Proceedings of the 2nd International Conference on Inventive Communication and Computational Technologies (ICICCT 2018)
IEEE Xplore Compliant - Part Number: CFP18BAC-ART; ISBN:978-1-5386-1974-2

Automatization of Coloring Grayscale Images using Convolutional Neural Network

Ms. Shweta Salve
Department of Computer Engineering
MCT's Rajiv Gandhi Institute of Technology
Mumbai, India
shwetasalve15@gmail.com

Ms. Twinkle Shah
Department of Computer Engineering
MCT's Rajiv Gandhi Institute of Technology
Mumbai, India
shahtwil28@gmail.com

Ms. Vishakha Ranjane
Department of Computer Engineering
MCT's Rajiv Gandhi Institute of Technology
Mumbai, India
vishakharanjane10@gmail.com

Prof. Sumitra Sadhukhan
Department of Computer Engineering
MCT's Rajiv Gandhi Institute of Technology
Mumbai, India
sumitrassadhukhan@gmail.com

***Abstract*—The paper is about coloring a grayscale image automatically using deep learning techniques. Colorization of images has many applications such as in CCTV Cameras, Astronomical photography, and electron microscopy, photographs of ancient history. In this proposed system, we will design and implement system which color the images automatically with the help of training images. There consists of two types of images one is grayscale image that is input/test image and second one is colored image that is training image then by doing appropriate deep learning models we get the required target image. Our objective is to build use friendly system which colorize the system in faster and in an efficient way.**

Keywords— Deep learning, CNN, Inception-ResNet-V2, Keras, Tensorflow, Anaconda, GPU.

I. INTRODUCTION

Coloring grayscale images can have a larger impact on a wide variety of domains, for instance, electron microscopy, astronomical photography and improvement of surveillance feeds. The information content of a grayscale image is limited, thus coloring the image can provide more insights about its semantics. For instance, a shirt cannot be distinguished as red or blue in case of grayscale images. So in order to obtain a better understanding of a grayscale image we need to colorize them. In deep learning, models such as Inception ResNet V2 also used in [10] are usually pre-trained using colored image datasets. When applying these networks on grayscale images, a prior colorization step can help improve the results. Designing and implementing an effective and accurate system that automates this process still remains a challenging task.

II. RELATED WORK

Various attempts have been made in order to get robust and satisfying colorized grayscaled images.

Anat Levin et al [1] presented a manual colorization process whereby the user has to just assign colors to the grayscale image and the pixel with same value and intensity will get the color. It isn't like other manual techniques where the user has to assign color to each and every pixel or to worry about object boundaries. Also, there is no need of any precise image segmentation technique. This manual technique saves time and doesn't give unwanted colors.

L.Yatziv et al [2] proposed a different method to colorize the grayscale images. On the basis of various concepts of luminance-weighted chrominance blending and fast intrinsic distance computations, robust and high-quality results of images and video are obtained with reduced computational cost and minimized complexity.

A novel method was presented by Bekir Karlik et al [3] on coloring the grayscale images. Various combinations of artificial neural networks and some image processing algorithms were developed to transfer colors from a selected source image to a target image.

Aurelie Bugeau and Vinh-Thong Ta et al. [4] proposed a simple patch based image colorization method where a general distance selection framework for color prediction have been used, using a training image. But the resultant image is not that realistic and image is desaturated.

Raj Kumar Gupta et al [5] proposed a method for colorizing gray images using semantically similar reference images. This method works at resolution of super pixels, in which we extract a variety of features from the reference and the target images.

The drawback of model proposed by Rasoul Kabirzadeh, Patrick Blaes et al. [6] is that it does not distinguish objects globally. For example, the algorithm has difficulty in differentiation a mountain and sky.

In [7] the input color data comes from a source image considered as a reference image. Variational formulation, energy minimization along with patch based method has been used to automatically select set of best possible colors.

A convolutional-neural-network-based system is presented by Jeff Hwang et al [8] that colorizes black and white images without direct human assistance. This system uses concept on baseline regression model. The final classification based model built generates colorized images that are significantly more aesthetically-pleasing than those created by the baseline based model.

