the number of pixels in the spatial domain of the image, meaning the image in both the spatial and Fourier (frequency) domains has the same size.

For a square image of size $N \times N$, the formula for the two-dimensional DFT is:

$$F(k,l) = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} f(i,j)e^{-2\pi(\frac{ki}{N}+\frac{li}{N})}$$

where $f(a,b)$ is the image in the spatial domain, and the exponential term is the basis function corresponding to each point $F(k,l)$ in the Fourier domain. This equation can be interpreted as: the value of each point $F(k,l)$ is obtained by summing the products of the spatial image with the corresponding basis function.

Similarly, the formula for transforming from the Fourier domain back to the spatial domain can be derived. The inverse Fourier transform will then be:

$$f(a,b) = \frac{1}{N^2}\sum_{b=0}^{N-1} P(k,b)e^{-2\pi i\frac{lb}{N}}$$

Where

$$P(k,b) = \frac{1}{N}\sum_{a=0}^{N-1} f(a,b)e^{-2\pi i\frac{ka}{N}}$$

Using these two formulas, the spatial domain image is first transformed into an intermediate image using N one-dimensional Fourier transforms. This intermediate image is then transformed into the final image, again using N one-dimensional Fourier transforms. Representing the expression for the two-dimensional Fourier transform as a series of $2N$ one-dimensional transforms reduces the number of required calculations.

However, even with these improvements for computational efficiency, the regular one-dimensional DFT has a computational complexity of N^2. The computational complexity can be reduced to $NlogN$ by applying the Fast Fourier Transform (FFT). This improvement is very significant for large images.

The Fourier transform method produces good results but can also lead to distortions and effects like ringing and halos.

6. Image upscaling methods using AI

Modern scaling methods often use neural networks that analyze images and generate new pixels, which seamlessly integrate with the surrounding pixels and the overall image. This process is known as "super resolution." It involves training a machine learning model on a large set of high-quality images so that it learns to generate new pixels that perfectly fit the image.

As a result, methods using neural networks can create images that look much more natural and visually appealing than those created by classical scaling methods. Furthermore, the advantage of artificial intelligence is its ability to detect and correct other image issues, such as noise reduction, increased sharpness, and color enhancement. This leads to even more realistic images.

One of the main advantages of Convolutional Neural Networks (CNNs) [2] in this field is their ability to capture complex patterns and details of the image due to their architecture. This ability allows the network to create high-quality scaled images that retain more information and look more natural compared to traditional interpolation methods like bicubic or linear interpolation. CNNs can learn from large amounts of data, enabling them to optimize results and find the best ways to recover details during scaling. This is particularly important for applications where image quality is critical, such as in medicine or digital content creation.

However, using CNNs for image scaling also has its drawbacks. First, training such a network requires a large amount of data and computational resources. This can be a challenging task, especially for organizations with limited resources. Additionally, the training process can be time-consuming, affecting the time to deploy such solutions in real systems. Another issue is the possibility of generating artifacts or unwanted details that may appear in the scaled images, especially if the network was improperly trained or if it encounters data that differs from what it was trained on.

Despite these challenges, CNNs remain a powerful tool for image scaling, especially when quality and detail are key requirements. They offer significantly better performance compared to traditional methods, although they require substantial resources to achieve the best results.

7. Generative Adversarial Networks for image upscaling

To begin with, let's generally consider what generative adversarial networks (GANs) [3] are and what they can be used for. Generative adversarial networks (GANs) are a class of neural networks that consist of two main components: a generator and a discriminator.

Generator: This is a neural network that takes input data (such as signals, images, sound, etc.) and tries to generate data that is as similar as possible to real data.

Discriminator: This is a neural network that takes input data and its main task is to distinguish real data from the data generated by the generator.

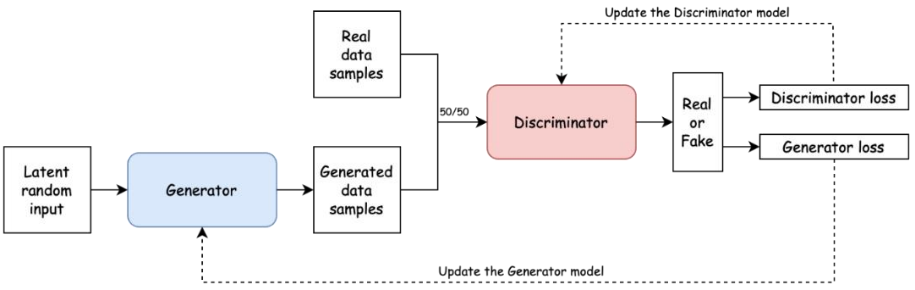


Fig. 3 GAN architecture

During the training process of a GAN, the generator and discriminator continuously improve through competition with each other. The generator tries to fool the discriminator by creating increasingly realistic data, while the discriminator aims to improve its ability to distinguish real data from generated data. As a result of this training, the generator learns to create highly realistic data.

