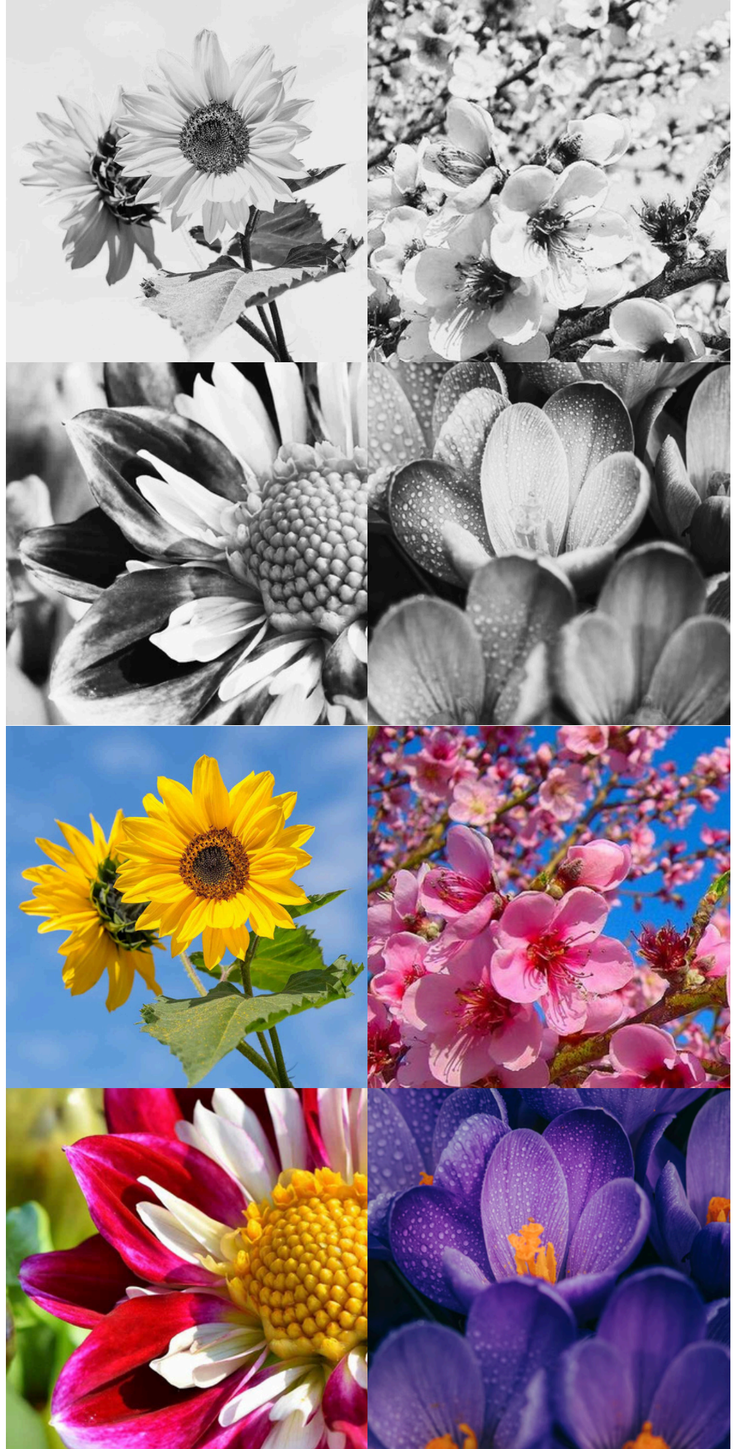
Image Colorization

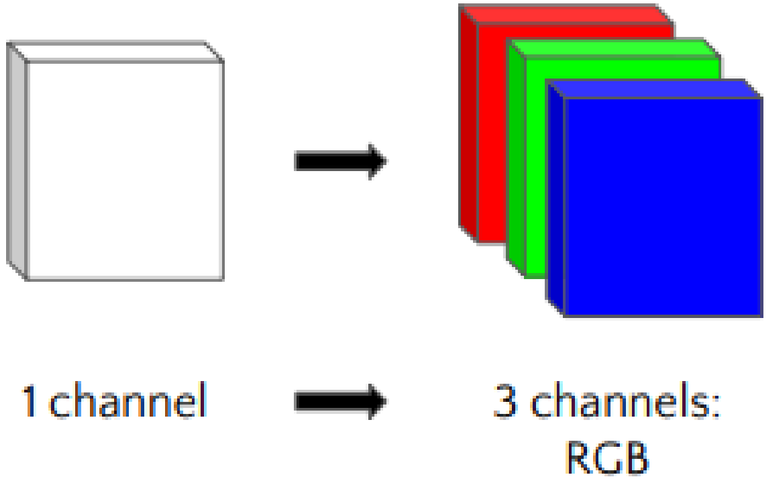
Final Project for Vision and Cognitive Systems
AY: 2023 - 2024

Sofia Bazzan
Diletta Pasin



Introduction

The image **Colorization** TASK involves predicting plausible color channels for a given grayscale image, transforming it into a realistic colored version. The main issue that makes this topic challenging is the multi-modal nature of the problem



