Report Assignment 3

Course: Generative AI Models (2AMU20)
Responsible Lecturer: Jakub Tomczak

Full Name	Student ID
Francesco Brescia	2118874
Luca Mainardi	2014602
Matteo Nana	1867512

Eindhoven, July 11, 2024

1 | Introduction

In recent years, generative deep learning models have demonstrated remarkable potential in various image processing tasks, such as super-resolution and colorization. These models enhance and transform images, making them visually appealing and detailed, which is crucial for applications ranging from photography to medical imaging.

This project aims to convert low-resolution grayscale images into high-resolution colored images by combining two advanced generative models. Specifically, we utilize a conditional Generative Adversarial Network (cGAN) for the colorization process and a Standard Generative Adversarial Network (GAN) for the super-resolution task. By integrating these two models, we seek to transform 64x64 grayscale images into 256x256 colored images, effectively enhancing both resolution and color quality.

For the colorization model, we trained on a subset of the COCO dataset, comprising 10,000 images. The COCO dataset, known for its diverse and complex scenes, provides an excellent basis for training a robust colorization model. While, for the super-resolution task we train the model on the PASCAL dataset, also using 10,000 images, to improve the spatial resolution of the colorized images.

Despite achieving notable improvements, our models still exhibit artifacts and lack vibrancy in colors, indicating a wide range for further refinement. To evaluate the performance of our models, we conducted both quantitative and qualitative experiments. The quantitative assessment was conducted using the Structural Similarity Index (SSIM) to compare the predicted images with the ground truth images, this metric provides an objective measure of similarity between the images. Additionally, a qualitative evaluation through visual inspection helped identify defects and areas for improvement.

2 | Literature Review

This chapter aims to provide a comprehensive overview of the current state-of-the-art in image super-resolution and colorization. The review is primarily based on two seminal papers: "A Deep Journey into Super-resolution: A Survey" [1] and "Deep Learning for Image Colorization: Current and Future Prospects" [13].

2.1 | Image colorization

Image colorization, the process of adding RGB color values to grayscale images, plays a crucial role in improving visual perception and is widely applied in various fields of computer vision, such as image recognition and object detection. It is also essential in specialized areas like animation scene design, historical photograph restoration, infrared imaging, and remote sensing. Our project focuses primarily on the natural colorization of images, which can be categorized into indoor and outdoor scenes. Indoor scenes include environments like bedrooms and restaurants, while outdoor scenes encompass natural environments such as the sky, ocean, mountains, grasslands, and deserts, as well as man-made structures like buildings, billboards, and transportation. This category also includes living beings like humans and animals. Natural image colorization is challenging due to the variety of objects and scenes it includes, this methods often require large datasets of real color images for training, which are difficult to obtain. Additionally, colorizing historical images is complex due to the difficulty of capturing realistic historical colors and details, given the lack of large reference datasets and the extinction of objects that existed in old photographs.

2.1.1 | RGB vs Lab

As you might know, when we load an image, we get a rank-3 (height, width, color) array with the last axis containing the color data for our image. These data represent color in RGB color space and there are 3 numbers for each pixel indicating how much Red, Green, and Blue the pixel is.

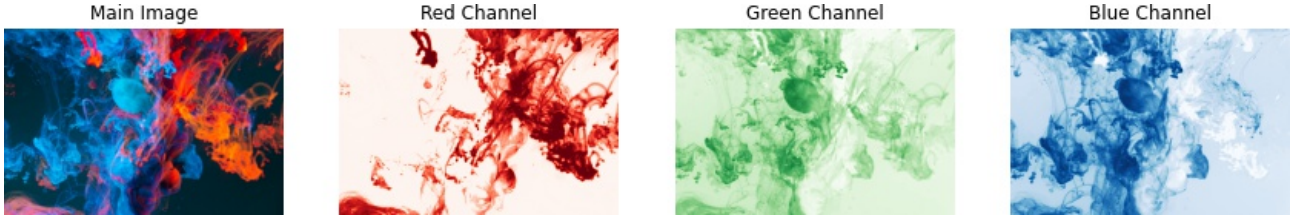

Figure 2.1: RGB Space

In Lab color space, we have again three numbers for each pixel but these numbers have different meanings. The first number L, encodes the lightness of each pixel, ranging from 0 (black) to 100 (white). The a and b channels represent chromaticity, where negative a* signifies green, and positive a* signifies red. Similarly, negative b* denotes blue, while positive b* denotes yellow [3].

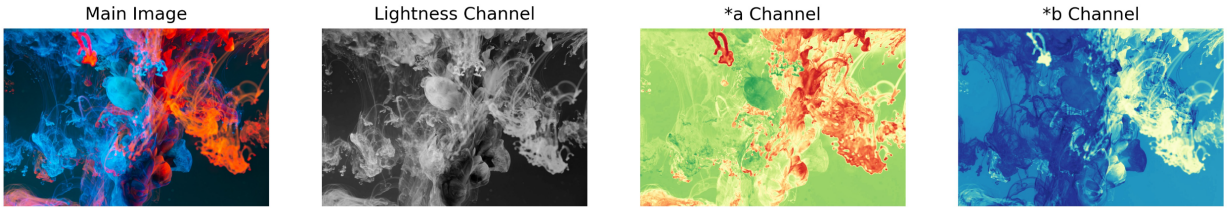

Figure 2.2: Lab Space

It's important to say that in almost all the papers that deal with colorization task, researchers decided to use the Lab color space instead of RGB space to train the models and there are a couple of reasons for this choice. To train a model we should give it a grayscale image and hope that it will make it colorful. When using Lab, we can give the L channel to the model (which is the grayscale image) and want it to predict the other two channels (a, b) and after its prediction, we concatenate all the channels and we get our colorful image. But if you use RGB, you have to first convert your image to grayscale, feed the grayscale image to the model and hope it will predict 3 numbers for you which is a way more difficult and unstable task due to the many more possible combinations of 3 numbers compared to two numbers. If we assume we have 256 choices (in a 8-bit unsigned integer image this is the real number of choices) for each number, predicting the three numbers for each of the pixels is choosing between 256^3 combinations which are more than 16 million choices, but when predicting two numbers we have about 65000 choices.