The GAN architecture for image scaling consists of the same two components mentioned above: the generator and the discriminator. The generator takes a low-resolution image as input and attempts to create a high-resolution version of it. The discriminator's task is to determine whether the high-resolution image given to it is real or generated by the generator. Thus, during the training process, the generator learns to create highly realistic high-resolution images based on low-resolution images.

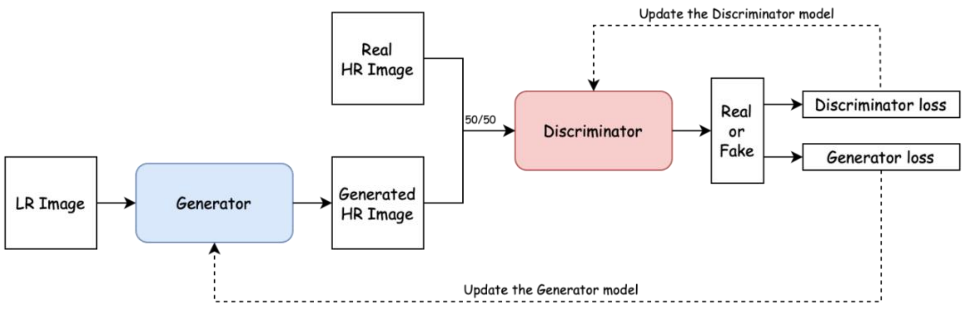


Fig. 4 GAN architecture for image upscaling

Now let’s consider loss function. In machine learning, a loss function, also known as a cost function or objective function, is a measure that quantifies the difference between the predicted values of the model and the actual values of the data. Essentially, it shows how well the model performs on a given dataset. The goal of training a machine learning model is to minimize this loss function, thereby improving the model's accuracy. Loss functions come in various forms depending on the type of machine learning task being performed.

The loss function for GANs used for image scaling is called the perceptual loss function [4], which consists of two components: content loss and adversarial loss.

Loss function:

$$l^{SR} = l_X^{SR} + 10^{-3} l_{Gen}^{SR}$$

where l_X^{SR} – content loss; l_{Gen}^{SR} – adversarial loss.

Now let’s discuss content loss function in detail. To compare two images using a discriminator, it is essential to select the appropriate loss function. This function will be the content loss function. We will next examine the mechanism by which the discriminator can compare two images (one generated by the generator and the real one).

In the context of image scaling, a pixel-wise comparison approach is typically used to compare two images. This usually involves using simple loss functions such as Mean Squared Error (MSE) or Mean Absolute Error (MAE), where the difference between corresponding pixels in the real and generated images is calculated. The main drawback of pixel-wise comparison is that it does not account for high-level semantic information in the images; it focuses only on the values of individual pixels. This can lead to blurriness in the results, especially in complex tasks like image scaling, where fine details are crucial.

Therefore, due to the limitations of pixel-wise comparison, we will use another method based on VGG (Visual Geometry Group) [5] in image scaling. The main idea of this method is to use feature maps from different layers of convolutional neural networks (CNNs) that have been pre-trained on a large dataset. This approach helps capture differences in the perception of two images, such as discrepancies in content and style, which are not always noticeable at the pixel level. By extracting feature maps from both the real and generated images, we can compute the difference in high-level features, such as textures, edges, patterns, shapes of objects, etc.

Table 1

Comparison of Pixel-wise Similarity and Perceptual Similarity

Criteria	Pixel-wise Similarity	Perceptual Similarity
Basis for comparison	Individual pixels	High-level features
Method of obtaining elements for comparison	Direct extraction of individual pixels from the image	Feature extraction using a neural network
Computational complexity	High computation speed	Low computation speed
Alignment with human perception	May not align with human perception	Aligns with human perception

Currently, there are two popular versions: VGG16 and VGG19. They are built with the same architecture but differ in the depth of the neural network. VGG16 has 16 layers, while VGG19 has 19 layers (including convolutional and fully connected layers).