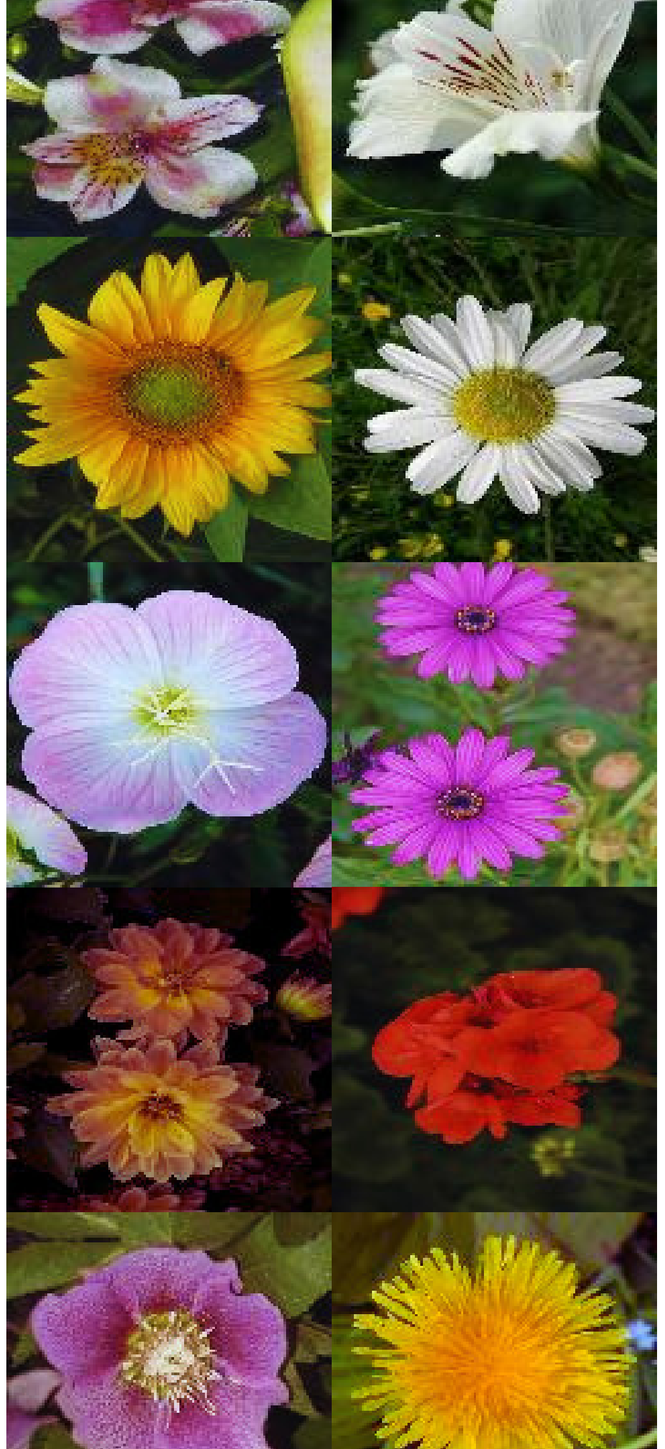
Dataset

Dataset

We used the the **102 Category Flower Dataset**, composed by 8189 colored images of flowers and plants with different dimensions (at least 500 x 500 px).

The images within the dataset exhibit considerable diversity in scale, pose, and lighting, capturing variations within categories and similarities across closely related ones.

The decision to focus on flowers for this task was influenced by the diverse forms and colors they exhibit, rendering them particularly intriguing for our purposes.



