International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 06 Issue: 07 | July 2019 www.irjet.net p-ISSN: 2395-0072

Coloring Greyscale Images using Deep Learning

Akshay Patil[1], Asmit Save[2], Varun Patil[3], Vanessa Dsouza[4]

[1,2,3,4]*Student, Dept. of Computer Engineering, St. Francis Institute of Technology, Mumbai, Maharashtra, India*

Abstract - *Traditional or manual coloring of greyscale images requires significant user interaction in the form of placing numerous color scribbles, looking at related images, or performing segmentation. Even powerful softwares of today's era can sometimes take about a month to color a black and white image. This is due to the complex shades of colors present in the image. In this paper, we instead, propose a fully automated data-driven approach for coloring of grayscale images. For this, we built a model using deep learning that can predict colors in a greyscale image. The approach uses image dataset and deep Convolutional Neural Networks (CNN). The model is trained using training dataset and the trained model is further tested using test data images. The pixel deviation from original images is computed and results are compared.*

***Key Words*: Image dataset, Convolutional Neural Networks, Deep learning, Pixel deviation.**

1. INTRODUCTION

Coloring a black and white image is difficult job to be done manually. The process needs a lot of advanced software to analyse brightness levels and predict colors. The process is tedious as it takes a lot of time and manual efforts. The time, in some cases, can be as long as a month. Thus, to prevent all these efforts and haphazard, the objective of the proposed work is to develop a model that is able to color black and white images with better speed. The model uses deep learning using Convolutional Neural Networks (CNN) for the proposed system. Due to very less availability of automatic coloring software's, this idea paves the way for better vision towards picture colorization.

1.1 Dataset Description and Transformation

The dataset for the model is an image dataset of 1300 images. It is a mixture of .jpeg and .png format images. A total 1000 images are used for training the model and 300 images are used for model testing. The images vary in resolution and hence are not suitable directly for training and testing. Thus, the images are all resized to 256 x 256 pixels. The test images are converted into black and white format for prediction and result evaluation phase.

1.2 Model Selection and Description

Since the data is in the form of image dataset, the model requires frequent backtracking to predict colors between layers to minimise predicted errors. Thus we select Convolutional Neural networks (CNN). CNN uses less pre-processing as compared to other image classification algorithms. CNN does not depend on the prior knowledge and human effort in feature design making it suitable for the model. The model consists of 16 layers: 1 input layer, 14 hidden layers and 1 output layer.

2. HARDWARE AND SOFTWARE REQUIREMENTS

Requirements are the minimal configurations of a device and softwares required for the model to work properly and efficiently.

2.1 Hardware requirements

- Graphics Processing Unit (GPU).
- Intel Core i3 processor or above.

2.2 Software requirements

- Windows 7 or above / Linux.
- Python 2.7 or above.
- Jupyter Notebook.

3. METHODOLOGY

The proposed system aims to generate acceptable results for colored versions of greyscale images. To accomplish this, the training image dataset is provided as an input to the model. The model learns to predict the colors of pixels using CNN. After the model is trained, the black and white test image dataset is provided as input to the model. The model colors the images.

The performance is analyzed in the evaluation section where the maximum, mean and median pixel deviations are computed. A new image black and white can be now input to the model for coloring. The workflow of the system is shown in the diagram below.

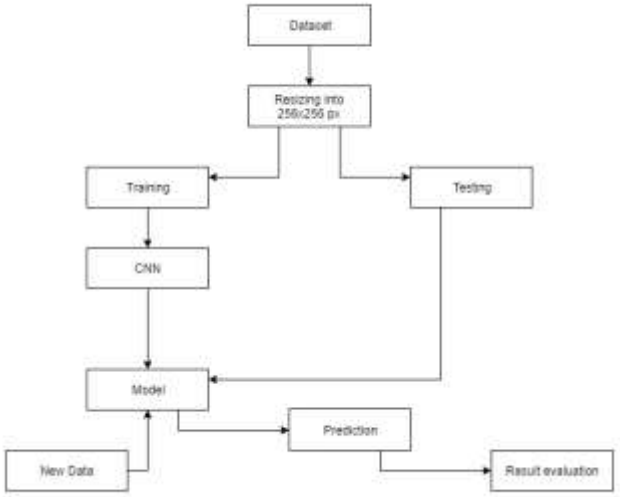

Fig -1: Block Diagram

3.1 Working

The training dataset images are in RGB format. Every color image consists of these three layers: Red(R), Green(G) and Blue(B). These color layers determine the image color as well as brightness. The following figure depicts how the 3 RGB layers look separately.

Fig -2: RGB image format

The system converts this RGB images into Lab format. The system uses Lab color space where L stands for luminance and a, b are just letters. Lab is used over RGB because Euclidean distance in Lab is more similar to how humans perceive color differences. Luminance describes how dark or light a color is. L varies between 0 and 100. 0 represents total darkness and 100 represents maximum brightness. A describes whether a color is towards green or magenta, b describes whether a color is towards blue or yellow. The values of and b ranges between -128 to 127. The following figure shows a and b layers of Lab format.

