International Journal of User-System Interaction 13(2) 2020, 110-120

Automatic black and white image colorization using generative adversarial networks

Gheorghe-Cosmin Petre[1], Ștefan Trăușan-Matu[1,2]

[1] University Politehnica of Bucharest
313 Splaiul Independenței, Bucharest, Romania

[2] Romanian Academy Research Institute for Artificial Intelligence "Mihai Drăganescu"
Calea 13 Septembrie nr. 13, Bucharest, Romania
E-mail: petregcosmin@gmail.com, stefan.trausan@upb.ro

Abstract. This paper presents an approach to automatically colorize black and white images using artificial intelligence. Because different colors can be used to paint the same object, a machine learning algorithm can learn these colors during the training phase and generate realistic images. The solution is based on generative adversarial networks, a machine learning framework that uses two neural networks (the generator and the discriminator) to generate new data. The two models are trained simultaneously using the GPU support offered by Google Colab, the generator trying to deceive the discriminator with different colorizations methods and the discriminator classifying the images received from the generator as synthetic images. After the training phase, the resulting generator model can produce colorful versions of grayscale images from different datasets used as input. For a better comparison of the results, both qualitative and quantitative methods are used for the evaluation of the trained models, and also a Turing test, the algorithm obtaining a 23% chance of misleading the participants into choosing the generated image from pairs of real and synthetic images.

Keywords: image generation, artificial intelligence, machine learning, convolutional neural networks, generative adversarial network

DOI: 10.37789/ijusi.2020.13.2.4

1. Introduction

Artificial intelligence has become very popular over the past few years, as a result of the development of both hardware components and machine learning algorithms. The original purpose of the artificial intelligence algorithms was to be able to develop systems that can solve complex problems in the same manner as humans do, according to Krzysztof (2018). In time, having data

from various domains and fast processing methods, the application of these algorithms extended to more complex use cases, like voice recognition, autonomous cars, and even medical diagnosis.

The solution proposed in this paper aims to solve the automatic colorization problem, the resulting application producing realistic images that can lead users to the point of not being capable to distinguish between a real-world image or a synthetically generated one. The implementation uses machine learning algorithms, considering the property of the objects to be represented using multiple colors: for example in a landscape image, the color of the sun is a combination of red, orange, and yellow; any of these colors can be used separately for authentic representation.

The use cases of the solution are not limited to producing realistic images of black and white images - the system can be further developed for interesting applications in the art domain and cinematography: repainting popular portraits and colorizing symbolic movies or recordings of historical events.

The paper continues with related work. The third section presents details of the implementation of the proposed approach, followed by a section dedicated to the results of some experiments. The final section of the paper contains some conclusions and future work

2. Related work

In the past, several approaches were using traditional methods, one of them being proposed by Levin (2004). His solution consists of using an interactive annotation with different colors for the segments of an image. One segment of the image contains the pixels that have similar intensities and for all of them, the colorization algorithm should use the chosen color for this region.

One important disadvantage of using traditional approaches is related to the time lost for human intervention. Also, manual intervention can lead in some cases to human errors - actions performed by humans that result in unexpected behavior or failure. In Levin's case, the annotations are very important for the final result. A person has to choose a color for every region in the image and if a bad color is chosen for a segment, the whole image would look unrealistic.

2.1. Colorful Image Colorization

Zhang and Isola (2016) proposed a solution for automatic colorization using deep convolutional networks, the resulting system being able to produce colorful versions of black and white images represented using the CIELAB color space. For the neural network suggested in their approach, it is used a VGG architecture proposed by Simonyan (2014). The architecture is slightly modified to accept as input only the L channel from the CIELAB representation of an image (the L channel represents the lightness - for a black and white image this channel contains all the information) and to generate the two components (a* and b*) that contains the color information. The colored image results from the concatenation of the two generated components and the L used as input.

