

Article

Color Normalization Through a Simulated Color Checker Using Generative Adversarial Networks

Albert Siré Langa ^{1,*} , Ramón Reig Bolaño ¹ , Sergi Grau Carrión ¹ and Ibon Uribe Elorrieta ^{2,*}

¹ Faculty of Science Technology and Engineering (FCTE), Universitat de Vic–Universitat Central de Catalunya, 08500 Vic, Spain; ramon.reig@uvic.cat (R.R.B.); sergi.grau@uvic.cat (S.G.C.)

² Diabetic Foot Unit, Facultad de Medicina, Instituto de Investigación Sanitaria del Hospital Clínico San Carlos, Universidad Complutense de Madrid, 28040 Madrid, Spain

* Correspondence: alberto.sire@uvic.cat (A.S.L.); iuribe@ucm.es (I.U.E.)

Abstract: Digital cameras often struggle to reproduce the true colors perceived by the human eye due to lighting geometry and illuminant color. This research proposes an innovative approach for color normalization in digital photographs. A machine learning algorithm combined with an external physical color checker achieves color normalization. To address the limitations of relying on a physical color checker, our approach employs a generative adversarial network capable of replicating the color normalization process without the need for a physical reference. This network (GAN-CN-CC) incorporates a custom loss function specifically designed to minimize errors in color generation. The proposed algorithm yields the lowest coefficient of variation in the normalized median intensity (NMI), while maintaining a standard deviation comparable to that of conventional methods such as Gray World and Max-RGB. The algorithm eliminates the need for a color checker in color normalization, making it more practical in scenarios where inclusion of the checker is challenging. The proposed method has been fine-tuned and validated, demonstrating high effectiveness and adaptability.

Keywords: color normalization; color calibration; color checker; generative adversarial network; machine learning



Academic Editor: George A. Tsihrintzis

Received: 26 March 2025

Revised: 14 April 2025

Accepted: 18 April 2025

Published: 25 April 2025

Citation: Langa, A.S.; Bolaño, R.R.; Carrión, S.G.; Elorrieta, I.U. Color Normalization Through a Simulated Color Checker Using Generative Adversarial Networks. *Electronics* **2025**, *14*, 1746. <https://doi.org/10.3390/electronics14091746>

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

The use of digital cameras introduces variability in image colors due to external factors. According to color theory [1], the surface color of an object depends on both surface reflectance and the spectral power distribution of the illumination. Consequently, color normalization for digital pictures is an essential preprocessing step in applications such as computer vision systems.

Color normalization aims to generate color image data with identical or similar perceptual response when evaluated by human operators [2]. The goal is to make images comparable and enhance the performance of downstream tasks like segmentation, classification, or feature extraction.

Various methods have been proposed for color normalization. Traditional methods such as the Gray World Assumption [3] rely on illumination assumptions. This method is a white balance technique based on the premise that, on average, a scene contains a neutral gray. This assumption holds when the scene includes a well-balanced distribution of colors. In such cases, the average reflected color is presumed to represent the illumination color. Consequently, the illumination's color cast can be estimated by analyzing the average color of the scene and comparing it to gray.

Additionally, regression-based methods have been widely explored for color normalization. One prominent example is root-polynomial regression, proposed by Finlayson et al. [4], which provides a robust polynomial approach to model the relationship between input and target colors. Other regression methods, such as least squares, have also been successfully investigated, as discussed in [5].

There are other color normalization methods that rely on the use of a color checker. By utilizing a color checker, colors are defined independently of the light source, illumination conditions, and the image capturing device. This ensures consistent and accurate color representation across varying imaging environments. An example of a color checker is the Pantone PCNCT-CARD (PANTONE) (see Figure 1), which is a compact and portable color calibration tool designed for creative professionals. It features a selection of essential Pantone colors arranged on a durable card, ideal for on-the-go color matching and verification. It comprises 4 ArUco markers [6] and 72 color patches.



Figure 1. Example of color checker: PANTONE PCNCT-CARD.

Linear methods that utilize color checkers for image normalization are based on the derivation of a color correction matrix (CCM) [7]. These methods involve capturing the input image to be normalized, ensuring that a color checker is included in the scene. The RGB values of the color patches on the color checker are then extracted from the input image. The CCM is computed as the transformation matrix that maps the color patterns of the input image to those of a reference color checker. Once the CCM is obtained, it is applied to the RGB values of every pixel in the input image, achieving color normalization.

Recent advancements have focused on the application of neural networks for color normalization. Initial studies [8] demonstrate that neural network algorithms outperform simple linear regression in achieving better color correction accuracy. Further, ref. [9] highlights how neural networks improve performance by learning complex, non-linear mappings between input and normalized colors.

Generative adversarial networks (GAN) [10] have also been employed for color normalization tasks. A GAN is a deep learning framework consisting of two neural networks: a generator that creates fake data and a discriminator that evaluates its authenticity compared to real data.

This progression from traditional methods like Gray World to advanced neural network and GAN-based techniques reflects the growing sophistication of color normalization approaches in image processing.

This paper introduces a GAN-based method (GAN-CN-CC algorithm) that eliminates the need for a physical color checker by simulating its role in color normalization. The

proposed method initially employs a machine learning algorithm (ML-CN-CC algorithm) and a PANTONE color checker to achieve color normalization. Subsequently, a generative adversarial network is trained to replicate the initial algorithm without requiring the use of the PANTONE color checker. Finally, a comprehensive validation is conducted by comparing the proposed method against state-of-the-art techniques. This approach enables the normalization of colors in any image captured, regardless of the lighting conditions or camera type.

2. Materials and Methods

In this section, we outline the materials and methodology employed to generate color-normalized images based on a reference PANTONE, eliminating the need for the physical PANTONE.

2.1. Materials for the Algorithm Based on Machine Learning and Color Checker (ML-CN-CC)

To build and test the ML-CN-CC color normalization algorithm with a color checker, a dataset comprising 1508 images with PANTONE references was used (1532×2048 and 1200×1600 pixels). These images comprise a wide variety of different scenes and were captured using various mobile devices under diverse lighting conditions (Table 1).

Table 1. List of devices used to capture the images.

Device
Vivo V40 Lite
Redmi Note 10 Pro
Samsung Galaxy A23 5G

2.2. Materials for the Algorithm Based on GAN for Color Normalization Simulating the Use of a Color Checker (GAN-CN-CC)

The performance of generative adversarial networks (GANs) is highly dependent on the quality and diversity of the training dataset, which must encompass a broad spectrum of lighting conditions and scenes.

To train the GAN network, a dataset of 38,022 image pairs was created. First, all images from the dataset described in Section 2.1 were processed using the ML-CN-CC algorithm. After this, the images for which the ML-CN-CC algorithm produced optimal results (loss value ≤ 0.07) were converted into $256 \times 256 \times 3$ patches, which is the input size for the GAN network. In this way, 38,022 image pairs were obtained, each consisting of a $256 \times 256 \times 3$ input image and a $256 \times 256 \times 3$ image processed with the ML-CN-CC algorithm (see Figure 2). The GAN-CN-CC network was then trained using this dataset of paired images.

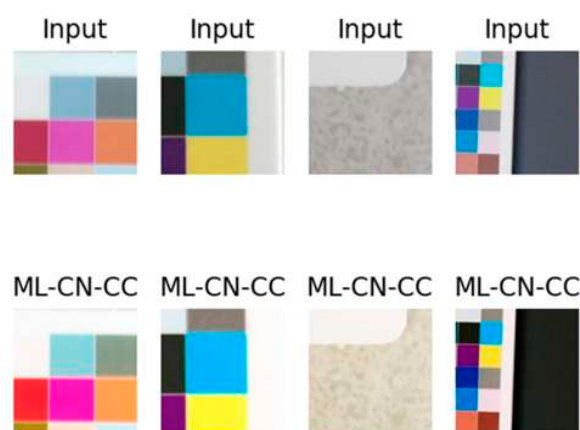


Figure 2. Pairs of input images and ML-CN-CC normalized images, which serve as the input for the GAN-CN-CC network.