Fig -3: Lab image format

In order to predict the values of a and b we use various convolutional filters. The function of each filter is to determine and interpret what is seen in the particular picture. The filters are responsible for extracting information from the pictures. The network either creates a new image from a filter or combines several filters into one image. In case of a convolutional neural network, each filter is adjusted automatically to help with the intended outcome. The system begins stacking hundreds of filters and narrow them down into two layers, a and b layers. The final prediction of lab color space looks like the figure below. Thus, the images are all resized to 256 x 256 pixels. The test images are converted into black and white format for prediction of colors.

3.2 Activation function: Rectified Linear Unit (ReLU)

The model is based on deep learning with Convolutional Neural Networks. The predicted colors can have high error rate and may require frequent backtracking between layers of the model. As a result, ReLU is used as activation function. The function is one of the mostly widely used functions in deep learning. ReLU can backtrack between layers of the model thus making it most suitable for the proposed system.

- ReLU : f(x)=max(0,x), where x ∈ R.

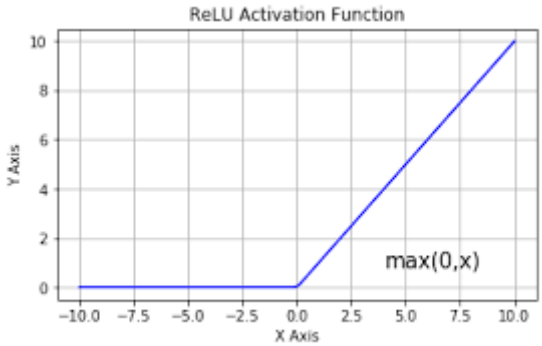

Fig -2: Graphical Representation: ReLU

4. RESULTS

The test data results are compared and evaluated against the actual images. Just by printing the images side by side, we observe a slightly less saturation in predicted images. The images are converted into numpy matrix format using numpy library in Python. The pixel difference between the images

can now be calculated just by subtracting the numpy matrices. The figure below is a visual representation of pixel difference.

Fig -3: Pixel difference between actual and predicted colors

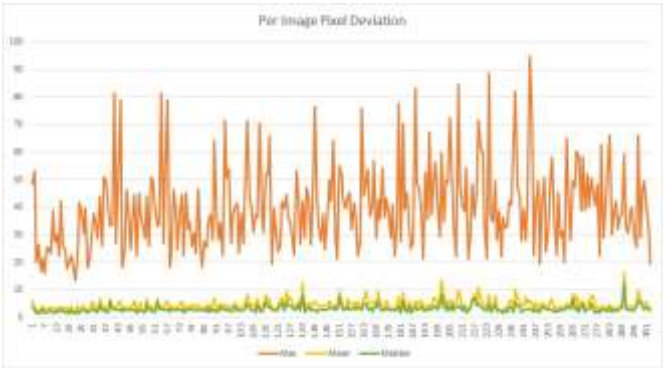

Chart -1: Pixel deviation chart

The pixel difference for all the 300 test images is calculated for evaluation. We have computed 3 types of pixel differences between actual and computed image: Maximum pixel deviation, mean pixel deviation and median pixel deviation. These deviations are computed using numpy functions in numpy library of Python. The matrix contains the difference between the actual and predicted image. We compute the maximum, mean and median of these values.

Maximum pixel deviation: It the pixel that has deviated the most from the actual image. In other words, it is the most wrongly predicted color. It can be as small as 0 or as high as 100.

Mean pixel deviation: It is the mean value of the difference matrix. It gives the average deviation of the predicted image from the actual image.

Median pixel deviation: It is median value of the difference matrix. It gives the median deviation of the predicted image from the actual image.

From the results obtained, we observe that the mean and max pixel deviations graph lines almost converge with each other. Thus the system predicts correct color for most of the pixels. The outliers or the pixels wrongly colored are captured using the maximum graph line. In some cases out of 300, they are sometimes observed to be a very low number slightly more than 10 whereas in other cases, they are observed to be high above 90.

Thus, there are pixels that have not been properly predicted looking at max line but the mean and median lines show that overall image prediction is acceptable.

5. CONCLUSIONS

The model fulfills its purpose of coloring black and white images. The system solves the problem faced by manual coloring softwares. From the results obtained, it is observed the output images differ slightly from the actual images. The difference is noticeable but the colors obtained are acceptable. With more images for training, the pixel deviations can be reduced even further. The results imply that although few pixels have high error rate, overall image is as close to original image.

The proposed work is a new approach towards picture colorization and the idea can be used to replace manual picture coloring softwares which are most commonly used today.

The future scope of the proposed work could be to use the model with object detection for better accuracy and coloring video formats.

ACKNOWLEDGEMENT

The authors can acknowledge any person/authorities in this section. This is not mandatory.

REFERENCES

[1] Chen, Y., Luo, Y., Ding, Y., & Yu, B., "Automatic Colorization of Images from Chinese Black and White Films Based on CNN", International Conference on Audio, Language and Image Processing (ICALIP),2018. pp. 1171-1175.

[2] Salve, S., Shah, T., Ranjane, V. & Sadhukhan, S., "Automatization of Coloring Grayscale Images Using Convolutional Neural Network", Second International Conference on Inventive Communication and Computational Technologies (ICICCT), 2018.