Vision and Cognitive Systems Report

Mahir Selek
mahir.selek@studenti.unipd.it

Joi Berberi
joi.berberi@studenti.unipd.it

Abstract

Image colorization is the process of adding color to grayscale or black and white images. It involves using algorithms to analyze the image and assign appropriate colors to different parts of the image. The goal is to create a realistic and visually appealing representation of the original scene. This process can be achieved through various techniques, including deep learning models trained on large datasets of colored images. Image colorization is a complex task that combines computer vision, machine learning, and artistic interpretation to transform monochromatic images into vibrant and engaging visuals.

1. Introduction

Deep learning (DL) has the ability to achieve complicated cognitive tasks exceeding the performance of humans. Because the performance of DL algorithms, such as convolutional neural network(CNN), gives great results compared to other machine learning machine learning.

This project analyzes that implementation of this paper [2] for image colorization. In this application, how outputs are obtained due to the structural differences of different architectures will be examined in detail.

1.1. Motivations Behind the Study

Deep learning has revolutionized the task of colorizing black and white images. Previously reliant on human effort and coding, the process can now be completed end-to-end using AI. This advancement eliminates the need for extensive human input and manual programming, making colorization more efficient and seamless.

By comparing the output of each model, we can see which model is more unique at capturing the features that we are interested in, and which model produces more interesting and visually appealing results

1.2. Overview

The paper titled "Image-to-Image Translation with Conditional Adversarial Networks," commonly known as pix2pix, presented a versatile solution for various image-to-image tasks in deep learning, including colorization. This approach employed two types of losses: L1 loss, which facilitated a regression-based approach, and an adversarial (GAN) loss, which enabled unsupervised learning and addressed the problem effectively. The combination of these losses allowed for a powerful and flexible framework to tackle image colorization and other related tasks.

In this project, our initial focus will be on replicating the methodology proposed by the authors in the paper. We will carefully implement their techniques and reproduce their results. Building upon their work, we will introduce a generator model and make strategic modifications to the training approach. These enhancements aim to achieve remarkable outcomes while minimizing the dataset requirements. By doing so, we aim to push the boundaries of image colorization and further optimize the process for practical applications

2. Related Works

Over the past few years, numerous deep learning-based solutions have emerged for image colorization. The Colorful Image Colorization paper [6] took a classification-based approach to address this problem while also acknowledging the inherent uncertainty involved (e.g., a car can have multiple valid color possibilities, making color selection uncertain).

Conversely, another paper tackled colorization as a regression task, incorporating additional modifications. Each approach has its own advantages and disadvantages. However, in this article, we will adopt a distinct strategy, presenting an alternative approach to image colorization that aims to offer unique benefits.

3. Dataset

The COCO "Common Objects in Context" dataset is a widely used and comprehensive dataset [3] in the field of computer vision. It is designed for various tasks, including object detection, segmentation, and image captioning. While the COCO dataset is not specifically tailored for image colorization, it still offers several advantages for this task.

3.1. Aim to use COCO dataset

Figure 1. COCO

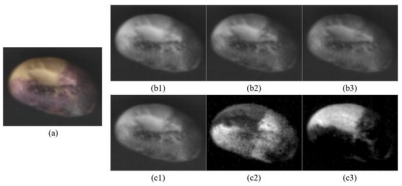

Figure 2. Cielab

- Extensive Content: With over 200,000 images, the COCO dataset offers a diverse range of object categories and scenes, providing abundant training data for image colorization,
- High-Quality Annotations: The dataset includes detailed annotations such as object bounding boxes and segmentation masks, offering additional contextual information that can assist in accurate colorization,
- Realistic Scenarios: Comprised of real-world images, the COCO dataset reflects natural color distributions, lighting conditions, and object appearances, enabling models to learn from realistic scenarios.

4. Colorization Problem

When we load an image, it is represented as a rank-3 array, where the dimensions correspond to height, width, and color. The color data is encoded in the RGB color space, with three numbers (Red, Green, and Blue) indicating the intensity of each color component for every pixel.

Alternatively, in the Lab color space, each pixel is still represented by three numbers. However, their meanings differ from RGB. The first channel, L, represents the Lightness of each pixel. Visualizing this channel results in a black and white image, as shown in the Figure 2 of the row below. The a and b channels in the Lab color space encode the green-red and yellow-blue characteristics of each pixel, respectively. These channels provide additional information about the color properties of the image. Visualizing each channel separately allows us to observe the distinct aspects captured by the Lab color space, as depicted in the following image.