2.3. Methods for the Algorithm Based on Machine Learning and Color Checker (ML-CN-CC)

Before creating the GAN network that normalizes images by simulating the use of a color checker, the algorithm that normalizes images based on the use of the color checker must first be created. For that, we propose a method based on the use of a machine learning algorithm in conjunction with a PANTONE color checker (referred to as the ML-CN-CC method).

Initially, the color checker was detected within the Reference PANTONE image and positioned vertically (Figure 3). This detection was accomplished using the function *aruco.detectMarkers* from the ArUco library [6]. This function allows for the detection of the four corner markers of the PANTONE and, therefore, the PANTONE itself.

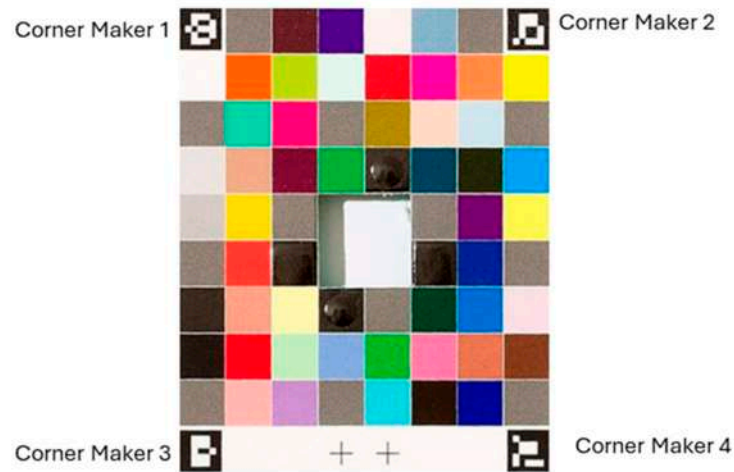


Figure 3. Detected Reference PANTONE and corner markers.

Subsequently, the non-repeated color patches of the Reference PANTONE were detected. To do this, the width of the detected PANTONE was divided by 8 (the number of horizontal patches), and the color value of each patch was detected, resulting in a total of 54 distinct colors, which are the numbered ones in Figure 4. This process was repeated for the 10 rows of the PANTONE. The RGB values of these colors were stored in a 54×3 matrix (Reference PANTONE matrix, Figure 4).

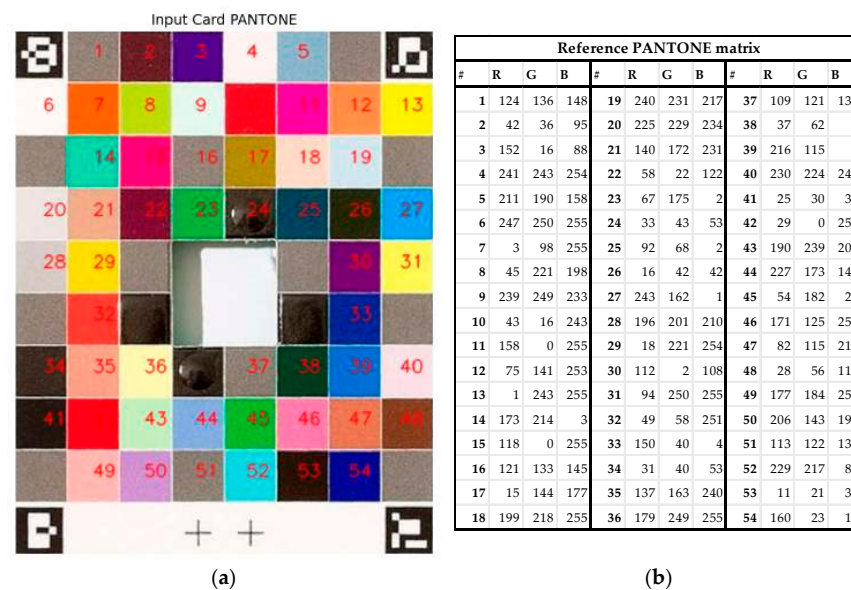


Figure 4. (a) Reference PANTONE distinct colors patches. Those colors (a total of 54 of 72 colors) are indicated with a number in red (b) Reference PANTONE Matrix.

Subsequently, the input image (Figure 5) underwent the same processing, beginning with the detection of markers, followed by vertical positioning and the identification of color patches (Figure 6). The RGB values of the 54 colors were stored in a 54×3 matrix (Input Image Matrix, Figure 6).



Figure 5. Input image.

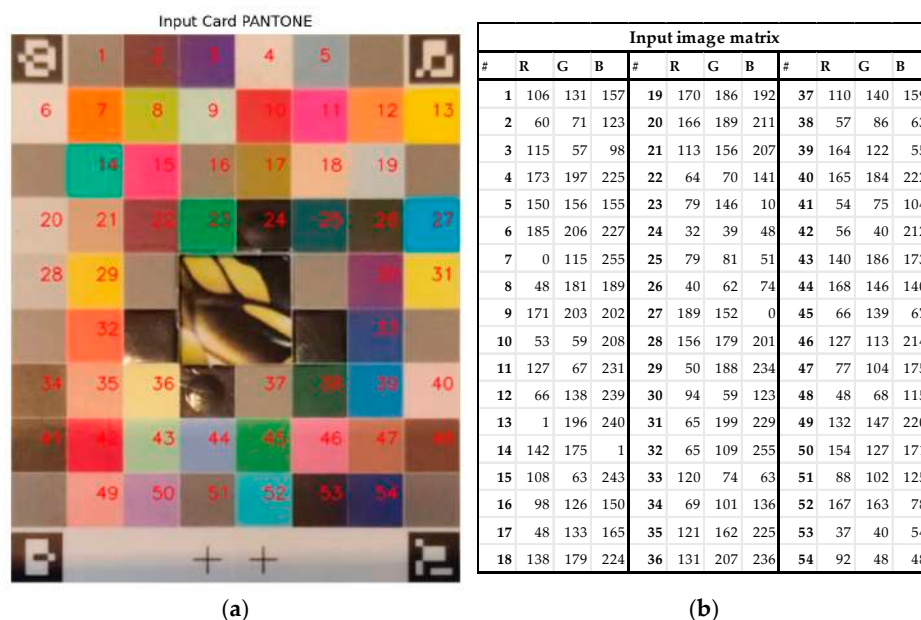


Figure 6. (a) Input image PANTONE distinct colors patches. Those colors (a total of 54 of 72 colors) are indicated with a number in red (b) Input Image Matrix.

The machine learning algorithm was then trained to convert the RGB colors of the Input Image Matrix into the colors of the Reference PANTONE Matrix. In other words, the training data for the network consisted of the 54×3 RGB colors of the PANTONE detected

in the input image along with the corresponding 54×3 RGB colors of the PANTONE detected in the Reference PANTONE image.

The proposed machine learning network is a multi-output regression model. It is designed with an input layer consisting of 54×3 inputs, followed by four hidden layers that progressively transform the input data, and then concludes with an output layer. The first three dense layers employ linear activation functions, focusing on data transformation without introducing nonlinearity. The fourth dense layer utilizes the ReLU activation function, enabling the network to model more complex relationships. The output layer (3 outputs, 3 RGB colors) is equipped with a custom activation function specifically tailored for regression tasks, ensuring accurate mapping of the input to the desired output. It is worth noting that this network model based on 4 hidden layers (Figure 7) delivered the best results across the various iterations performed.

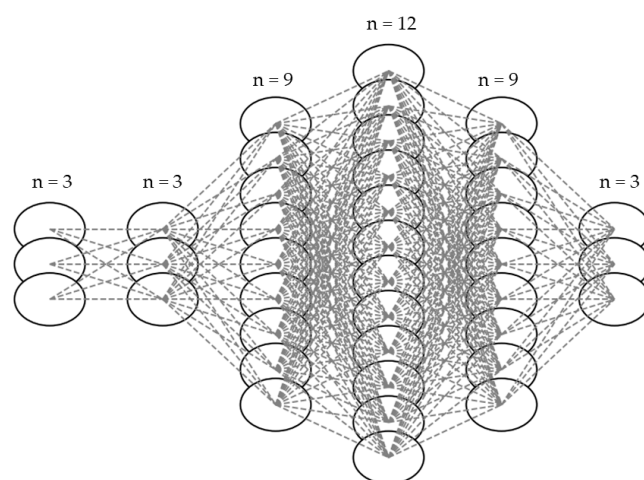


Figure 7. General Structure of the ML-CN-CC network. Four dense layers and an output layer (3 outputs, 3 RGB colors), ensuring accurate mapping of the input to the desired output.

