# Image Colorization with CNN

In [ ]:
```
import numpy as np
import pandas as pd # used for data processing and reading csv
import os
import cv2
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from tensorflow.keras.optimizers import Adam
from keras import backend as K
from keras.layers import Conv2D,MaxPooling2D,UpSampling2D,Input,BatchN
from keras.layers.merge import concatenate
from keras.models import Model
from keras.preprocessing.image import ImageDataGenerator
import keras
#from tensorflow import set_random_seed
import tensorflow.compat.v1 as tf

tf.set_random_seed(123)
session_conf = tf.ConfigProto(intra_op_parallelism_threads=1, inter_op
sess = tf.Session(graph=tf.get_default_graph(), config=session_conf)
tf.keras.backend.set_session(sess)
tf.set_random_seed(2)
np.random.seed(1)
from google.colab import drive
drive.mount('/content/drive')
```

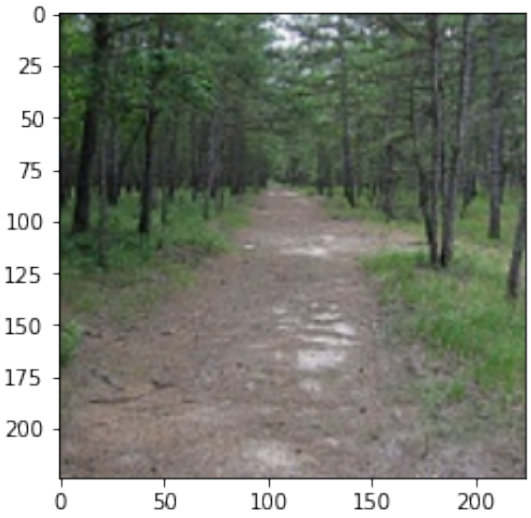
```
Mounted at /content/drive
```

Loading the image path

In [ ]:
```
ImagePath="/content/drive/MyDrive/Deep Learning/"
```

In [ ]:
```
img = cv2.imread(ImagePath+"20056.jpg")
img = cv2.cvtColor(img,cv2.COLOR_BGR2RGB)
img = cv2.resize(img, (224, 224))
plt.imshow(img)
img.shape
```

Out[4]: (224, 224, 3)



In [ ]:
```
HEIGHT=224
WIDTH=224
ImagePath="/content/drive/MyDrive/Deep Learning/"

def ExtractInput(path):
    X_img=[]
    y_img=[]
    for imageDir in os.listdir(ImagePath):
        try:
            img = cv2.imread(ImagePath + imageDir)
            img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
            img = cv2.cvtColor(img, cv2.COLOR_RGB2GRAY)
            img = cv2.cvtColor(img, cv2.COLOR_GRAY2RGB)

            img = img.astype(np.float32)
            img_lab = cv2.cvtColor(img, cv2.COLOR_RGB2Lab)
            img_lab_rs = cv2.resize(img_lab, (WIDTH, HEIGHT)) # resize
            img_l = img_lab_rs[:,:,0] # pull out L channel
            img_ab = img_lab_rs[:,:,1:]#Extracting the ab channel
            img_ab = img_ab/128
            X_img.append(img_l)
            y_img.append(img_ab)
        except:
            pass
    X_img = np.array(X_img)
    y_img = np.array(y_img)

    return X_img,y_img
```

In [ ]:
```
X_,y_ = ExtractInput(ImagePath) # Data-preprocessing
```

In [ ]:
```
X_train,X_val,y_train,y_val = train_test_split(X_,y_, random_state=42)
```

