Development of Colorization of Grayscale Images using CNN-SVM

Abdallah Abualola[1] , Teddy Surya Gunawan[1,2][0000-0003-3345-4669],
Mira Kartiwi[3], Eliathamby Ambikairajah[2], Mohamed Hadi Habaebi[1]

[1] Electrical and Computer Engineering Department, International Islamic University Malaysia
[2] School of Electrical Engineering & Telecommunications, UNSW, Australia
[3] Information Systems Department, International Islamic University Malaysia
tsgunawan@iium.edu.my

Abstract. Nowadays, there is a growing interest in colorizing many grayscales or black and white images dating back to before the colored camera for historical and aesthetic reasons. Image and video colorization can be applied to historical images, natural images, astronomical photography. This paper proposes a fully automated image colorization using a deep learning algorithm. First, the image dataset was selected for training and testing purposes. A convolutional neural network (CNN) was designed with several layers of convolutional and max pooling. Support Vector Machine (SVM) regression was used at the final stage. The proposed algorithm was implemented using Python with Keras and Tensorflow libraries in Google Colab. Results showed that the proposed system could predict the colored image from the training process's learning knowledge. A survey was then conducted to validate our findings.

Keywords: grayscale image, color image, convolutional neural networks, SVM regression.

1 Introduction

Nowadays, we have many grayscale or black and white images that date back to before the colored cameras. Converting grayscale images into colored images is a complex, multilayered process that requires high computational requirements. The goal is to convert one channel of image data, i.e., grayscale image, to a Red-Green-Blue (RGB) image, a three-channel image data. The colorization problem has two extents that have to be modeled, and they are the input space, which represents the feature of the grayscale images.

Many types of research have been conducted in converting grayscale images to colored images. A semi-manual colorization method was proposed in [1], which still requires users to assign the right color to a specific area or patch. Neural networks and several image processing techniques were proposed in [2]. Three neural networks output were combined into one output, which transfers colors from the source image to the target image. Similar reference images were used in [3], but it was limited to superpixel representation where it extracted features from reference and target images. It supported spatial coherence but had inaccuracy when it comes to thin objects' boundaries.

Image colorization based on modeling was proposed in [4], but the model can not distinguish global objects. In [5], the best colors were selected using energy minimization, variational formulation, and a patch-based method, in which the color data was extracted from a reference image. In [6], 22 convolutional layers were used with the ImageNet dataset [7] to predict each pixel histogram to deal with the multi-model technique. A Generative Adversarial Network (GAN) [8] has been used to colorize the SUN dataset's natural grayscale images [9]. Even though many researchers and developers created various models and achieved pleasing results, however there quality and computational time could be further improved. There are many proposed solutions, but the most immense challenge is automating the process where no user interference is required, especially for a considerable number of images.

2 Proposed Colorization using CNN-SVM

The goal that needs to be achieved is to use a grayscale image as an input, a one-dimension image, and convert it to a three-dimensional RGB image. Fig. 1 shows the proposed algorithm. First, the input images are down-sampled to 256×256 resolution to make the training more straightforward and reduce the memory requirement. Regardless of the small resolution, the network should learn smooth colorization, as observed in [6]. To colorized output, a conventional spline interpolation algorithm is used. After having the input, the network converts the image to the CIELab color space. This color space is easy to deal with, and it is designed to match the human eye perception of images. Lab color space consists of luminance (L), the grayscale part of an RGB image. At the same time, a and b are the color channels. The a channel controls colors between green and red, and b channel controls colors between blue and yellow.

First, the RGB color space can be transformed to CIEXYZ color [10] as defined in CIE standards and is shown in Eq. (1). Then, the CIEXYZ color can be transformed into CIELab color space, as shown in Eq. (2).

