DOI:10.48047/IJFANS/V11/I12/194

A Deep Learning-based Approach for Colorization of Grayscale Images and Videos

Rama Devi Gunnam[1], Assistant Professor, Department of CSE,
Vasireddy Venkatadri Institute of Technology, Nambur, Guntur Dt., Andhra Pradesh.

Gurram Harini[2], **Bapathu Anitha Reddy**[3], **Gurram Bhumika**[4], **Bikki Sai Pavan Kumar**[5]
[2,3,4,5] UG Students, Department of CSE,
Vasireddy Venkatadri Institute of Technology, Nambur, Guntur Dt., Andhra Pradesh.
[1,2,3,4,5] ramarajesh95@gmail.com[1], harinigurram2001@gmail.com[2], bapathuanitha14@gmail.com[3], bhumikagurram12@gmail.com[4], saipavanbikki02@gmail.com[5]

Abstract

Colorization is the process of converting grayscale photos into colorful ones that are more visually appealing. Previously, a wide range of colorization techniques has been developed, which require the involvement of the human brain which consumes a lot of time and energy. In today's world, there are many procedures that will automatically convert the grayscale image to a color image. Most of the conversion techniques incorporate elements of deep learning, machine learning, and art. This study gives a novel technique for coloring grayscale images that makes use of GAN and U-Net model characteristics. By using this technique, the model is able to learn how to colorize images from a trained U- Net. Additionally, the Fusion layer is used to combine the global priors for each class with the local information finds for each class, which are based on small image patches. This produces colorization outcomes that are more attractive on a visual level. Finally, the results of the method were obtained by doing an evaluation based on user research and comparing it to the state-of-the-art.

Keywords: GAN, U-Net, RGB, Lab, deep learning, colorization

Introduction

Humans are easily able to decide which color should be assigned to each element in a picture by using the knowledge they already have about the relevance of the objects depicted. Additionally, people have the ability to predict the colors of objects based on a certain level of subjective emotion. This is one of the most difficult task for a machine to complete because it calls for the machine to simulate both human knowledge and emotion. Due to this, older colorization systems frequently required human input in some manner .

A rank-3 array containing the image's height, width, and color is returned when we load a photo. The color information for our image is contained on the last axis of this array. Each

Research paper © 2012 IJFANS. All Rights Reserved, UGC CARE Listed (Group -I) Journal Volume 11, Iss 12, Dec 2022

individual pixel is associated with three numbers, and these numbers are used to represent color in the RGB color space [3]. We again utilize three integers to represent a single pixel in the L*a*b color space, but this time, each of these numbers stands for a different property. The distributions of the colors green, red, and yellow-blue that make up each pixel are encoded by the a and b channels [1]. The following graphic (Fig.1) presents an analysis of the L*a*b color space, which separates each channel so that it can be considered independently.

Problem Identification

All of the colorization papers we read and colorization tools we looked at used the L*a*b color space to train the models rather than the RGB color space. In order to teach a model to colorise pictures, we need to start by showing it a grayscale image, then wait with bated breath as the model makes an effort to colorize the picture, and finally evaluate the results. We can provide the model with the L channel, which is the grayscale image, and ask it to forecast the other two channels, *a and *b, when we make use of the L*a*b technique. As soon as it has made its prediction, we concatenate all of the channels, and what we get is an image full of vibrant colors. However, if you choose to work with RGB, you will first need to convert your image to grayscale. This is because RGB works with colors rather than shades of grey. After that, you will need to provide the model with the grayscale image, and then cross your fingers and hope that it provides you with three numbers to work with. Working with three numbers instead of two is a significantly more difficult and unreliable task. This is due to the fact that there are a significantly greater number of possible combinations with three numbers than there are with two numbers.