In [ ]:
```
print()
```

In [ ]:
```
K.clear_session()
def InstantiateModel(in_):
    model_ = Conv2D(16,(3,3),padding='same',strides=1)(in_)
    model_ = LeakyReLU()(model_)
    model_ = Conv2D(32,(3,3),padding='same',strides=1)(model_)
    model_ = LeakyReLU()(model_)
    model_ = BatchNormalization()(model_)
    model_ = MaxPooling2D(pool_size=(2,2),padding='same')(model_)

    model_ = Conv2D(64,(3,3),padding='same',strides=1)(model_)
    model_ = LeakyReLU()(model_)
    model_ = BatchNormalization()(model_)
    model_ = MaxPooling2D(pool_size=(2,2),padding='same')(model_)

    model_ = Conv2D(128,(3,3),padding='same',strides=1)(model_)
    model_ = LeakyReLU()(model_)
    model_ = BatchNormalization()(model_)

    model_ = Conv2D(256,(3,3),padding='same',strides=1)(model_)
    model_ = LeakyReLU()(model_)
    model_ = BatchNormalization()(model_)

    model_ = UpSampling2D((2, 2))(model_)
    model_ = Conv2D(128,(3,3),padding='same',strides=1)(model_)
    model_ = LeakyReLU()(model_)
    model_ = BatchNormalization()(model_)

    model_ = UpSampling2D((2, 2))(model_)
    model_ = Conv2D(64,(3,3), padding='same',strides=1)(model_)
    model_ = LeakyReLU()(model_)

    concat_ = concatenate([model_, in_])

    model_ = Conv2D(64,(3,3), padding='same',strides=1)(concat_)
    model_ = LeakyReLU()(model_)
    model_ = BatchNormalization()(model_)

    model_ = Conv2D(32,(3,3),padding='same',strides=1)(model_)
    model_ = LeakyReLU()(model_)

    model_ = Conv2D(2,(3,3), activation='tanh',padding='same',strides=

    return model_
```

In [ ]:
```
Input_Sample = Input(shape=(HEIGHT, WIDTH,1))
Output_ = InstantiateModel(Input_Sample)
Model_Colourization = Model(inputs=Input_Sample, outputs=Output_)
```

In [ ]:
```
LEARNING_RATE = 0.001
Model_Colourization.compile(optimizer=Adam(lr=LEARNING_RATE),
                            loss='mean_squared_error')
Model_Colourization.summary()
```

```
Model: "model"
__________________________________________________________________________________
 Layer (type)                   Output Shape         Param #     Conn
ected to
==================================================================================
 input_1 (InputLayer)           [(None, 224, 224, 1  0           []
                                )]

 conv2d (Conv2D)                (None, 224, 224, 16  160         ['in
put_1[0][0]']
                                )

 leaky_re_lu (LeakyReLU)        (None, 224, 224, 16  0           ['co
nv2d[0][0]']
                                )

 conv2d_1 (Conv2D)              (None, 224, 224, 32  4640        ['le
aky_re_lu[0][0]']
                                )

 leaky_re_lu_1 (LeakyReLU)      (None, 224, 224, 32  0           ['co
nv2d_1[0][0]']
                                )

 batch_normalization (BatchNorm  (None, 224, 224, 32  128        ['le
aky_re_lu_1[0][0]']
 alization)                     )

 max_pooling2d (MaxPooling2D)   (None, 112, 112, 32  0           ['ba
tch_normalization[0][0]']
                                )
```

```
 conv2d_2 (Conv2D)              (None, 112, 112, 64  18496       ['ma
x_pooling2d[0][0]']
                                )

 leaky_re_lu_2 (LeakyReLU)      (None, 112, 112, 64  0           ['co
nv2d_2[0][0]']
                                )

 batch_normalization_1 (BatchNo  (None, 112, 112, 64  256        ['le
aky_re_lu_2[0][0]']
 rmalization)                   )

 max_pooling2d_1 (MaxPooling2D)  (None, 56, 56, 64)  0           ['ba
tch_normalization_1[0][0]']

 conv2d_3 (Conv2D)              (None, 56, 56, 128)  73856       ['ma
x_pooling2d_1[0][0]']

 leaky_re_lu_3 (LeakyReLU)      (None, 56, 56, 128)  0           ['co
nv2d_3[0][0]']

 batch_normalization_2 (BatchNo  (None, 56, 56, 128)  512        ['le
aky_re_lu_3[0][0]']
 rmalization)

 conv2d_4 (Conv2D)              (None, 56, 56, 256)  295168      ['ba
tch_normalization_2[0][0]']

 leaky_re_lu_4 (LeakyReLU)      (None, 56, 56, 256)  0           ['co
nv2d_4[0][0]']

 batch_normalization_3 (BatchNo  (None, 56, 56, 256)  1024       ['le
aky_re_lu_4[0][0]']
 rmalization)

 up_sampling2d (UpSampling2D)   (None, 112, 112, 25  0           ['ba
tch_normalization_3[0][0]']
                                6)

 conv2d_5 (Conv2D)              (None, 112, 112, 12  295040      ['up
_sampling2d[0][0]']
                                8)

 leaky_re_lu_5 (LeakyReLU)      (None, 112, 112, 12  0           ['co
nv2d_5[0][0]']
                                8)

 batch_normalization_4 (BatchNo  (None, 112, 112, 12  512        ['le
aky_re_lu_5[0][0]']
 rmalization)                   8)
```

```
 up_sampling2d_1 (UpSampling2D)  (None, 224, 224, 12  0           ['ba
tch_normalization_4[0][0]']
                                8)

 conv2d_6 (Conv2D)              (None, 224, 224, 64  73792       ['up
_sampling2d_1[0][0]']
                                )

 leaky_re_lu_6 (LeakyReLU)      (None, 224, 224, 64  0           ['co
nv2d_6[0][0]']
                                )

 concatenate (Concatenate)      (None, 224, 224, 65  0           ['le
aky_re_lu_6[0][0]',
                                )                                 'in
put_1[0][0]']

 conv2d_7 (Conv2D)              (None, 224, 224, 64  37504       ['co
ncatenate[0][0]']
                                )

 leaky_re_lu_7 (LeakyReLU)      (None, 224, 224, 64  0           ['co
nv2d_7[0][0]']
                                )

 batch_normalization_5 (BatchNo  (None, 224, 224, 64  256        ['le
aky_re_lu_7[0][0]']
 rmalization)                   )

 conv2d_8 (Conv2D)              (None, 224, 224, 32  18464       ['ba
tch_normalization_5[0][0]']
                                )

 leaky_re_lu_8 (LeakyReLU)      (None, 224, 224, 32  0           ['co
nv2d_8[0][0]']
                                )

 conv2d_9 (Conv2D)              (None, 224, 224, 2)  578         ['le
aky_re_lu_8[0][0]']

==================================================================
==============================
Total params: 820,386
Trainable params: 819,042
Non-trainable params: 1,344
__________________________________________________________________
______________________________

/usr/local/lib/python3.7/dist-packages/keras/optimizer_v2/adam.py:105
: UserWarning: The `lr` argument is deprecated, use `learning_rate` i
```

```
nstead.
  super(Adam, self).__init__(name, **kwargs)
```