In this paper, it is highlighted a problem that appears in many alternatives of automatic colorization: the resulting images are desaturated, because of the loss functions inspired by regression problems, the error is calculated as the Euclidean distance between the real and the generated value for each pixel. To solve the desaturation issue, Zhang and Isola (2016) considered the automatic colorization a classification problem, split the color space in bins of different colors, and used a cross-entropy loss function, obtaining for each pixel a probability vector over the possible colors selected initially based on the dataset. For the value of one pixel, it is chosen the color with the highest probability from the vector obtained in the previous step.

Figure 1. Effects of color rebalancing - 1st column images without rebalancing, 2nd column images with rebalancing, 3rd column real images – Zhang and Isola (2016)

Another modification made to avoid the desaturated colors was implemented using the color rarity in the loss function at training. In this way, the values of the pixels composing the background do not affect the intensity of other colors from smaller objects.

The system is trained and validated with images from the ImageNet dataset and for the evaluation of the results they considered the accuracy obtained by a pre-trained VGG classifier on the images generated for the validation set. The accuracy dropped from 68.3% for real images to 56% for the synthetic versions and the difference is not excessive, meaning that the colorizations contain sufficient information to identify objects with reasonable accuracy (from the classifier perspective).

For the same purpose of evaluating the results, Zhang and Isola (2016) performed a Turing test (Turing, 1950) as follows: they asked the participants to choose the fake image from 40 pairs consisting of both generated and real versions of the images. The algorithm succeeded in deceiving the participants in 32% of the cases, a decent percentage considering that in the case of two identical images the resulting percentage would be 50%.

2.2. ColorUNet

In 2018, researchers from Stanford University came with a solution based on the U-Net architecture. Billaut (2018) tried to obtain satisfactory results using a network created from scratch inspired by how the layers are organized in the U-Net and the goal was to obtain realistic images, not necessarily identical to the real ones.

This time, for the color space, the authors have chosen the YUV color space, which has the representation of the channels similar to the CIELAB space: Y represents the brightness of the image, U, and V displaying the color information.

The space color is decomposed into groups of colors, more exactly the most 32 frequent colors from the ImageNet and Sun datasets. If a color is not part of the most frequent ones, it is used the closest one. In this way, the control over the generated colors is more precise.

In the present paper, the colorization is considered a classification problem and for the loss function, the most adequate solution was cross-entropy like in the article proposed by Zhang and Isola (2016). Furthermore, in the

prediction of a pixel is also used the rebalancing of the colors, the weight for a value depending on the rarity of the color.

For better control of the tone presented in the generated images, Billaut (2018) introduced in the formula used for prediction an additional parameter T - the temperature. This parameter affects the intensity of the colors and can be used to hide artifacts of wrong colors.

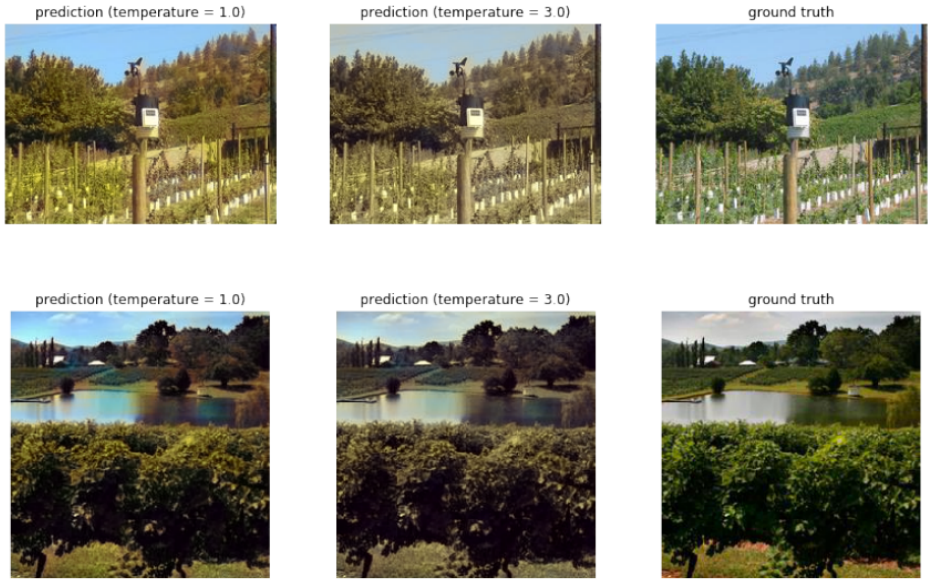

Figure 2. Differences in the results determined by the temperature parameter – Billaut (2018)