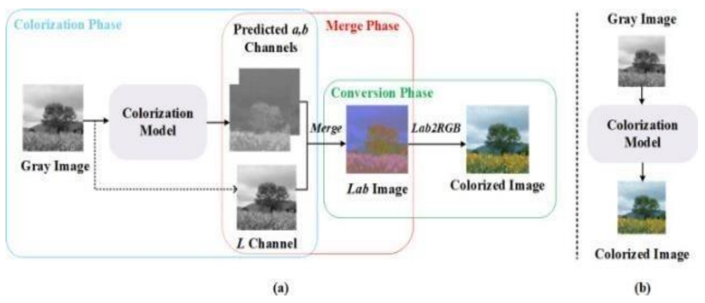

Fig.1. Phases of colorization

Literature survey

One of the most popular techniques for coloring images is to cover it in user-specific colored scribbles. Using this technique, A method was devised by Levin et al.[2] which utilizes optimization algorithms to convert a grayscale image into a colorized one. By resolving a quadratic cost function created from the brightness disparities between a pixel and its neighbours, this was accomplished.

Huang et al.[1] modified the method to prevent the colors from overflowing into the object edges in the image. Yatziv and Sapiro[3] introduced a fast colorization method that utilizes weighted geodesic distances and chrominance mixing .

In addition, there are some deep learning-based methods that have been proposed to address long-range propagation in image recolorization. For example, Zhang et al.[4] proposed a deep recurrent neural network for colorization that uses a novel method for global reasoning to propagate information between pixels. Huang et al.[1] proposed a multi-scale deep neural network that incorporates both local and global information to achieve long-range propagation in image recolorization . These deep learning-based methods have shown promising results in achieving accurate and natural-looking image recolorization with long-range propagation. These techniques however, significantly rely on to human input and require trail and error in order to get a satisfying result. Unlike scribble-based methods, examples-based colorization approaches rely on a reference image to determine appropriate colors for the input image that are similar in nature. Color transfer techniques are frequently used to recolor color images Reinhard et al.[5] proposed a method for automatic color correction of images based on statistics of the image colors. Tai et al.[6] introduced a color transfer method that utilizes global and local statistics of the input and target images. Pitie et al.[7] proposed a non-parametric approach to color transfer that utilizes a patch-based representation of images. Wu et al.[8] introduced an automatic colorization method that utilizes sparse user input to propagate colors throughout the image.

Mapping functions are utilized in these approaches to transfer the color distribution of a reference image to the input image, following color statistics computation in both images. By comparing the brightness and texture information between photographs. It seems like there have been many advancements and improvements made to the colorization process over time, with various techniques and strategies utilized. The integration of supervised classification algorithms and global optimization techniques that consider multimodality and predicted possible colors for each pixel is certainly promising for further improving the accuracy and quality of colorized images.

Gupta et al. [9] used super pixel segmentation to divide the input image and the reference image into multiple super pixels, and then performed feature matching and space voting to match super pixels between the two images. The matched super pixels were used to transfer color from the reference image to the input image. The time-consuming effort of providing appropriate reference photos that are equivalent to the original image is necessary

Research paper © 2012 IJFANS. All Rights Reserved, UGC CARE Listed (Group -I) Journal Volume 11, Iss 12, Dec 2022

for these methods, nevertheless. One of the advantages of using reference-based colorization methods is that they can handle lighting discrepancies between the input and reference images. This is because the colorization is based on the reference image, which provides information about the desired colors regardless of the lighting conditions. Therefore, as long as the reference image contains the necessary color information, the colorization will be robust to lighting variations in the input image.

They employ significantly less training data than our method, which severely restricts the kinds of photos it may be applied to. Moreover, the use of a strong segmentation model is essential for image segmentation. Nevertheless, in cases where the image lacks any segmentation classes, the results are usually below par because of the method's heavy reliance on segmentation. A large dataset and end-to-end learning enable us to expand our model across a range of image formats.

In recent years, convolutional neural networks (CNNs) have become the standard approach for many image-related tasks, including super resolution, image generation, image segmentation, and more. CNNs are effective because they can learn hierarchical representations of the input data, allowing them to capture complex features at multiple levels of abstraction.