The content loss function using VGG is defined as follows:

$$l_{VGG|i,j}^{SR} = \frac{1}{W_{i,j} + H_{i,j}} \sum_{x=1}^{W_{i,j}} \sum_{y=1}^{H_{i,j}} (\varphi_{i,j}(I^{HR})_{x,y} - \varphi_{i,j}(G_{\theta G}(I^{LR}))_{x,y})^2$$

Where:

$\varphi_{i,j}$ represents the feature map obtained from the j j-th convolutional layer before the i i-th pooling layer using max pooling.

$G_{\theta G}(I^{LR})$ is the generated high-resolution image.

I^{HR} is the original high-resolution image.

$W_{i,j}$ and $H_{i,j}$ denote the dimensions of the corresponding feature maps in VGG.

Essentially, this formula calculates the mean squared error (MSE) for the feature maps extracted from the generated high-resolution image and the true high-resolution image.

Now let's discuss adversarial loss. In Generative Adversarial Networks (GANs), a crucial component is the adversarial loss function. This function evaluates how well the generator can deceive the discriminator by creating images that look realistic and indistinguishable from real images. The main idea is that the generator aims to minimize this loss, while the discriminator aims to maximize it.

The generator strives to create images that closely resemble real ones, meaning they are high-resolution and realistic. The better the generator fools the discriminator, the lower the adversarial loss. Conversely, the discriminator learns to distinguish real images from those created by the generator, trying to increase the adversarial loss for the generator.

The adversarial loss function is mathematically expressed as:

$$l_{Gen}^{SR} = \sum_{n=1}^{N} -\log D_{\theta D}(G_{\theta G}(I^{LR}))$$

Where:

N is the number of samples in the dataset.

$G_{\theta G}(I^{LR})$ is the high-resolution image generated by the generator from the low-resolution image I^{LR}.

$D_{\theta D}$ is the probability that the discriminator recognizes this image as real.

In other words, $D_{\theta D}(G_{\theta G}(I^{LR}))$ represents the probability that the image generated by the generator $G_{\theta G}(I^{LR})$ is considered a real image by the discriminator. The generator tries to maximize this probability, reducing the value of the adversarial loss, while the discriminator tries to minimize it, improving its ability to recognize fake images.

Thus, the interaction between the generator and the discriminator in GANs is a zero-sum game, where one side tries to reduce the loss, and the other tries to increase it. This process encourages both networks to continually improve, ultimately leading to the generator creating highly realistic images.

Now let's consider architecture of generator.

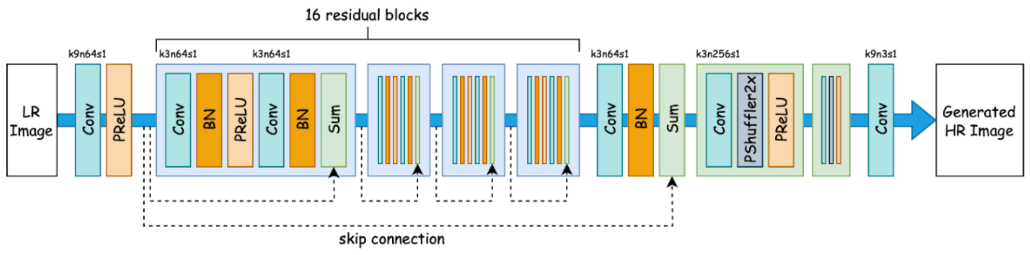


Fig. 5 Architecture of generator

Initially, a low-resolution image is fed into the generator. The process begins with an initial convolutional transformation, which includes a convolutional layer with 9 filters of size 64 and a stride of 1, followed by the Parametric ReLU activation function.

Next, the image passes through 16 residual blocks. Each residual block contains a convolutional layer with 3 filters of size 64 and a stride of 1, batch normalization, the

Parametric ReLU activation function, and a residual connection that adds the input signal to the block's output.

Following these 16 blocks, there is a final residual block consisting of a convolutional layer with 3 filters of size 64 and a stride of 1, batch normalization, and the addition of the signal from the previous blocks.

Afterwards, the image goes through two pixel shuffle blocks. Each pixel shuffle block includes a convolutional layer with 3 filters of size 256 and a stride of 1, a pixel shuffle operation to double the resolution, and the Parametric ReLU activation function.

At the end of the neural network, a final convolutional layer with 9 filters of size 3 and a stride of 1 generates the high-resolution output image.