The LAB color space offers advantages for colorization tasks compared to the RGB color space. While hallucinating the color of an RGB image requires identifying three channels, the LAB color space utilizes one channel to represent the luminosity of the image (ranging from 0 to 100) and two channels (a and b) to define the RGB colors (ranging from -128 to 127). Therefore, when given the L channel of an image as input, the network only needs to estimate the two color channels and combine them for colorization.

4.1. Example

Color space conversion from RGB to CIE L*a*b* for easily quantifying the visual differences between colors.

(a) A RGB image of soybean seed is decomposed into (b1) red, (b2) green and (b3) blue color components as well as converted to (c1) L*, (c2) a* and (c3) b* color channels.

Overall, the Lab color space offers perceptual uniformity, separate color and brightness channels, device independence, and robustness to lighting variations. These characteristics make it a valuable choice for various image processing tasks, especially those involving color manipulation and analysis.

5. Models

5.1. Pix2Pix

In our project, the pix2pix framework plays a significant role in the image colorization task. Pix2pix provides a powerful solution for image-to-image translation, and one of the specific tasks it addresses is colorization. By leveraging the conditional generative adversarial network (GAN)[1] architecture of pix2pix, you can train a generator model to convert grayscale images to their corresponding colored versions.

The relationship between pix2pix and our project lies in the utilization of its architecture and training methodology to achieve accurate and realistic image colorization. The generator network in pix2pix learns to map grayscale images to colored images, while the discriminator network evaluates the quality and authenticity of the generated colorized images. This adversarial training process helps the generator improve its ability to produce convincing and visually pleasing results.

5.2. GANs

In our approach, we will utilize a conditional GAN (Generative Adversarial Networks) along with an additional loss function called L1 loss. The GAN [1] consists of a generator and a discriminator model that work collaboratively to solve the problem at hand.

Specifically, our generator model takes a grayscale image (1-channel image) as input and generates a 2-channel image, representing the *a and *b channels in the Lab color space. On the other hand, the discriminator model takes these two generated channels, concatenates them with the input grayscale image, and determines whether the resulting 3-channel image is real or fake. To train the discriminator effectively, it is exposed to real images (3-channel images in

the Lab color space) that are not produced by the generator, teaching it to recognize genuine images.

The ”condition” in our GAN refers to the grayscale image that serves as input for both the generator and discriminator. This condition is provided to both models, and we expect them to consider this information during the learning process.

Mathematically, let x represent the grayscale image, z denote the input noise for the generator, and y represent the desired 2-channel output from the generator (which can also represent the 2 color channels of a real image). Additionally, G represents the generator model and D represents the discriminator. The loss function for our conditional GAN can be expressed as follows:

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

Figure 3. GAN Loss

5.3. U-Net Architecture

The U-Net[5] is convolutional network architecture for fast and precise segmentation of images.This architecture consists of an encoder-decoder structure with skip connections. It is particularly useful for image colorization due to its ability to capture detailed spatial information and generate accurate colorized outputs.

The U-Net architecture is characterized by its symmetric and expansive structure. It consists of an encoder path, which gradually reduces the spatial dimensions of the input image while extracting high-level features, and a decoder path, which expands the feature maps and recovers the spatial information. Skip connections are incorporated between corresponding encoder and decoder layers, allowing the network to preserve fine-grained details and aid in accurate colorization.

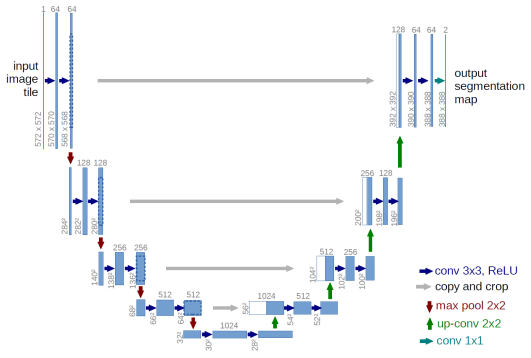

Figure 4. U-Net Architecture

The relationship between the U-Net architecture and image colorization lies in its capability to handle the complex mapping between grayscale input images and corresponding color outputs. By leveraging the U-Net architecture, the model can effectively learn to extract meaningful features from the grayscale images and generate plausible colorizations that align with the input.