For example, in super resolution, CNNs can learn to map low-resolution images to high-resolution images by learning the underlying patterns in high-resolution data. Similarly, in image segmentation, CNNs can classify each pixel in an image based on its features, allowing for more accurate and efficient segmentation .
Moreover, CNNs can process images of any resolution, making them highly scalable and adaptable to various tasks. Some CNN architectures, such as U-Net, are designed to incorporate images at different scales, enabling them to learn both local and global features in the input data.

Eigen and Fergus proposed a joint task network for depth estimation, surface normal estimation, and semantic labelling, where all three tasks are trained simultaneously. The network is based on a multi-scale architecture that processes features at different scales and then combines them for prediction. Wang et al. proposed a multi-scale and multi-context deep neural network for salient object detection, which combines features from different scales and contexts to capture both local and global information. The network consists of multiple branches, each processing features at a different scale and context, and a fusion module that combines the features for prediction. processing. Fusion has also been applied on a variety of scales. In contrast, our fusion technique integrates local characteristics into the global picture feature vector.

A global feature vector for the entire image can improve efficiency by utilizing the class labels. In some cases, it can also improve accuracy by capturing important information about the entire image, such as overall texture or color composition. This approach has been utilized in various computer vision tasks, including image classification, object detection, and semantic segmentation. For example, the Spatial Pyramid Pooling (SPP) network proposed by He et al [10]. uses global max pooling to generate a fixed-length feature vector for an image that can be used for image classification and object detection.

We are the first to offer a system that facilitates end-to-end learning by merging global and local features, as far as we are aware.

Methodology

The use of structures in image modelling, such as regression or per-pixel classification, can lead to challenges in image-to-image translation. These approaches consider the output space to be "unstructured," where each output pixel is considered to be independent of every other output pixel given the input image. To address this issue, Conditional GANs introduce a structured loss function that penalizes the overall configuration of the output [7]. By doing so, the generator G is less likely to deceive the discriminator. Additionally, both the generator and discriminator take into account the input edge map, which distinguishes them from unconditional GANs.

Various techniques have been developed to address the issue of loss in image modelling, including conditional random fields, SIM metric, feature matching, nonparametric losses, convolutional pseudo-prior, and covariance-based losses. Among these techniques, the conditional GAN is particularly notable for its ability to learn a loss function that penalizes any deviation from the target structure. Image-conditional models, including the GAN, have been successfully applied in generating product photos, synthesizing images from sparse annotations, and predicting images from normal maps. In some studies, the GAN was used unconditionally for image-to-image mappings, but other variables such as L2 regression were incorporated to make the output dependent on the input [6]. These techniques have been shown to produce positive outcomes in various image applications, including inpainting, future state prediction, user-directed image alteration, style transfer, and super resolution.

We employ a convolutional "Patch GAN" classifier as our discriminator, and a "U- Net"-based architecture as our generator, which only penalizes structure at the size of individual picture patches. This is in contrast to prior studies. Initially, it was suggested to record local style statistics using a notion similar to Patch GAN. Here, we demonstrate that this method works for a larger variety of issues and look at the impact of adjusting the patch size.

Implementation

As previously mentioned, we will construct a conditional GAN and utilize the L1 loss function. A GAN is comprised of a generator model and a discriminator model, which work together to solve problems. In our example, the generator model creates a picture with two channels, one channel representing a and the second channel representing b, starting with a grayscale image, which is an image with only one channel. The discriminator then assesses this new three-channel image to decide whether it is real or fraudulent depending on the combination of all three channels. Starting with these two freshly formed channels, the discriminator concatenates them with the input grayscale image before assessing them. For the discriminator to recognize that these pictures are genuine, they need to see some real pictures (again, three-channel pictures in Lab color space). These images should be used to convey this information to the discriminator.