In [ ]:
```
def GenerateInputs(X_,y_):
    for i in range(len(X_)):
        X_input = X_[i].reshape(1,224,224,1)
        y_input = y_[i].reshape(1,224,224,2)
        yield (X_input,y_input)
checkpoint_filepath = '/content/drive/MyDrive/CNN/checkpoint'
model_checkpoint_callback = tf.keras.callbacks.ModelCheckpoint(
    filepath=checkpoint_filepath,
    save_weights_only=True,
    monitor='val_acc',
    mode='max',
    save_best_only=True)
Model_Colourization = keras.models.load_model('/content/drive/MyDrive/
Model_Colourization.fit_generator(GenerateInputs(X_,y_),epochs=40,verb
Model_Colourization.save('/content/drive/MyDrive/CNN/Model_Colourizati
```

```
Epoch 1/40

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:14: User
Warning: `Model.fit_generator` is deprecated and will be removed in a
future version. Please use `Model.fit`, which supports generators.


40/40 [==============================] - 80s 2s/step - loss: 6.2425e-
06
Epoch 2/40
40/40 [==============================] - 74s 2s/step - loss: 5.5684e-
06
Epoch 3/40
40/40 [==============================] - 73s 2s/step - loss: 1.9193e-
06
Epoch 4/40
40/40 [==============================] - 73s 2s/step - loss: 1.4038e-
05
Epoch 5/40
40/40 [==============================] - 73s 2s/step - loss: 6.8002e-
06
Epoch 6/40
40/40 [==============================] - 73s 2s/step - loss: 1.5060e-

06
Epoch 7/40
40/40 [==============================] - 73s 2s/step - loss: 9.5100e-
06
Epoch 8/40
40/40 [==============================] - 72s 2s/step - loss: 3.9833e-
06
Epoch 9/40
```

```
Epoch 9/40
40/40 [==============================] - 73s 2s/step - loss: 9.4687e-
06
Epoch 10/40
40/40 [==============================] - 73s 2s/step - loss: 1.1352e-
05
Epoch 11/40
40/40 [==============================] - 73s 2s/step - loss: 6.8041e-
06
Epoch 12/40
40/40 [==============================] - 73s 2s/step - loss: 1.8981e-
05
Epoch 13/40
40/40 [==============================] - 73s 2s/step - loss: 1.7091e-
05
Epoch 14/40
40/40 [==============================] - 73s 2s/step - loss: 6.8975e-
06
Epoch 15/40
40/40 [==============================] - 73s 2s/step - loss: 2.9812e-
06
Epoch 16/40
40/40 [==============================] - 73s 2s/step - loss: 2.4602e-
06
Epoch 17/40
40/40 [==============================] - 73s 2s/step - loss: 3.0610e-
06
Epoch 18/40
40/40 [==============================] - 73s 2s/step - loss: 8.2604e-
06
Epoch 19/40
40/40 [==============================] - 73s 2s/step - loss: 2.7190e-
05
Epoch 20/40
40/40 [==============================] - 73s 2s/step - loss: 3.3542e-
06
Epoch 21/40
40/40 [==============================] - 73s 2s/step - loss: 4.9227e-
06
Epoch 22/40
40/40 [==============================] - 73s 2s/step - loss: 1.1562e-
05
Epoch 23/40
40/40 [==============================] - 73s 2s/step - loss: 1.0428e-
05
Epoch 24/40
40/40 [==============================] - 72s 2s/step - loss: 7.1335e-
06
Epoch 25/40
40/40 [==============================] - 73s 2s/step - loss: 7.2008e-
06
```

```
06
Epoch 26/40
40/40 [==============================] - 73s 2s/step - loss: 1.1901e-
05
Epoch 27/40
40/40 [==============================] - 73s 2s/step - loss: 1.1012e-
05
Epoch 28/40
40/40 [==============================] - 73s 2s/step - loss: 2.0832e-
06
Epoch 29/40
40/40 [==============================] - 73s 2s/step - loss: 2.4532e-
06
Epoch 30/40
40/40 [==============================] - 73s 2s/step - loss: 1.1199e-
05
Epoch 31/40
40/40 [==============================] - 73s 2s/step - loss: 7.7890e-
05
Epoch 32/40
40/40 [==============================] - 73s 2s/step - loss: 4.7251e-
05
Epoch 33/40
40/40 [==============================] - 73s 2s/step - loss: 2.3196e-
06
Epoch 34/40
40/40 [==============================] - 73s 2s/step - loss: 1.2447e-
06
Epoch 35/40
40/40 [==============================] - 73s 2s/step - loss: 2.0690e-
06
Epoch 36/40
40/40 [==============================] - 73s 2s/step - loss: 8.4212e-
07
Epoch 37/40
40/40 [==============================] - 73s 2s/step - loss: 6.4166e-
07
Epoch 38/40
40/40 [==============================] - 73s 2s/step - loss: 5.5359e-
07
Epoch 39/40
40/40 [==============================] - 73s 2s/step - loss: 3.4691e-
06
Epoch 40/40
40/40 [==============================] - 73s 2s/step - loss: 1.5790e-
06
```