The training process employs the Adam optimizer (Adaptive Moment Estimation), a widely used optimization algorithm in neural networks. In this study, the optimizer was configured with a learning rate of 0.001, ensuring efficient and stable convergence during training.

Regarding hyperparameters, the network was trained for 150 epochs with a batch size of 8.

Finally, the model employs mean absolute error (MAE) as its loss function, which is widely used in regression problems. The MAE aims to minimize the average absolute difference between the model's predictions and the true values, providing a straightforward measure of prediction accuracy.

It is worth noting that, for each image, a dedicated ML-CN-CC algorithm was trained, capable of converting the test image's PANTONE colors into the reference PANTONE colors (as previously explained).

2.4. Methods for the Algorithm Based on GAN for Color Normalization Simulating the Use of a Color Checker (GAN-CN-CC)

The ML-CN-CC algorithm provides consistent results in the color normalization process supported by a color checker. However, there are environments where the use of such physical markers is not practical (i.e., clinical environments where the use of a physical marker may interfere with the assessment and treatment of a lesion).

For this reason, the method we present below is based on the use of a generative adversarial network that allows for simulation of the ML-CN-CC method without a color

checker. Instead of relying on a color checker, our GAN network leverages its ability to learn complex mappings and transformations from the image data itself.

This method, named GAN-CN-CC, is based on a Pix2Pix GAN [11]. A Pix2Pix GAN is a generative adversarial network designed for paired image-to-image translation tasks. In our case, the image-to-image translation problem is defined as follows: the input domain consists of a set of standard RGB images of a scene captured under an unknown light source, with or without the inclusion of a color checker. The output domain comprises the corresponding color-normalized images, simulated as if processed by the ML-CN-CC algorithm.

GAN-CN-CC consists of two main neural networks that compete with each other. First, we have the Generator (G), whose goal is to convert the input image into a processed image using the ML-CN-CC algorithm. The Generator's architecture is based on a U-Net type network [12], where the Encoder component encodes the input image, while the Decoder component decodes that information to generate an image that mimics the style of the output domain. In this sense, the U-Net architecture of the Generator is composed of seven encoder–decoder blocks. Each block includes both an encoder and a decoder component. The U-Net generator receives input images of size $256 \times 256 \times 3$. The encoder consists of a series of convolutional layers with 64, 128, 256, 512, 512, 512, and 512 filters, progressively increasing the feature complexity. These convolutional layers are integrated with activation functions, batch normalization, and dropout layers, enhancing both learning capacity and generalization. The decoder is composed of transposed convolutional (deconvolutional) layers, each followed by activation and batch normalization layers. The decoder uses filters in the following order: 512, 512, 512, 256, 128, and 64, effectively reducing the dimensionality of the feature maps and reconstructing the spatial resolution.

On the other hand, we have the Discriminator (D), whose goal is to distinguish between “real” images (from the training dataset) and “fake” images (generated by the Generator). It is based on a PatchGAN [11] architecture that does not evaluate the entire image but instead classifies small patches of the image as real or fake and then weighs the final classification. In the Discriminator, leaky ReLU activation is applied in the convolutional layers with batch normalization. The convolutional layers of the Discriminator use 64, 256, and 512 filters, maintaining progressively increasing feature map depth for discriminative learning.

The training of the GAN-CN-CC network is based on a competitive approach between the Generator and the Discriminator, as described in Figure 8.

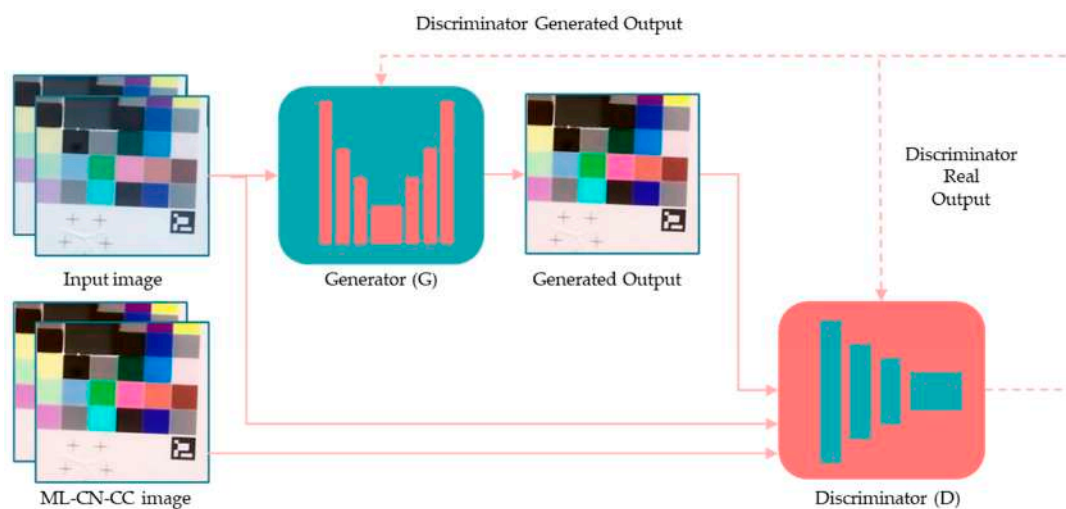


Figure 8. Diagram of the GAN-CN-CC Network. A GAN-CN-CC network is a generative adversarial network that learns to translate images from one domain to another. It consists of a U-Net-based Generator that converts an input image into a target-style image and a PatchGAN Discriminator that distinguishes real images from generated images.

During the Generation process, an input image is fed into the Generator, which produces a generated output image.

Additionally, the input image along with the generated output is fed into the Discriminator, resulting in the Discriminator-generated output value. The Discriminator also receives the input image along with the image processed using ML-CN-CC (real image), producing the Discriminator real output value. Based on these values, loss functions are applied to enhance the competition between the Generator, which aims to create images that are not recognized as “fake” by the Discriminator, and the Discriminator, which strives to differentiate between the generated (“fake”) and real images.

To achieve this, the Generator of the GAN-CN-CC network uses a combination of three loss functions to improve the quality of the generated images. First, there is the Adversarial Loss, which ensures that the images generated are realistic enough to fool the Discriminator. The basic formula for this loss function for the Generator is the Binary Cross-Entropy:

$$\mathcal{L}_{BCE}(G) = -(m \cdot \log(D(G(x))) + (1 - m) \cdot \log(1 - D(G(x)))) \quad (1)$$

where:

m is 1

x is the input image

$G(x)$ is the output of the Generator (generated image)

$D(G(x))$ is the output of the Discriminator when the input is a generated image

On the other hand, there is the Reconstruction Loss (L1 Loss), which calculates the difference between the generated image and the expected real image to maintain the structural fidelity of the image. The formula is as follows:

$$\mathcal{L}_{L1}(G) = \mathbb{E}_{x,y}[\|y - G(x)\|] \quad (2)$$

where:

\mathbb{E} is the expected value, it represents the weighted average of all possible outcomes of a random variable X , according to its probability distribution.

y is the input image processed with ML-CN-CC

x is the input image

$G(x)$ is the output of the Generator (generated image)

Finally, we have the Color Loss (a loss not considered in the original Pix2Pix GAN), which aims to achieve maximum similarity between the colors of the generated image and the real image. The formula for this loss function is as follows:

$$\mathcal{L}_{CL}(G) = \mathbb{E}_{x,y}[\|\Delta E00(y_{LAB}, G(x)_{LAB})\|] \quad (3)$$

where:

\mathbb{E} is the expected value, it represents the weighted average of all possible outcomes of a random variable X , according to its probability distribution.

y_{LAB} is the input image processed with ML-CN-CC and converted to Lab space color

x is the input image

$G(x)_{LAB}$ is the output of the Generator converted to Lab space color

$\Delta E00$ is the CIE Delta E 2000 difference between the real image and the generated image.