To gain a deeper understanding, let's examine the variables involved in a conditional Generative Adversarial Network (GAN). We can assume that the grayscale image is represented by the variable X, the input noise for the generator is represented by Z, and the generator's output on the two channels of interest (or the two cold channels of a real image) is represented by the variable Y. In this context, we can use the letters G and D to denote the discriminator and generator models, respectively. Based on this information, we can subtract a certain amount from our conditional GAN to improve its performance.

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

The condition that will be applied to both players in this game is shown by the fact that both models have the value x assigned to them. You would assume that we will feed the generator a "n"- dimensional vector of random noise, but in fact, we will use dropout layers within the generator architecture to introduce the noise.

The early loss function aids in creating realistic-looking, vibrant photos. Nevertheless, To provide additional assistance to the models and add some supervision to our task, we incorporate the L1 loss function (also referred to as mean absolute error) which measures the difference between the projected colors and the actual colors.

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

Research paper © 2012 IJFANS. All Rights Reserved, UGC CARE Listed (Group -I) Journal Volume 11, Iss 12, Dec 2022

Although the model can learn to colorize images using only the L1 loss function, it may adopt a conservative approach and often opt for colors such as "grey" or "brown" when faced with difficulties in selecting the perfect color. The model may utilize these colors to minimize the L1 loss. Nonetheless, the model can still colorize images even with only the L1 loss function, as it is comparable to he blurring effect of L1 or L2 loss in super resolution tasks. The L1 loss function is preferred over the L2 loss function because it has a lesser impact on producing greyish-looking images. Therefore, our combined loss function is as follows:

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

The λ coefficient is used to balance the effects of the two losses on the overall loss. It is important to note that the L1 loss is not included in the discriminator loss.

We are now using 8,000 images from the COCO dataset that were uploaded from PyTorch for training purposes. As a result, just 0.6% of the total set used in the paper is represented by our training set. The dataset contains a wide range of environments and places, giving the model the capacity to colorize.

After reading an RGB image and converting it to Lab color space, the first (grayscale) channel and the color channels were split as our inputs and targets for the models, respectively. The images were then horizontally flipped and resized. The data loaders were then created after that.

Then, to act as the generator for our generative adversarial network, we constructed a U-Net (GAN). The specifics of the code are outside the scope of this article, but the most important concept to grasp is that it builds the U-Net iteratively by starting with the middle section (the lower portion of the U shape) and adding down- and up-sampling modules to the left and right of that middle module, respectively, until it reaches the input module and the output module. To help you better understand what is going on in the code, take a look at the following graphic (Fig.2):

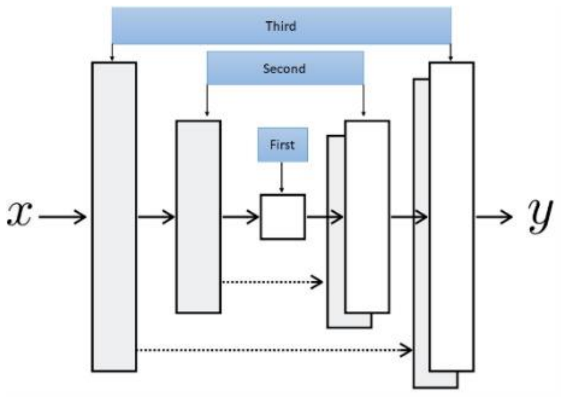

Fig.2. U-Net architecture

Research paper © 2012 IJFANS. All Rights Reserved, UGC CARE Listed (Group -I) Journal Volume 11, Iss 12, Dec 2022

The diagram's blue rectangles show the order in which the code generates the linked modules. Despite the fact that the U-Net we will build has more levels than what is depicted in this diagram, it serves as a good example. Moreover, keep in mind that the method descends eight layers, Suppose we start with a 256 by 256 image. In the U-Net, we first obtain a 1 by 1 (256 / 28) image, which is then up-sampled to generate a 256 by 256 image. Our discriminator has a rather straightforward architectural design. The first and last blocks don't employ normalization, and the final block doesn't have an activation function (it is embedded in the loss function we will use).