Gustav Larsson et al [9] presented an automatic colorization process whereby the model uses deep convolutional network which can help in semantic parsing as well as localization and it predicts color histograms at every image location. Semantic segmentation and edge detection techniques are not accurate while coloring images so deep convolutional network pre trained model is used which adapts itself for image classification. This method gives us visually appealing colored images.

III. PROPOSED SOLUTION

Our system aims to obtain a better yet accurate colored version of the grayscale image. In the existing system the results turned out to be quite good for some of the images, generating near-photorealistic pictures. For instance, natural elements such as the oceans seem to be well recognized. However, specific objects are not always well colored.

So to remove the disadvantage from the existing system our system uses one of most remarkably powerful network known as Inception ResNet V2. It is one of the top and high quality classifier developed by Google.

A. System Architecture

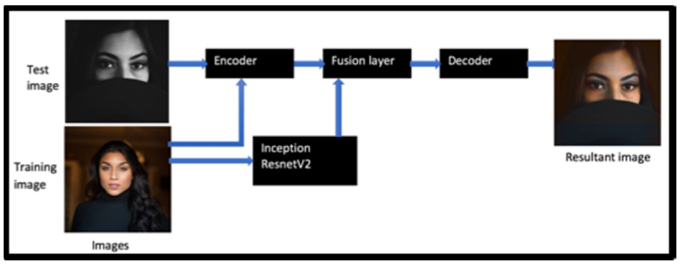

Fig. 1. Architecture design of the system

Our system model is a deep Convolution Neural Network (CNN) which consists of mainly 4 components.

1. Encoder
2. Feature Extractor
3. Fusion layer
4. Decoder

Images are first loaded into the model.

These images are in the RGB color space format initially. Now the system converts this RGB color space into Lab color space. 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. L channel describes how dark or light a color is. L varies between 0 and 100. 0 means total darkness and 100 means maximum light. 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. 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.

Fig. 2. Lab color space prediction of a test image.

Now further, the system adjusts the settings for the images by image generator. In this way each image will be different. Thus increasing the learning rate. The shear_ range tilts the image to the left or right, and the other settings are zoom, rotation and horizontal-flip. Hence a number of images are generated according to the settings made. Further extraction of the black and white layers for the later processing is done.

The feature extraction and the encoder component works in parallel.

B. Comparison of various deep learning networks

Following is the classification performance comparison on ImageNet. (Single crop, single model).

Networks	Top-1 accuracy (%)	Top-5 accuracy (%)
VGG-16	76.0	90.1
ResNet-152	77.0	93.3
Inception V3	**78.2**	**94.1**

TABLE I. COMPARISON OF TOP-1 AND TOP-5 ACCURACIES OF VARIOUS NETWORKS

From the above table Inception V3 is found to be much accurate than other networks. So we compared the various Inception network on the basis of their accuracy.
The Inception-ResNet-V2 architecture is more accurate than other models, as shown in the table below, which shows the Top-1 and Top-5 validation accuracies on the ILSVRC 2012 image classification based on a single crop of the image.

Networks	Top-1 accuracy (%)	Top-5 accuracy (%)
Inception-ResNet-V2	**80.4**	**95.3**
Inception V3	78.0	93.9
ResNet 152	76.8	93.2
ResNet V2 200	79.9	95.2

TABLE II. COMPARISON OF TOP-1 AND TOP-5 ACCURACIES OF VARIOUS INCEPTION NETWORKS

Since, the Top-1 and Top-5 accuracies of Inception ResNet V2 are maximum and error rates are minimum as compared to the other networks, the model will be trained by using this classifier network.

C. Feature Extraction

The next step is feature extraction and classification. We are using Inception ResNet V2 classifier to serve our purpose. The figure below shows the schematic diagram of Inception ResNet V2. It is one of the most powerful classifiers. It is a pre-trained neural network model which is trained on 1.2 million images. It does the work of extracting the features and classifying the patterns. It is more accurate than the previous versions.