Data Preprocessing

Standardizing image dimensions to 128x128 pixels

Random hue shift

↓

Provide a varied representation of colors

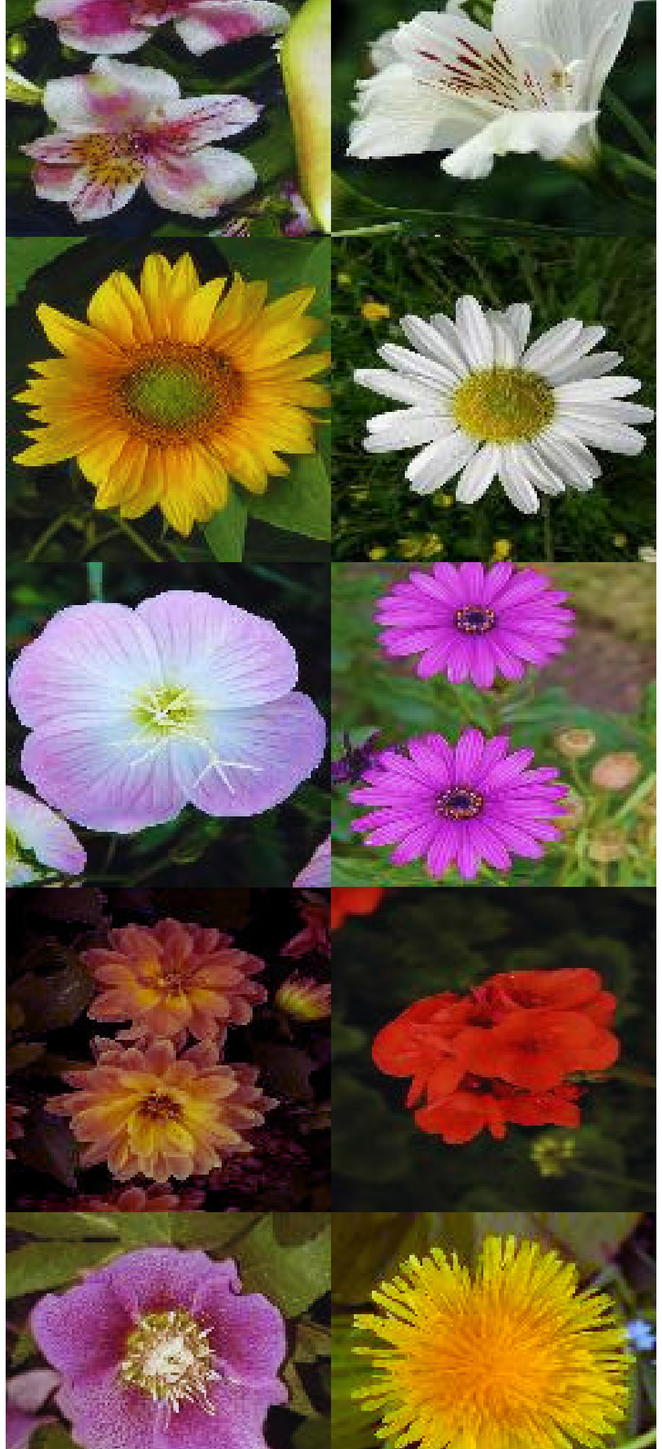
Pixel normalization

We divided our dataset into 60% for training, 20% for validation, and 20% for the test set.

Models

Models Structures

In order to find the best suited model for the problem, we tried many different popular architectures.



Convolutional Auto-Encoder

Deep CNN

U-net

ResNet

Dense Net

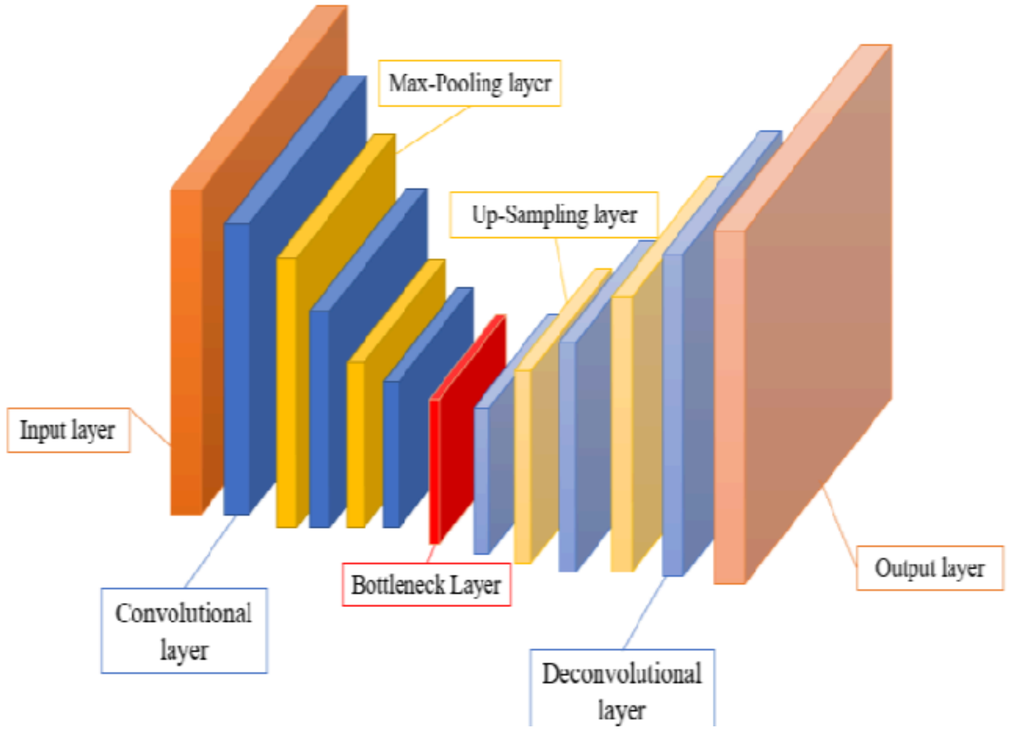
Pix2Pix

MLE based

In our image colorization task, we employ the **Mean Squared Error** (MSE) loss function to gauge the disparity between the colorized images generated by our model and the ground truth colored images. This choice of loss function is advantageous as it quantifies the **average squared difference** between the **predicted** and **actual color values**. By minimizing the MSE loss, our model strives to accurately reproduce the desired color distribution, leading to visually appealing and realistic colorized images. The MSE loss is a natural fit for image colorization tasks, providing a straightforward metric to optimize model performance and ensure faithful color reproduction.

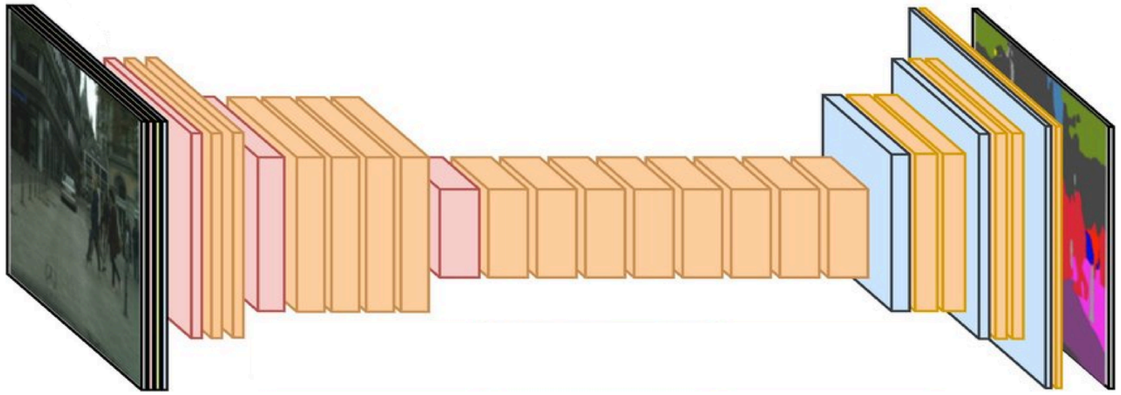
$$J(\theta) = \mathbb{E}_{x,y \sim p_{data}}[(f_\theta(x) - y)^2]$$

