Automatic Image Colorization

Karan MT19025 | Sachin MT19028 | Shivani Kumar PhD19010

***Abstract*—**In this work, we investigate the problem of colorization of grayscale images. This problem is ill-defined and underconstrained which is why earlier works in this domain have relied on heavy user interaction. We first explored Deep Colorization, which uses multi-scale features and deep neural networks for colorization. It consists of two separate networks, one for higher-level feature learning and another for colorization. After that, we explored Colorful Image Colorization, which uses a single Convolutional neural network and models the problem as an end-to-end learning task. In our analysis, we observed that both algorithms result in colorization outputs that are biased towards a specific color palette with no flexibility of generating multiple color images from a single grayscale input. In the end, we explored PixColor, which has the advantage of providing flexibility of sampling multiple color images.

***Index Terms*—**Image Colorisation, Deep Learning

I. Introduction

Given a grayscale image it is a daunting task, even for a human, to visualise it in color. See Figure 1 for examples. However, a human may try to find semantic clues like texture and world knowledge to assign colors to objects. For example, grass is mostly green or the sky is mostly blue. But these clues may also fail sometimes, as can be seen in Figure 1 (b) where any color may be assigned to the couch. Thus, in this work, we focus on assigning a plausible set of colors to the image which may or may not be the same as the ground truth.

A. Motivation

There exists a wealth of photographic images, from antique photography to low-resolution video, which lack color information. As discussed above, assigning color to a grayscale image is an uncertain task as there is often no "correct" attainable color for it. For example, the orange color of the couch in Figure 1 (b) can be identified as blue as they would be indistinguishable when photographed in black and white. Due to this indeterminate nature of the problem, earlier image colorization techniques used to rely on human interaction. Several vintage movies have been colorized entirely by hand, requiring every single frame to be inspected at great financial and time cost. Here we study some deep learning techniques, to assign aesthetically believable colors to grayscale images. The applications of such method allow for new appreciation of old, grayscale photographs and cinema, along with better interpretation of modern black and white images such as CCTV footage, etc.

B. Related Work

In 2015, Zezhou Cheng et al. [1] proposed an automatic image colorization algorithm using deep neural networks using different scale input features. They used patches as lower level features, daisy descriptors as middle scale, and pixel-level class segmentation as higher-order features. These features are then fed to another deep neural network for colour prediction. Another significant contribution was by Richard Zhang et al. [2], as they modelled the image colorization as multi-nominal instead and thus learned the distribution of the possible colours for each pixel, and scaled the loss at training time to emphasize rare colours. A general solution for exemplar-based colorization, trained on "synthetic" grayscale images by removing the chrominance channels from colour images was presented by Mingming He and co [3], which can be extended to colorize legacy movies by independently colorizing each frame and then temporally smoothing the colorized results with the method. Though these models are SoTA in their on terms, however, they have limitation like difficulties in identifying object in unusual colours, fail to predict appropriate colour for less semantic areas and do not perform well when luminance disparities between images. We have studied these experiments in detail and will be bringing out the shortcomings.

Problem Statement: Given an input lightness channel $X \in R^{H*W*1}$, our objective is to learn a mapping $Y' = F(X)$ to the two associated color channels $Y \in R^{H*W*2}$, where H, W are image dimensions.

II. Methodology

A. Deep Colorization

In 2015, Zezhou Cheng et al. [1] proposed an automatic image colorization algorithm using deep neural networks using different scale input features. For each pixel, they used a patch of size $[7 * 7]$ as lower order, daisy descriptors of size 32 (4 locations, 8 orientation) as middle order, and pixel-level class segmentation as higher-order features. These features are then fed to another deep neural network for color prediction. The Colorization network takes a vector of length 135(49 for lower, 36 for middle, and 50 for higher level) as input and returns a vector of length 2 which estimates the chrominance channel for that pixel in YUV color space as it reduces the correlation between channels. For optimization, stochastic gradient descent was used, with mean square error as the optimization function, $L_2(.,.)$:

$$L_2(Y', Y) = \frac{1}{2}\Sigma_{h,w}||Y_{h,w} - Y'_{h,w}||_2^2$$

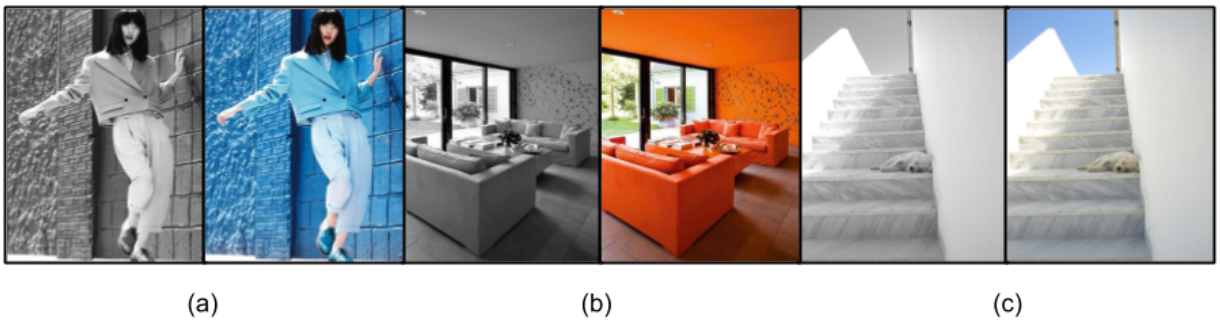

Fig. 1. Grayscale and their corresponding color images

B. Colorful Image Colorization

Another significant contribution was by Richard Zhang et al. [2], as they modeled the image colorization as multi-nominal instead and thus learned the distribution of the possible colors for each pixel. They scaled the loss at training time to emphasize rare colors. In this approach, a CNN model is trained to map from a grayscale input to a distribution over quantized color value outputs.

It is also optimized using stochastic gradient descent but L_2 norm is not used as it is not robust to the inherent ambiguity and multimodal nature of the colorization problem. If an object can take on a set of distinct ab values, the optimal solution to the Euclidean loss will be the mean of the set. This averaging effect yields grayish, desaturated results. Hence, another objective function is proposed.

Objective Function: The ab output space was quantized into bins with grid size 10 and Q was kept 313 for all colors in-gamut. For a given input lightness channel X, we learn a mapping, $Z' = G(X)$ to a probability distribution over possible colors $Z \in [0,1]^{H*W*Q}$, where Q is the number of quantized ab values. To compare predicted Z' against ground truth, we define function $Z = H_{gt}^{-1}(Y)$, which converts ground truth color Y to vector Z, using a soft-encoding scheme. We then use multinomial cross entropy loss $L_{cl}(.,.)$, defined as:

$$L_{cl}(Z', Z) = -\Sigma_{h,w} v(Z_{h,w}) \Sigma_q Z_{h,w,q} log(Z'_{h,w,q})$$

where $v(.)$ is a weighting term that can be used to rebalance the loss based on color-class rarity. Finally, we map probability distribution Z' to color values Y' with function $Y' = H(Z')$.

The function H, which maps the predicted distribution Z' to point estimate Y' in ab space, is defined as the annealed-mean of the distribution:

$$H(Z_{h,w}) = E[f_T(Z_{h,w})], f_T(z) = \frac{exp(log(z)/T}{\Sigma_q exp(log(z_q)/T)}$$

After this, class rebalancing is used in order to enhance rare colors in the images.

Class Rebalancing: A major part of almost all natural images is occupied by backgrounds such as sky, dirt, ocean, etc. This results in the distribution of the ab values to be biased towards low ab values. If this property of natural images is not accounted for in the loss function, the resultant images will be dominated by desaturated ab values. Thus, there is a need of reweighting the loss of each pixel at train time on the basis of pixel rarity. Each pixel is weighed by a factor of $w \in R^Q$, based on its closest ab bin.

$$v(Z_{h,q}) = w_{q^*}, \text{ where } q^* = argmax_q Z_{h,w,q}$$

To obtain smoothed emperical distribution $p' \in \Delta^Q$, the emperical probability of colors in the quantized ab space $p \in \Delta^Q$ from the full ImageNet training set is estimated and is smoothed with a Gaussian kernel G_σ. The distribution is then mixed with a uniform distribution with weights $\lambda \in [0,1]$, inverted, and normalized so that the weighting factor is 1 on expectation.

$$q \propto ((1-\lambda)p' + \frac{\lambda}{Q})^{-1}, E[w] = \Sigma_q p'_q w_q = 1$$

Thus, rebalancing will lead to generation of more saturated and vivid images.