Since we will be using two models in parallel, we need to specify which model we are using. This is done in Tensorflow, the backend for Keras. Now in order to create batch we use the tweaked images. For these images to go into the Inception ResNet V2 classifier we need to convert them into the grayscale images again and resize them. Then we use the pre-processor to format the pixel and color values according to the model. In the final step, we run it through the Inception network and extract the final layer of the model i.e. the embed_input.

D. Encoder

The function of the encoder is to process h × w image and produce an output of h/8 × w/8 × 512 representation. To this output, it uses 8 convolutional layers with 3 × 3 filters. Padding is used to preserve the layer's input size.

The stride of 2 is applied on the first, third and fifth layer which halves the dimension of the output and thereby reduces the number of computations required. The input to the encoder is encoder_input and the output is encoder_output. The output of the encoder model that is encoder_output is then fused with the embed_input in the fusion layer.

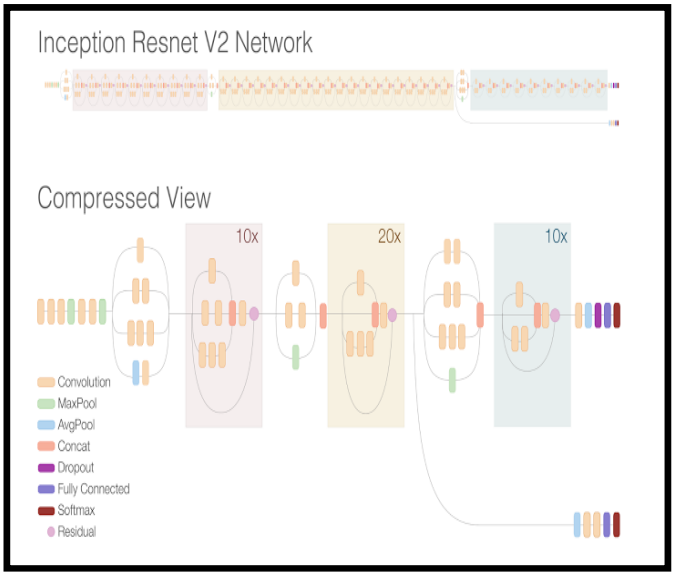

Fig. 3. Schematic diagram of Inception ResNet V2

E. Fusion Layer

The fusion layer takes the feature vector from Inception network, replicates it h x w/8 x 2 times and attaches it to the feature volume output of the encoder along the depth axis. This is illustrated in below figure [4]. This approach obtains a single volume with the encoded image and the mid-level features of shape H/8×H/8×1257. By mirroring the feature vector and concatenating it several times we ensure that the semantic information conveyed by the feature vector is uniformly distributed among all the regions of the image. Also, this solution is also robust to input image sizes. Finally, we apply 256 convolutional kernels of size 1×1, ultimately generating a feature volume of dimension H/8×W/8×256.

In the fusion layer, we multiply the 1000 category layer by 1024. Then we get 1024 rows with the final layer from the Inception model. This is again reshaped from 2D to 3D, a 32 x 32 volume with the 1000 category pillars. These are then combined together with the output from the encoder model.

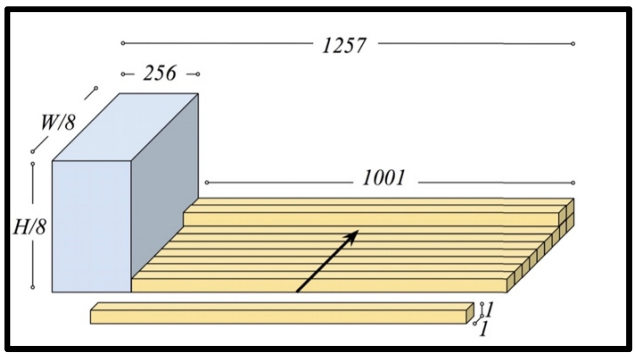

Fig. 4. Fusing the Inception embedding with the output of the layers of the encoder.

F. Decoder

The output of the fusion is then used as input in our Decoder model. Finally, the decoder takes this h/8×w/8×256 layers of volume and applies a series of convolutional and up-sampling layers to obtain a final layer with dimension h ×w ×2. Up-sampling is performed using the nearest neighbour method so that the output's height and width are twice the input's. The final output returned is decode_output.