In [ ]:
```
TestImagePath="/content/drive/MyDrive/Deep Learning/"
```

In [ ]:
```
def ExtractTestInput(ImagePath):
    img = cv2.imread(ImagePath)
    img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
    img = cv2.cvtColor(img, cv2.COLOR_RGB2GRAY)
    img_ = cv2.cvtColor(img, cv2.COLOR_GRAY2RGB)
    img_ = cv2.cvtColor(img_, cv2.COLOR_RGB2Lab)
    img_=img_.astype(np.float32)
    img_lab_rs = cv2.resize(img_, (WIDTH, HEIGHT)) # resize image to n
    img_l = img_lab_rs[:,:,0] # pull out L channel
    img_l_reshaped = img_l.reshape(1,224,224,1)

    return img_l_reshaped
```

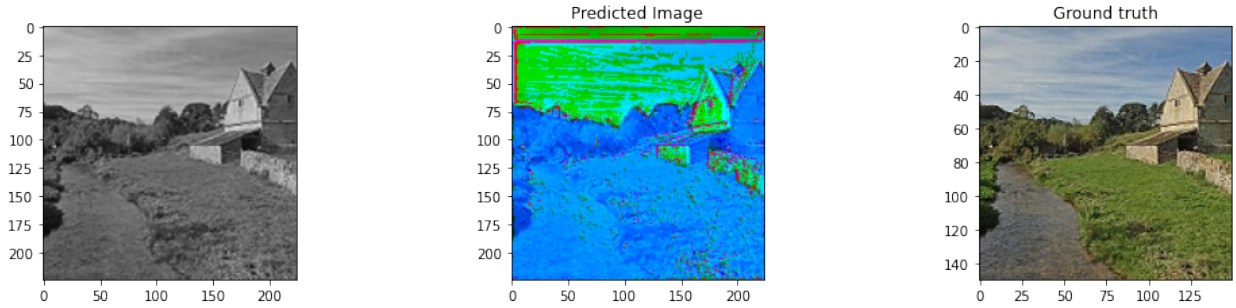
In [ ]:
```
ImagePath=TestImagePath+"20074.jpg"
image_for_test = ExtractTestInput(ImagePath)
Prediction = Model_Colourization.predict(image_for_test)
Prediction = Prediction*128
Prediction=Prediction.reshape(224,224,2)
```