Convolutional Auto-Encoder



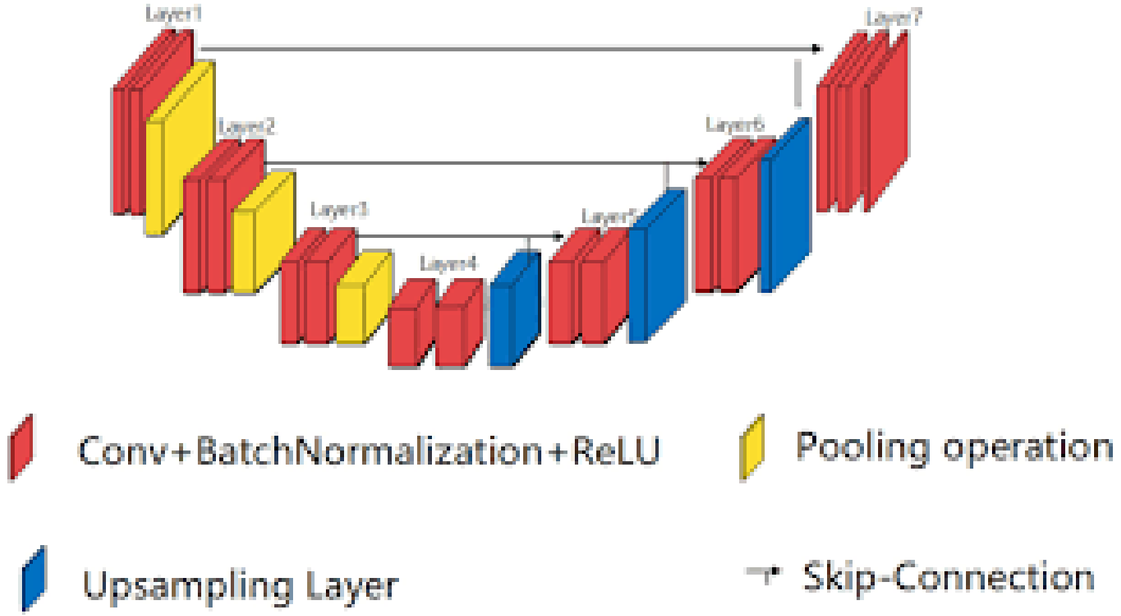
Our autoencoder comprises an encoder and a decoder. The **encoder** applies two blocks of **convolution**, **leaky ReLU**, and **max pooling** to compress the input black and white image. Meanwhile, the **decoder** initially performs **upsampling**, followed by blocks of **convolution**, **leaky ReLU**, and **batch normalization** to reconstruct the original image.

Deep CNN



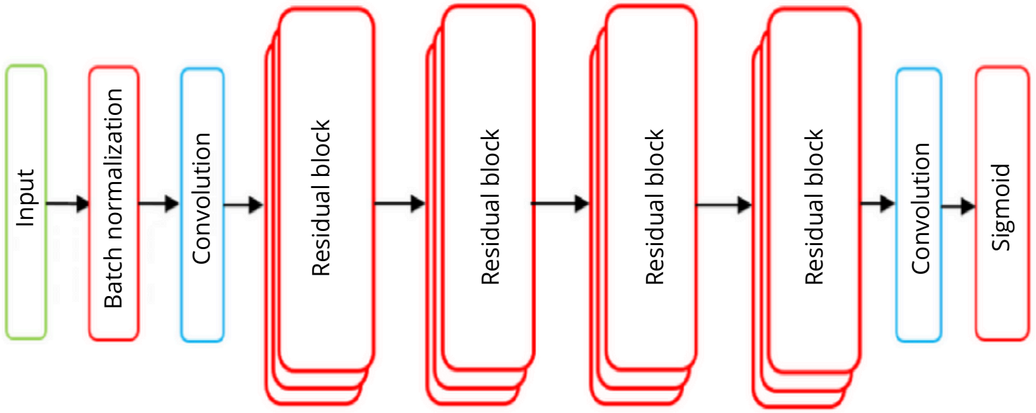
The Deep Convolutional Neural Network (CNN) model features a series of **repeated sequences** of **batch normalization**, **leaky ReLU**, and **convolutional layers**. These blocks are iterated multiple times to progressively extract features from the input. At the end, there is a **sigmoid** activation function applied to the output, which scales each channel's values between 0 and 1. Later, the images will be rescaled to represent RGB values.