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \frac{1}{0.17697} \begin{bmatrix} 0.49000 & 0.31000 & 0.20000 \\ 0.17697 & 0.81240 & 0.01063 \\ 0.00000 & 0.01000 & 0.99000 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{1}$$

$$\begin{aligned} L^* &= 116 f\left(\frac{Y}{Y_n}\right) - 16 \qquad & a^* &= 500\left(f\left(\frac{X}{X_n}\right) - f\left(\frac{Y}{Y_n}\right)\right) \\ b^* &= 200\left(f\left(\frac{Y}{Y_n}\right) - f\left(\frac{Z}{Z_n}\right)\right) \qquad & f(t) &= \begin{cases} \sqrt[3]{t} & \text{if } t > \delta^3 \\ \frac{t}{3\delta^2} + \frac{4}{29} & \text{otherwise} \end{cases} \end{aligned} \tag{2}$$

where $t = \frac{Y}{Y_n}$ and $\delta = \frac{6}{29}$. L channel values range from 0 to 100, and a and b range from -128 to 127. Because not every combination of these ranges becomes visible

RGB color, we consider only the range -110 to 110. However, some combinations do not map back to RGB. In these cases, the range changes to RGB space (0, 255) [11].

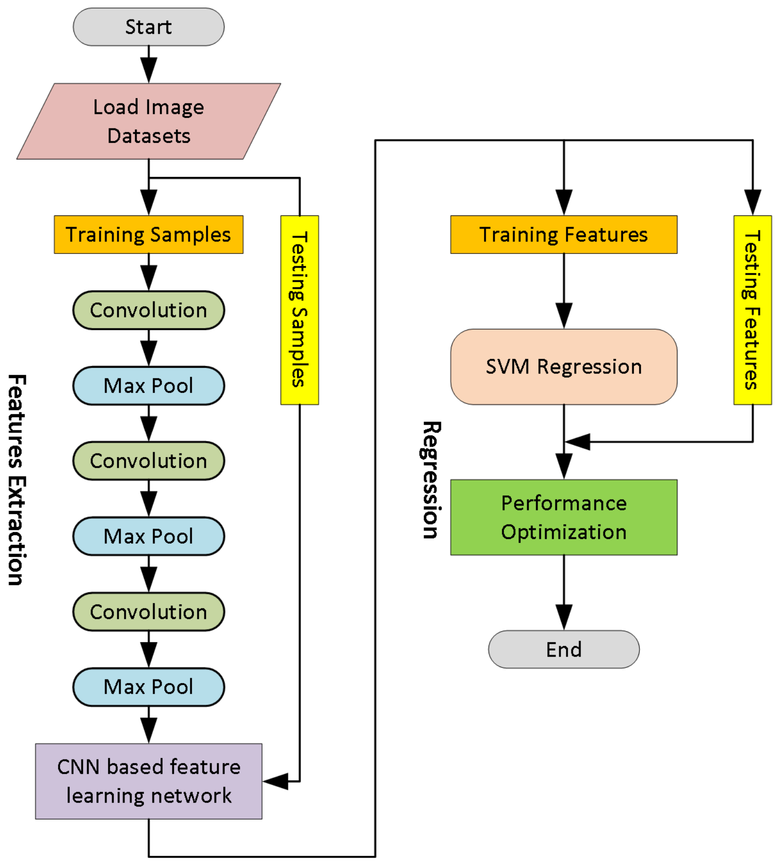

Fig. 1. Proposed Colorization Algorithmusing CNN-SVM

The approach to estimating a and b channels is to deal with the problem as a classification task. This approach is about estimating every pixel color histogram for the target image. The color histogram is equal to the probability distribution of the network prediction for every pixel. The SoftMax activation function will do the normalization of the output to give an appropriate distribution.

After estimating ab channels, we upsampled the channels using spline interpolation to get the original dimensions. An extra convolutional layer will be inserted after the prediction layer is activated by the SoftMax function for colorization in Lab space. The prediction calculation will be in the form of:

$$ab = \sum_i p_i \cdot q_i \tag{3}$$

where p is the predicted probabilities, and q is the canonical 2D ab points selected on the grid.