As concerns the network design, researchers employ various strategies to build network structures over time. These methods can be classified into CNN-based, GAN-based and Transformer-based.

2.1.2 | CNN-based methods

With the advancement of deep learning, CNNs have significantly impacted image processing due to their strong feature learning capabilities. CNN-based colorization methods are becoming increasingly prevalent and achieving notable results. Cheng et al. [5] proposed the first deep learning image colorization (DLIC) method, which divides reference images into clusters using an adaptive graph clustering method and trains a colorization model for each cluster. For a given grayscale image, the method identifies the closest clusters and corresponding pre-trained models. Feature descriptors for each pixel are input into the neural network to output the chromaticity, which is then combined with grayscale pixel values to produce the colorized image. A joint bilateral filter is used for final adjustments. Although effective for natural images, this method's reliance on reference images limits its application[10].

2.1.3 | GAN-based methods

Following the advancements in CNNs, generative adversarial networks (GANs) have shown great promise in the field of image colorization. Introduced in 2014, GANs have revolutionized computer vision with their powerful generative capabilities [6]. Evolving from the original model to conditional GANs (cGANs), CycleGANs, and most recently StyleGAN3, GAN-based colorization methods have achieved impressive results[11]. These methods vary primarily in the generator structure and loss function, often using classic generative models like cGANs and CycleGANs to enhance suitability for image colorization. Generator structures in GAN-based colorization models can be classified into single-stream, dual-stream, and multi-stream networks. While GANs have improved the generative quality, they still face challenges in retaining detailed features. Autoencoders (AEs), which consist of an encoder and a decoder, are often combined with other modules like ResNet blocks to enhance feature extraction capabilities in image colorization models [2]. However, the resulting colorized images may still lack satisfactory underlying features.

2.1.4 | U-Net methods

The U-Net architecture has also made significant contributions to image colorization [12]. Originally proposed for image segmentation, U-Net adds layer-by-layer connections between encoders and decoders, forming a U-shaped structure. In colorization models, skip connections help deconvolution processes by fusing low-level and high-level features, improving the quality of colorized images. Modules like ResNeXt and ResNet blocks are often integrated into U-Net structures. For instance, DenseNet blocks have been combined with U-Net for colorizing near-infrared face images, while Lee and Lee used ResNeXt blocks in a U-Net structure for enhanced image colorization. The encoder-decoder structure tends to extract global features, which are more suitable for global shape transformations. However, local guidance is crucial in image colorization to accurately separate target boundaries. The conditional concatenation module inputs auxiliary information (e.g., text descriptions, noise, category labels) layer by layer in the generative network to enhance color diversity. For example, grayscale images have been used as conditions in each layer of the network for continuous supervision, and Gaussian noise has been introduced into each layer to increase color diversity.

2.1.5 | Transformer-based methods

Building upon the foundation laid by CNNs, GANs, and U-Nets, transformer-based methods have also been explored for image colorization. Transformers, originally proposed by Vaswani et al. [14] as a sequence-to-sequence model for machine translation, have shown remarkable performance in a variety of tasks, including computer vision, audio processing, and even in fields like chemistry and life sciences. Manoj et al. [8] introduced Transformers into the image colorization task for the first time, achieving high-fidelity image colorization. Their method uses a conditional self-regression Transformer to generate a low-resolution coarse-colorized image, which is then up-sampled through two completely parallel networks to obtain high-resolution fine color images. More recently, Casey et al. [4] introduced the Animation Transformer (AnT) for the colorization of animation images. This method uses a Transformer-based architecture to learn the spatial and visual relationships between segments across a sequence of images, providing a practical and advanced AI-assisted colorization method for professional animation workflows.

2.1.6 | Future challenges

Despite significant advancements, several challenges remain in the field of image colorization. Ensuring color consistency, handling diverse and complex scenes, and reducing artefacts are ongoing areas of research. Future directions include improving model robustness, developing self-supervised learning techniques, and exploring the use of larger and more diverse datasets.

2.2 | Super-resolution

Image super-resolution (SR) is the process of converting low-resolution images into high-resolution images with better visual quality and refined details. It plays a crucial role in various applications, including large computer displays, HD televisions, mobile devices, object detection in scenes, face recognition in surveillance videos, medical imaging, remote sensing, astronomical imaging, and forensics. Super-resolution is a challenging problem due to its ill-posed nature, where multiple solutions can exist for the same low-resolution image. Reliable prior information is typically required to constrain the solution space. Additionally, higher up-scaling factors increase the complexity of recovering missing details, making the assessment of output quality non-trivial. Recent advances in deep learning have significantly enhanced the performance of SR models, particularly through the use of Convolutional Neural Networks (CNNs).

2.2.1 | Residual learning methods

Various strategies have been developed to improve the performance and efficiency of these models. One such strategy is residual learning, which utilizes skip connections in the network design to prevent gradients from vanishing, enabling the construction of very deep networks. This approach has significantly improved SR performance by learning the high-frequency details between the low-resolution input and the high-resolution ground truth. Residual learning approaches can be categorized into single-stage and multi-stage networks. Enhanced Deep Super-Resolution (EDSR) and Cascading Residual Network (CARN) are notable examples of single-stage networks. EDSR modifies the ResNet architecture to remove Batch Normalization layers and ReLU activation outside residual blocks, achieving substantial improvements. However, it requires significant computational resources. CARN employs ResNet Blocks with local and global cascading modules, enhancing

information propagation through multi-level representation and numerous shortcut connections, making it efficient but complex in design.

