

ArIES

Image Colorization Project SpecDoc

Team members:

Ashish Singh

Jinal Shah

Tanmay Gupta

Mentors:

Alakh Verma

Harsh Surya

Image Colorization Project

Overview:

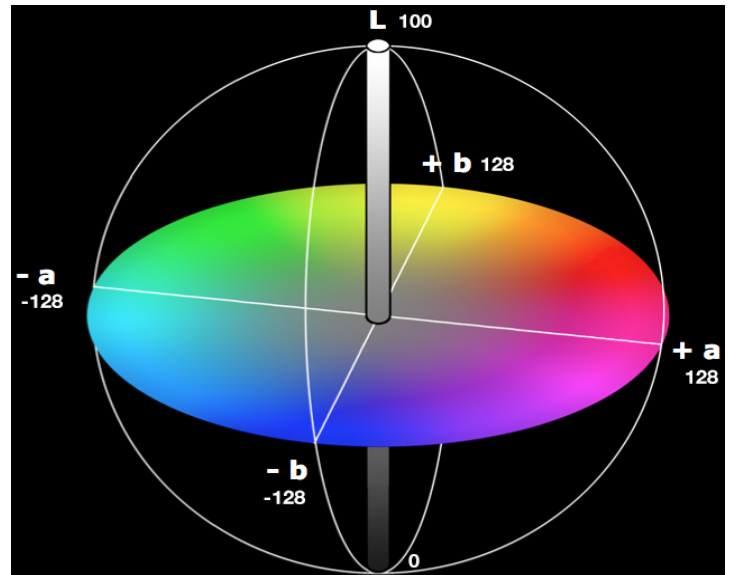
Deep Learning is an upcoming subset of machine learning which makes use of artificial neural networks. These are inspired by the human brain and its immense structure and function. Here we make use of it to colorize black and white images.

Methodology:

Taking input as a black and white image, this model tries to produce a colorized image. We are using TensorFlow and Keras API. Our model is trained using Google Colab.

1. To train the network, we start off with image datasets made up of colourful pictures. The datasets used here are CIFAR10 and Landscape Dataset. Then we convert all images from the RGB color space to the Lab color space. Just like RGB color space, Lab is an alternate color space in which the three channels represent:-

- The *L channel* represents light intensity only.
- The *a channel* encodes green-red color.
- The *b channel* encodes blue-yellow color.



2. Since the *L channel* can encode intensity, we use it as the input in our network in the grayscale format. We put all 'L' values in an array called 'X'.
3. We then use our model to let the network predict the respective *a and b channel* values. The 'ab' values are divided by 128 (feature scaling) so as to reduce values from $[-127, 128]$ to $(-1, 1]$ to make the learning process quicker and more efficient. These are now put in an array called 'Y'.

```

X=[]
Y=[]
for img in train[0]:
    try:
        lab = rgb2lab(img)
        X.append(lab[:, :, 0])
        Y.append(lab[:, :, 1:] / 128 )
    except:
        print('error')
X = np.array(X)
Y = np.array(Y)
X = X.reshape(X.shape+(1,))
print(X.shape)
print(Y.shape)

(2388, 64, 64, 1)
(2388, 64, 64, 2)

```

4. Our model is a Convolutional Neural Network (CNN) which trains on 'X' as feature inputs and 'Y' as target values. The model consists of 11 convolutional layers. There are no pooling layers in our network. Changes in resolution are achieved using striding and upsampling layers. The kernels are sized 3x3.