In summary, the generator's overall workflow can be described as follows. The low-quality input image is processed by an initial convolutional layer. It then passes through 16 residual blocks, each containing convolutional layers, batch normalization, and the PReLU activation function. Next, it goes through resolution enhancement blocks using PixelShuffle operations. Finally, the image is processed by a final convolutional layer to produce a high-quality, high-resolution image.

Now let’s consider architecture of discriminator.

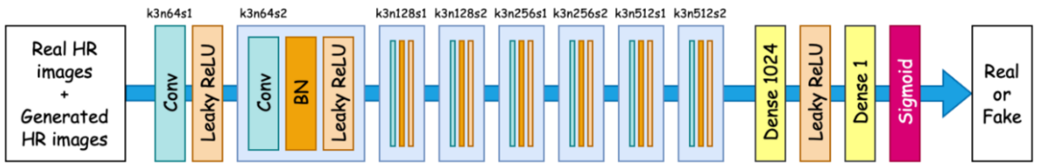


Fig. 6 Architecture of discriminator

This architecture of the discriminator is designed to classify images as real or generated. The discriminator is a crucial part of the generative adversarial network (GAN) system. Let's examine its components in detail.

The discriminator takes high-resolution images as input, which can be either real or generated by the generator. The process begins with a convolutional transformation, involving an initial convolutional layer with 3 filters of size 64 and a stride of 1, followed by the Leaky ReLU activation function.

Next, a convolutional block processes the image. This block contains a convolutional layer with 3 filters of size 64 and a stride of 1, batch normalization, and the Leaky ReLU activation function. Subsequent blocks process larger feature maps and include convolutional layers with varying parameters.

After the convolutional blocks, the data passes through fully connected layers. The first fully connected layer contains 1024 neurons and processes the output from the previous layer for further classification, followed by the Leaky ReLU activation function. The final layer contains a single neuron that reduces the output to a single value for classification.

At the very end of the discriminator network, there is a layer that uses the sigmoid activation function, which converts the output value to a range between 0 and 1. This value is interpreted as the probability that the input image is real (closer to 1) or generated (closer to 0).

In summary, the overall workflow of the discriminator can be described as follows. The high-quality input image passes through a series of convolutional layers, with each layer extracting different features. After the convolutional layers, the data goes through fully connected layers for further processing. Finally, the output layer with a sigmoid activation function generates the probability that the image is real or generated. This architecture enables the discriminator to effectively distinguish between real and generated images, aiding in training the generative network to create more realistic high-quality images.

8. Implementation and training GAN for image upscaling

For implementation I used Python programming language and Keras and TensorFlow libraries. Combining Keras and TensorFlow allows leveraging the benefits of both libraries. Keras is used for the rapid creation and prototyping of models, while TensorFlow provides the necessary flexibility and performance for training and deploying them. This combination is ideal for developing complex deep learning models, such as Generative Adversarial Networks (GANs), which are used for image scaling.

Table 2

Characteristics of computer for training

Component	Characteristic
Operating system	Windows
CPU	AMD Ryzen 7 5700x
RAM	32 GB

9. Results

Let's consider the following images. These images display the results of upscaling using different methods. The initial low-resolution image (LR Image) shows a dog's nose with noticeable pixelation and loss of detail.

The nearest-neighbor method also exhibits significant pixelation, as it simply duplicates the nearest pixels, resulting in sharp but rough edges without additional detail. Bilinear interpolation somewhat improves the smoothness of the image but remains blurry and doesn't add any new information, which is evident in the lost details of the nose and surrounding areas.

Bicubic interpolation provides better smoothness and fewer artifacts compared to the bilinear method, but still retains some degree of blurriness. The Lanczos method shows slightly better sharpness and detail preservation but can sometimes create a ringing effect around the edges. The most impressive results are demonstrated by the GAN method implemented in this work. This method not only interpolates pixels but also attempts to restore textures and fine details, making the image more realistic and natural. Compared to other methods, the image obtained with GAN has the best sharpness, more natural contours, and textures. The dog's nose looks significantly more detailed than with other methods, lacking the coarse pixel artifacts characteristic of the nearest-neighbor method and reducing the blurriness observed in bilinear and bicubic interpolations.