IV. CONCEPTS USED

A. Convolutional Neural Network

It is a feed forward artificial NN that is applied to analyse the visual imagery. It is inspired by biological processes. It uses little pre-processing as compared to image classification algorithms. The major advantage of using CNN is that it does not depend on the prior knowledge and human effort in feature design.
Types of layers:

1. *Input Layer*: This layer holds the raw input of image with width 32, height 32 and depth 3.
2. *Convolution Layer:* This layer computes the output volume by computing dot product between all filters and image patch. Suppose we use total 12 filters for this layer we'll get output volume of dimension 32 x 32 x 12.
3. *Activation Function Layer:* This layer will apply element wise activation function to the output of convolution layer. Some common activation functions are ReLU: max (0, x), Sigmoid: 1/ (1+e^-x), Tanh, Leaky ReLU, etc.
4. *Pool Layer:* This layer is periodically inserted in the covnets and its main function is to reduce the size of volume which makes the computation fast reduces memory and also prevents from overfitting.
5. *Fully Connected layer:* This layer is regular neural network layer.

B. Activation Functions

Activation function decides whether a neuron should be activated or not in a neural network.

ReLU:
The ReLU function stands for Rectified Linear Unit. It is the most widely used activation function. It is non-linear therefore it has the ability to back propagate the errors. The significance of ReLU activation function is that it does not activate all the neurons at the same time.
Following figure is the graphical representation of ReLU activation function.

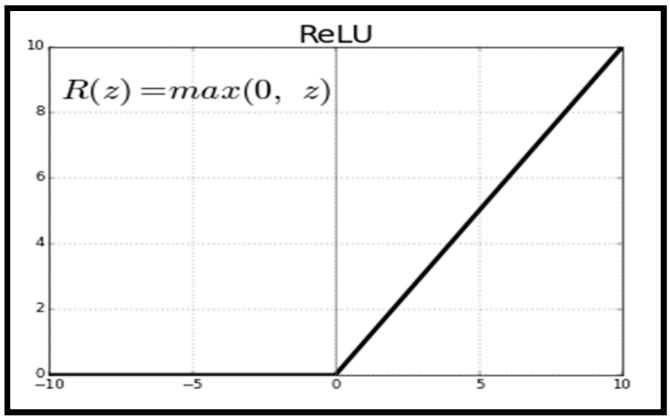

Fig. 5. Graphical representation of ReLU activation function.

Tanh:
It is known as the Hyperbolic Tangent activation function. Tanh takes a real-valued number but suppresses it into a range between -1 and 1. It minimizes the errors.
Following figure is the graphical representation of tanh activation function.

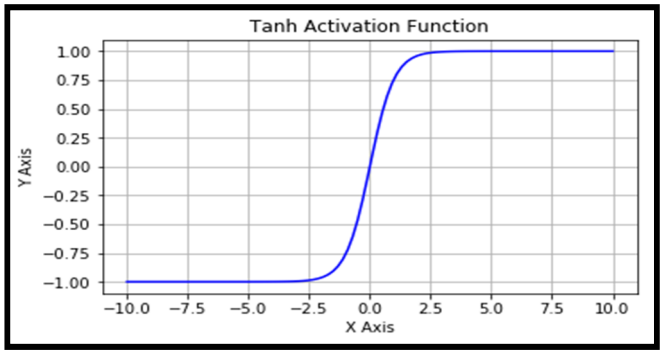

Fig. 6. Graphical representation of Tanh activation function.

V. SYSTEM REQUIREMENTS

Hardware requirements:

a. Dual core processor
b. INTEL core i3 processor or above.
c. GPU

Software Requirements:

a) Windows 7 or above/Linux
b) Python 2.7 or above / Anaconda 2.7 or above
c) Keras
d) Tensorflow
e) Jupyter notebook