Model: "sequential_3"		
Layer (type)	Output Shape	Param #
=====		
conv2d_36 (Conv2D)	(None, 64, 64, 32)	320
conv2d_37 (Conv2D)	(None, 64, 64, 64)	18496
conv2d_38 (Conv2D)	(None, 32, 32, 64)	36928
conv2d_39 (Conv2D)	(None, 32, 32, 128)	73856
dropout_3 (Dropout)	(None, 32, 32, 128)	0
conv2d_40 (Conv2D)	(None, 16, 16, 128)	147584
conv2d_41 (Conv2D)	(None, 16, 16, 256)	295168
conv2d_42 (Conv2D)	(None, 16, 16, 128)	295040
conv2d_43 (Conv2D)	(None, 16, 16, 128)	147584
up_sampling2d_6 (UpSampling2D)	(None, 32, 32, 128)	0
conv2d_44 (Conv2D)	(None, 32, 32, 64)	73792
up_sampling2d_7 (UpSampling2D)	(None, 64, 64, 64)	0
conv2d_45 (Conv2D)	(None, 64, 64, 32)	18464
conv2d_46 (Conv2D)	(None, 64, 64, 16)	4624
conv2d_47 (Conv2D)	(None, 64, 64, 2)	290
=====		
Total params: 1,112,146		
Trainable params: 1,112,146		
Non-trainable params: 0		

5. It uses the *ADAM* optimizer with a *learning rate* of *0.0003*. The loss function used is *mean squared error*. It also includes a Dropout layer to prevent overfitting.

```
opt = tf.keras.optimizers.Adam(learning_rate=0.0003)
model.compile( loss='mse' , metrics=['accuracy'])
```

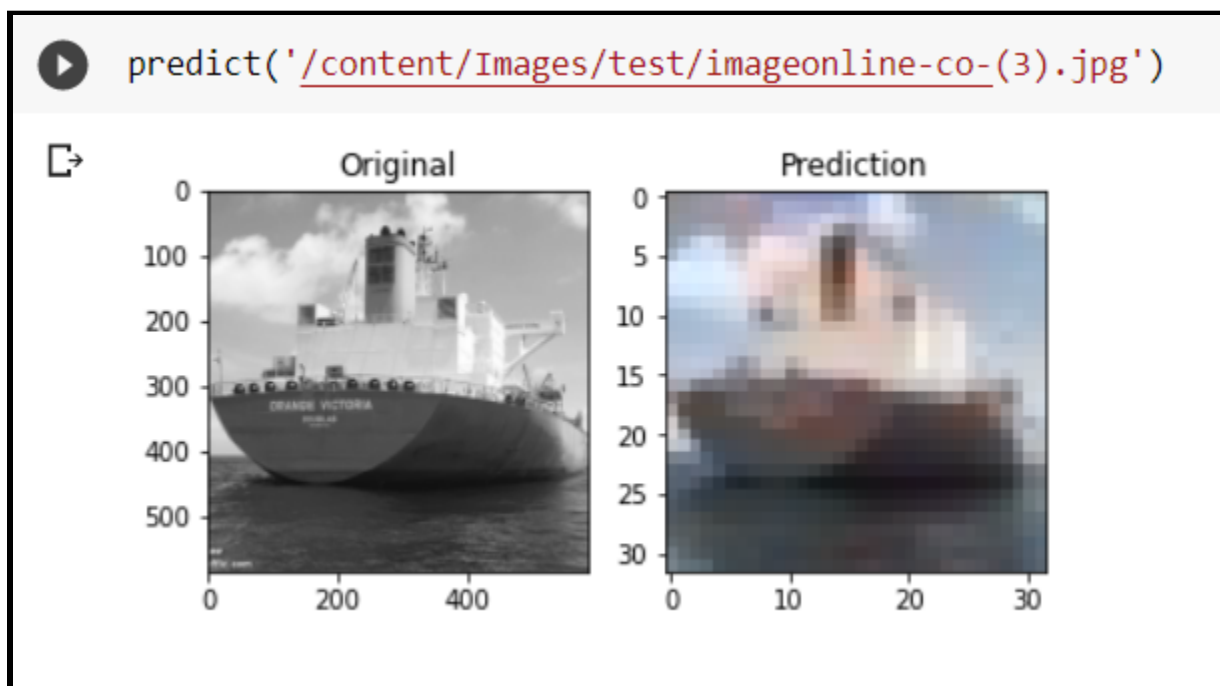
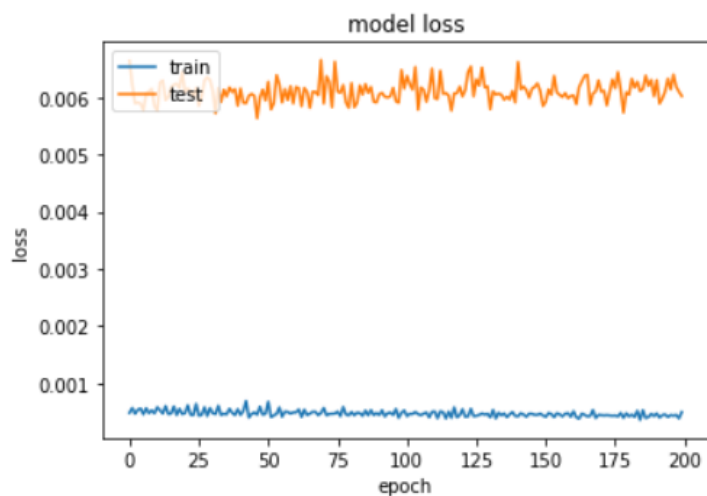
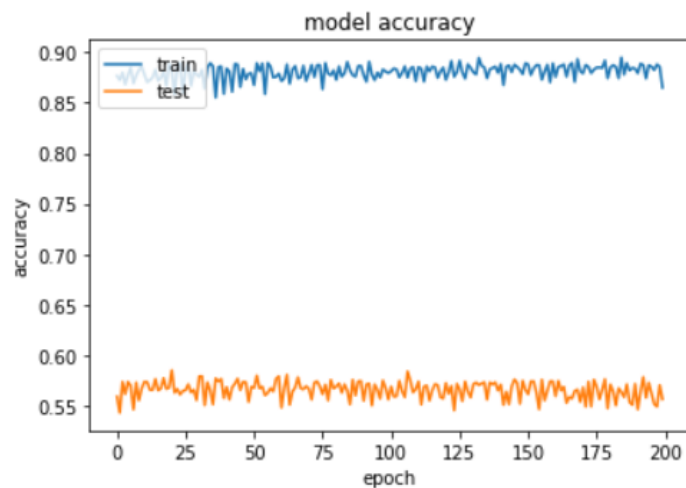
6. 10% of the dataset is used as the validation set. Our network trains for 200 epochs.

```
model.fit(X,Y,validation_split=0.1, epochs=200, batch_size=100)
```