C. Pixel Recursive Colorization

In Colorful Image Colorization, we modelled the colorization problem using a mapping, $Z' = G(X)$, where $Z \in [0,1]^{H*W*Q}$. In PixColor [4], it is modelled in generative fashion i.e. the output of i^{th} pixel is not just conditioned on X but also on the previous outputs.

$$p(y|X) = \prod_i p(y_{i,Cr}|y_{1:i-1}, X) * p(y_{i,Cb}|y_{i,r}, y_{1:i-1}, X) \tag{1}$$

where $y_{i,Cr}$, $y_{i,Cb}$, is the value of Cr and Cb channel of i^{th} pixel respectively, and Cb and Cr denotes blue and red differences respectively in YCbCr color space. By modelling it in generative fashion, it provides two advantages over Colorful Image Colorization, (i) we can sample different colorful images from single grayscale input, and (ii) sampling should mitigate the biasness towards specific chrominance range.

1) Feature Extraction: Given $X \in [H, W]$, we first extracts the features Y_1 using Resnet-101 of size $[\frac{H}{4}, \frac{W}{4}, 1024]$. These features are then passed into adaption network and uses three convolution layers to adapt the features required by pixelcnn. The output from the adaption network is of size $[\frac{H}{4}, \frac{W}{4}, 64]$, and is fed into conditional pixelcnn.

2) Conditional Image Generation Using PixelCNN: The Conditional PixelCNN consists of two variations of convolutional layers,

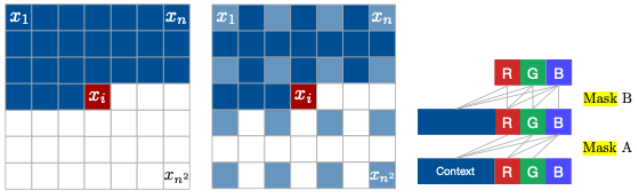

Fig. 2. Mask function

Mask Convolutional Layer: It functions in the same way as that of normal convolutional layer but it masks weights of the convolutional layer as shown in the figure 3. The mask functions prohibits sample x_i from using any information about the future samples ($x_{i+1:N}$).

Gated Convolutional Layer: In the original pixelcnn [5], only masks layers were defined and even though it is faster during training, PixelRNN outperforms in terms of quality of output generation. In Conditional PixelCNN [6], the authors claimed that multiplicative units present in the lstm cell was the reason behind PixelRNN outperforms PixelCNN along short range dependencies. Thus, they proposed gated convolutional layers which has multiplicative units $\tanh$ and σ, to mimic the lstm. As shown in the figure it uses multiple Mask Convolutional layers (shown in Green), followed by a split function(shown in blue).

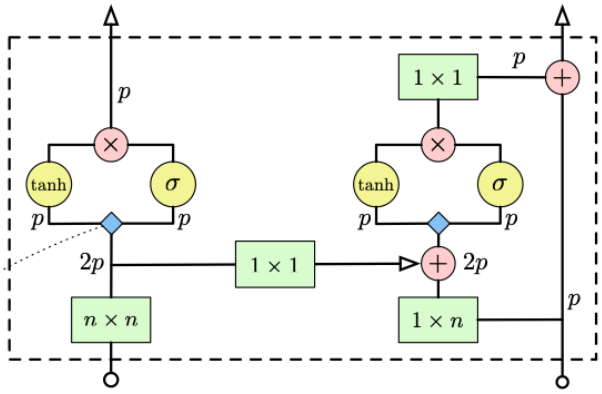

Fig. 3. Gated Convolutional Layer

The output of adaption network is fed into A-type mask convolutional layers, followed by multiple Gated Convolutional layers and B-type mask convolutional layer. It is important to use A-type layers for input, as B type conditions the output on itself which is not present in the input.

3) Sampling Images during Test times: The training of PixColor is same as that of any other end-to-end trainable architecture but during testing, for each pixel i, we don't condition the next pixel on its output directly but we do weighted sampling based on the probability of classes of output.