To demonstrate the applicability of the solution in different use cases, an extension of the system was implemented to colorize black and white videos by adding at the current network a new layer to stabilize the colors that take into consideration past information, more exactly the colorizations obtained for previous frames. In contrast to the naive method of splitting the video into frames and colorizing the frames individually, the new temporal layer brings the benefit of generating a more realistic colorized video, because in the same scene (and for the same objects) the colors returned by the algorithm are very similar.

3. Implementation

We propose herein an application for automatic colorization based on generative adversarial networks (GAN), introduced by Goodfellow (2014). The solution consists of two convolutional neural networks, the generator and the discriminator, these two components competing one against each other to generate colored images. The purpose of the generator is to create new realistic colorful images and the discriminator interprets and estimates the probability that the results from the generator are fake or real.

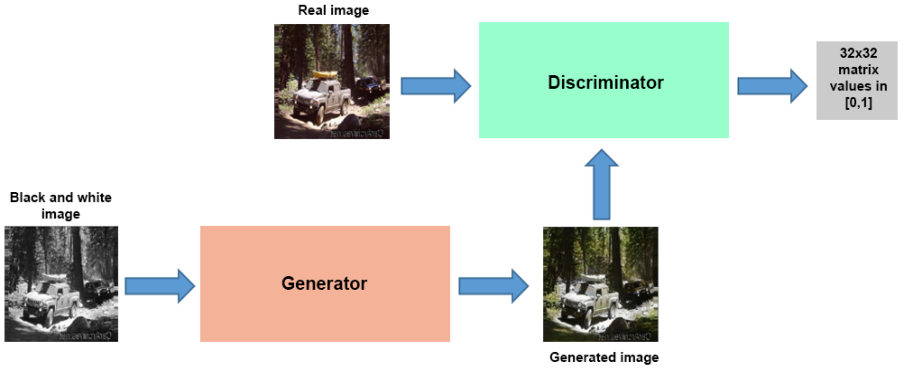

Figure 3. Inputs and outputs for the two components of the proposed GAN

For the implementation of the generator, we used a modified U-Net architecture, convolutional layers, and transposed convolutional layers for the network. The generator receives as input the L channel of a 256x256 resolution image and produces the a* and b* color components from CIELAB space.

For the discriminator, we used a PatchGAN suggested by Isola (2017) for image-to-image translations using adversarial networks. The difference from a normal convolutional network is given by the mapping of the input to a square matrix, instead of a single scalar. A value in the matrix returned represents the probability that a patch with 34x34 pixels from the image is real (the patches are overlapped).

The discriminator has an important role in defining the loss functions during the training phase. For every image from the training dataset, we obtain two matrices from the discriminator by providing as input both the real

and the generated image. For the generator, the error consists of the difference between the matrix resulted from the generated image and a matrix with all the values 1, adding the mean for the difference matrix of the pixel matrices for the generated and real image. For the discriminator's error, we chose the binary cross-entropy between the two generated matrices and two matrices, one with all the values 1 and the other with all the values 0.

The dataset selected for the training and validation consists of images from Places[12], more exactly from the nature category. Places is used mainly for visual cognition and contains approximately 10 million images divided into 400 categories (eg: nature, art, sports). The images have resolutions of 256x256 or 512x512. For this project, we used images with resolution of 256x256 and we split them as follows: 15.000 for training and 3.000 for validation. The dataset size was chosen taking into consideration the time needed for the training phase and it was also influenced by the environment meant for training.