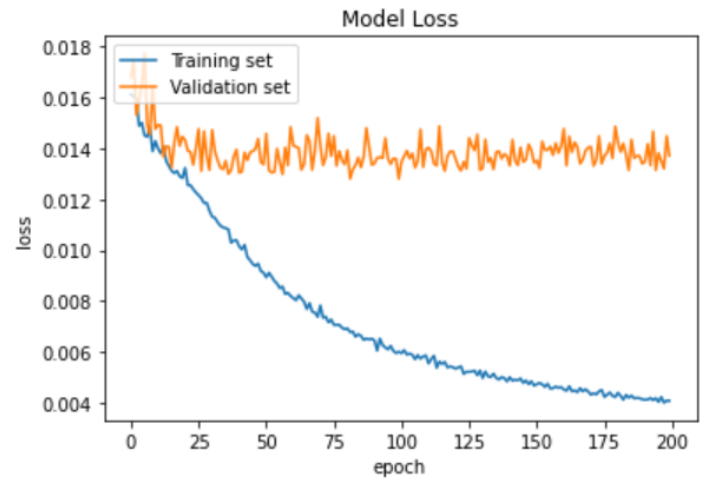
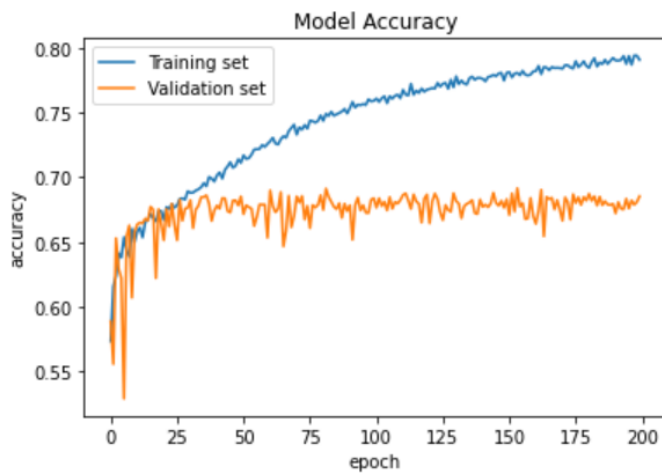
Finally, we define a function which an image as an input and outputs its original, grayscale and predicted colourized form as predicted by our model.