III. Experiments

A. Datasets

1) ImageNet: The original network of Colorful Image Colorization is trained on 1.3M images from the ImageNet training set [7]. The best model for this method is hosted on the web[1] and is used to get the results for the state of the art of this model. However, to experiment with the architecture of the model, we downloaded 20k images from the ImageNet dataset (40 classes with 500 images each) using the imagenet-downloader[2].

2) ADE20K: This dataset [8] has been designed for scene parsing, and spans diverse annotations of scenes, objects, and parts of objects. We used this dataset for training the models based on Deep Colorization and PixColor. The training set has 20K images with segmentation labelling over 50 classes and testing set has 2K images.

B. Evaluation

1) Deep Colorization: The input to the network is a vector of 131 length, standard normalization and the colorization network has 3 hidden layers with the ReLU activation function. We trained it over a subset of ADE20K dataset over all images which are of size $[256, 256]$. We trained three different models, one without any higher order features[A], one which uses higher order features from the segmentation algorithm[B], and the one in which ground truth segmentation were used for comparison. We observe the performance of these models in order $A < B < C$, as expected.

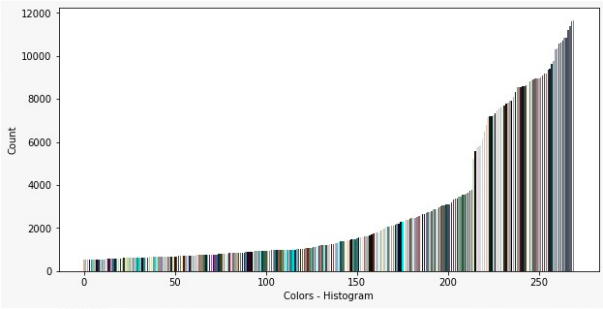

Fig. 5. Histogram of prominent colors present in ADE20K Dataset

The results can be seen in Figure 4, Model A is not able to detect the sky well in first figure and in second figure it

