Automatic Image Colorization

A Project Report

submitted by

Pavan C M (13990)

as part of

Machine Learning for Signal Processing (E9 205)

undertaken during

August - December 2016

DEPARTMENT OF ELECTRICAL COMMUNICATION ENGINEERING

INDIAN INSTITUTE OF SCIENCE BANGALORE - 560012

December 17, 2016

TABLE OF CONTENTS

CHAPTER 1
Introduction

1.1 Problem definition

This project investigates the problem of colorizing grayscale images. Colorization is fundamentally an ill posed problem mainly due to the loss of information across dimensions when a colour image goes to grayscale version. The main challenge arises as various colors can give rise to same grayscale values. Mathematically the problem is estimating 3 dimensions (RGB or YUV color space) from single dimension. It normally requires human assistance to achieve artifact free color images.

In this project an attempt has been made to come with methods to colorize images without human assistance. The algorithm works by a training a model on a large corpus of images and then using the developed model to colorize grayscale images. The motivation to use deep models come from therecent success achieved by deep learning techniques on large datasets. Deep learning has been succesfully applied to various classification, recognition and regression problems [1] [2]. This project formulates colorization as a regression problem and neural networks are employed to solve regression. A large image database is used for training the model.

1.2 Previous work

Previous work regarding colorization can be divided into two, scribble based colorization and example based colorization.

In Scribble based colorization, user is required to provide some colorful scribbles and based on the scribble an algorithm predicts the colors of the image [3].
In Example based colorization, the color information from a reference image is transferred to target grayscale image. The reference image can be either user supplied [4] or web supplied example images [5].

The method implemented in this project is an extension of the second method where in a large image dataset is provided to the algorithm and the model transfers colors by considering the observed patterns in the provided dataset.

CHAPTER 2
Model and Implementation

2.1 Model

This project is the implementation of Deep colorization proposed by Cheng,Yang et.al [6] This model employs a multi layered perceptron network with 3 hidden layers to perform colorization. All the images from the dataset are resampled so that their length and width are less than 250. This is mainly done to reduce the computational overhead of predicting large number of pixels otherwise. Model is designed to predict values in YUV color space. YUV model is employed as it has least correlation among the components and it involves predicting only U and V values, as Y component itself represents grayscale image.

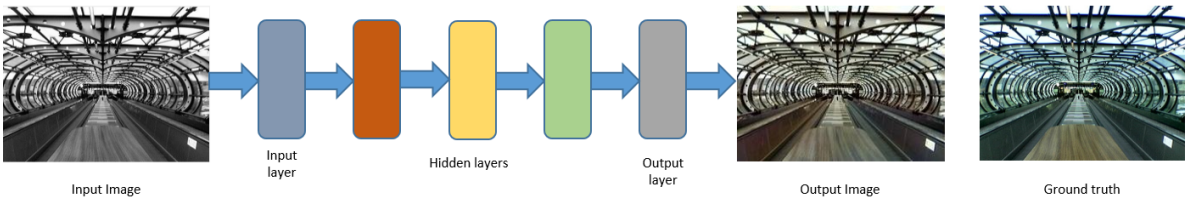

Figure 2.1: Overview of the colorization model

Three types of features are obtained from the input image at every pixel and these features are used for training the model. The number of units in hidden layer is perceptually equal to that of the input layer. The number of neurons in input layer is equal to the dimension of the feature descriptor. Since this is a multi layer perceptron, every neuron in output or hidden layer is connected to every neuron in the preceding layer.

$$y_j^l = f(w_{jo}^l b + \sum_{i>0} w_{ji}^l y_i^{l-1}) \tag{2.1}$$

where w_{ji}^l is the weight of the connection between j^{th} neuron in the l^{th} layer to the i^{th} neuron in the $(l-1)^{th}$ layer, b is the bias and $f(z)$ is the activation function. In this model ReLU is activation function used.

2.2 Feature Descriptor

Choice of features plays a significant role in this model. The features are divided into low level,mid level and high level features. While training, all features are concatenated to a single vector. Let x^L, x^M, x^H represent low,mid and high level features, the $x = [x^L; x^M; x^H]$

2.2.1 Low-level patch feature

Low level features are patch features obtained by considering $7x7$ patch around every pixel of interest. Patch feature is mainly employed due to it's better performance in low texture regions when compared to other features. Figure 2.2 shows significance of patch feature. Patch feature affects the performance in low texture regions. In Figure 2.2 sky is a low texture region which has artifacts when colorized, the artifacts increase in absence of patch feature.

(a) Input Image (b) -patch PSNR=18dB (c) +patch PSNR=19.1dB (d) Ground Truth

Figure 2.2: Figures illustrating effect of patch feature

2.2.2 Mid-level DAISY feature

DAISY is a fast local descriptor for dense matching [7]. DAISY decriptor is influenced by SIFT features except that DAISY feature is computed at every pixel. DAISY feature performs well in high texture regions but decreases performance in low texture regions. This model uses DAISY decriptor with 4 locations and 8 orientations at each location.