The $\Delta E00$ color difference between a sample color (L_2, a_2, b_2) and a reference color (L_1, a_1, b_1) is:

$$\Delta E00 = \sqrt{\left(\frac{\Delta L'}{K_L S_L}\right)^2 + \left(\frac{\Delta C'}{K_C S_C}\right)^2 + \left(\frac{\Delta H'}{K_H S_H}\right)^2 + R_T \left(\frac{\Delta C'}{K_C S_C}\right) \left(\frac{\Delta H'}{K_H S_H}\right)} \quad (4)$$

where:

- $\Delta L'$ Lightness difference.
- $\Delta C'$ Chroma difference.
- $\Delta H'$ Hue difference.
- K_L, K_C, K_{H-} Parametric weighting factors. We will take $K_L = 2, K_C = 1$ and $K_H = 1$
- S_L, S_C, S_H Scaling functions for lightness, chroma, and hue.
- R_T Rotation term to account for interactions between chroma and hue.

Here, $\Delta E00$ is a standardized color difference metric to measure perceptual differences between two colors. Thus, $\Delta E00$ accounts for perceptual non-uniformities and includes corrections for chroma, lightness, and hue interactions. For a detailed explanation of the $\Delta E00$ formula and its parameters, please refer to [13].

In this way, the Generator's total loss function is as follows:

$$\mathcal{L}_{GANCNCC}(G) = \mathcal{L}_{BCE}(G) + \mathcal{L}_{L1}(G) + \mathcal{L}_{CL}(G) \quad (5)$$

Regarding the Discriminator, it takes the generated image and the real ML-CN-CC-normalized image as concatenated inputs, applies downscale layers to reduce dimensionality and extract features, and produces an output in the form of a probability map indicating if the generated image is real or not.

The Discriminator uses a loss function that is a combination of two. On the one hand, there is the Real Loss, which is as follows:

$$\mathcal{L}_{REAL}(D) = -(m \cdot \log(D(y))) + (1 - m) \cdot \log(1 - D(y)) \quad (6)$$

where:

m is 1

y is the input image processed with ML-CN-CC

On the other hand, there is the Fake Loss, which has the following formula:

$$\mathcal{L}_{FAKE}(D) = -(m \cdot \log(D(G(x)))) + (1 - m) \cdot \log(1 - D(G(x))) \quad (7)$$

where:

m is 0

x is the input image

$G(x)$ is the output of the Generator (generated image)

$D(G(x))$ is the output of the Discriminator when the input is a generated image

In this way, the Discriminator's total loss function is:

$$\mathcal{L}(D) = \mathcal{L}_{REAL}(D) + \mathcal{L}_{FAKE}(D) \quad (8)$$

Through iterative training, the GAN-CN-CC network progressively improves the Generator's ability to normalize colors without explicit reference to a physical color checker, while the Discriminator becomes increasingly adept at identifying inaccuracies.

The training process employs the Adam optimizer with a 2×10^{-4} learning rate and, regarding hyperparameters, the network was trained for 110 epochs with a batch size of 32. It is worth noting that the network was trained without a validation dataset, and its testing was performed with images that were not seen during training.

3. Results

3.1. Results for the Algorithm Based on Machine Learning and Color Checker (ML-CN-CC)

Figures 9–11 illustrate the results of the ML-CN-CC training process for one input image, comparing the RGB values of the Input Image Matrix processed by the ML-CN-CC algorithm (Normalized Input Matrix) with the corresponding RGB values of the PANTONE Reference Matrix. As can be seen, the ML-CN-CC algorithm is capable of accurately converting the 54 PANTONE values from the input image to the 54 values of the Reference PANTONE.

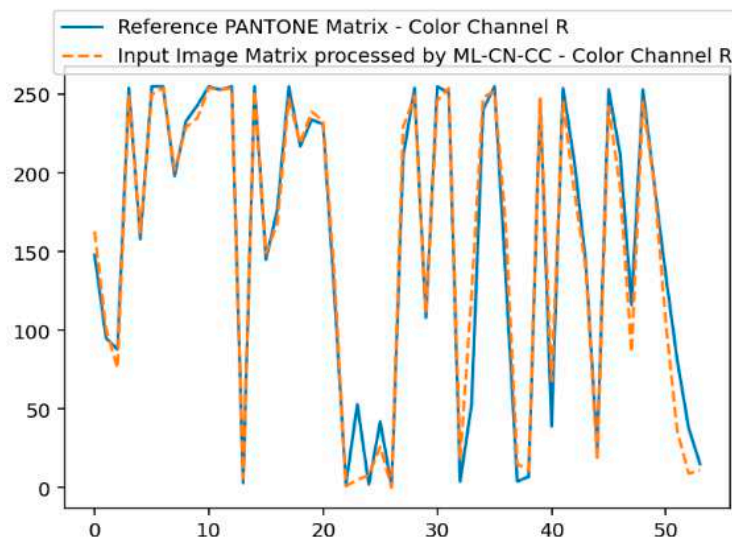


Figure 9. Comparison of the 54 values of channel R for the matrix resulting from processing the Input Image Matrix with the ML-CN-CC algorithm and the Reference PANTONE Matrix.

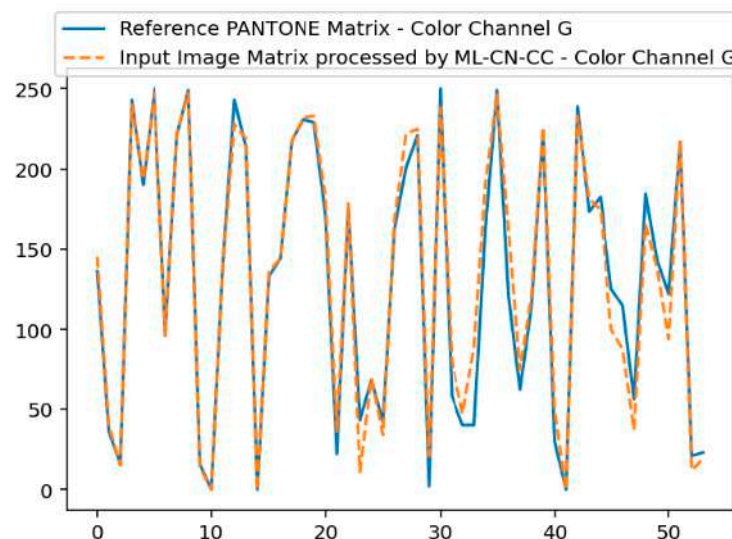


Figure 10. Comparison of the 54 values of channel G for the matrix resulting from processing the Input Image Matrix with the ML-CN-CC algorithm and the Reference PANTONE Matrix.

Subsequently, the trained model is applied to all pixels of the input image to normalize the color of the entire image (Normalized image, see Figures 12 and 13).

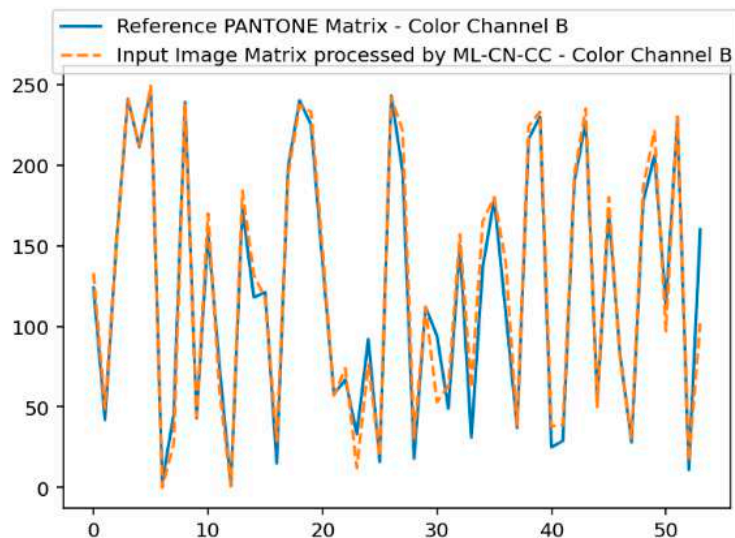


Figure 11. Comparison of the 54 values of channel B for the matrix resulting from processing the Input Image Matrix with the ML-CN-CC algorithm and the Reference PANTONE Matrix.



Figure 12. (a) Input image and (b) its version processed using the ML-CN-CC algorithm (Normalized image).