Results:

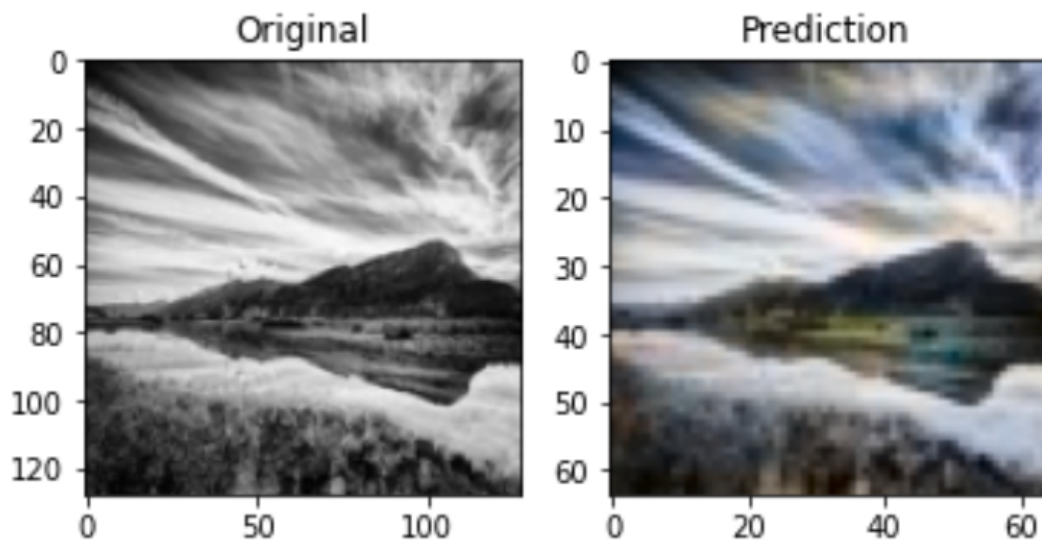
1. Ship:



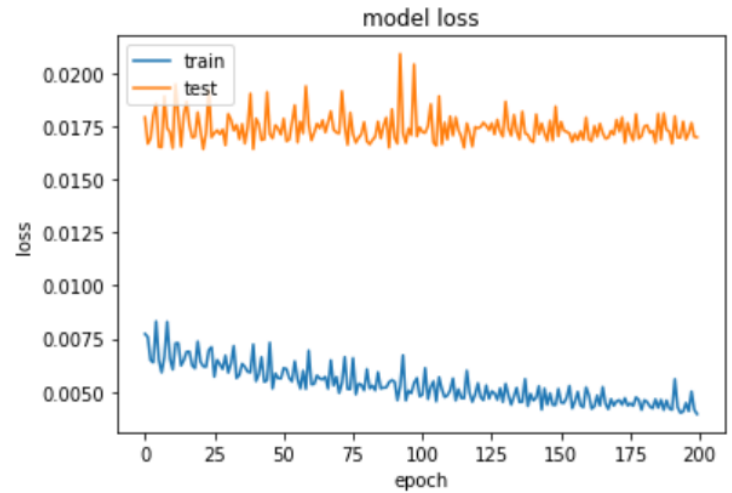
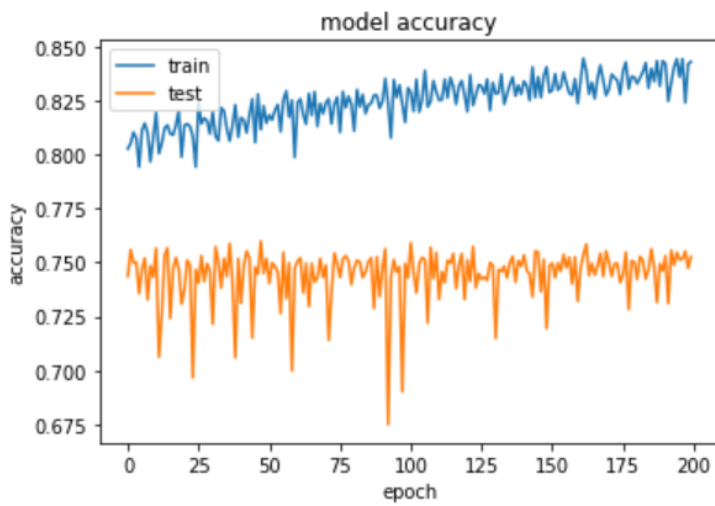
2. Landscape:



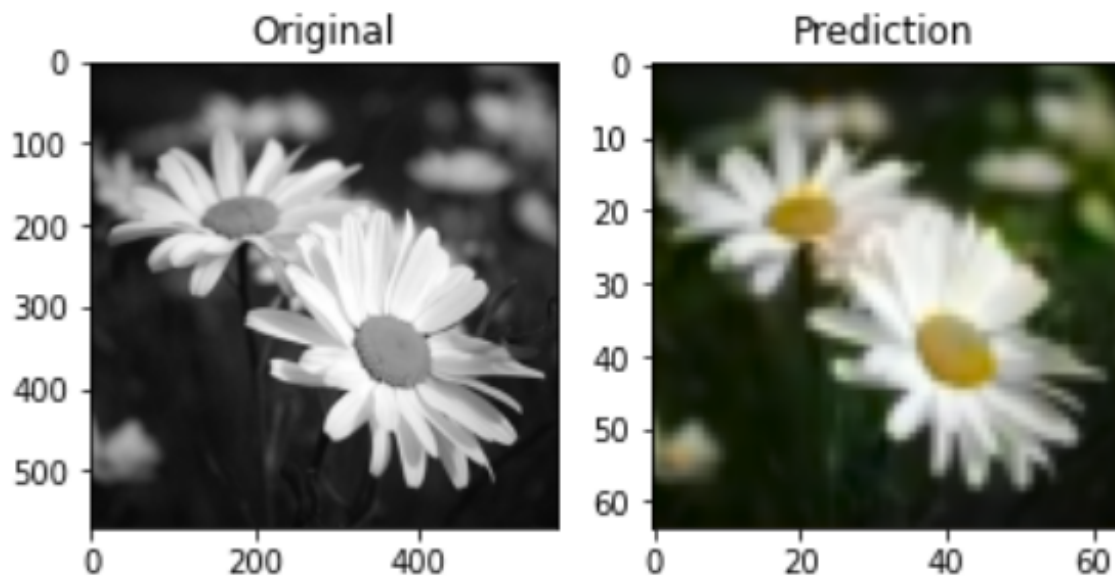
```
predict('/content/Test-images/download.jpg')
```



