TensorFlow Implementation and analysis of Image Colorization

Tushar Jain

1 Introduction

I have implemented the CNN based image colorization technique as suggested by Satoshi, Edgar and Hiroshi in their 2016 Paper named as *"Let there be Color! : Joint End-to-end Learning of Global and Local Image Priors for Automatic Image Colorization with Simultaneous Classification (Satoshi Iizuka, 2016)"*. I have tried to analyze its result on various datasets.

2 Model architecture

I have used the same model architecture as suggested in the original paper with a Low-Level features extraction layer whose output is send to Mid-Level feature extraction and Global feature extraction outputs from both are than fused together in a fusion layer and then sent for deconvolution[1].

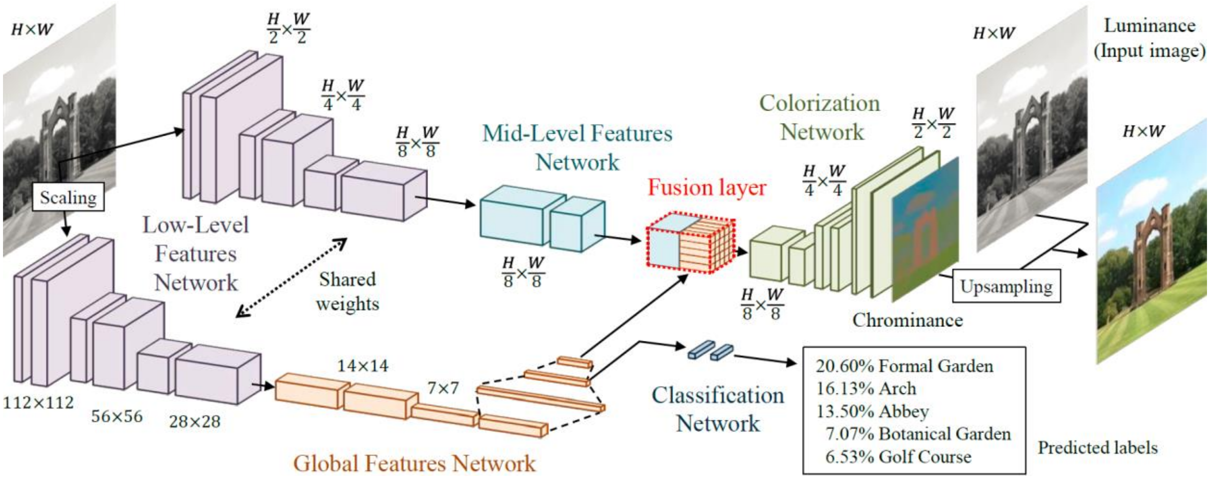


Figure 1 Model Architecture (Satoshi , Edgar, SIGGRAPH 2016)

[1] In original paper it Up Sampling layer was used but I have used 2D deconvolution.

2.1 Layer specification

Low-Level Features Network

Type	Kernel	Stride	Outputs
Conv2D	3 × 3	2 × 2	64
Conv2D	3 × 3	1 × 1	128
Conv2D	3 × 3	2 × 2	128
Conv2D	3 × 3	1 × 1	256
Conv2D	3 × 3	2 × 2	256
Conv2D	3 × 3	1 × 1	512

Colorization Network[2]

Type	Kernel	Stride	Outputs
Fusion	-	-	256
Conv2DTranspose	3 × 3	1 × 1	128
Conv2DTranspose	3 × 3	2 × 2	64
Conv2DTranspose	3 × 3	1 × 1	64
Conv2DTranspose	3 × 3	2 × 2	32
Conv2DTranspose	3 × 3	2 × 2	2

Global-Level Features Level

Type	Kernel	Stride	Outputs
Conv2D	3 × 3	2 × 2	512
Conv2D	3 × 3	1 × 1	512
Conv2D	3 × 3	2 × 2	512
Conv2D	3 × 3	1 × 1	512
Flatten	-	-	-
Dense	-	-	1024
Dense	-	-	256
Dense	-	-	512

Mid-Level Features Network

Type	kernel	stride	output
Conv2D	3×3	1×1	512
Conv2D	3×3	1×1	256

2.2 Batch Normalization

In order to improve the Learning Time after every convolutional layer I added Batch normalization layer. I found a significant decrease in the learning time.

3 Dataset

1. **CelebA dataset**(Large-Scale Celeb Faces Attributes Dataset (Liu, 2015)) Dataset Contains more than 200k images of celebrity faces from different countries. I used 1000 images from them as training data 200 as validation data and 200 as test data.
2. **Linnaeus 5 dataset** (Chaladze, 2017) .Dataset contains two parts Test and Train, Train contains 6000 classes divide equally between 5 classes and Test contains 2000 images divided equally among 5 classes. I used images of *'flower'* class from the dataset because they were more colorful and thus giving model more colors to learn

[2] Structure of colorization network is different from original network

4 Training

4.1 CIE L*a*b* Color Space

Before giving the image as a model to the input I converted the image into CIELAB color space and then used the Lightness value(L) as input which is effectively gray scaled image than output channels were selected 2 as given in model architecture. This restricts us to predict only two channels. Predicting three channels would have been difficult for our model

4.2 Accuracies

1. For CelebA dataset : Total 200 epochs with 100 steps per epoch and batch size 50.

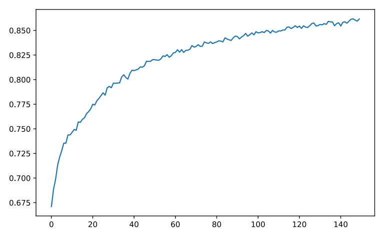


Figure 2 Accuracy vs number of epochs 0 to 150 epoch(CelebA)