A loss function in every neural network calculates prediction errors and obtains useful gradients to be used in the update function to update the network's weights. Since the process is predicting probability distribution, the Kullback-Leibler divergence function is applied. This function act as a distance function of two probability distributions. Moreover, the loss function can be enhanced by using a prior boosting reweighing factor called class rebalancing. The loss function becomes:

$$L(\hat{P}, P) = -\sum_{h,w} v(P_{h,w}) \sum_q (\log(\hat{P}_{h,w,q}) - \log(P_{h,w,q})) \tag{4}$$

where $\hat{P}$ is the color histogram of network prediction for pixels, P is the color histogram of the ground truth image, h,w is the indices of the pixels, and v is the weighting term which is defined by [12]:

$$v(P_{h,w}) = w_q^*, q^* = \arg\max P_{h,w,q}$$
$$w \approx \frac{1}{(1-\lambda)r + \frac{\lambda}{313}} \tag{5}$$

where r is the empirical prior distribution of ab pairs calculated over the training set, and λ is a term used to create a mixture of the training set prior probability and uniform distribution.

3 Results and Discussion

3.1 Experimental Setup

The dataset used in training is ImageNet, which has around one million images [7]. These images were split into three groups, training, validation, testing. Around 10000 images were used for validation, 10000 images were used for testing, and the rest was used for training. This separation is to ensure unbiased results. Each image had to go through some preparations. First, convert the image at full resolution to CIELab color space and pull out the L channel. Second, resize the original image to the network input size, which is 224×224. Then pull back the resized L channel into the network.

In this research, the Python programming language will be used. Python will be applied to an open-source library for neural networks, including Keras and Tensorflow, using Google Colab. For training purposes, the most common dataset ImageNet will be used. ImageNet dataset contains RGB and grayscale images with over 10 thousand categories and over 3 million images. It is considered the most suitable dataset that can be used for training and testing.

3.2 Colorization Experiments

Fig. 2 shows the example of successful image coloring cases alongside the grayscale input and the ground truth image. It can be observed that some colors are not similar to the original images, but that is still acceptable where the output images are still considered realistic. Fig. 3 presents an impressive result that has been achieved on images from outside the dataset (the testing process). However, the proposed system still can not correctly colorize some types of images. The possible reason is the lack of variety of colors in the training images, which leads to a slight bias towards some colors.