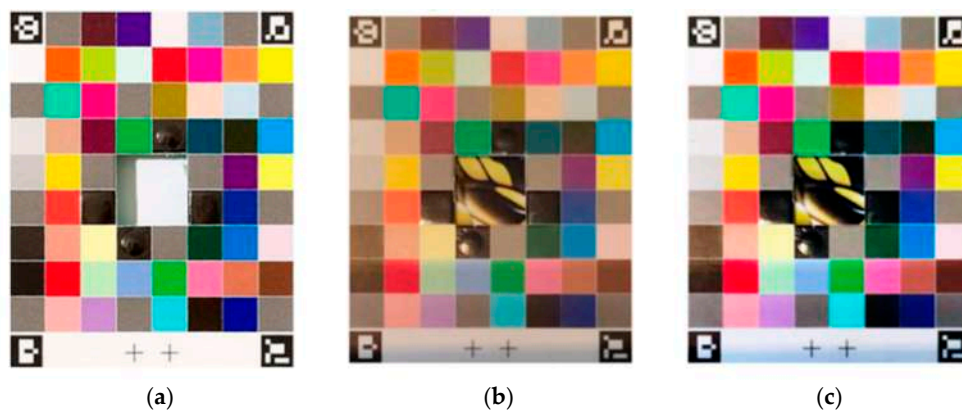


Figure 13. (a) Reference PANTONE, (b) Input Image PANTONE and (c) its version processed using the ML-CN-CC algorithm (Normalized PANTONE).

The loss function of the neural network is presented in Figure 14.

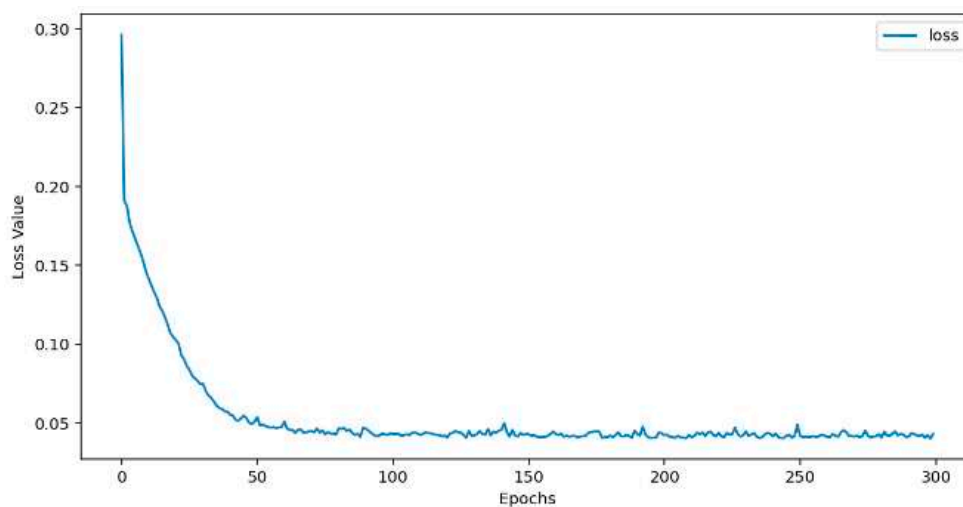


Figure 14. Loss function for the ML-CN-CC neural network.

Additionally, as further validation, the 54 distinct colors of the Reference PANTONE are compared with those from the Normalized PANTONE for 500 test images. The comparison is performed using the mean squared error (MSE), which measures the average squared difference between the normalized values produced by the model and the reference PANTONE values. To achieve this, the MSE value is calculated by comparing the 54 RGB values of the two matrices, measuring the squared difference between the corresponding elements, and then averaging those values, resulting in the values shown in Figure 15. The mean value obtained is 0.0406.

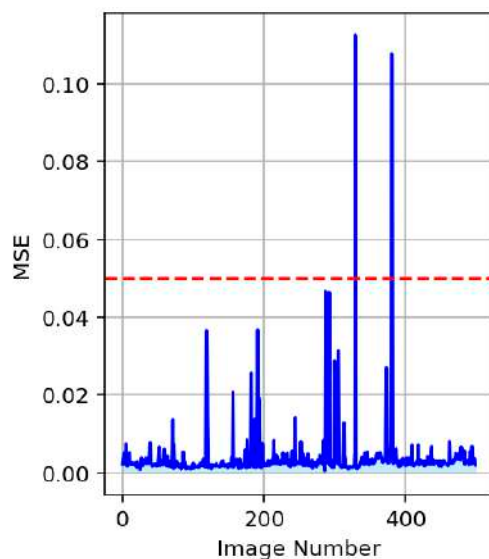


Figure 15. Mean squared error (MSE) for 500 test images.

Moreover, this comparison is also performed using the CIE Delta E 2000 formula described previously. The values of ΔE_{00} can range from 0 to infinity, where:

- $\Delta E_{00} \leq 1$: ideal for imperceptible differences.
- $1 < \Delta E_{00} \leq 3$: acceptable in most applications.

Figure 16 shows the results of the analysis made for 500 test images. For each image, ΔE_{00} is calculated between the 54 reference PANTONE colors and the 54 normalized PANTONE colors, and then the average value is calculated. Since the color difference was

defined in the CIE Lab color space, we first transferred the color values in RGB color space to CIE Lab color space and then calculated the color differences. The mean value obtained is 2.13694.

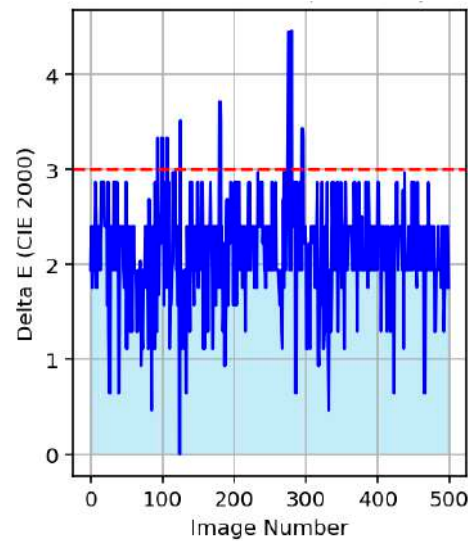


Figure 16. Mean value of the ΔE_{00} for 500 test images.

Finally, a double-check is performed by normalizing the Reference PANTONE image using ML-CN-CC, expecting the resulting normalized image to be identical to the original. To evaluate this, the structural similarity index (SSIM) [14] between the Reference PANTONE and the Normalized PANTONE is calculated. SSIM is a metric widely used to evaluate image quality by comparing a generated or modified image with a reference image. Unlike the mean squared error (MSE), SSIM measures the perceived similarity between images by considering structural characteristics, luminance, and contrast rather than point-to-point differences.

An SSIM value of 1.0 indicates that the images are identical in terms of structure, luminance, and contrast. Values between 0.8 and 1.0 suggest high similarity, where the images are nearly indistinguishable. Values between 0.5 and 0.8 indicate moderate similarity, with perceptible differences but overall structural resemblance. Values from 0.0 to 0.5 indicate low similarity, with significant differences between the images, while an SSIM value of 0.0 indicates no similarity at all.

In this case, the obtained SSIM value is 0.9778, further confirming the proper performance of the ML-CN-CC network (see Figure 17).

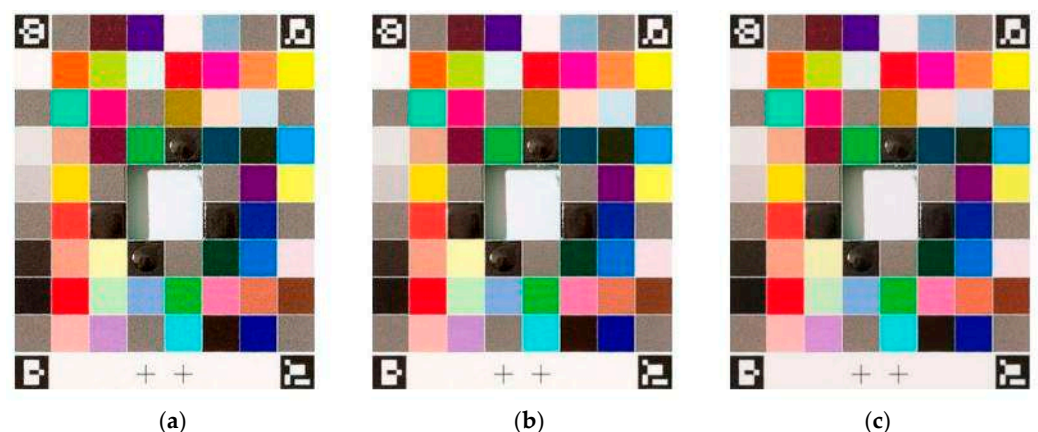


Figure 17. (a) Reference PANTONE (b) Input Image PANTONE (equal to Reference PANTONE) and (c) Normalized PANTONE.