U-Net



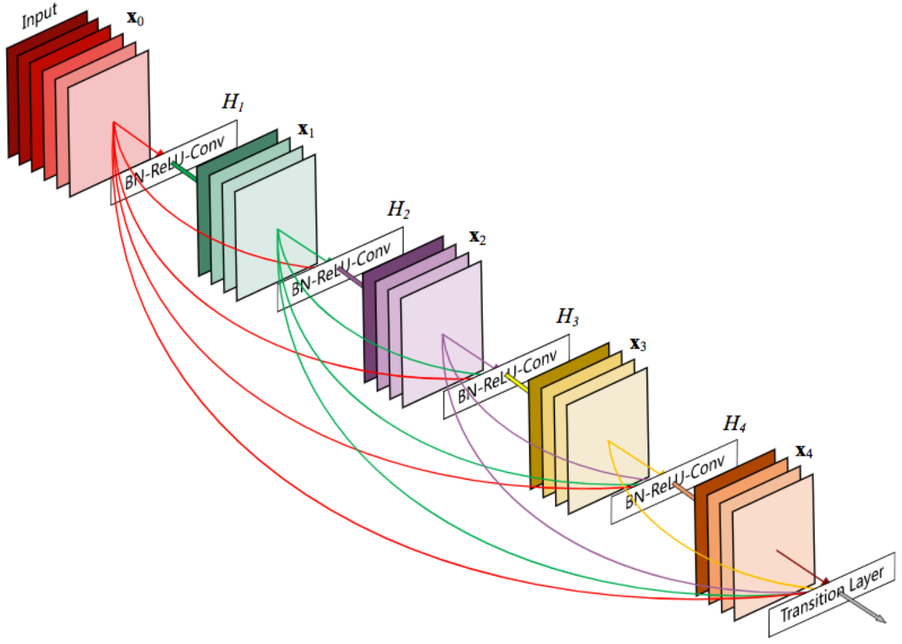
The UNet model architecture consists of a series of **repeated blocks**, each starting with a **convolution**, followed by **batch normalization** and **leaky ReLU** activation. Subsequently, **max pooling** is applied, and the output is combined with the original input using a **residual layer**. This process is **repeated multiple times**, followed by **upsampling** and another series of repeated blocks. Finally, the information is decoded using a **convolutional layer** with a **sigmoid** activation function.

Res-Net



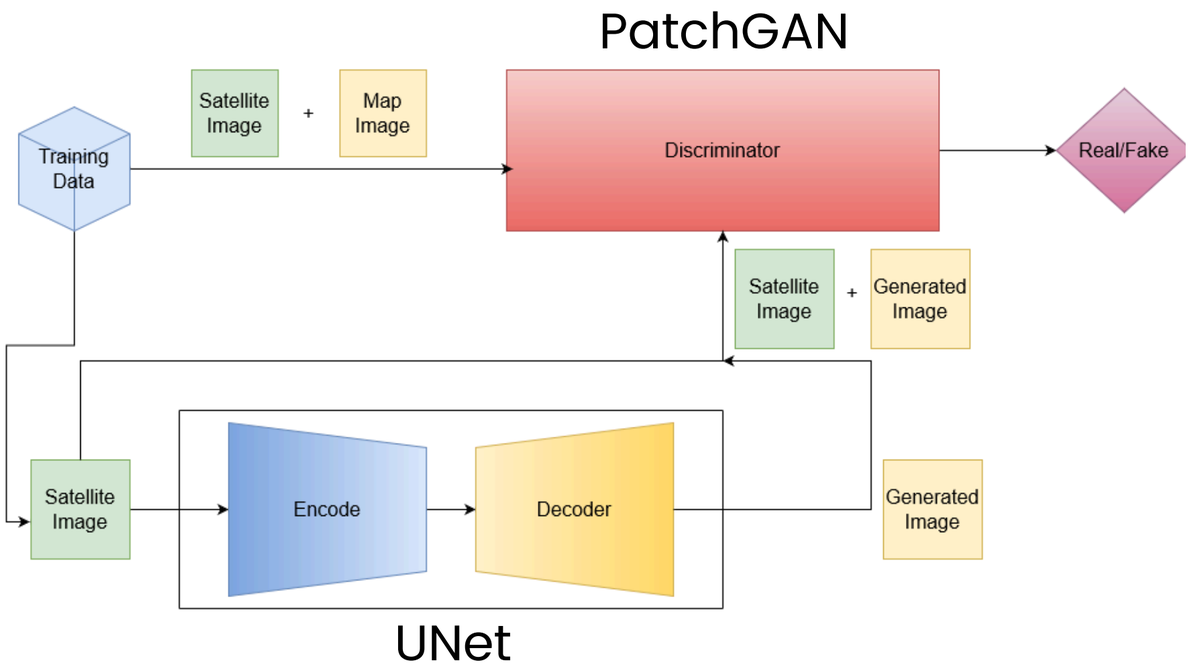
The ResNet model architecture is built on the concept of residual learning. It comprises a series of **residual blocks**, each containing **convolutional layers**, **batch normalization**, and **Leaky ReLU activation**. The key feature is the residual block, which includes the addition of the original input to the output of the convolutional layers. This design enables the model to learn the residual during training, addressing the vanishing gradient problem and facilitating the training of deep networks.

Dense Net



In our DenseNet implementation, **convolutional blocks** composed of convolution, batch normalization, and Leaky ReLU activation are **iteratively concatenated together**. **Each block receives input** from the **previous blocks**, allowing for extensive feature reuse and gradient flow throughout the network. This dense connectivity pattern ensures that each layer captures and incorporates information from all preceding layers, leading to a rich and expressive feature hierarchy.