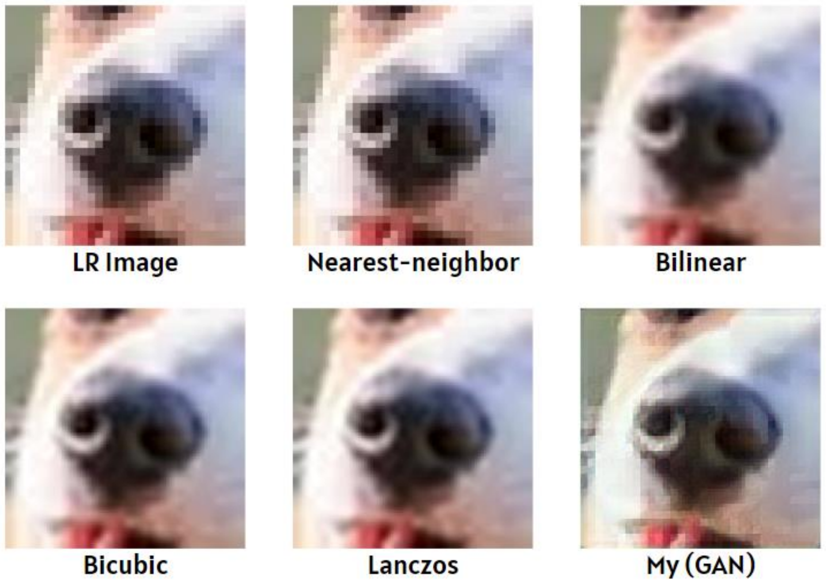


Fig. 7 Dog's nose

Let's consider another image. This set of images shows the results of upscaling using different methods, featuring a part of a cat's ear. The initial low-resolution image (LR Image) demonstrates the cat's ear with noticeable pixelation and blurriness, making it difficult to distinguish individual fur strands.

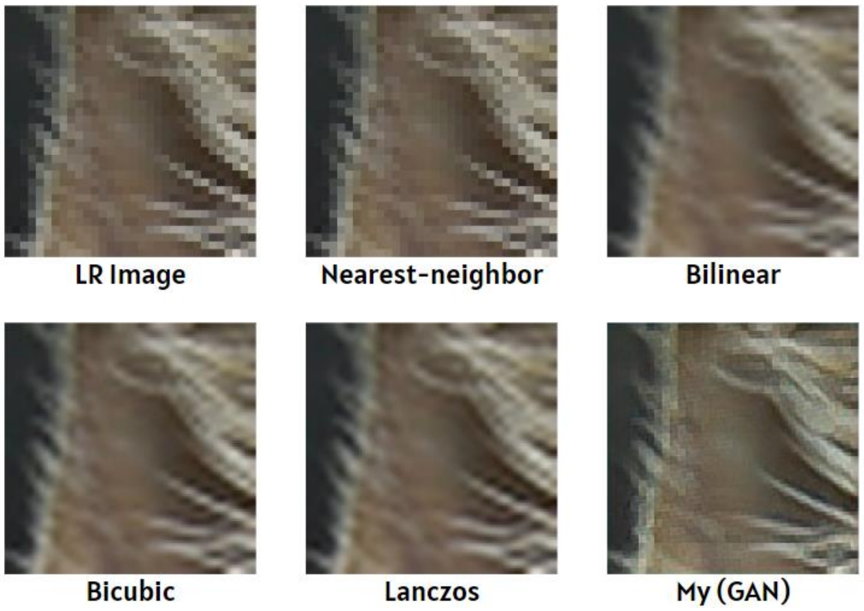


Fig. 8 Cat's ear

The nearest-neighbor method preserves pixel sharpness but creates rough, clear edges without additional details, resulting in a loss of texture and natural appearance of the cat's fur. Bilinear interpolation makes the image smoother, but the fur strands remain blurry and less defined, which does not improve the overall image quality.

Bicubic interpolation provides better smoothness than bilinear and slightly improves the appearance of individual details, but it still cannot restore the fine details of the fur strands. The Lanczos method offers better sharpness and detail compared to the previous methods, but a certain degree of blurriness remains, and the fur strands do not look realistic enough.

The GAN method shows a significant advantage over the other methods. It not only scales the image but also restores details with high precision. The image obtained using GAN has clear and natural contours of the fur strands, which look more realistic and detailed. This is especially noticeable in the individual strands, which retain their individuality and texture, unattainable by other methods. GAN eliminates blurriness and adds depth and textural complexity to the image, making it more natural.