3.2. Results for the Algorithm Based on GAN for Color Normalization Simulating the Use of a Color Checker (GAN-CN-CC)

Once the GAN-CN-CC network is trained, only the Generator is used to produce images that simulate the ML-CN-CC processing from the input images. For this, the input image, with or without a marker, enters the Generator, which processes it and produces an output image that simulates the processing of the ML-CN-CC algorithm (Figure 18).

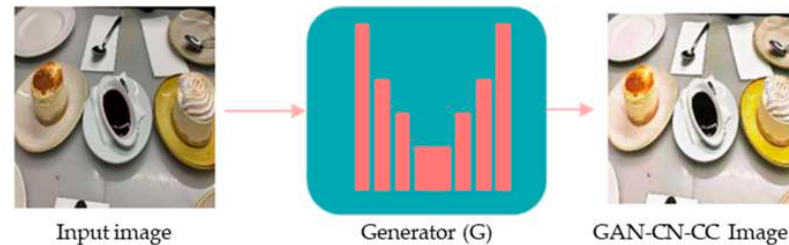


Figure 18. Generator network. The U-Net Generator is responsible for producing images that replicate the ML-CN-CC processing based on the input images.

To validate that the Generator balances realism and fidelity effectively, the performance of our network is validated using quantitative and qualitative methods:

Quantitative evaluation

- Loss functions.
- Pixel-by-pixel MSE differences.
- Structural similarity index.
- Fréchet Inception Distance (FID) [15], which compares the distributions of features extracted from real and generated images using a pre-trained inception network, measuring the similarity of their means and covariances. A small FID value indicates the generation of high-fidelity images.
- Number of parameters (NoP) reflects the number of parameters in networks.
- Floating point of operations (FLOPs) evaluates the computation cost of networks.

Qualitative evaluation

- Visual inspection of generated images to assess realism and adherence to input conditions.

The quantitative performance of the GAN-CN-CC network was evaluated using various metrics, including Generator loss (adversarial loss, L1 loss, and color loss) as well as the Discriminator loss (real loss and fake loss) (Figures 19–21).

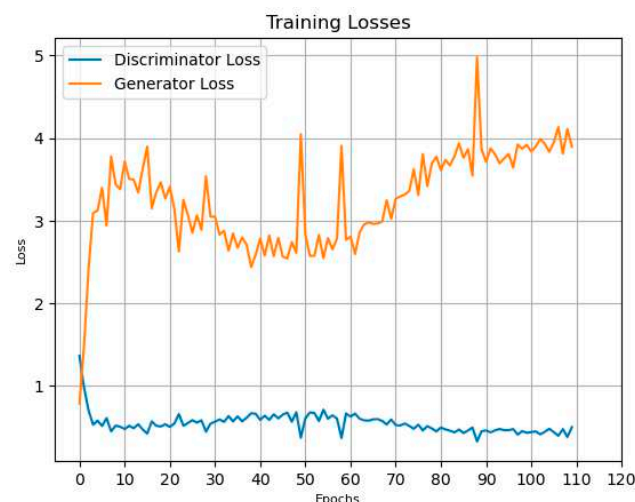


Figure 19. Loss function evolution (Generator loss and Discriminator loss).

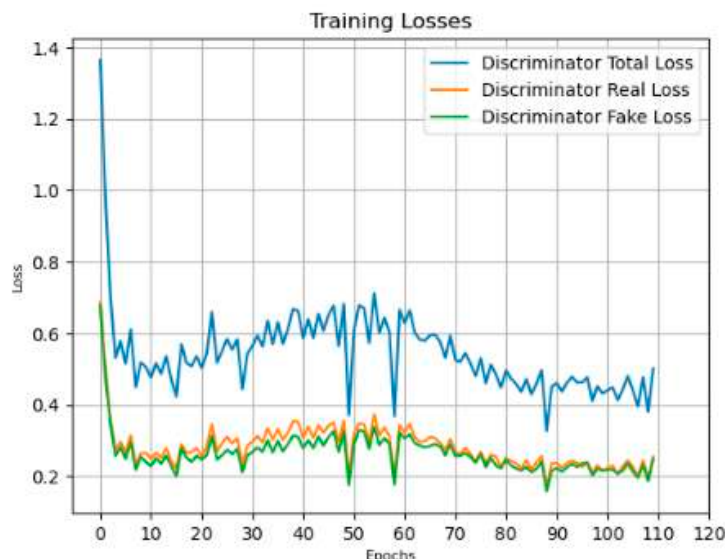


Figure 20. Loss function evolution (Discriminator loss, Discriminator real and fake loss).

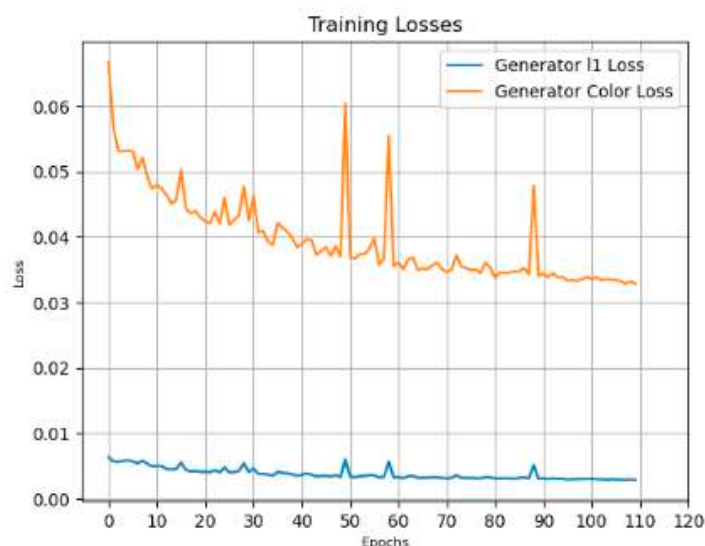


Figure 21. Loss function evolution (Generator L1 loss and Color loss).

To validate the performance of the GAN-CN-CC network from a quantitative perspective, we continue comparing a dataset of 60 test image pairs consisting of the image normalized with ML-CN-CC using a color checker and the one generated by the GAN-CN-CC Generator without a color checker. As explained previously, the input images were not used in the training of the GAN-CN-CC network. For this comparison, we used the pixel-by-pixel MSE difference, and the results are shown in Figure 22. The mean value obtained is 0.03053.

Next, we measured the SSIM index for the 60 pairs of images, as shown in Figure 23. The mean value obtained is 0.65956.

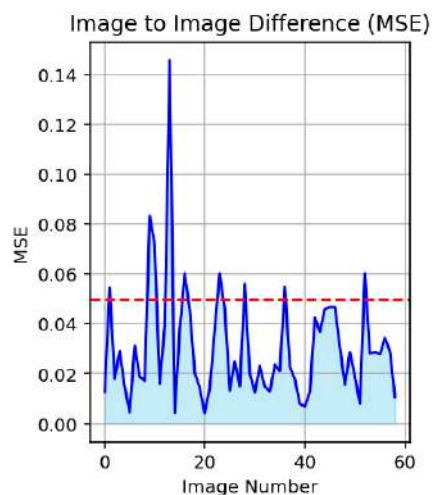


Figure 22. Pixel-by-pixel MSE differences between the ML-CN-CC and the GAN-CN-CC images (60 pairs of test images).