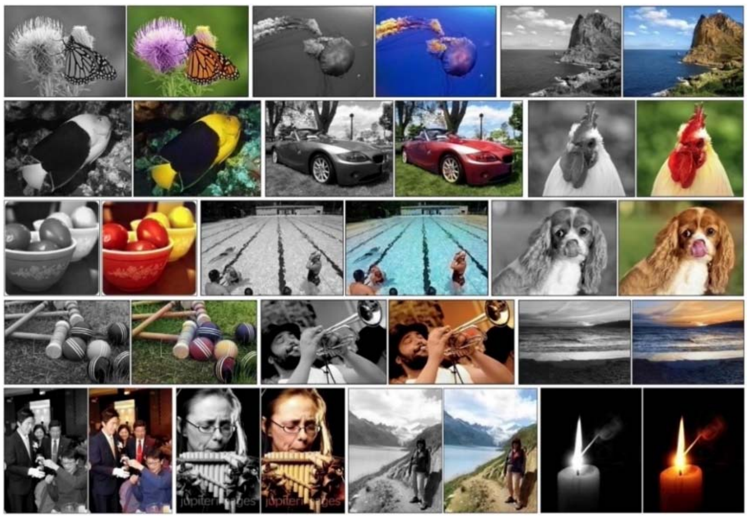

Fig. 2. Example of Succesful Colorization in the Testing Process

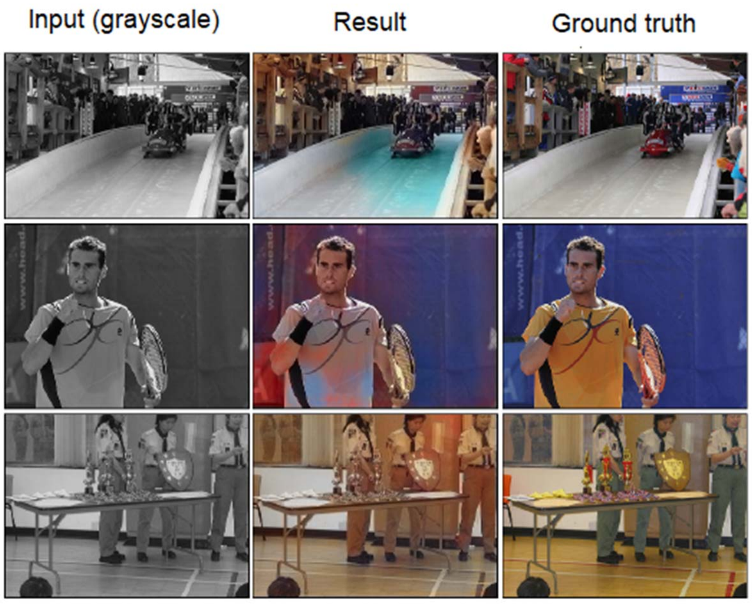

Fig. 3. Example of Unsuccessful Colorization Process

3.3 Historical and 4K Natural Image Colorization Experiment

Some old grayscale historical images have been used to view the results and decide if the output is realistic to test the real-life images. Fig. 4 presents some colored legacy images that have been colored using the proposed system. Moreover, 4K high-resolution grayscale images were colorized using our proposed system, and the result is shown in Fig. 5.

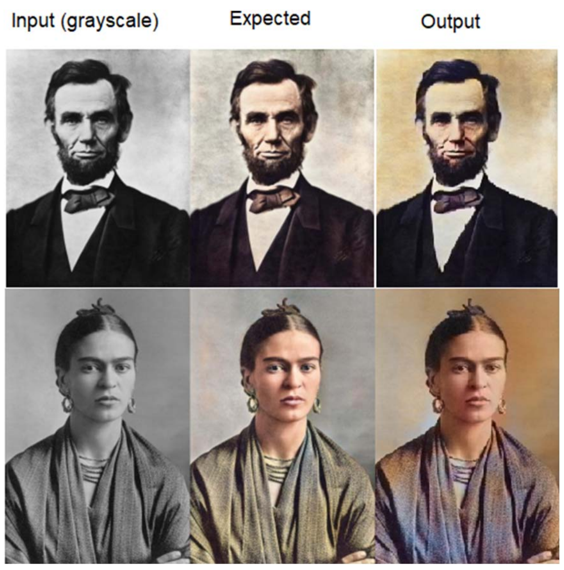

Fig. 4. Results of Colorization of Legacy Images

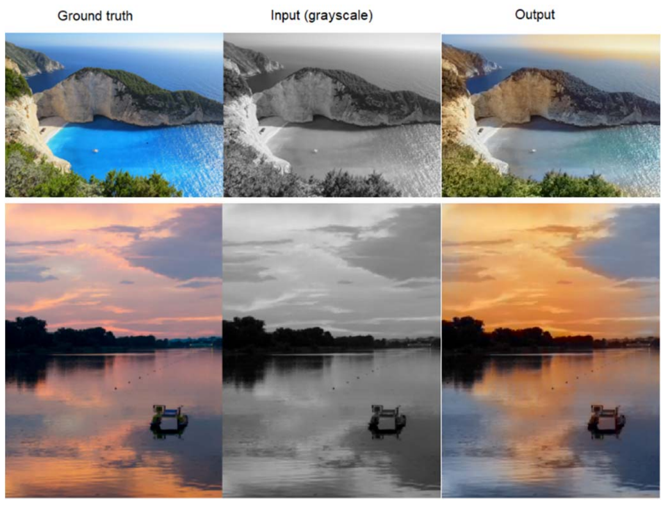

Fig. 5. Example of 4k Natural Images Colorization

3.4 Performance Comparison

Fig. 9 shows the difference between our proposed system results and the algorithm proposed in [13]. Besides, Table 1 shows a detailed comparison between our proposed system and the deep colorization module.