Building on the concept of residual networks, multi-stage residual networks involve multiple subnets trained in succession, offering further refinements. Examples include FormResNet and Balanced Two-Stage Residual Networks (BTSRN). FormResNet builds upon DnCNN, using a formatting layer with Euclidean and perceptual loss followed by a DiffResNet layer similar to DnCNN, providing marginal improvements but with increased complexity. BTSRN uses a low-resolution stage followed by a high-resolution stage, employing residual blocks with 1×1 convolutional layers for feature map projection, effectively balancing performance and resource usage.

2.2.2 | Recursive networks

Another innovative approach are recursive networks, which use either recursively connected convolutional layers or recursively linked units to progressively break down the SR problem into simpler sub-tasks. Deep Recursive Convolutional Network (DRCN) and Deep Recursive Residual Network (DRRN) are prominent examples. DRCN recursively applies convolution layers, keeping the number of parameters constant, which is efficient but might struggle with very high-resolution tasks. DRRN combines residual learning with recursive blocks to achieve deep architectures with reduced complexity, maintaining a balance between depth and computational cost.

2.2.3 | Attention-based methods

Attention-based models introduce another layer of sophistication by selectively attending to certain features at each layer, allowing for more efficient feature representation and improved SR performance. Residual Channel Attention Network (RCAN) and Densely Residual Laplacian Attention Network (DRLN) are prime examples. RCAN uses a recursive residual design with channel attention mechanisms, resulting in better convergence and performance but higher computational complexity. DRLN employs a modular architecture with densely connected residual units and cascading connections, offering state-of-the-art results with a relatively lower number of convolutional layers but higher parameter count.

2.2.4 | GAN-bsed methods

Generative Adversarial Networks (GANs) take a different route by employing a game-theoretic approach where a generator creates SR images that a discriminator cannot distinguish from real high-resolution images. This results in high perceptual quality SR images, although the PSNR values might be lower. Notable GAN-based SR methods include SRGAN, EnhanceNet, SRFeat, and ESRGAN. SRGAN proposed using an adversarial objective function promoting SR outputs that lie close to the manifold of natural images. It uses a multi-task loss formulation combining MSE loss, perceptual similarity metric, and adversarial loss, achieving perceptually similar high-dimensional images but potentially introducing artifacts. EnhanceNet focuses on creating faithful texture details in high-resolution SR images. It employs perceptual loss and texture matching loss alongside adversarial training, resulting in realistic and perceptually better outputs, though it may create visible artifacts in highly textured regions. SRFeat uses an additional discriminator to generate high-frequency structural features rather than noisy artifacts, offering enhanced texture details but at a higher computational cost. ESRGAN builds upon SRGAN by incorporating dense blocks and an enhanced discriminator, achieving better visual results with global residual connections and dense connections, but it still lags in quantitative measures compared to some other methods.

2.2.5 | Future challenges

Despite significant progress, several open research questions remain. Incorporating priors can enhance SR performance in cases with limited training data, such as medical imaging. Developing new objective functions and perceptual metrics is essential, as existing measures like ℓ_1 and ℓ_2 do not always correlate with perceptual quality. Unified solutions for multiple degradations, unsupervised SR approaches, tackling higher SR rates, and handling arbitrary SR rates are promising research directions. Additionally, addressing real vs. artificial degradation is crucial for practical applications, as most existing models are trained on synthetically degraded images that do not match real-world scenarios.

3 | Approach

In this project, we propose a combined approach for image colorization and super-resolution using Generative Adversarial Networks (GANs). Specifically, we utilize a conditional GAN (cGAN) for the colorization task and

a GAN-based model for super-resolution. Our implementation draws on methodologies discussed in notable papers such as "Image-to-Image Translation with Conditional Adversarial Networks" by Isola et al. [7] and "Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network" by Ledig et al. [9].

3.1 | Image Colorization

Our approach to image colorization is inspired by the pix2pix framework, which uses a conditional Generative Adversarial Network (cGAN) to learn the mapping from a grayscale input image to a colored output image.

3.1.1 | Network Architecture

The architecture of our image colorization model is based on the U-Net structure, known for its effectiveness in image-to-image translation tasks. The U-Net architecture consists of an encoder-decoder structure with skip connections, allowing the network to capture both low-level and high-level features efficiently.

Generator: The generator network follows the U-Net architecture, which includes an encoder-decoder structure with skip connections. The encoder part of the network is responsible for downsampling the input grayscale image to a lower-dimensional feature space (in our case, due to the small dimensions of our initial image, they are just 3 steps). This is achieved thigh-resolutionough a series of convolutional layers, each followed by a ReLU activation function and a max-pooling layer. The encoder progressively reduces the spatial dimensions of the input while increasing the depth of the feature maps, capturing essential features necessary for colorization.

$$E(x) = f_L(\cdots f_2(f_1(x)))$$

where x is the input grayscale image, f_i represents the i-th convolutional layer followed by ReLU and max-pooling, and L is the total number of layers in the encoder.

The decoder part of the network is responsible for upsampling the encoded features back to the original image size, introducing color information in the process. This is accomplished thigh-resolutionough a series of transposed convolutional layers (also known as deconvolutional layers), each followed by a ReLU activation function. The skip connections from the encoder to the decoder ensure that fine details are preserved during the upsampling process.

$$D(z) = g_1(g_2(\cdots g_L(z)))$$

where z is the encoded feature map, g_i represents the i-th transposed convolutional layer followed by ReLU, and L is the total number of layers in the decoder.