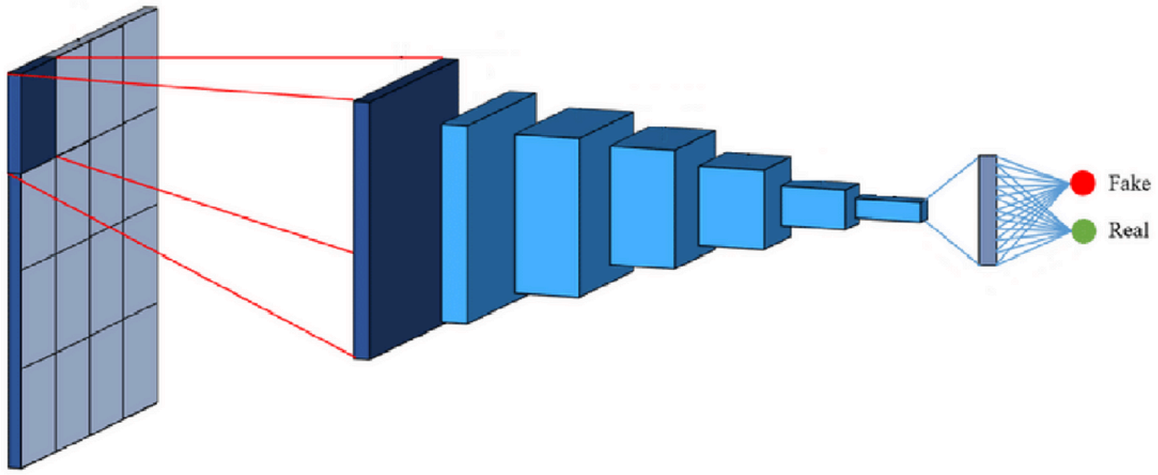
Pix2Pix



The Pix2Pix model is a conditional Generative Adversarial Network (cGAN), conditions the generation of the output image on an additional input, typically a b/w image.

Pix2Pix comprises a **generator** and a **discriminator**. The generator, implemented as a **UNet**, aims to produce **realistic output images** from input images. The discriminator, based on **PatchGAN**, **evaluates** the **realism** of the **generated images**, distinguishing between real and fake images to guide the training process.

PatchGAN



The **PatchGAN** discriminator employs a series of **convolutional layers** followed by **batch normalization** and **leaky ReLU** activation. It focuses on evaluating **local image patches** rather than entire images, providing detailed feedback to the generator. This approach enables the discriminator to effectively determine the realism of the generated images, enhancing the overall quality of the output.

Evaluation Metrics

Metrics for colorization evaluation

Regression Metrics:
MAE (Mean Absolute Error)
and MSE (Mean Squared Error)

PSNR (Peak Signal-to-Noise Ratio):
images are treated as signals, and the PSNR is then calculated

SSIM (Structural Similarity Index):
similarity estimated using statistics of the pixel distributions

Colorfulness:
Combines the mean and standard deviation across opponent color channels (yellow-blue and red-green), in the given RGB image.

Inadequacy of metrics

PSNR:

PSNR is based on the MSE, thus suffers of the same problems

MAE/MSE:

MAE/MSE operates under the assumption of a single correct output for each pixel, which may not align with the nature of the colorization problem

Inadequacy of metrics

Colorfulness:

if an image predominantly consists of black, the metric will penalize it due to its perceived lack of color, even if the model-generated image is, in reality, accurate.

SSIM:

captures only statistics of the output, is not able to represent the credibility of the recolorization

The “Turing Test”

	Colorfulness	PSNR	SSIM	MSE	MAE
Autoenc	15916.6	45697.3	347.5	5081997	**5885808**
Deep NN	**22045.0**	**45700.4**	428.8	5080131	5976311
Dense net	20934.3	45683.7	377.0	5091630	5902259
Resnet	21575.1	45700.1	401.7	5080246	5945178
Unet	20723.7	45657.8	**434.7**	**5061775**	5896118
Pix2Pix	16742.3	45695.1	429.0	5083569	6003876

Since those considerations, we decided to rely on the “Turing Test”, evaluating results visually on a subset of the test-set.

Results

Results

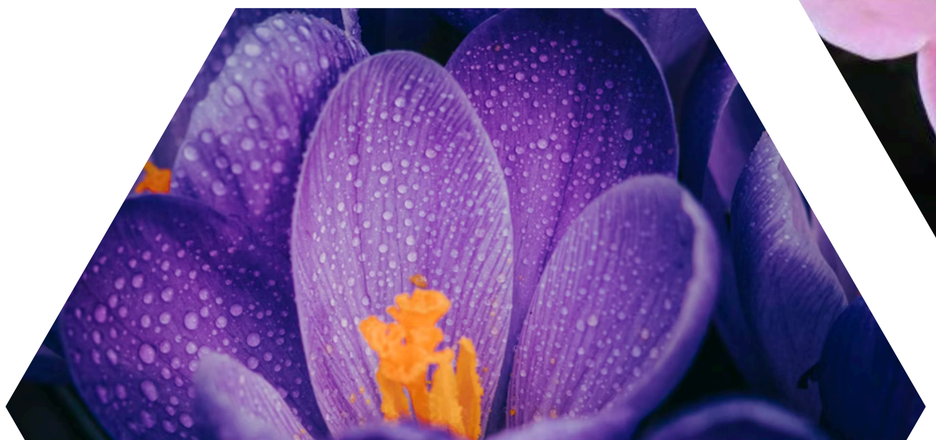
Models are difficult to evaluate, however it is evident that the **UNet** and **Pix2Pix** models consistently produced colorized images with higher fidelity and realism compared to all the other implemented models.