[1]https://algorithmia.com/algorithms/deeplearning/ColorfulImageColorization

[2]https://github.com/mf1024/ImageNet-datasets-downloader

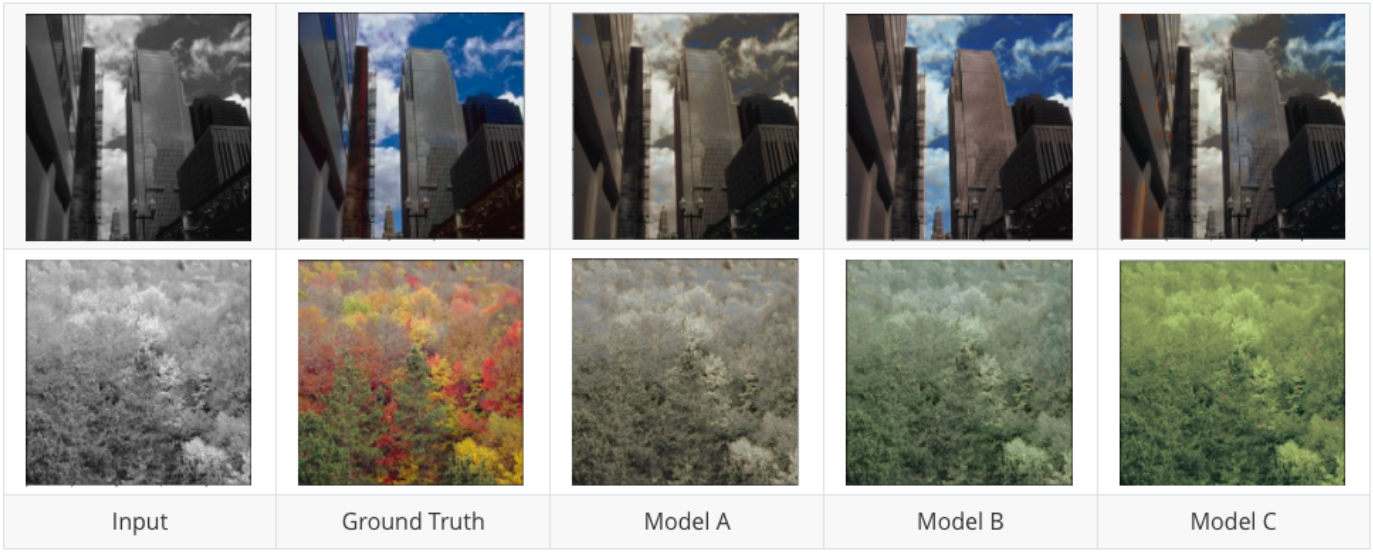

Fig. 4. Results from Deep Colorization

TABLE I
PSNR & AVERAGE PIXEL ACCURACY

Model	PSNR	Pixel Accuracy(%)
A	68.88	68.88%
B	69.44	69.97%
C	69.42	75.88%
CIC(Pre-trained)	-	71.1%
CIC(Rebalancing)	-	67.7%
CIC(CE)	-	63.3%
Ground Truth	-	79%

gets biased towards shade of gray, which is more prominent in the dataset, as shown in fig 5. The model B performs better than the A, and is able to distinguish the sky(and bushes) but its performance is directly affected by the quality of Image Segmentation, just like the second figure in which it has generated black color in the sky between two building. In order to compare the performance quantitatively, we calculated the Peak signal to ratio (PSNR) for all models and obtain quite similar score for each model but on our observations the model C should outperforms model B and A by a huge margin. We came to conclusion that, PSNR can not be a good measure of comparison. Thus we calculated Average Pixel Accuracy for each model. This score of testing is defined in Colorful Image Colorization [2]. The generated images from colorization network is passed into segmentation network for comparative anaysis.

C. Colorful Image Colorization

We used ImageNet to train the proposed model for two settings- (i) using cross-entropy loss instead of the new objective function and (ii) using the new objective function but without rebalancing. The results for pretrained model can be seen in Table III-B1. We analysed the performance of pretrained model by retrieving results for different types of images. Some of the results can be seen in Figure 6. Figure 6 (a) shows the good cases obtained from the model while the Figure 6 (b) shows the not so good cases obtained.

However, the training data could not be analysed separately as it was not available (no information about which images of ImageNet were actually used). It is evident from the images obtained that whenever the model fails to identify the ”correct” color, it is mostly identifying the color *brown* indicating that there is some color biaseness in the training dataset towards the color brown. This behaviour can be seen in Figure 6 (b).

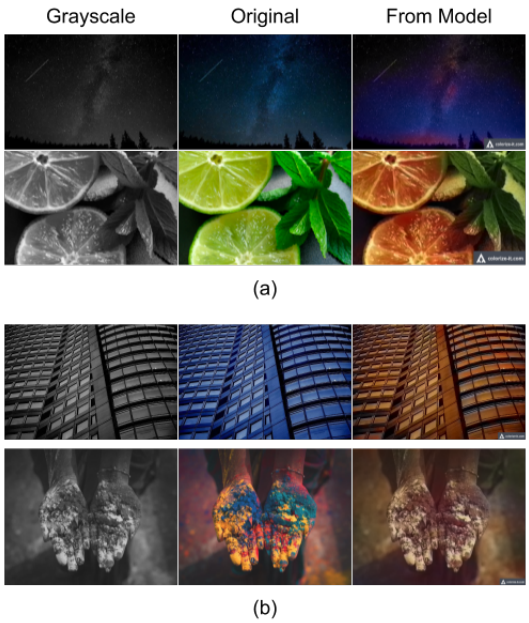

Fig. 6. Colorful Image Colorisation: Best Model Results

1) Model using cross-entropy loss: For the other two settings to be trained, we make use of the dataset that we collected using the imagenet-downloader of 20k images. The CNN model was trained on this dataset but instead of using the author's newly suggested objective function, normal cross-entropy loss without any rebalancing was used for this setting.

The results with this setting can be seen in Figure 7. The results obtained were such that there was a green tint in all the generated images.

Fig. 7. Colorful Image Colorisation: Model using cross-entropy loss results

2) Model with new objective function: The model described is executed using the new objective function without rebalancing. This experiment was carried out to understand the importance of rebalancing in the whole architecture. The results with this setting as well as the comparison with the best model can be seen in Figure 8. The images obtained are somewhat believable but are achieving lower saturation when compared to the whole model, implying the importance of rebalancing.