Skip connections are used to concatenate the feature maps from the encoder to the corresponding layers in the decoder. This helps in preserving the spatial information lost during downsampling and improves the quality of the generated images.

$$s_i = \text{concat}(f_i(x), g_{L-i}(z))$$

where concat denotes the concatenation operation, $f_i(x)$ is the feature map from the i-th layer of the encoder, and $g_{L-i}(z)$ is the feature map from the $(L-i)$-th layer of the decoder.

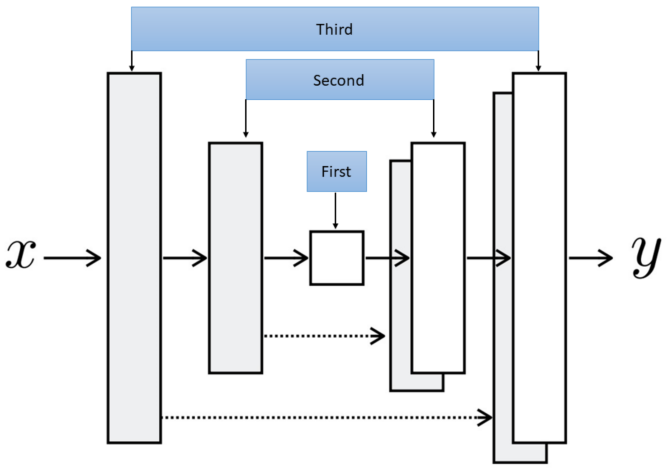

Figure 3.1: Colorization Generator

Discriminator: The discriminator network is designed to distinguish between real and generated color images. It takes as input both the grayscale image and the corresponding color image (either real or generated) and outputs a probability indicating whether the color image is real or fake. The discriminator architecture is a series of convolutional layers followed by LeakyReLU activations and batch normalization, which progressively downsample the input and extract high-level features for binary classification.

$$D(x, y) = \sigma(h_L(\cdots h_2(h_1(\text{concat}(x, y)))))$$

where x is the input grayscale image, y is the corresponding color image, h_i represents the i-th convolutional layer followed by LeakyReLU and batch normalization, and σ denotes the sigmoid activation function.

3.1.2 | Loss Functions

In this approach two losses are used: L1 loss, which makes it a regression task, and an adversarial (GAN) loss, which helps to solve the problem in an unsupervised manner.

As we know, in a GAN we have a generator and a discriminator model which learn to solve a problem together. In our setting, the generator model takes a grayscale image (1-channel image) and produces a 2-channel image, a channel for a and another for b. The discriminator takes these two produced channels and concatenates them with the input grayscale image and decides whether this new 3-channel image is fake or real. Of course the discriminator also needs to see some real images (3-channel images again in Lab color space) that are not produced by the generator and should learn that they are real.

It is of fundamental importance to add that since we are implementing a *conditional GAN* both the generator and discriminator should see a condition that influences them during the training phase, in our case, it is represented by the grayscale image.

The adversarial loss encourages the generator to produce realistic color images that can fool the discriminator. It is defined as:

$$\mathcal{L}_{\text{cGAN}} = \mathbb{E}_{\mathbf{x},\mathbf{y}\sim p_{data}(\mathbf{x},\mathbf{y})}[\log D(\mathbf{x}, \mathbf{y})] + \mathbb{E}_{\mathbf{x}\sim p_{data}(\mathbf{x}),\mathbf{z}\sim p_{\mathbf{z}}(\mathbf{z})}[\log(1 - D(\mathbf{x}, G(\mathbf{x}, \mathbf{z})))]$$

where x is the grayscale image (condition given to both models), z is the input noise for the generator, and y is the 2-channel output we want from the generator (it can also represent the 2 color channels of a real image). Also, G is the generator model and D is the discriminator.

The earlier loss function helps to produce good-looking colorful images that seem real, but to further help the models and introduce some supervision in our task, we combine this loss function with L1 Loss of the predicted colors compared with the actual colors:

$$\mathcal{L}_{\text{L1}}(G) = \mathbb{E}_{\mathbf{x},\mathbf{y},\mathbf{z}}[\|y_i - G(x, z)\|_1]$$

where N is the number of training examples, y_i is the ground truth color image, and $G(x, z)$ is the predicted color image generated by the network.

If we use L1 loss alone, the model still learns to colorize the images but it will be conservative and most of the time uses colors like "gray" or "brown" because when it doubts which color is the best, it takes the average and uses these colors to reduce the L1 loss as much as possible (it is similar to the blurring effect of L1 or L2 loss in super resolution task). Also, the L1 Loss is preferred over L2 loss (or mean squared error) because it reduces that effect of producing gray-ish images. So, our combined loss function will be:

$$G^* = \arg\min_G \max_D [\mathcal{L}_{\text{cGAN}} + \mathcal{L}_{\text{L1}}(G)]$$

3.1.3 | Training Procedure

The dataset used for training and evaluation is made of 10,000 images, from the COCO dataset, which contains images with diverse scenes. The dataset was split into 8,000 images for training and 2,000 images for validation, with an additional 20 images reserved for final evaluation. Images were resized to 64×64 and then converted from RGB to Lab space.

The training procedure involves several key steps to ensure effective learning and validation. The model architecture consists of a generator and a discriminator, both optimized using Adam optimizers with learning rates of 2×10^{-4} and betas set to 0.5 and 0.999. During training, for each batch of images (16 in this case), the grayscale L channel is passed thigh-resolutionough the generator to produce the colorized ab channels. The discriminator is then trained to distinguish between the real and fake colorizations. Specifically, the fake images (concatenation of L and generated ab channels) and real images (concatenation of L and ground truth ab channels)

are used to compute the discriminator loss, which is a sum of the losses from real and fake predictions using a mean square error loss. The generator is updated by freezing the discriminator's weights and backpropagating the combined cGAN loss and L1 loss (weighted by a factor of 100) thigh-resolutionough the generator. This process helps in refining the generator to produce more realistic colorizations.