2.2.3 High-level Semantic features

Patch and DAISY features mainly produce neighbouring effect on the pixel in textured and non texture regions. Semantic features are obtained by segmenting image into semantic categories. An example ogf segmentation is shown in Figure 2.3.

(a) Original Image

(b) Segmented Image

Figure 2.3: Figures illustrating segmentation

Segmentation is done by using state of the art segmenting algorithm proposed in [8]. Colorization can be considered as a semantic aware process, where the color of the pixel depends on what category it belongs to (e.g a sky and water can have only blue color). The above segmentation algorithm returns a probability vector of length N, where every element in the vector gives probability of the pixel belonging to that category out of the N categories. The probability vector is used as a high level descriptor.

2.3 Dataset

The colorization neural network is trained on 1285 images and a total of 10 semantic categories are used (e.g building,forest etc). Feature decriptor is a vector with 141 dimensions with 49 as low-level, 32 as mid-level and 60 as high-level features. As 141 features are extracted at every pixel the training samples represents a extremely large set of samples resulting in very slow training of the model. Training of 1000 images with 50 epochs nearly took 11 hours on a system equipped with NVidia GT710 GPU.

CHAPTER 3
Observations and Results

3.1 Experimental results

Some of the colorized images obtained from this model is shown in Figure 3.1.

(a) Field (b) Pasture (c) Beach (d) Building

(e) 22dB (f) 22.7dB (g) 16.5dB (h) 19.2dB

(i) Ground Truth (j) Ground Truth (k) Ground Truth (l) Ground Truth

Figure 3.1: Results of the colorization model. First row represents input grayscale image. Second row corresponds to the colorization obtained from the trained model. The values below the image is their respective PSNR values. Third row is the ground truth images

Peak Signal to Noise ratio is used as a performance metric for evaluating colored images. PSNR is a measure derived from mean square error. Hence higher PSNR implies the image is more near to the original image in mean square sense which doesn't necessarily mean the image has a superior quality. An example is shown in Figure 3.2 where 2 images with similar PSNR have considerable difference in their appearance.

3.2 Choice of Color Space

In the above model YUV color space was used for training and testing the model. The motivation to use YUV is that it was used extensively during the initial days of color telivision as it required to add 2 additional chominance channels (U and V) to existing luminance (Y) so that existing black and white telivison sets could be easily converted to colored ones. In this project a small experiment was carried out to check whether use of other spaces improves performance.

(a) Input Image (b) RGB trained (c) YUV trained (d) Ground Truth

Figure 3.2: Variation with color spaces

In the images shown in Figure 3.2 the RGB trained image has a PSNR of 16.2dB and YUV trained model has 16dB. But physically there is a significant variation. Hence PSNR is nit a good candidate for assessing the quality of colored image with respect to ground truth. Structural Similarity Index (SSIM) which considers human perception is a better choice for evaluating the performance.

3.3 Effect of Training data

The trained model is heavily dependent on training data as color transfer solely depends on what sort of training data is used for training. Effect of size of training data on the performance is show in Figure 3.3. From the graph it is evident that the images are colored more accurately with increasing data size.

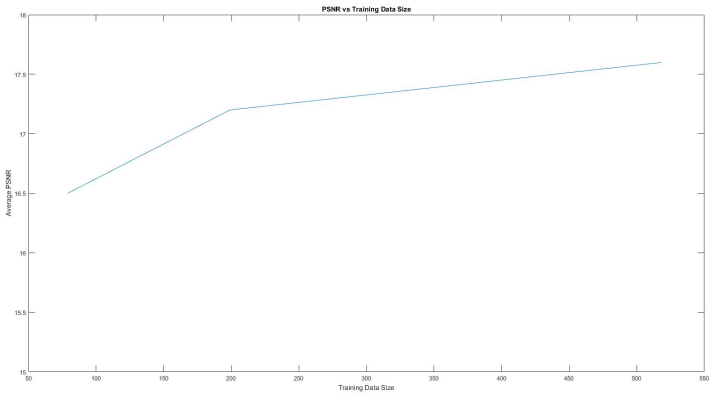

Figure 3.3: Variation of PSNR with training data

3.4 Limitations

As this model is machine learning based approach, it comes with it's own limitations.

- The semantic feature used in the model is obtained from a fully convolutional neural network(FCN) trained for segmentation with color images. This FCN gives best results when used on color images. This can be observed by comparing Figure 2.3 and Figure 3.4. Hence segmentation is not always accurate.

(a) Grayscale Image (b) Segmented with grayscale image (c) Segmented with color image

Figure 3.4: Figures illustrating segmentation with grayscale image

- There is a need for a robust post processing as there appears some extraneous artifacts in colored images. Artifacts are more prominent in low texture regions and near the edges as observed in Figure 3.5.

(a) Image 1 (b) Ground truth 1 (c) Image 2 (d) Ground Truth 2

Figure 3.5: Figure showing artifacts in colored images

- Since features are extracted at every pixel, the training is extremely slow.
- The above model fails in case of synthetic images, as the behaviour of synthetic images is different to that of natural images.