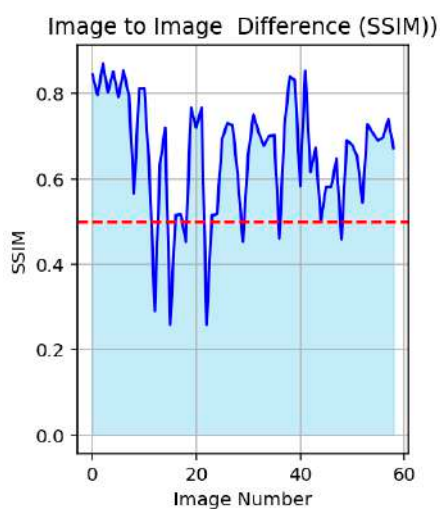


Figure 23. SSIM index between the ML-CN-CC image and the GAN-CN-CC image (60 pairs of test images).

The Fréchet Inception Distance (FID) is also analyzed. Since this metric requires a large number of images, it is evaluated for the entire training set, resulting in a value of 35.71962. A low FID score (<50) like the one obtained indicates that the generated images are similar to the real ones, both in quality and diversity.

Finally, the Number of Parameters (NoP) and floating point operations (FLOPs) are the quantitative metrics used to measure model efficiency. The values obtained for the GAN-CN-CC Generator network are 54,426,115 (NoP) and 12.09657 (GigaFLOPs), respectively.

From a qualitative perspective, Figure 24 presents examples of outputs generated by the GAN-CN-CC Generator, illustrating its ability to perform accurate color normalization across diverse input conditions. To accomplish this, the following method was used: First, an image was captured with a color checker. Then, this image was processed using the ML-CN-CC and GAN-CN-CC algorithms. Next, the same image was taken under the same lighting conditions but without the color checker. Finally, this image was processed using the previously created ML-CN-CC algorithm and the GAN-CN-CC algorithm.

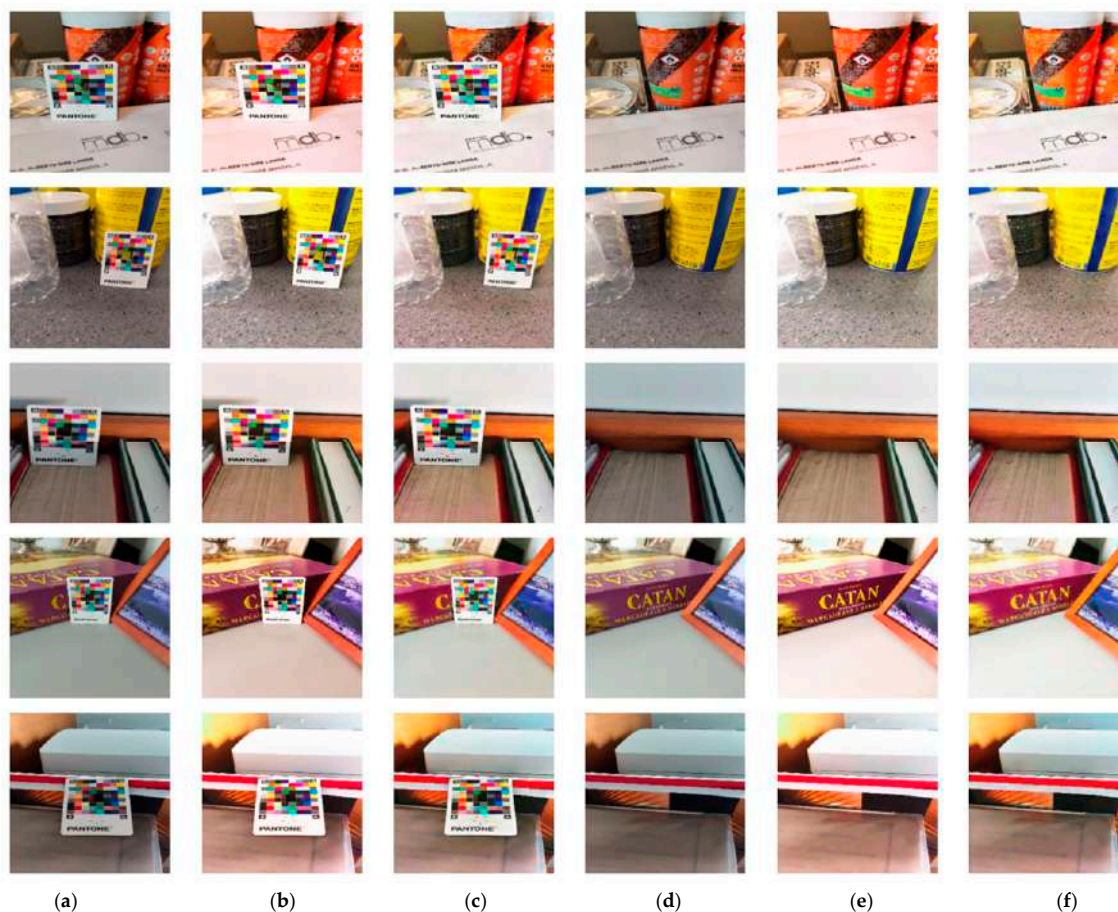


Figure 24. Example of images processed with the ML-CN-CC and GAN-CN-CC algorithms. (a) Input image with color checker (b) Image “a” processed by ML-CN-CC algorithm (c) Image “a” processed by Generator (d) Same input image without color checker (e) Image “d” processed by ML-CC-CN algorithm created in “b” (f) Image “d” processed by Generator.

As we can see in Figure 24, GAN-CN-CC shows limitations in generating accurate white and black tones. This is likely due to the fact that such extreme values lie at the boundaries of the input data normalization range, and the generator does not explore or reproduce these boundary regions effectively during training.

3.3. Comparative Evaluation of GAN-CN-CC with State-of-the-Art Color Normalization Methods

This section presents a comparative analysis of the proposed GAN-CN-CC method against standard color normalization techniques. First, we compare it with the well-known Gray World method [3], which assumes that the average color of a natural scene should tend toward a neutral gray. In other words, the mean values of the R, G, and B channels should be approximately equal. When this condition is not met—for example, if the blue channel has a significantly higher mean than the others—the algorithm interprets this as a blue color cast and adjusts each channel proportionally to compensate for the imbalance.

Second, we consider the Max-RGB method [16], which assumes that, in a well-balanced image, the brightest pixel in each color channel (R, G, B) should represent the white point under the scene’s illumination. Based on this assumption, the method rescales each channel so that the maximum intensity becomes equal across all channels, followed by a common scaling to reconstruct the balanced image.

To assess the performance of the GAN-CN-CC normalization process in comparison to the aforementioned methods, we employ the normalized median intensity (NMI) metric.

NMI is used to quantify the intensity variation within a population of images after the normalization process and is commonly applied to evaluate color consistency [17].

A set of images is considered more consistent when the standard deviation of the NMI (NMI SD) and the coefficient of variation of the NMI (NMI CV)—defined as the standard deviation divided by the mean—are minimized. Therefore, the lower the NMI SD and NMI CV, the more consistent and reliable the color normalization method is considered to be.

As shown in Table 2, the proposed GAN-CN-CC method achieves the lowest coefficient of variation (CV), while maintaining a standard deviation (SD) comparable to conventional methods. In terms of execution time, all evaluated approaches are computationally efficient, ranging from approximately 0.4 ms for Max-RGB to 68 ms for GAN-CN-CC.

Table 2. Quantitative metrics used to compare the GAN-CN-CC normalization method with current state-of-the-art color normalization techniques. GWO refers to Gray World, and MRGB refers to Max-RGB. NMI SD and NMI CV represent the standard deviation and the coefficient of variation of the normalized median intensity (NMI), respectively. It is important to note that lower values of these metrics indicate better color normalization.

Method	Exec Time (s)	NMI SD	NMI CV
Original images	-	0.17778	0.32832
GWO	0.00140	0.17880	0.33284
MRGB	0.00043	0.18752	0.38284
GAN-CN-CC	0.06809	0.19622	0.32603

The visual performance of the compared methods is shown in Figure 25. As observed in the same figure, GAN-CN-CC exhibits more homogeneous image intensity across different samples.



Figure 25. Images processed by the different methods: (a) Original image, (b) Gray World, (c) Max-RGB, (d) GAN-CN-CC.