Training progresses over multiple epochs (30 in this case), with both generator and discriminator losses monitored. Validation is performed after each epoch to evaluate model performance on unseen data. The best model is saved based on the lowest validation loss observed. Regular visualization of results from random validation batches ensures qualitative assessment, aiding in the early detection of overfitting or underfitting issues. This procedure effectively balances the adversarial training dynamics, promoting the generation of high-quality, realistic colorized images while ensuring stability and convergence of the GAN framework.

3.2 | Super-Resolution

Our approach to super-resolution is based on the SRGAN framework, which uses a Generative Adversarial Network (GAN) to learn the mapping from low-resolution images to high-resolution images.

3.2.1 | Network Architecture

The architecture of our super-resolution model is based on the SRGAN framework, which includes a generator and a discriminator network.

Generator: The generator network is designed to upscale low-resolution images and follows the architecture proposed in the Super-Resolution Generative Adversarial Network (SRGAN) framework.

The Generator network initiates with an initial convolutional layer which processes the input low-resolution image. The core of the Generator is composed of a series of residual blocks. Residual blocks are fundamental in deep learning models due to their ability to mitigate the vanishing gradient problem, allowing the network to be trained effectively even at great depths. Each residual block consists of two convolutional layers with small 3x3 kernels, followed by batch normalization and ReLU activations. The architecture of these blocks ensures that the network can learn complex representations while maintaining stability and ease of training. Mathematically, a residual block can be represented as:

$$\mathbf{y}_i = \mathbf{x}_i + \mathcal{F}(\mathbf{x}_i, \{W_{i1}, W_{i2}\})$$

where $\mathcal{F}$ denotes the function learned by the residual block.

Following the stack of residual blocks, the network includes an intermediate convolutional layer that serves to consolidate the features extracted by the residual blocks. Next, the Generator incorporates upsampling blocks to increase the spatial resolution of the feature maps. Each upsampling block consists of a transposed convolutional layer (also known as *Deconvolution*) to expand the spatial dimensions without losing the learned feature information. The mathematical operation of an upsampling block can be depicted as:

$$\mathbf{y}_{up} = \text{PReLU}(\text{PixelShuffle}(\text{Conv2D}(\mathbf{y}_{in}, W_{up})))$$

where $\mathbf{y}_{in}$ is the input to the upsampling block.

The final stage of the Generator network is a convolutional layer that adjusts the number of output channels to match the desired high-resolution image. This layer is followed by a sigmoid activation function which ensures that the output pixel values are within the appropriate range. The overall transformation performed by the Generator can be summarized as:

$$G(\mathbf{x}; \theta_G) = \sigma(\text{Conv2D}(\mathbf{y}_{up}, W_{final}) + b_{final})$$

where θ_G represents the collective parameters of the network.

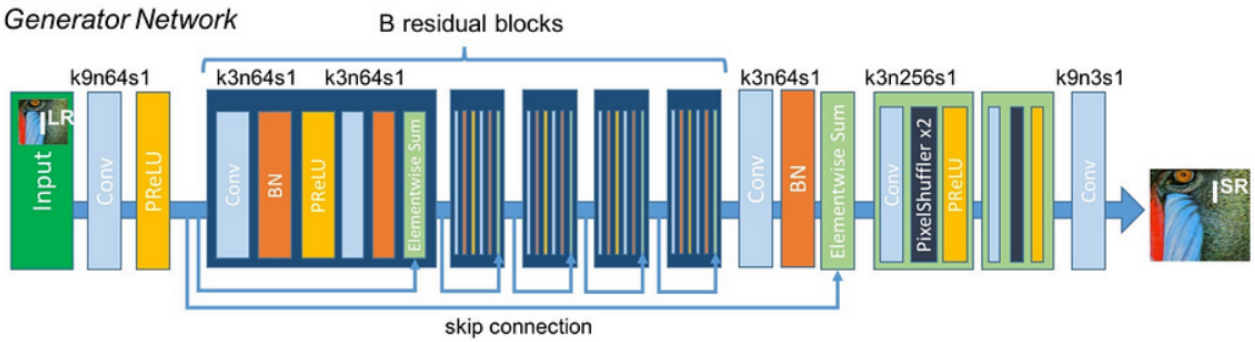

Figure 3.2: Super-Resolution Generator

Discriminator: The Discriminator's primary role is to distinguish between real high-resolution images and those generated by the Generator, thereby ensuring the Generator produces outputs that are indistinguishable from real images.

The Discriminator network begins by processing the input image thigh-resolutionough a series of convolutional blocks. Each convolutional block consists of a convolutional layer followed by batch normalization (except in the first block) and a LeakyReLU activation function. The design of these blocks allows the Discriminator to progressively extract and learn hierarchical features from the input image, which are crucial for effectively distinguishing real images from generated ones. This process can be mathematically represented as:

$$\mathbf{y}_i = \text{LeakyReLU}(\text{BN}(\text{Conv2D}(\mathbf{x}_{i-1}, W_i) + b_i))$$

where $\mathbf{y}_i$ is the output of the i-th block, $\mathbf{x}_{i-1}$ is the input to the i-th block, and W_i and b_i are the weights and biases of the convolutional layer, respectively.

The final part of the Discriminator consists of two fully connected layers. The first fully connected layer reduces the dimensionality of the flattened feature vector to 1024, followed by a LeakyReLU activation function. The second fully connected layer further reduces the dimensionality to a single value, representing the probability that the input image is real. The mathematical operations of the final layers can be expressed as:

$$\mathbf{z} = \text{LeakyReLU}(\mathbf{W}_{fc1}\mathbf{y} + \mathbf{b}_{fc1})$$
$$D(\mathbf{x}) = \sigma(\mathbf{W}_{fc2}\mathbf{z} + \mathbf{b}_{fc2})$$

where $\mathbf{W}_{fc1}$ and $\mathbf{b}_{fc1}$ are the weights and biases of the first fully connected layer, $\mathbf{W}_{fc2}$ and $\mathbf{b}_{fc2}$ are the weights and biases of the second fully connected layer, and σ denotes the sigmoid activation function.