3.5 Performance of Linear Regression

Since colorization is a regression problem, an attempt was made to evaluate the performance of colorization with linear regression using the same features used in the previous model. Mathematically linear regression can be represented as

$$y = w^T x + b \tag{3.1}$$

where w represents weights calculated using training data, x is training data and b reprsents bias. An experiment was carried out using linear regression across 6 semantic categories and the result is shown in figure 3.6. It is evident from the figure though linear regression model has lower performance in most of the cases, it closely follows the performance of that of neural network. However the figure represents only the average performance, the individual performance may vary. An instance where linear model is better is shown in Figure 3.7. The above analysis is limited to the training data corresponding to 6 semantic categories.

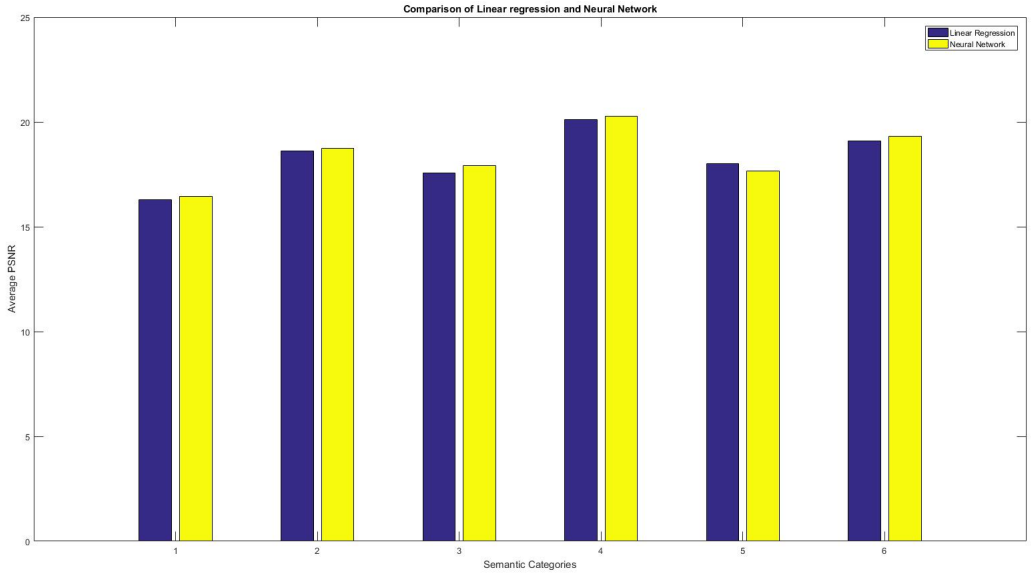

Figure 3.6: Comparison of linear regression and neural network

(a) Input Image (b) linear regression PSNR-20dB (c) Neural Network PSNR-19.3dB (d) Ground Truth

Figure 3.7: Comparison of Linear Regression and Neural Network modelled images

3.6 Conclusion

This project was an attempt towards achieving full automated image colorization. Baseline implementation consisted of training a deep neural network model using pre-defined features. The performance of this model was evaluated and shortcomings of the model were analyzed. Towards the end an attempt was made to model using linear regression which also showed a similar performance on the limited data on which it was evaluated. A more detailed analysis on applicability of linear models can be carried out as a future direction, along with support vector regression(SVR) model. However colorization is an ill posed problem which doesn't have a definite answer. It's difficult to regain exact color information lost however complex the model and this limitation applies to all state of the art colorization methods. This can be well understood by Figure 3.8. Both colorized images are possible outcomes. So given grayscale image, humans tend to believe whichever color image (b or c) is shown first, as the true color image.

(a) Input Image (b) Image 1 (c) Image 2

Figure 3.8

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems, pages 10971105, 2012.

[2] W.W. Ouyang and X. Wang. Joint deep learning for pedestrian detection. In ICCV, pages 20562063. IEEE, 2013

[3] A. Levin, D. Lischinski, and Y. Weiss. Colorization using optimization. In ACM SIGGRAPH 2004 Papers, pages 689694, 2004

[4] T. Welsh, M. Ashikhmin, and K. Mueller. Transferring color to greyscale images. ACM Trans. Graph., 21(3):277280, July 2002.

[5] A. Y.-S. Chia, S. Zhuo, R. K. Gupta, Y.-W. Tai, S.-Y. Cho, P. Tan, and S. Lin. Semantic colorization with internet images. In TOG, volume 30, page 156. ACM, 2011

[6] Cheng, Zezhou, Qingxiong Yang, and Bin Sheng. "Deep colorization." Proceedings of the IEEE International Conference on Computer Vision. 2015.

[7] E. Tola, V. Lepetit, and P. Fua. A fast local descriptor for dense matching. In CVPR, pages 18. IEEE, 2008.

[8] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. arXiv preprint arXiv:1411.4038, 2014