Conclusion. Classical methods, such as bicubic and bilinear interpolation, remain popular due to their simplicity and speed, but they are limited in the quality of the results they produce. Modern approaches, especially those based on artificial intelligence and neural networks, achieve better results in detail restoration and overall image quality improvement. In particular, the use of Convolutional Neural Networks (CNNs) and Generative Adversarial Networks (GANs) is promising. Hybrid methods that combine classical and modern approaches can provide optimal results. For instance, combining interpolation methods with neural network approaches allows for achieving high accuracy and processing speed. The current state of research in image scaling is characterized by the active development of new AI-based technologies. Further research and the development of new methods remain relevant to ensure high-quality scaled images in the context of increasing volumes of visual content and higher demands for its processing.

Examining the aspects of image scaling highlights its crucial role in fields such as computer graphics, web development, medicine, and astronomy, where it ensures accurate reproduction and analysis of visual data. In these fields, scaling helps achieve optimal display and provides a better user experience.

It has been established that existing scaling methods, though effective, have limitations such as loss of details or artifacts. Therefore, continuous improvement of scaling algorithms is important to meet growing demands. The advantages of scaling lie in its flexibility and detail preservation, but drawbacks such as artifacts and quality loss need to be avoided. The further development of scaling technologies promises the use of artificial intelligence to create more accurate algorithms.

Special attention has been given to the application of Generative Adversarial Networks (GANs) as one of the advanced and effective approaches. Analyzing various loss functions has revealed their critical role in model training. The content loss function helps preserve key image details, while the adversarial loss function enhances image realism by training the generator to compete with the discriminator.

The architecture of the generator, responsible for creating scaled images, and the discriminator, which evaluates their quality, has been described. The structure and principles of the generator's operation ensure efficient scaling with high accuracy and detail. The discriminator, in turn, has been examined as a key element in distinguishing between real and synthesized images.

Based on the obtained results, it can be concluded that the GAN method proves significantly more effective compared to traditional image scaling methods, providing higher quality and detail in the final result. The GAN method implemented in this work showed significantly better results even in conveying the finest details, such as hair on a cat's fur. The images produced by the developed method turned out to be significantly more realistic and visually appealing compared to those obtained using classical image scaling methods.

References

[1] Chetan Suresh, Ravi Saini, Sanjay Singh, Anil Kumar Saini. A Comparative Analysis of Image Scaling Algorithms. 2013. URL: https://doi.org/10.5815/ijigsp.2013.05.07
[2] Chao Dong, Chen Change Loy, Kaiming He, Xiaoou Tang. Image Super-Resolution Using Deep Convolutional Networks. 2014. URL: https://doi.org/10.48550/arXiv.1501.00092
[3] Christian Ledig, Lucas Theis, Ferenc Huszar, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, Wenzhe Shi. Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network. 2017. URL: https://doi.org/10.48550/arXiv.1609.04802
[4] Justin Johnson, Alexandre Alahi, Li Fei-Fei. Perceptual Losses for Real-Time Style Transfer and Super-Resolution. 2016. URL: https://doi.org/10.48550/arXiv.1603.08155
[5] Karen Simonyan, Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. 2014. URL: https://doi.org/10.48550/arXiv.1409.1556

Масштабування зображень з використанням генеративних змагальних мереж

Ярослав Лис, Адріан Наконечний

Мета цієї статті полягає у розробці та вдосконаленні алгоритмів масштабування зображень, спрямованих на збереження деталей і візуального вигляду при зміні розміру зображення. Об'єктом дослідження цієї статті є алгоритми масштабування зображень, зокрема їх вдосконалення для забезпечення отримання високоякісних зображень. Проведено аналіз артефактів, які виникають під час масштабування зображень. Додатково проаналізовано різноманітні алгоритми масштабування зображень. Особлива увага приділяється алгоритмам і методам масштабування зображень, які використовують машинне навчання і нейронні мережі. У цій статті запропоновано програмну реалізацію методу масштабування зображень за допомогою генеративних змагальних мереж (GAN). Детально описано архітектуру GAN і обчислено функції втрат для генератора і дискримінатора. Результати цієї статті показують, що вдосконалені алгоритми на основі GAN дозволяють досягти високоякісного масштабування зображень з мінімальною втратою деталей. Програмна реалізація алгоритмів показала ефективність і можливість масштабування зображень в реальному часі, що є важливим для різноманітних практичних застосувань, таких як медична візуалізація, обробка відео та інші області, де якість зображення є критичною.

Ключові слова: обробка зображень, масштабування зображень, нейронні мережі, генеративні змагальні мережі, генератор, дискримінатор.

Отримано 06.07.2025