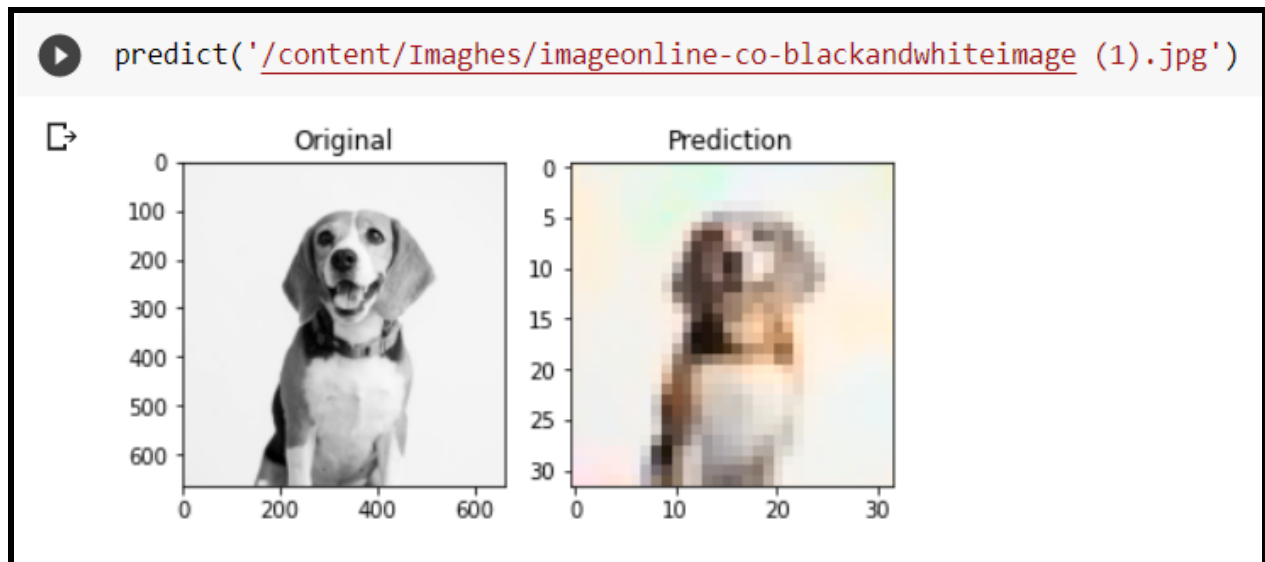
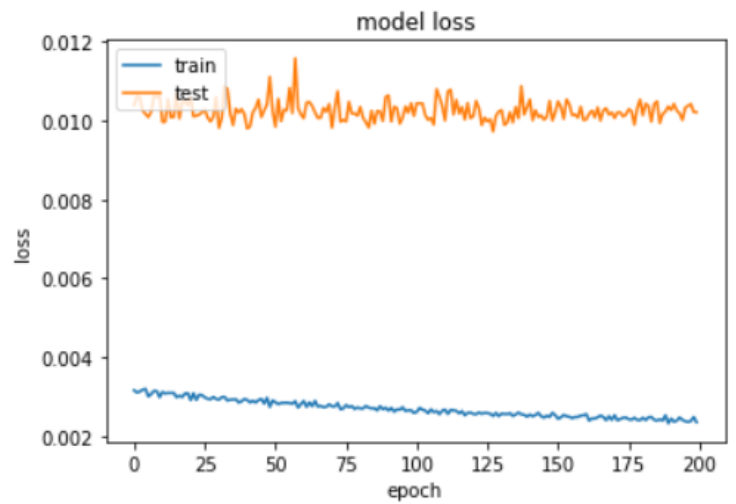
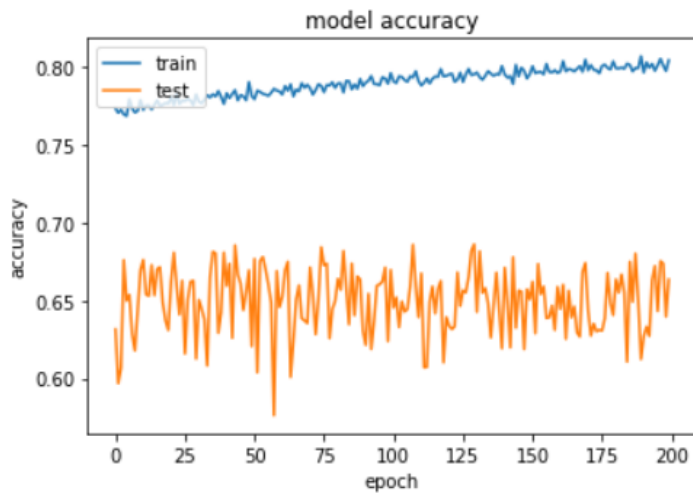
3. Flower:



```
predict('/content/fool/test/daisy.jpg')
```



4. Dog:



References:

1. Colorful Image Colorization Research Paper by *Richard Zhang, Phillip Isola, Alexei A. Efros*:
<https://arxiv.org/abs/1603.08511>
2. Datasets used:
 - a. <https://www.kaggle.com/arnaud58/landscape-pictures>
 - b. <https://www.cs.toronto.edu/~kriz/cifar.html>
3. <https://www.pyimagesearch.com/2019/02/25/black-and-white-image-colorization-with-opencv-and-deep-learning/>
4. <https://www.youtube.com/watch?v=VyWAvY2CF9c>