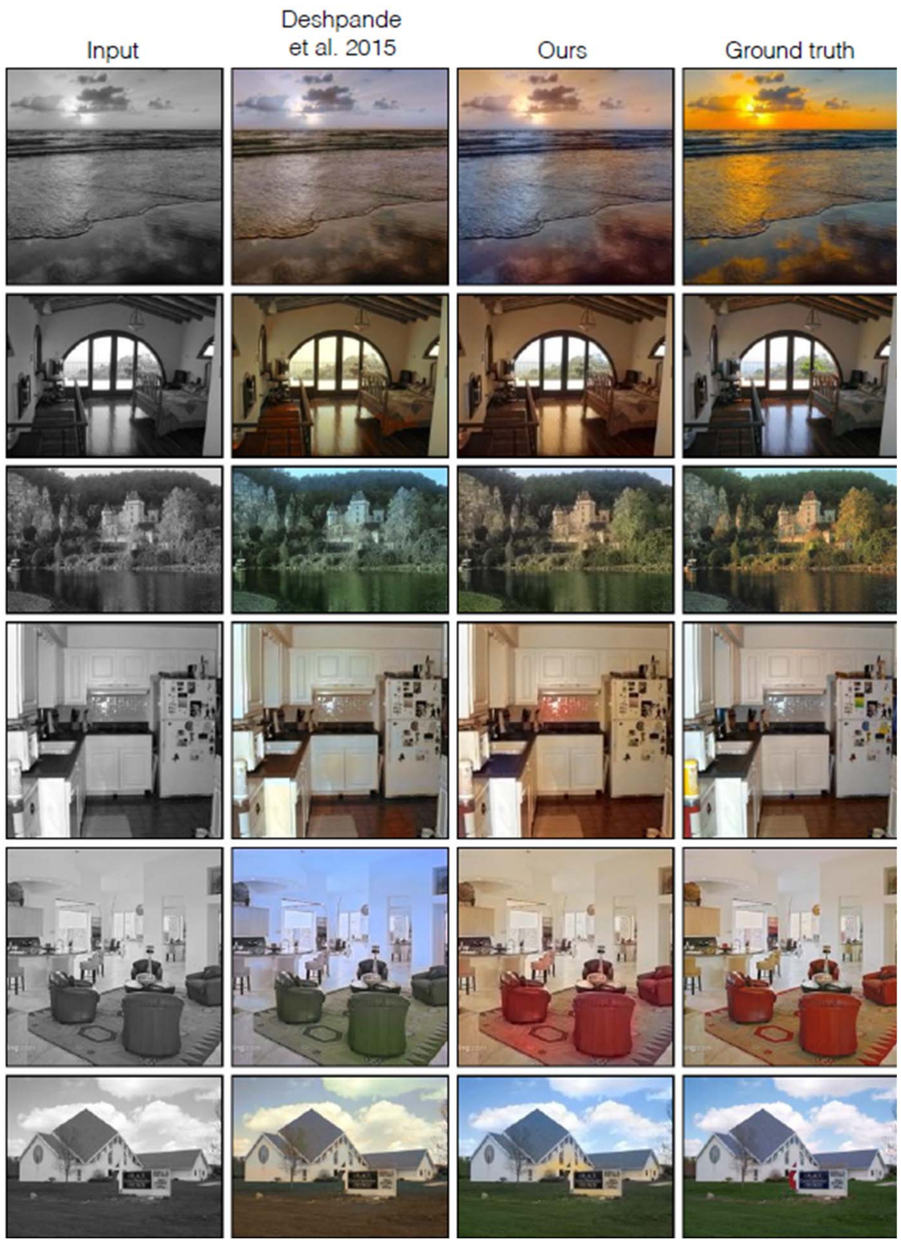

Fig. 6. Performance Comparison

Table 1. Performance Comparison

	Deep Colorization [13]	**Proposed System**
Algorithm	• Extract Feature sets • Fully Convolutional Network • Three layers of neural network regressor • Joint bilateral filter	• CNN and SVM regression
Learning	• Extract features • Train FCN • Train regressor	• Train CNN and SVM from pixel to color distribution. • Tune single parameters on the validation
Dataset	• Sun dataset • 2688 images for Training • 1334 images for testing • Limited variety of scenes	• ImageNet dataset • 1 million images for Training • 10000 images for testing • Diverse categories and scenes
Run-time	• 4.9 seconds per image • MATLAB implementation	• 21.1 ms perimage • Caffe (Python) on K40 GPU

A survey has been done on 7 engineering students. The survey included 15 colorization images as the output from our proposed system. Alongside these images, the ground truth images were presented as well. The first part of the survey asks participants which image is the original image using the first eight out of 15 images. The second part of the survey asks participants to determine whether the presented image (the last seven out of 15 images) is real or synthetic (colorized by our algorithm).