Fig. 8. Colorful Image Colorization: Model with new Objective function

D. PixColor

During training, input to the network is random cropped and standard normalized luminance of size $[256, 256]$, which provides features of size $[28, 28, 64]$ to the PixelCNN. We used the same setting [4], i.e. one mask-A convolutional layers, followed by 10 Gated-B Convolutional layers and two mask-B layers with 64 output features, as the channels output is discretized into 32 levels. We were able to train the network for 30K iterations with batch size of 8, and learning rate of 0.002 and clipped gradients with norm greater than 2. The results from the network is presented in figure **??**

Fig. 10. Two Different Samples from Same Gray Scale Image

We were able to sample different color images from a single grayscale input, but due to limiting size of ADE20K the model keeps on overfitting after 30K iterations9 and thus some sampled output generated non-natural images Figure 10, and 11

IV. Conclusion

Our project work has been concentrated on learning models, their approaches and understanding their shortcoming and thereon attempt improvements in possible way. We could indulge ourselves into deep neural network paradigm and discovered the art of colouring images from gray scale input. These novel methods studied by us are SoTA and have excelled themselves in far extent to the expectation amongst the community. Further, limitation presented by us in these models can become a path for future studies and discovering another SoTA in the field of Deep Colorization.

V. Individual Contribution

	Karan	Shivani	Sachin
Literature Review	✓	✓	✓
Problem Statement	✓	✓	✓
Algorithm Designing	✓	✓	-
Coding	✓	✓	✓
Results	✓	✓	✓