3.2.2 | Loss Functions

To train the super-resolution model, we use a combination of adversarial loss and perceptual loss.

The adversarial loss encourages the generator to produce realistic high-resolution images that can fool the discriminator. It is defined as:

$$\mathcal{L}_{\text{GAN}} = \mathbb{E}_y[\log D(y)] + \mathbb{E}_x[\log(1 - D(G(x)))]$$

where x is the input low-resolution image, y is the real high-resolution image, and $G(x)$ is the generated high-resolution image.

The perceptual loss is based on the activations of a pre-trained VGG19 network. Instead of comparing pixel values directly, the perceptual loss compares high-level feature representations, ensuring that the generated images are perceptually similar to the ground truth. The perceptual loss is defined as:

$$\mathcal{L}_{\text{perceptual}} = \mathbb{E}_{\mathbf{x},\mathbf{y}}[\|\phi(G(\mathbf{x})) - \phi(\mathbf{y})\|_2^2]$$

where ϕ denotes the feature extraction function of the VGG19 network, $G(\mathbf{x})$ is the generated high-resolution image, and $\mathbf{y}$ is the ground truth high-resolution image.

The total loss for the Generator during training is a combination of the adversarial loss and the perceptual loss:

$$G^* = \arg\min_G \max_D [\mathcal{L}_{\text{perceptual}} + \lambda \mathcal{L}_{\text{GAN}}]$$

where λ is a scaling factor that balances the importance of the GAN loss relative to the perceptual loss. In our implementation, λ is set to 10^{-3}.

3.2.3 | Training Procedure

The dataset used for training and evaluation is the PASCAL VOC dataset, from which we selected 10,000 images with diverse scenes. The dataset was split into 8000 images for training, 2000 images for validation. Additionally, we chose 20 images for testing. Images were down sampled to create low-resolution-high-resolution pairs for training the super-resolution model (in this case 64×64 - 256×256).

The training procedure involves several key steps to ensure effective learning and validation. We begin by initializing the generator and discriminator networks, optimizing them using the Adam optimizer with a learning rate of 2×10^{-4} and betas of 0.5 and 0.999. During each training epoch, we process batches of low-resolution images through the generator to produce high-resolution outputs. The training involves two key phases: updating the generator and updating the discriminator.

For the generator update, we compute a weighted sum of the perceptual loss that combines with adversarial loss to ensure that the generated images are realistic and accurate, and this is then backpropagated to update the generator's parameters.

For the discriminator update, the discriminator evaluates both the real high-resolution images and the generated images. We calculate the discriminator loss as the sum of losses for real and fake predictions, which is then backpropagated to update the discriminator's parameters. This helps the discriminator become more adept at distinguishing between real and generated images, thus providing better feedback to the generator.

After each epoch, we validate the model using the validation set and at the same time save the model with the best performance based on the lowest validation loss.

4 | Empirical Study

In this section, we detail the experiments conducted to evaluate the performance of our combined image upscaling and colorization model. We compare our results with existing models and discuss both quantitative and qualitative assessments.

4.1 | Quantitative Evaluation

To rigorously evaluate the quality of the images generated by our combined colorization and super-resolution model, we employed the Structural Similarity Index Measure (SSIM). SSIM is a perceptual metric that quantifies image quality by comparing the structural similarity between the generated images and the original high-resolution images. This metric is particularly well-suited for our task as it assesses the visual impact of three characteristics of an image: luminance, contrast, and structure.

Before calculating SSIM, it is essential to preprocess the images to ensure consistency. Both the images generated by our model and the original high-resolution images are resized to 64×64 pixels. Additionally, the images are normalized to maintain a consistent comparison basis.

The SSIM index is computed as follows:

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

where x and y are the generated and high-resolution images, respectively. μ_x and μ_y represent the mean intensity values of x and y, σ_x^2 and σ_y^2 denote the variances of x and y, and σ_{xy} indicates the covariance of x and y. Constants c_1 and c_2 are included to stabilize the division with weak denominators and are typically set to $c_1 = (k_1 L)^2$ and $c_2 = (k_2 L)^2$, where L is the dynamic range of the pixel values (e.g., 255 for 8-bit grayscale images) and k_1 and k_2 are small constants (e.g., $k_1 = 0.01$, $k_2 = 0.03$).

The SSIM index provides a value between -1 and 1, where 1 indicates perfect structural similarity between the generated and original images. Higher SSIM values correspond to images that are more perceptually similar to the reference images.

We selected SSIM for our evaluation because it considers perceptual phenomena, such as the human visual system's sensitivity to structural information. Unlike simple pixel-by-pixel comparisons, SSIM focuses on the perceptual similarity between images, making it a more relevant and robust metric for assessing the quality of both colorization and super-resolution.

By focusing on the structural similarities between the images, we ensure that the evaluation is aligned with human visual perception, leading to a more accurate assessment of the generated images' quality.

The results of this evaluation on some images from the test set are reported in Figures 4.1 to 4.4.

Figure 4.1: Generated image of an interior room with SSIM score of 0.615.

Figure 4.2: Generated image of an interior room with SSIM score of 0.537.