RESULT ANALYSIS

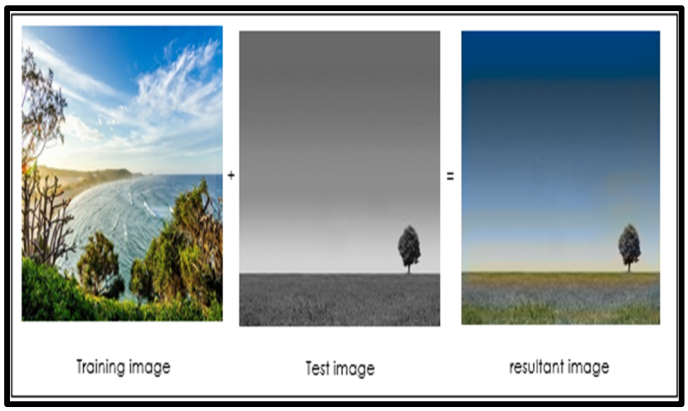

Fig. 7. Resultant colorized image obtained from the training and test image.

In the above figure, first image is the training image and second is the test image. Test image is the grayscale image. By appropriate training and colorization techniques we get the resultant colorized image.

Fig. 8. Resultant colorized image obtained from the training and test image at various epochs.

In the above figure,results are shown at various epochs .As we have trained the neural network more, more desirable results have been obtained.

Sr. no.	No. of Training images	No. of Test images	Time taken	No. of Epochs given	CPU/GPU
1	1	1	50 mins	1000	CPU
2	1	1	7 min 30 sec	1000	GPU
3	10	8	50 mins	1000	GPU

TABLE III. COMPARISON OF TIME COMPLEXITY WITH CPU/GPU FOR THE RESULTS

In the above table,we have compared the time taken by the system to colorize the grayscale image. As the number of training image increases, time taken to obtain result also increases.We obtain faster results if we use GPU.

CONCLUSION

Through this system the objective to colorize the grayscale images automatically using CNN and deep learning techniques has been fulfilled. This system solves the problems presented by the previous papers making this system if not completely, yet full proof and able to differentiate between two different objects or surroundings in a picture. We have trained the model using CNN and Inception ResNet V2 model.

Furthermore, it performs well with given ill posed nature of the problem and shows promise that realistic results are available. By using GPU we are able to obtain the output faster and with much more accuracy.

The future scope of this system could be to decrease the time taken to colorize images, applying it to a video, using different data sets for obtaining more accuracy or implementing it with another model.

REFERENCES

[1] Anat Levin,Dani Lischinski and Yair Weiss, "Colorization using Optimization", *Proceedings of ACM SIGGRAPH ,New York* ,vol. 23 issue 3,pp. 689-694, 2004.

[2] L.Yatziv and G Sapiro ,"Fast Image and Video Colorization using chrominance blending", *IEEE Transaction on Image Processing*, vol. 15 issue.5, pp. 1120-1129,2006.

[3] Bekir Karlik and Mustafa Sarioz," Coloring grayscale image using Artificial Neural Networks", *Adaptive Science & Technology*, ICAST 2009. *2nd International Conference* on. IEEE, 2009.

[4] Aurelie Bugeau and Vinh-Thong Ta ,"Patch-based Image Colorization. Pattern Recognition" *21st International Conference on Pattern Recognition (ICPR)*, 2012.

[5] Raj Kumar Gupta, Alex Yong-Sang Chia, Deepu Rajan, Ee Sin Ng and Huang Zhiyong ," Image colorization using similar images",*20th ACM International Conference on Multimedia*, pp. 369-378, 2012.

[6] Austin Sousa,Rasoul Kabirzadeh,Patrick Blaes "Automatic Colorization of Grayscale Images" *Department of Electrical Engineering, Stanford University*, 2013.

[7] Aurelie Bugeau, Vinh-Thong Ta, and Nicolas Papadakis, "Variational Exemplar-based Image Colorization", 2013.

[8] Jeff Hwang and You Zhou, "Image colorization with Deep Convolutional Neural Networks", 2016.

[9] Gustav Larsson, Michael Maire and Gregory Shakhnarovich," Learning representations for automatic colorization", vol.1, 2017.

[10] Federico Baldassarre, Diego Gonz´alez Mor´ın and Lucas Rod´es-Guirao, "Deep Koalarization: Image Colorization using CNNs and Inception-Resnet-v2" , *KTH Royal Institute of Technology*, 2017.