Why Adversarial Loss visually outperforms Regression

Traditional regression-based loss functions in image colorization may struggle to capture the diversity of plausible colorizations, leading to suboptimal results. In contrast, Pix2Pix leverages Generative Adversarial Networks (GANs) to learn the target distribution more flexibly.

By incorporating an **adversarial loss term**, Pix2Pix encourages the generator to produce colorized images that not only match ground truth but also resemble realistic color distributions found in natural images. This results in visually convincing colorizations with rich, diverse color palettes.

While the GAN loss **encourages realism**, it may introduce some deviation from ground truth colors. Balancing adversarial and regression losses is crucial for achieving accurate yet visually appealing colorizations.

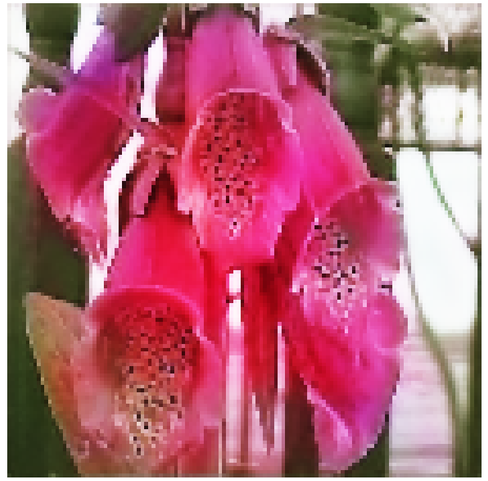
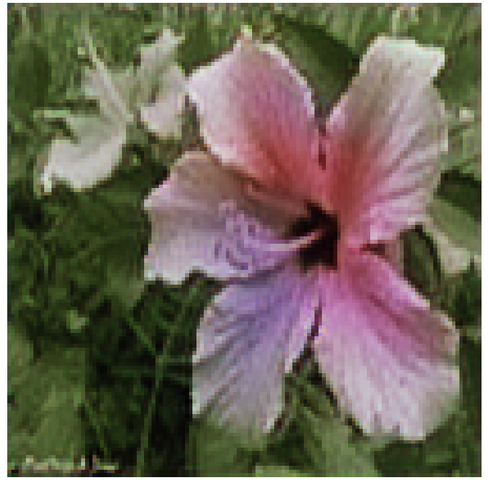


Failure Cases

Inconsistent Coloring

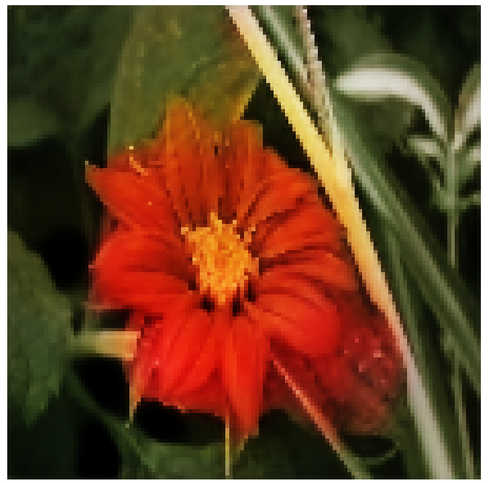
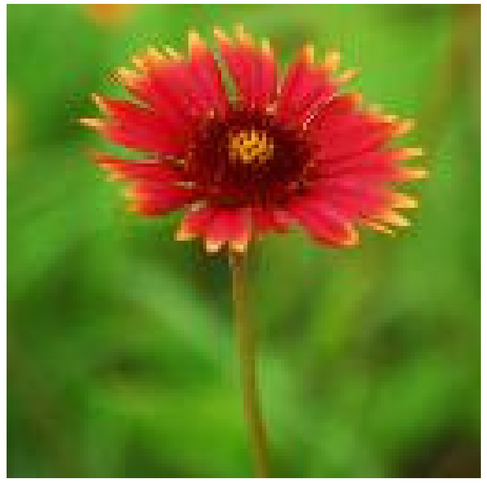
Spatially inconsistent colors, such as patches of different colors in the background or on the depicted object.

In this case, it is the "least serious" error, since the images are still plausible.

	Original	Generated
UNet		
Pix2Pix		

Color Bleeding

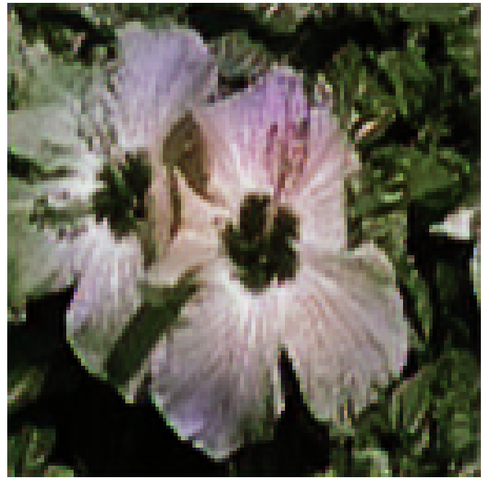
The color of an object (in our case, the flowers) spills over its intended boundary.