4. Discussion

As observed in the literature, color normalization is a key process for facilitating image segmentation and/or classification [18]. Numerous algorithms exist to perform this normalization process, some of which are based on the use of a color checker, which serves as a useful benchmark for illuminant estimation algorithms [19]. However, color checker-based algorithms require the use of a physical marker, which is not suitable or practical in certain environments.

In this study, we propose a color normalization method based on simulating a color checker using a modified Pix2Pix GAN network with a loss function adapted to optimize color similarity (GAN-CN-CC network). This approach achieves the benefits of a color checker-based illuminant estimation algorithm without the need for a physical marker.

The methodology involves first creating a machine learning algorithm based on a color checker (ML-CN-CC), which is then simulated without the color checker using a Pix2Pix GAN network.

The ML-CN-CC algorithm demonstrates the following performance metrics for 500 test images: 0.0406 (mean value of the mean squared error) and 2.13694 (mean value of the ΔE_{00}).

The results obtained also demonstrate adequate performance of the GAN-CN-CC network in simulating the behavior of the ML-CN-CC algorithm without a calibrator. Specifically, the following performance metrics are obtained for 60 test image pairs: 0.03053 (mean pixel-by-pixel MSE difference), 0.65956 (mean SSIM). Additionally, the FID index for the entire training set is 35.71962.

In terms of efficiency and computational cost of the GAN-CN-CC model, the Number of Parameters (NoP) is 54,426,115, and the floating point operations (FLOPs) amount to 12.09657 GigaFLOPs.

From a qualitative perspective, visual inspection of the generated images confirms realism and adherence to input conditions.

In terms of comparison with state-of-the-art color normalization methods (Gray World and Max-RGB), the GAN-CN-CC method achieves the lowest NMI coefficient of variation while maintaining a standard deviation of NMI comparable to conventional approaches. Regarding the execution speed of the GAN-CN-CC algorithm, it is approximately 68 milliseconds.

As a point of improvement, future work should focus on developing a GAN-CN-CC network capable of working with images of resolutions higher than 256×256 pixels, thereby eliminating the need to reduce the input image size for the algorithm, which results in a loss of resolution.

On the other hand, GAN-CN-CC struggles to generate white and black colors. That is because extreme values like these are at the boundaries of the input data normalization range, and the GAN-CN-CC generator does not explore or reproduce these boundary values in a sufficiently efficient way.

5. Conclusions

This study highlights the effectiveness of the GAN-CN-CC model as a replacement for traditional color checkers by replicating their functionality in color normalization through learned patterns.

The use of GAN-CN-CC enables fully automated and scalable color normalization workflows, making it particularly suitable for processing large volumes of images. Unlike traditional approaches, which require a physical color checker for each image, our GAN-based solution can perform real-time corrections, offering significant advantages for live analysis and video processing.

However, a notable limitation of GAN-CN-CC is its black-box nature, which contrasts with the transparency of physical color checkers. This necessitates careful validation to ensure reliability in critical applications, such as clinical imaging.

In summary, GAN-CN-CC offers a data-driven, automated, and scalable alternative for color normalization, particularly in contexts where physical tools are impractical or infeasible, paving the way for more efficient and flexible workflows.

Author Contributions: Conceptualization, A.S.L. and I.U.E.; methodology, A.S.L.; software, A.S.L.; validation, A.S.L.; investigation, A.S.L.; writing—original draft, A.S.L.; writing—review and editing, R.R.B. and S.G.C.; supervision, R.R.B. and S.G.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The raw data supporting the conclusions of this article will be made available by the authors on request.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Nassau, K. Fundamentals of color science. In *Azimuth*; Nassau, E.K., Ed.; North-Holland: Amsterdam, The Netherlands, 1998; Volume 1, pp. 1–30. [\[CrossRef\]](#)
2. Salvi, M.; Branciforti, F.; Molinari, F.; Meiburger, K.M. Generative models for color normalization in digital pathology and dermatology: Advancing the learning paradigm. *Expert Syst. Appl.* **2024**, *245*, 123105. [\[CrossRef\]](#)
3. Buchsbaum, G. A spatial processor model for object colour perception. *J. Frankl. Inst.* **1980**, *310*, 1–26. [\[CrossRef\]](#)
4. Finlayson, G.D.; Mackiewicz, M.; Hurlbert, A. Color correction using root-polynomial regression. *IEEE Trans. Image Process.* **2015**, *24*, 1460–1470. [\[CrossRef\]](#) [\[PubMed\]](#)
5. Kucuk, A.; Finlayson, G.; Mantiuk, R.; Ashraf, M. Comparison of Regression Methods and Neural Networks for Colour Corrections. In *London Imaging Meeting*; Society for Imaging Science and Technology: Cambridge, MA, USA, 2022; Volume 3, pp. 74–79.
6. Garrido-Jurado, S.; Muñoz-Salinas, R.; Madrid-Cuevas, F.J.; Marín-Jiménez, M.J. Automatic generation and detection of highly reliable fiducial markers under occlusion. *Pattern Recognit.* **2014**, *47*, 2280–2292. [\[CrossRef\]](#)
7. Senthilkumaran, V. Color correction using color checkers. In Proceedings of the First International Conference on Combinatorial and Optimization, ICCAP 2021, Chennai, India, 7–8 December 2021.
8. Cheng, H.D.; Cai, X.; Min, R. A novel approach to color normalization using neural network. *Neural Comput. Appl.* **2009**, *18*, 237–247. [\[CrossRef\]](#)
9. MacDonald, L.; Mayer, K. Camera Colour Correction using Neural Networks. In *London Imaging Meeting*; Society for Imaging Science and Technology: Cambridge, MA, USA, 2021; Volume 2, pp. 54–57.
10. Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative adversarial networks. *Commun. ACM* **2020**, *63*, 139–144. [\[CrossRef\]](#)
11. Isola, P.; Zhu, J.Y.; Zhou, T.; Efros, A.A. Image-to-image translation with conditional adversarial networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 1125–1134.
12. Ronneberger, O.; Fischer, P.; Brox, T. U-net: Convolutional networks for biomedical image segmentation. In Proceedings of the Medical Image Computing and Computer-Assisted Intervention—MICCAI 2015: 18th International Conference, Munich, Germany, 5–9 October 2015; Springer International Publishing: Berlin/Heidelberg, Germany, 2015; pp. 234–241.
13. Luo, M.R.; Cui, G.; Rigg, B. The development of the CIE 2000 colour-difference formula: CIEDE2000. *Color Res. Appl.* **2001**, *26*, 340–350. [\[CrossRef\]](#)
14. Wang, Z.; Bovik, A.C.; Sheikh, H.R.; Simoncelli, E.P. Image quality assessment: From error visibility to structural similarity. *IEEE Trans. Image Process.* **2004**, *13*, 600–612. [\[CrossRef\]](#) [\[PubMed\]](#)
15. Yu, Y.; Zhang, W.; Deng, Y. *Frechet Inception Distance (FID) for Evaluating GANs*; China University of Mining Technology Beijing Graduate School: Beijing, China, 2021; Volume 3.
16. Gijssen, A.; Gevers, T.; Van De Weijer, J. Computational color constancy: Survey and experiments. *IEEE Trans. Image Process.* **2011**, *20*, 2475–2489. [\[CrossRef\]](#)
17. Salvi, M.; Branciforti, F.; Veronese, F.; Zavattaro, E.; Tarantino, V.; Savoia, P.; Meiburger, K.M. DermoCC-GAN: A new approach for standardizing dermatological images using generative adversarial networks. *Comput. Methods Programs Biomed.* **2022**, *225*, 107040. [\[CrossRef\]](#) [\[PubMed\]](#)

18. Sudhamsh, G.V.S.; Rashmi, R.; Girisha, S. Enhancing Histopathological Image Analysis: A Study on Effect of Color Normalization and Activation Functions. In Proceedings of the International Conference on Computation of Artificial Intelligence & Machine Learning, Jaipur, India, 18–19 January 2024; Springer Nature: Cham, Switzerland, 2024; pp. 220–232.
19. Hemrit, G.; Finlayson, G.D.; Gijzen, A.; Gehler, P.; Bianco, S.; Funt, B.; Drew, M.; Shi, L. Rehabilitating the colorchecker dataset for illuminant estimation. *arXiv* **2018**, arXiv:1805.12262. [[CrossRef](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.