The discriminator employed in this instance was a "Patch" Discriminator. By using a straightforward discriminator, the model's output is a single number known as a scaler that conveys how much the model believes the input, in this case the entire image, is real (or fake). For each patch of the input image, which is typically 70 by 70 pixels, a patch discriminator generates a single number.
Each of these patches is evaluated by the model to see if it is a fake or not. Although the model's output form in this instance is 30 by 30, this does not necessarily imply that our patches are also 30 by 30. The actual patch size can be estimated by calculating the receptive field of each of the 900 output numbers (30 multiplied by 30 and determining that it is 70 by 70 in our example.

Results & Conclusion

We have obtained the dataset from COCO, which consists of 20,000 images, of which 8000 were utilized for the training process and 2000 were utilized for the testing procedure. The proposed model, a GAN with U-net being the backbone for the generator, performed brilliantly when it came to converting grayscale images to color images as they appear in the following graphic (Fig.3).

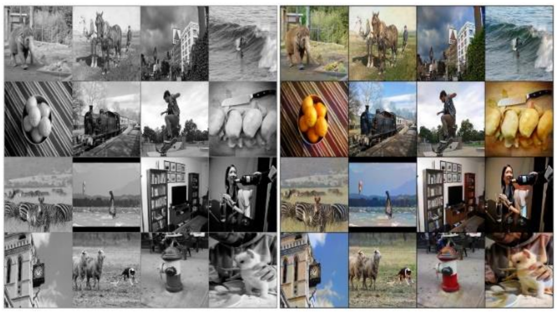

Fig.3. Colorized images

Future Scope

The loss functions can be further optimized to increase the resolution of the grayscale videos.

References

[1] H.Yi , T.Yi , C.Cheng and W.Wen , "An adaptive edge detection based colorization algorithm and its applications", 2005 International Conference on Multimedia, DOI:10.1145/1101149.1101223.

[2] A. Levin, D. Lischinski, and Y. Weiss, "Colorization using optimization," ACM Trans. Graph., vol. 23, no. 3, pp. 689–694, 2004.

[3] L. Yatziv and G. Sapiro, "Fast image and video colorization using chrominance blending," IEEE Trans. Image Process., vol. 15, no. 5, pp. 1120–1129, May 2006.

[4] Zhang B, Li J, Lü Q. Prediction of 8-state protein secondary structures by a novel deep learning architecture. *BMC Bioinformatics*. 2018;**19**(1):293.

[5] S. Olsen, R. Gold, A. Gooch and B. Gooch, "Recovering color from black and white photographs," 2010 IEEE International Conference on Computational Photography (ICCP), 2010, pp. 1-8, doi: 10.1109/ICCPHOT.2010.5585088.

[6] Tai Y W, Jia J, Tang C K. Local color transfer via probabilistic segmentation by expectation-maximization. in IEEE Computer Society Conf on Computer Vision and Pattern Recognition 2005;1:p.747-754.

[7] F. Pitie, A.C. Kokaram, and R. Dahyot, "Automated colour grading using ´

colour distribution transfer," Computer Vision & Image Understanding, vol. 107, no. 1–2, pp. 123–137, 2007.

[8] B. Sheng, H. Sun, S. Chen, X. Liu, and E. Wu, "Colorization using the rotation-invariant feature space," IEEE Comput. Graph. Appl., vol. 31, no. 2, pp. 24–35, Mar./Apr. 2011.

[9] R.Gupta , D.Rajan , E.Sin , A. Yong , "Image Colorization Using Similar Images, " 2012 International Conference on Multimedia. DOI:10.1145/2393347.2393402.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun," Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition ,"arXiv:1406.4729v4[cs.CV] 23 April 2015.