The code implements a U-Net as the generator for our GAN. It constructs the U-Net by adding down-sampling and up-sampling modules iteratively to the middle part of the network. This process continues until the input and output modules are reached, resulting in a U-shaped architecture.

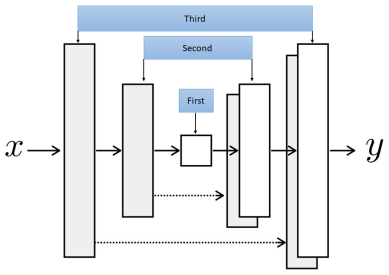

Figure 5. U-Net/GAN

The blue rectangles indicate the order in which the modules are constructed in the code. Although the depicted image doesn't represent the complete U-Net architecture, it provides an understanding of the process. Notably, the code involves building more layers, and specifically, it goes 8 layers down. Starting with a 256 by 256 image, the middle of the U-Net produces a 1 by 1 image (due to downsampling), which is subsequently upsampled to generate a 256 by 256 image with two channels.

6. Experiments

This experiment is based on the models that we defined before. The GANs consist of two main components: a generator model and a discriminator model. In the context of image colorization, the generator takes grayscale images as input and aims to produce colorized versions of those images. The discriminator, on the other hand, evaluates the generated colorized images and distinguishes them from real color images.

6.1. Implementation

Generator (U-Net): The U-Net architecture serves as the generator in the GAN framework. It takes grayscale images as input and generates colorized versions of those images. The U-Net is specifically designed to capture and leverage both global and local features through its expansive structure. It consists of down-sampling and up-sampling modules, creating a U-shaped architecture that helps preserve image details during the colorization process.

Discriminator: The discriminator model is an integral

part of the GAN framework. It evaluates the generated colorized images and distinguishes them from real color images. The discriminator is trained to classify whether an image is real or fake, providing feedback to the generator on how to improve the quality and realism of the generated colorizations.

6.2. Preprocessing

In the preprocessing step, we perform image resizing and horizontal flipping (only for the training set). We then read the RGB images and convert them to the Lab color space. The grayscale channel is separated as the input, while the color channels serve as the targets for our models. Finally, we create data loaders to handle the prepared data.

6.3. Loss Function

To enhance the colorization process, we combine the previous loss function with L1 Loss[7] (mean absolute error). This additional loss compares the predicted colors with the actual colors, providing supervision and improving the realism of the generated images. By combining these losses, we aim to achieve more accurate and visually appealing colorizations.

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

Figure 6. L1 Loss

However, if we rely solely on L1 loss, the model tends to adopt a conservative approach and often assigns colors such as "gray" or "brown" when uncertain. This behavior arises from the model's tendency to minimize L1 loss by averaging colors in cases of doubt. This effect is reminiscent of the blurring impact observed in L1 or L2 loss in super-resolution tasks.

To address this, we choose L1 loss over L2 loss (mean squared error) since it mitigates the production of gray-ish images. By incorporating L1 loss in our combined loss function, we encourage the model to produce more vibrant and diverse colorizations while still maintaining realism.

$$G^* = \arg\min_G \max_D \mathcal{L}_{cGAN}(G,D) + \lambda \mathcal{L}_{L1}(G)$$

Figure 7. Loss

where lambda is a coefficient to balance the contribution of the two losses to the final loss (of course the discriminator loss does not involve the L1 loss).

6.4. Results of Baseline Model

As observed, the baseline model's performance falls short in generating visually appealing colorizations. While it demonstrates basic comprehension of common objects such as sky and trees, it struggles to determine the appropriate colors for uncommon objects. Additionally, the model exhibits color spillovers and circular patches of color (as seen in the center of the first image in the second row), which is undesirable.

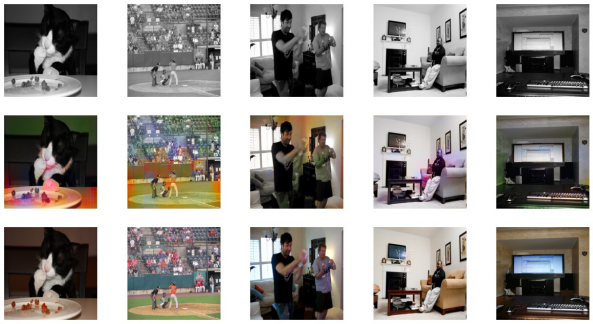

Figure 8. Baseline Results

6.5. Final Model

This article focuses on addressing the problem mentioned earlier by introducing a novel approach inspired by the Super Resolution literature. To overcome the challenge of starting with limited knowledge in the GAN training process, a separate pretraining phase is implemented for the generator in a supervised and deterministic manner [4].