	Original	Generated
UNet		
Pix2Pix		

Greenish Color

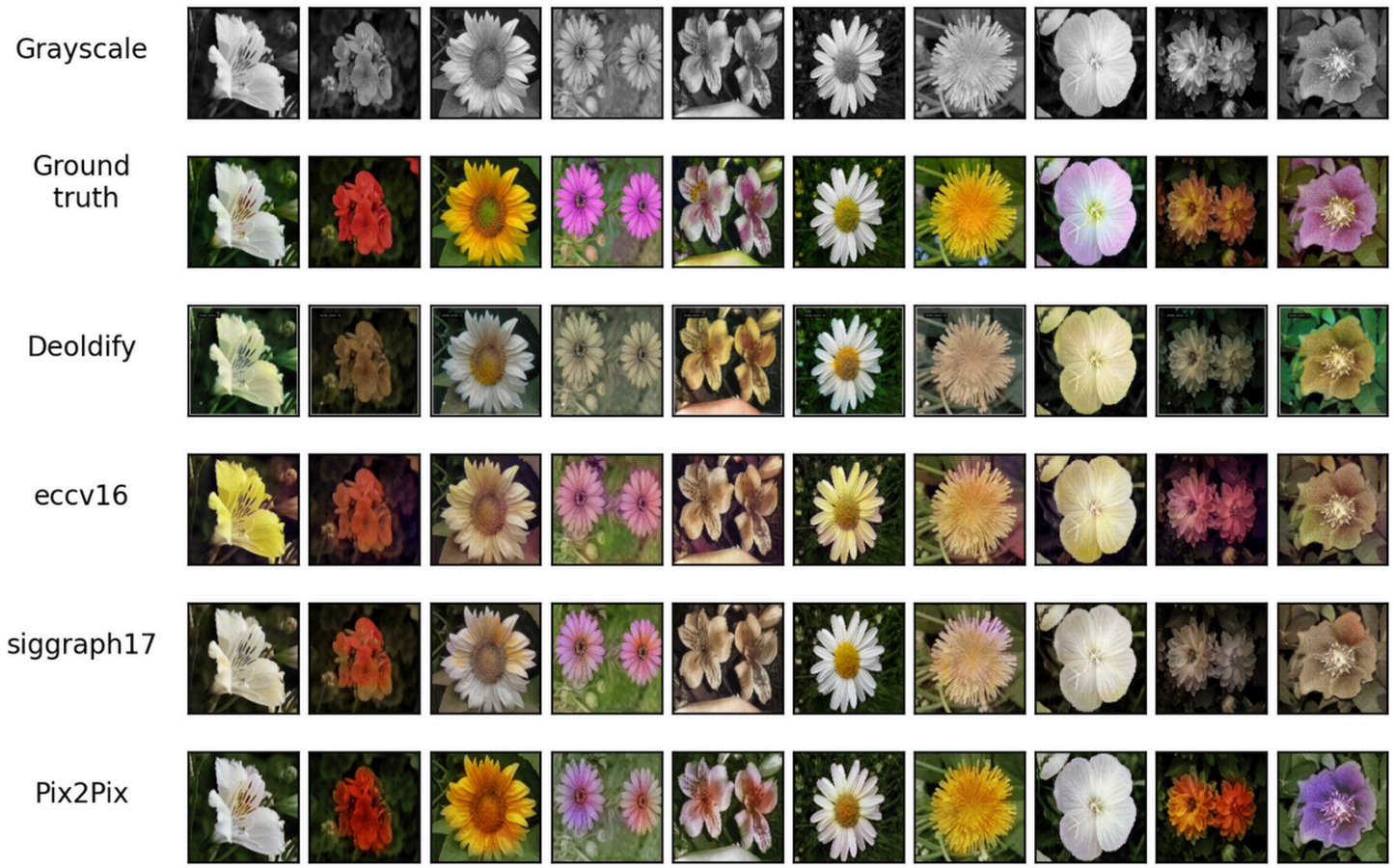
This is the most frequent case, where the image is predominantly colorized with shades of green.

Model's bias towards greenish colors, likely influenced by their prevalence in the training dataset

	Original	Generated
UNet		
Pix2Pix		

Comparision with state of art

We decided to compare Pix2Pix with the **DeOldify** model, which is again based on cGAN with a ”U-net” generator but also add a self-attention layer and spectral normalization to both the generator and discriminator, and with the **ecvv16** and **siggraph17**



Considered Variations

- **Contrast shift** as a data augmentation technique
- **BCE loss** instead of the MSE in the pix2pix generator
- At the beginning we tried to use **flower images** from the **ImageNet dataset** but yielded limited color diversity due to predominance of white and yellow flowers.

Conclusions

Our results indicate that among the different neural networks evaluated, the U-Net and the Pix2Pix architectures consistently demonstrated the best performances. However, in terms of overall performance, the Pix2Pix model emerged as the most effective, producing colorized images that closely resembled the ground truth.

Thank you for your attention

Hybrid

Our Pix2Pix hybrid training strategy employs two key loss functions to optimize the performance of the discriminator and generator models. The **generator loss** primarily relies on the **Mean Squared Error** (MSE) loss, ensuring that the generated colorizations closely match the ground truth images. On the other hand, the **discriminator loss** is driven by an **adversarial loss**, guiding the model to effectively differentiate between real and generated images.

- Generator loss: $J(\theta) = \mathbb{E}_{x,y\sim p_{data}}[(f_\theta(x) - y)^2]$

- Discriminator loss: $\text{Adv Loss (MSE)} = \frac{1}{N}\sum_{i=1}^{N}(\text{Preds}_{1_i} - 1)^2 + \frac{1}{N}\sum_{i=1}^{N}\text{Preds}_{0_i}^2$