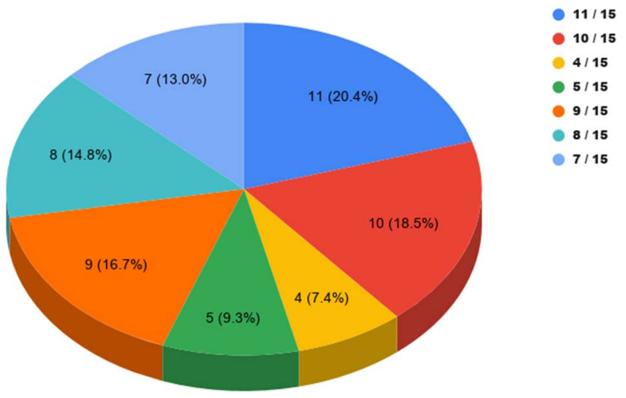

Fig. 7. Qualitative Analysis of the Colorized Images

Fig. 7 shows the summary of the people's opinions, where each question represents one point for guessing right. The highest score was 11 out of 15, while the lowest score is 4 out of 15. We are interested in the lower score, which means our proposed algorithm produces a realistic colorized image. If we take a score of 8 as the median (correctly identified original versus colorized images), we found that 3 participants favored our algorithm, 1 participant cannot decide, and 2 participants were not convinced with our proposed algorithm.

4 Conclusions and Future Works

This research has presented a method to colorize images using a deep convolutional neural network with well-chosen appropriate configuration parameters and functions. Python with Keras and Tensorflow libraries were used in the implementation. Google Colab with GPU was used for training and testing the proposed algorithms. Quantitative and qualitative analysis of the results showed that our algorithm could produce realistic colored images from the original grayscale images. Future works include training with other image datasets, different deep learning configurations, and algorithm optimization.

Acknowledgments

The authors would like to express their gratitude to the Malaysian Ministry of Education (MOE), which has provided research funding through the Fundamental Research Grant, FRGS19-076-0684 (FRGS/1/2018/ICT02/UIAM/02/4). The authors would like to acknowledge the International Islamic University and the University of New South Wales's support.

References

1. Bugeau, A. and V.-T. Ta. *Patch-based image colorization*. in *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*. 2012. IEEE.
2. Karlik, B. and M. Sariöz. *Coloring grayscale image using artificial neural networks*. in *2009 2nd International Conference on Adaptive Science & Technology (ICAST)*. 2009. IEEE.
3. Gupta, R.K., et al. *Image colorization using similar images*. in *Proceedings of the 20th ACM international conference on Multimedia*. 2012.
4. Sousa, A., R. Kabirzadeh, and P. Blaes, *Automatic Colorization of Grayscale Images.* Department of Electrical Engineering, Stanford University, 2013.
5. Bugeau, A., V.-T. Ta, and N. Papadakis, *Variational exemplar-based image colorization.* IEEE Transactions on Image Processing, 2013. **23**(1): p. 298-307.
6. Yu, F., et al., *Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop.* arXiv preprint arXiv:1506.03365, 2015.
7. Russakovsky, O., et al., *Imagenet large scale visual recognition challenge.* International journal of computer vision, 2015. **115**(3): p. 211-252.
8. Cao, Y., et al. *Unsupervised diverse colorization via generative adversarial networks*. in *Joint European conference on machine learning and knowledge discovery in databases*. 2017. Springer.

9. Xiao, J., et al. *Sun database: Large-scale scene recognition from abbey to zoo*. in *2010 IEEE computer society conference on computer vision and pattern recognition*. 2010. IEEE.

10. Gonzalez, R.C. and R.E. Woods, *Digital Image Processing, 4th Edition*. 2018: Pearson.

11. Ruderman, D.L., T.W. Cronin, and C.-C. Chiao, *Statistics of cone responses to natural images: implications for visual coding.* JOSA A, 1998. **15**(8): p. 2036-2045.

12. Zhang, R., P. Isola, and A.A. Efros. *Colorful image colorization*. in *European conference on computer vision*. 2016. Springer.

13. Deshpande, A., J. Rock, and D. Forsyth. *Learning large-scale automatic image colorization*. in *Proceedings of the IEEE International Conference on Computer Vision*. 2015.