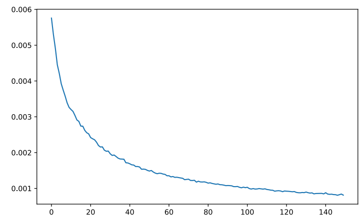
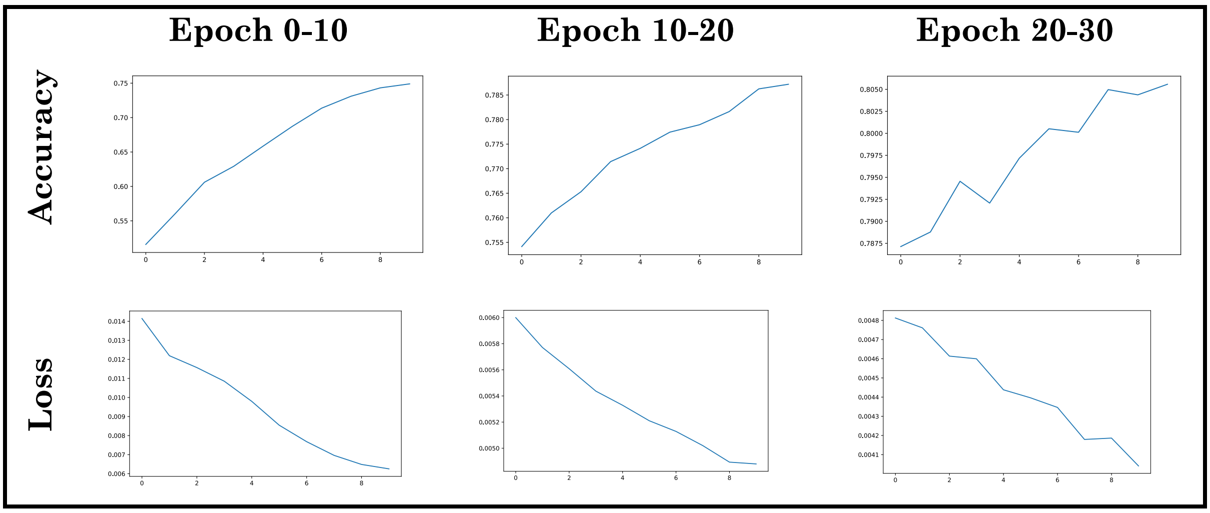


Figure 3 Loss vs Number of Epochs 0 to 150 epoch (CelebA)

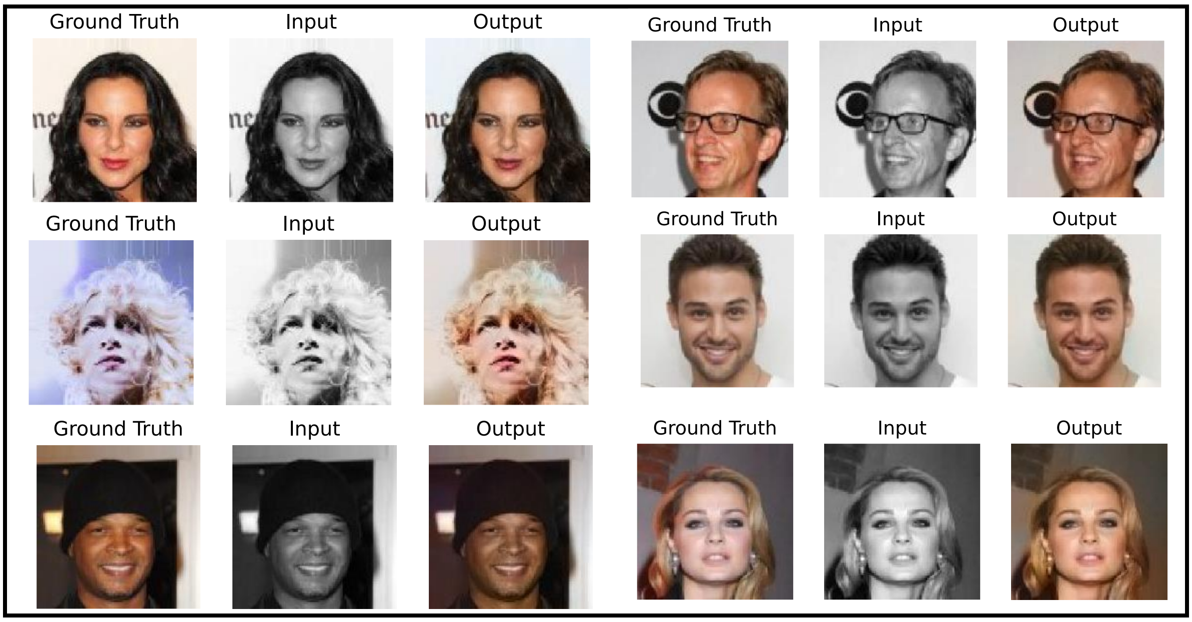
2. For Linnaeus 5 dataset: 30 epochs 100 steps per epoch with batch size 50 (1hr 20 min training time with Google Colab GPU)



5 Results

From Linnaeus 5

From CelebA



6 Works Cited

1. *Chaladze, G. K. (2017). Linnaeus 5 Dataset for Machine Learning. Retrieved from http://chaladze.com/l5/*

2. *Liu, Z. a. (2015). Deep Learning Face Attributes in the Wild.*
3. *Satoshi Iizuka, E. S.-S. (2016). Let there be Color!: Joint End-to-end Learning of Global and Local Image Priors for Automatic Image Colorization with Simultaneous Classification. ACM Transactions on Graphics (Proc. of SIGGRAPH 2016), 35(4), 110:1--100:11. Retrieved from http://iizuka.cs.tsukuba.ac.jp/projects/colorization/en/*