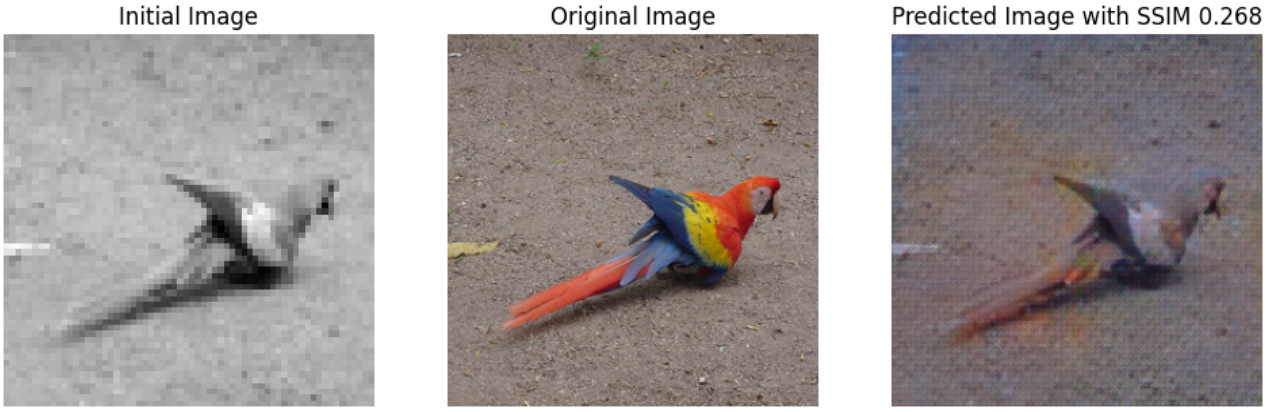

Figure 4.3: Generated image of an animal with SSIM score of 0.268.

Figure 4.4: Generated image of an animal with SSIM score of 0.246.

The SSIM scores indicate a varying degree of success in generating high-quality images. The first two images, depicting interiors of rooms, achieved SSIM scores of 0.615 and 0.537, respectively. These scores suggest that our model performs reasonably well in scenarios with well-defined structural features and consistent lighting conditions, typical of indoor scenes. The generated images maintain a good level of structural similarity to the original high-resolution images, with clear edges and recognizable objects.

In contrast, the third and fourth images, depicting animals, have significantly lower SSIM scores of 0.268 and 0.246, respectively. These lower scores highlight some of the model's weaknesses. The artifacts present in these images, indicate challenges in accurately colorizing and super-resolving images with complex textures and varying lighting conditions.

These results indicate that while our combined model can produce structurally coherent images in some cases, there is room for improvement, especially in handling complex textures and ensuring accurate colorization. Future work could focus on enhancing the model's ability to learn and reproduce fine details and textures, as well as improving the colorization process for more natural and realistic results.

4.2 | Qualitative Evaluation

In addition to quantitative evaluation, it is crucial to perform a qualitative assessment of the images generated by our combined colorization and super-resolution model. This section presents several examples to illustrate the strengths and weaknesses of our approach. To provide a comprehensive evaluation, we have selected a variety of images that highlight different aspects of the model's performance. Each figure demonstrates specific strengths and areas for improvement.

4.2.1 | Successful Prediction

Figures 4.5 and 4.6 show examples of successfully generated images. The colors are natural, and the details are well-preserved, demonstrating the model's capability to produce high-quality outputs.

Figure 4.5: Generated image of a cat with natural colors and well-preserved details

Figure 4.6: Generated image of a room interior with realistic colors and good level of detail

4.2.2 | Minor Artifact in Center

Figures 4.7 and 4.8 presents an image where a minor artifact is visible in the center. There is a noticeable shade of colors that slightly detracts from the overall quality but does not significantly ruin the image.

Figure 4.7: Generated image of an animal with SSIM score of 0.556

Figure 4.8: Generated image of an animal with SSIM score of 0.554.

4.2.3 | Significant Artifact in Center

In Figure 4.9, a significant artifact is present in the center of the image. Bright colors such as blue and purple appear unnaturally, ruining the image quality and highlighting a failure case of the model.

Figure 4.9: Image with a significant artifact in the center. Bright colors like green and purple appear unnaturally, some uncertainty with the colors.

4.2.4 | Issues with Vegetation Coloring

One consistent issue observed is the improper coloring of plants and grass. In Figures 4.10 and 4.11, the vegetation appears in incorrect colors or very light and faded green, failing to accurately represent natural scenes.

Figure 4.10: Image highlighting issues with vegetation coloring. The plants and grass appear in incorrect colors or very light and faded green.

Figure 4.11: Image highlighting issues with vegetation coloring.

4.2.5 | Discussion

The qualitative evaluation reveals several key insights into the performance of our model. The successful prediction demonstrates the model's potential to generate high-quality, realistic images. However, the presence of artifacts, especially in the center of some images, indicates areas where the model struggles. The minor artifacts may be due to subtle errors in the colorization or super-resolution processes, while significant artifacts suggest potential issues with the network's training or architecture.

The consistent problem with vegetation coloring points to a specific challenge in the model's ability to accurately colorize natural scenes. This could be due to the lack of sufficient training data representing diverse types of vegetation or limitations in the model's ability to generalize across different green hues.

5 | Conclusions

In this paper, we presented a comprehensive approach to image colorization and super-resolution by combining Conditional Generative Adversarial Networks (cGANs) and Super-Resolution Generative Adversarial Networks (SRGANs). Our method integrates the strengths of both cGANs for realistic colorization and SRGANs for enhancing image resolution, leveraging the powerful feature extraction capabilities of deep convolutional neural networks.

We implemented our image colorization model based on the pix2pix framework, which employs a cGAN to map grayscale images to color images, conditioning the generator on the input grayscale image to produce realistic results. The super-resolution model was implemented following the SRGAN framework, which introduces a perceptual loss function to generate photorealistic high-resolution images. Both models were trained and evaluated on the COCO and DIV2K datasets, respectively.