In [ ]:
```
plt.figure(figsize=(30,20))
plt.subplot(5,5,1)
img = cv2.imread(TestImagePath+"20074.jpg")
img_1 = cv2.cvtColor(img,cv2.COLOR_BGR2RGB)
img = cv2.cvtColor(img_1, cv2.COLOR_RGB2GRAY)
img = cv2.cvtColor(img, cv2.COLOR_GRAY2RGB)
img = cv2.resize(img, (224, 224))
plt.imshow(img)

plt.subplot(5,5,1+1)
img_ = cv2.cvtColor(img, cv2.COLOR_RGB2Lab)
img_[:,:,1:] = Prediction
img_ = cv2.cvtColor(img_, cv2.COLOR_Lab2RGB)
plt.title("Predicted Image")
plt.imshow(img_)

plt.subplot(5,5,1+2)
plt.title("Ground truth")
plt.imshow(img_1)
```

Out[21]: <matplotlib.image.AxesImage at 0x7fd8c48b7150>

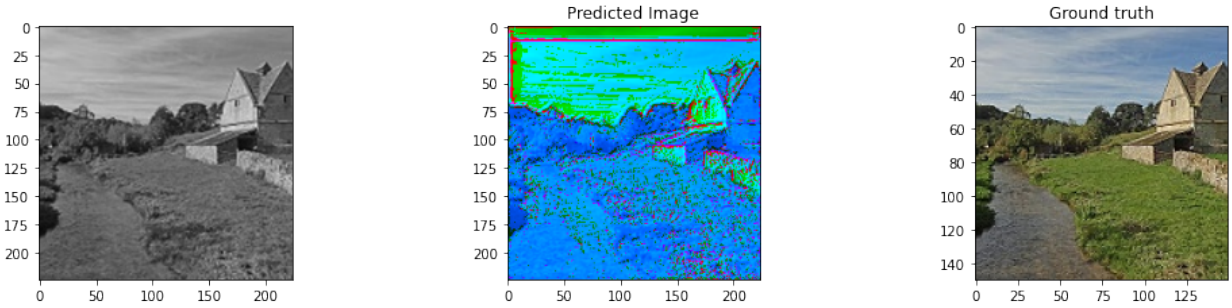


In [ ]:
```
plt.figure(figsize=(30,20))
plt.subplot(5,5,1)
img = cv2.imread(TestImagePath+"20074.jpg")
img_1 = cv2.cvtColor(img,cv2.COLOR_BGR2RGB)
img = cv2.cvtColor(img_1, cv2.COLOR_RGB2GRAY)
img = cv2.cvtColor(img, cv2.COLOR_GRAY2RGB)
img = cv2.resize(img, (224, 224))
plt.imshow(img)

plt.subplot(5,5,1+1)
img_ = cv2.cvtColor(img, cv2.COLOR_RGB2Lab)
img_[:,:,1:] = Prediction
img_ = cv2.cvtColor(img_, cv2.COLOR_Lab2RGB)
plt.title("Predicted Image")
plt.imshow(img_)

plt.subplot(5,5,1+2)
plt.title("Ground truth")
plt.imshow(img_1)
```