The solution was implemented using Python and Tensorflow. The platform used for training the models is Google Colab connected to Google Drive, where the initial images (the training and validation datasets) and also the resulted images (generated images) are stored. The hardware specifications provided by Google are as follows: Tesla K80 GPU - having 2496 CUDA cores and 12GB GDDR5 VRAM, one single-core Xeon Processors - 2.3Ghz with two threads, 16GB of RAM.

The task of selecting the number of epochs for GANs is a difficult task because if the training is stopped too early, the results can be inconclusive. This is why we used 20 epochs and saved the models every epoch.

To prevent the problems introduced by the GAN architecture revealed by Wiatrak (2019), we applied advanced training strategies, some of the ideas inspired by the work of Salimans (2016):

- We chose different learning rates for the generator and the discriminator improving both the performance and the stability of the system;
- The first step of the algorithm consists of shuffling images for consecutive epochs to reduce the possibility of overfitting;
- We used Adam optimizers to maintain the balance between the two components during the training phase (even if we used initially

[12] http://places2.csail.mit.edu/index.html

different learning rates, the optimizers can adjust these values to increase the stability of the solution);

- The discriminator of a GAN architecture tends to return extreme values - to solve this problem we implemented label smoothing replacing values of 1 with 0.9 in the loss functions of this component.

Therefore, the implementation combines the two ideas proposed by Zhang and Isola (2016) and Billaut (2018). We used a GAN architecture with a modified U-Net network for the generator and a PatchGAN for the discriminator. We additionally applied special training strategies to solve the problems generated from the GAN architecture and chose proper lose functions and network configurations aiming to obtain satisfactory results.

4. Results

The results' evaluation is a hard task especially for GAN based systems because the loss functions do not indicate the state of the entire application, but rather how well a component behaves compared to the other one. This is why we took advantage of both quantitative and qualitative evaluation methods. We did 4 experiments with 20 epochs each as mentioned before.

For the first evaluation method, we tried to reduce the models using a pre-trained model proposed by Szegedy (2015). This Inception model receives as input a colored image and returns a score (inception score) based on how realistic the image is. In most cases, the Inception model depends on multiple variables (contours, diversity, object's placements in the image), but in our case, it is used exclusively to evaluate the colorization. We sorted the models in descending order by score and selected the first 12 models.

As our second evaluation, we inspected manually the results generated for the models previously mentioned and we excluded the models trained using small numbers of epochs because there is a high probability that the artifacts can appear in the resulting images.

After selecting the best model based on a manual investigation (the model corresponding to experiment 4 epoch 18 in Figure. 4) we proposed a Turing test similar to Zhang and Isola (2016) to check how well the system can deceive humans. For this experiment, we chose 40 participants from different areas of studies (medical students, engineering students) with the age in the range of 20-28 years. It should be mentioned that they did not participate in

any other experiments of identifying false images.

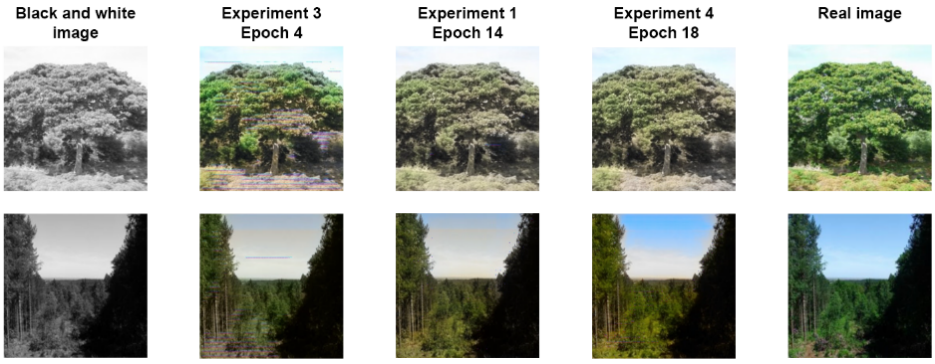

Figure 4. Results of the best 3 models selected using manual inspection

The participants had to choose fake images from 20 pairs of real and generated images. As a result, our model managed to mislead the candidates in 23% of the cases. Even if Zhang and Isola (2016) obtained a 32% chance of misleading the participants, that does not mean our solution is worse because the results are greatly influenced by the subjectivity of the partakers.