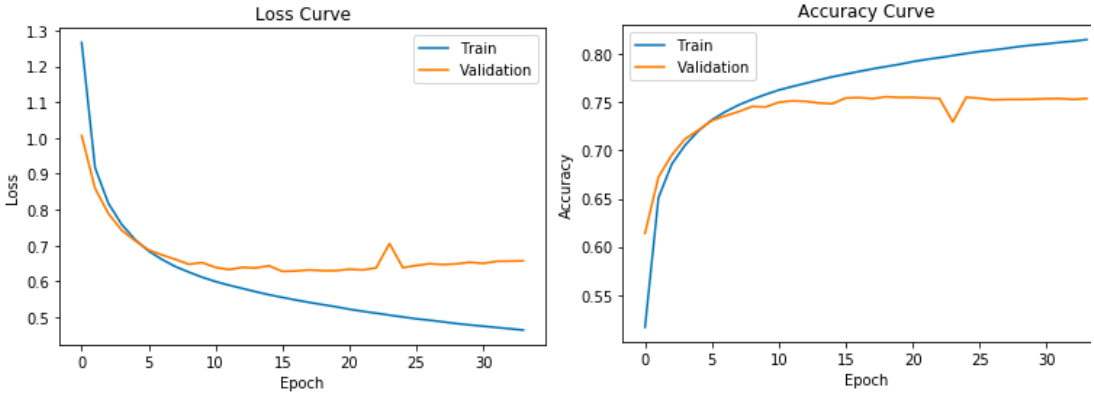

Fig. 9. Learning Graph - Train vs Validation

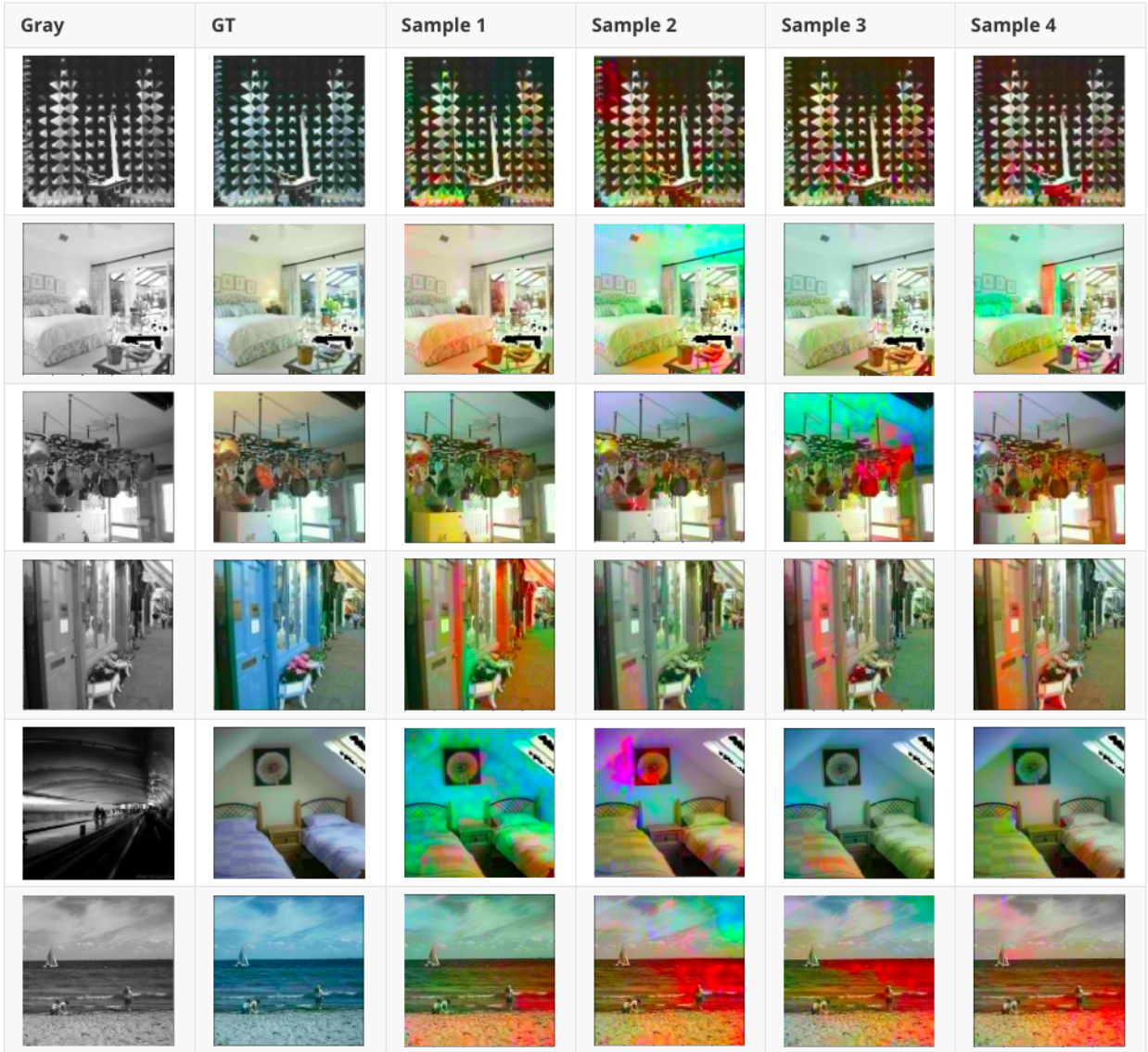

Fig. 11. Four Samples from the given GrayScale Images

REFERENCES

[1] Z. Cheng, Q. Yang, and B. Sheng, "Deep colorization," in *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 415–423.

[2] R. Zhang, P. Isola, and A. A. Efros, "Colorful image colorization," in *European conference on computer vision*. Springer, 2016, pp. 649–666.

[3] M. He, D. Chen, J. Liao, P. V. Sander, and L. Yuan, "Deep exemplar-based colorization," *ACM Transactions on Graphics (TOG)*, vol. 37, no. 4, pp. 1–16, 2018.

[4] S. Guadarrama, R. Dahl, D. Bieber, M. Norouzi, J. Shlens, and K. Murphy, "Pixcolor: Pixel recursive colorization," *CoRR*, vol. abs/1705.07208, 2017. [Online]. Available: http://arxiv.org/abs/1705.07208

[5] A. v. d. Oord, N. Kalchbrenner, and K. Kavukcuoglu, "Pixel recurrent neural networks," *arXiv preprint arXiv:1601.06759*, 2016.

[6] A. Van den Oord, N. Kalchbrenner, L. Espeholt, O. Vinyals, A. Graves *et al.*, "Conditional image generation with pixelcnn decoders," in *Advances in neural information processing systems*, 2016, pp. 4790–4798.

[7] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database," in *CVPR09*, 2009.

[8] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba, "Scene parsing through ade20k dataset," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.