Our quantitative evaluation utilized the VGG19 network to extract perceptually significant features, allowing us to compare the super-resolved and original high-resolution images more effectively. This approach provided a robust basis for evaluating the quality of our generated images, aligning well with human visual perception.

Qualitative evaluation revealed the model's capability to produce high-quality images, though certain artifacts and inaccuracies were noted. Specifically, minor and significant artifacts were observed in some images, and consistent issues with vegetation coloring highlighted areas for improvement.

5.1 | Future Work

While our approach has demonstrated promising results, there are several avenues for future work to enhance the performance and applicability of our models. One of the primary limitations of our current approach is the size of the dataset. Increasing the dataset size beyond the current 10,000 images will provide more diverse training examples, enabling the model to generalize better and produce more accurate results. Collecting a larger and more varied dataset will be a critical step forward.

Refining the model architecture can help mitigate the artifacts observed in the generated images. Exploring more advanced network designs, such as using attention mechanisms or deeper architectures, may improve the model's ability to capture intricate details and produce higher-quality images. Additionally, implementing more sophisticated data augmentation techniques can increase the diversity of the training data without the need for additional image collection. Techniques such as random cropping, rotation, color jittering, and synthetic data generation can help the model become more robust and versatile.

Incorporating domain-specific knowledge, such as understanding the typical colors of vegetation or other common objects, can help improve the colorization accuracy. Developing specialized loss functions or incorporating additional training objectives that focus on these domain-specific aspects can lead to better results. Fine-tuning the models with additional perceptual losses, such as the Learned Perceptual Image Patch Similarity (LPIPS) or other advanced perceptual metrics, can enhance the visual quality of the generated images. These metrics better align with human visual perception and can guide the model to produce more realistic and appealing images.

Exploring new evaluation metrics is another important direction for future work. Metrics such as Fréchet Inception Distance (FID), Perceptual Image Error (PIE), and Natural Image Quality Evaluator (NIQE) offer alternative ways to assess image quality. These metrics can provide insights into how well the images are perceived by humans and can help in fine-tuning the models to achieve better visual quality.

Finally, testing the models in real-world applications, such as film restoration or medical imaging, can provide valuable insights into their practical performance and limitations. Real-world testing can reveal specific challenges and areas where the models need further refinement.

References

[1] Saeed Anwar, Salman Khan, and Nick Barnes. "A Deep Journey into Super-resolution: A Survey". In: *ACM Computing Surveys* 53.3 (2020). DOI: 10.1145/3391406. URL: https://doi.org/10.48550/arXiv.1904.07523.

[2] Dor Bank, Noam Koenigstein, and Raja Giryes. "Autoencoders". In: *arXiv preprint arXiv:2003.05991* (2020). URL: https://doi.org/10.48550/arXiv.2003.05991.

[3] Robert R. Buckley and Edward J. Giorgianni. "CIELAB for Color Image Encoding (CIELAB, 8-Bit; Domain and Range, Uses)". In: *Encyclopedia of Color Science and Technology*. Ed. by Ming Ronnier Luo. New York, NY: Springer New York, 2016, pp. 213–221. ISBN: 978-1-4419-8071-7. DOI: 10.1007/978-1-4419-8071-7_14. URL: https://doi.org/10.1007/978-1-4419-8071-7_14.

[4] J. Casey et al. "Animation Transformer (AnT) for the colorization of animation images". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2021, pp. 2126–2135. URL: https://openaccess.thecvf.com/content/CVPR2021/papers/Casey_Animation_Transformer_AnT_for_the_Colorization_of_Animation_Images_CVPR_2021_paper.pdf.

[5] Z. Cheng, Q. Yang, and B. Sheng. "Deep colorization". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2015, pp. 415–423. DOI: 10.1007/s00521-018-3828-z.

[6] Ian Goodfellow et al. "Generative Adversarial Nets". In: *Advances in Neural Information Processing Systems (NeurIPS)* (2014), pp. 2672–2680.

[7] Phillip Isola et al. "Image-to-Image Translation with Conditional Adversarial Networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 1125–1134.

[8] Manoj Kumar, Dirk Weissenborn, and Nal Kalchbrenner. "Colorization Transformer". In: *International Conference on Learning Representations (ICLR)*. 2021. URL: https://arxiv.org/abs/2102.04432.

[9] Christian Ledig et al. "Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network". In: *arXiv preprint arXiv:1609.04802* (2017).

[10] Sudesh Pahal and Preeti Sehrawat. "Image Colorization with Deep Convolutional Neural Networks". In: *Advances in Communication and Computational Technology*. Ed. by Gurdeep Singh Hura, Ashutosh Kumar Singh, and Lau Siong Hoe. Vol. 668. Lecture Notes in Electrical Engineering. Springer, Singapore, 2021. DOI: 10.1007/978-981-15-5341-7_4.

[11] Ivan Miguel Pires, Petre Lameski, and Sonja Gievska. "GAN-Based Image Colorization for Self-Supervised Visual Feature Learning". In: *Sensors* 22.4 (2022), p. 1599. DOI: 10.3390/s22041599.

[12] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. "U-Net: Convolutional Networks for Biomedical Image Segmentation". In: *arXiv preprint arXiv:1505.04597* (2015). URL: https://doi.org/10.48550/arXiv.1505.04597.

[13] Naveed Siddique, Muhammad Imran, and Munir Ahmad. "Deep Learning for Image Colorization: Current and Future Prospects". In: *arXiv preprint arXiv:2101.10784* (2021). URL: https://doi.org/10.48550/arXiv.2101.10784.

[14] Ashish Vaswani et al. "Attention Is All You Need". In: *Advances in Neural Information Processing Systems* 30 (2017), pp. 5998–6008. URL: https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.