Moreover, for some pairs of images, the participants chose the generated images in more than 60%, highlighting the fact that this Turing test depends on the subjectivity of the persons and also on the images used.

5. Conclusions and future work

In this paper we proposed a solution for the automatic colorization using a GAN architecture, the generator is the component that solves our problem. Using mainly convolutional neural networks, we managed to obtain a system that can deceive humans in a reasonable number of cases. Besides the addressed problem, the generator can identify how unique an image is. For example, in Figure. 5 the fake image is different from the real image because in the dataset there are not many images (or maybe none) with a red ground and the generator tries to produce colorizations based on the features learned during the training phase.

Figure 5. The uniqueness of an image

Even if the main component of the solution is the generator, the discriminator can have interesting applications. It can be used separately to measure the quality of generated images using different algorithms designed for problems other than the one presented in this document, similar to how the Inception model works.

As an extension of the application, we would like to test the results produced for black and white videos. For a stabilized colorization we can add one temporal layer at the end of the generator as in the solution suggested by Billaut (2018) to take into consideration the colors used for previous frames.

As future developments, we need to investigate the results provided by the solution proposed using different categories of images or entire datasets (in our experiment we used a single category and approximately 15.000 images). Moreover, for the two components, we should try different configurations or even different architectures (for example the generator can be implemented using a VGG network).

References

Billaut Vincent, Matthieu de Rochemonteix, Thibault Marc. (2018). *ColorUNet: A*

convolutional classification approach to colorization. arXiv:1811.03120

Cios Krzysztof. (2018). *Deep Neural Networks - A Brief History*. Advances in Data Analysis with Computational Intelligence Methods pp.183-200

Goodfellow Ian, Pouget-Abadie Jean, Mirza Mehdi, Xu Bing, Warde-Farley David, Ozair Sherjil, Courville Aaron, Bengio Yoshua. (2014). *Generative Adversarial Networks*. NIPS'14: Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2 December 2014 Pp. 2672–2680

Isola Phillip, Zhu Jun-Yan, Zhou Tinghui, Efros Alexei. (2017). *Image-to-image translation with conditional adversarial networks*. Proceedings of the CVPR, 2017, pp. 5967–5976

Levin Anat, Lischinski Dani, Weiss Yair. (2004). *Colorization using optimization*. In ACM SIGGRAPH 2004 Papers. Association for Computing Machinery, New York, NY, USA, 689–694

Salimans Tim, Goodfellow Ian, Zaremba Wojciech, Cheung Vicki, Radford Alec, Chen Xi. (2016). *Improved techniques for training gans*. In Advances in Neural Information Processing Systems, pp. 2234–2242

Simonyan Karen, Zisserman Andrew. (2014). *Very Deep Convolutional Networks for Large-Scale Image Recognition*. arXiv:1409.1556

Szegedy Christian, Vanhoucke Vincent, Ioffe Sergey, Shlens Jonathon, Wojna Zbigniew. (2015). *Rethinking the Inception Architecture for Computer Vision*. arXiv:1512.00567

Turing Alan (1950) Computing machinery and intelligence, *Mind*, Volume LIX, Issue 236, pp 433–460

Wiatrak Maciej, Albrecht Stefano. (2019). *Stabilizing Generative Adversarial Network Training: A Survey*. arXiv:1910.00927

Zhang Richard, Isola Phillip, Efros Alexei. (2016). *Colorful Image Colorization*. arXiv:1603.08511