The pretraining consists of two stages. Firstly, the backbone of the generator, which corresponds to the down sampling path, is pretrained using a classification model on the ImageNet dataset. This step helps the generator gain a better understanding of visual features. Secondly, the entire generator is pretrained specifically for the colorization task using L1 loss, which aids in capturing accurate color information.

In this approach, a pretrained ResNet18 model serves as the backbone for the U-Net architecture. The second stage involves training the U-Net on the training set using only the L1 loss. Once this pretraining stage is completed, the training progresses to incorporate the combined adversarial and L1 loss, as described in the previous section.

New Generator:The pretrained weights of the ResNet18 architecture were loaded, and the model was modified by removing the last two layers, namely GlobalAveragePooling and a Linear layer used for ImageNet classification. This modified ResNet18 model serves as the backbone. Subsequently, DynamicUnet utilizes this backbone to construct a U-Net with the required output channels (2 channels in our case) and an input size of 256.

Pretraining:After 20 epochs of pretraining, we save the weights of the generator model. In the subsequent section, we utilize this pretrained generator as the main component in our GAN architecture and proceed with training the entire network, following the same methodology as before.

Results:The saved weights of the generator, which were obtained in the previous section, are loaded. These weights are then utilized in our MainModel class, ensuring that the generator is not randomly initialized. The model is trained for a reduced number of epochs, typically ranging from 10 to 20, in comparison to the 100 epochs conducted in the previous section where pretraining was not employed.

Figure 9. The Colorized Outputs

6.6. Comparison

During our experiments, We made an interesting discovery: the U-Net we constructed using the ResNet18 backbone demonstrates impressive image colorization capabilities even after pretraining with L1 Loss alone, prior to the final adversarial training stage. However, the model tends to be conservative in its color choices, often favoring grayish tones when uncertain about object identification or appropriate colors. Nonetheless, it excels at accurately colorizing common elements such as the sky, trees, and grass within the images.

To illustrate the substantial impact of adversarial training in our case, We present a comparison between the outputs of the U-Net without adversarial training and the U-Net with adversarial training.

The results of the pretrained U-Net with and without adversarial training:

Figure 10. Comparison

7. Conclusion

In conclusion, this image colorization project has been a journey of exploration and experimentation. We started by delving into the world of deep learning and its applications in colorizing black and white images. Throughout the project, we implemented various methods and techniques, incorporating concepts from papers such as pix2pix and U-Net with ResNet.

One of the key insights we gained from this project was the importance of pretraining the generator. By leveraging a pretrained ResNet18 backbone and training the U-Net with L1 Loss, we observed significant improvements in the colorization results. This approach not only enhanced the model's ability to capture common scene elements accurately but also provided a foundation for subsequent adversarial training.

Overall, this project underscores the power of deep learning and its potential in transforming image colorization tasks. It has broadened our understanding of generative adversarial networks, the significance of pretraining, and the importance of architectural choices in achieving desirable results. Moving forward, we envision further refinement and exploration of these methods to unlock even greater potential in the field of image colorization.

References

[1] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

[2] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017.

[3] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014.

[4] Mavra Mehmood, Nasser Alshammari, Saad Awadh Alanazi, Asma Basharat, Fahad Ahmad, Muhammad Sajjad, and Kashaf Junaid. Improved colorization and classification of intracranial tumor expanse in mri images via hybrid scheme of pix2pix-cgans and nasnet-large. *Journal of King Saud University-Computer and Information Sciences*, 34(7):4358–4374, 2022.

[5] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, pages 234–241. Springer, 2015.

[6] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *Computer Vision–ECCV 2016: 14th*

European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part III 14, pages 649–666. Springer, 2016.

[7] Hang Zhao, Orazio Gallo, Iuri Frosio, and Jan Kautz. Loss functions for image restoration with neural networks. *IEEE Transactions on computational imaging*, 3(1):47–57, 2016.