Out[16]: <matplotlib.image.AxesImage at 0x7fd8c48f6dd0>



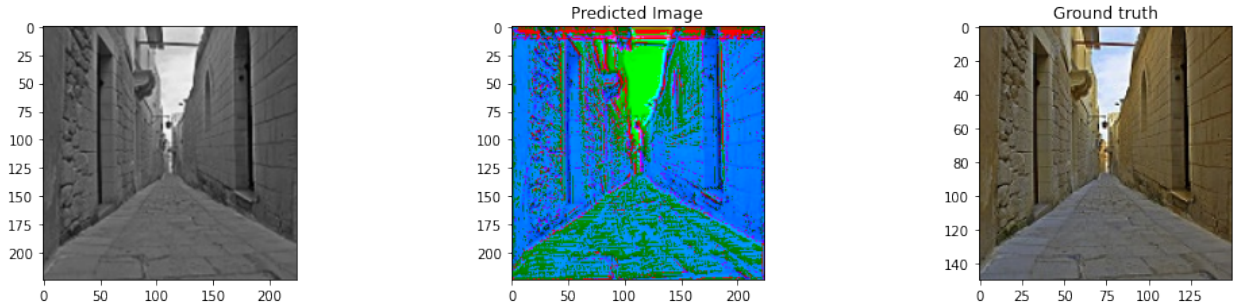

In [ ]:
```
ImagePath=TestImagePath+"20080.jpg"
image_for_test = ExtractTestInput(ImagePath)
Prediction_1 = Model_Colourization.predict(image_for_test)
Prediction_1 = Prediction_1*128
Prediction_1=Prediction_1.reshape(224,224,2)
```

In [ ]:
```
plt.figure(figsize=(30,20))
plt.subplot(5,5,1)
img = cv2.imread(TestImagePath+"20080.jpg")
img_1 = cv2.cvtColor(img,cv2.COLOR_BGR2RGB)
img = cv2.cvtColor(img_1, cv2.COLOR_RGB2GRAY)
img = cv2.cvtColor(img, cv2.COLOR_GRAY2RGB)
img = cv2.resize(img, (224, 224))
plt.imshow(img)

plt.subplot(5,5,1+1)
img_ = cv2.cvtColor(img, cv2.COLOR_RGB2Lab)
img_[:,:,1:] = Prediction_1
img_ = cv2.cvtColor(img_, cv2.COLOR_Lab2RGB)
plt.title("Predicted Image")
plt.imshow(img_)

plt.subplot(5,5,1+2)
plt.title("Ground truth")
plt.imshow(img_1)
```

Out[56]: <matplotlib.image.AxesImage at 0x7fe76d630250>



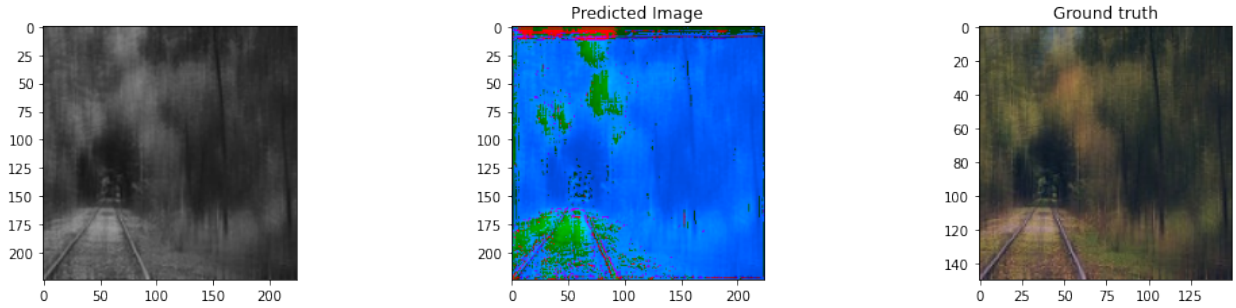

In [ ]:
```
ImagePath=TestImagePath+"20147.jpg"
image_for_test = ExtractTestInput(ImagePath)
Prediction_2 = Model_Colourization.predict(image_for_test)
Prediction_2 = Prediction_2*128
Prediction_2=Prediction_2.reshape(224,224,2)
```

In [ ]:
```
plt.figure(figsize=(30,20))
plt.subplot(5,5,1)
img = cv2.imread(TestImagePath+"20147.jpg")
img_1 = cv2.cvtColor(img,cv2.COLOR_BGR2RGB)
img = cv2.cvtColor(img_1, cv2.COLOR_RGB2GRAY)
img = cv2.cvtColor(img, cv2.COLOR_GRAY2RGB)
img = cv2.resize(img, (224, 224))
plt.imshow(img)

plt.subplot(5,5,1+1)
img_ = cv2.cvtColor(img, cv2.COLOR_RGB2Lab)
img_[:,:,1:] = Prediction_2
img_ = cv2.cvtColor(img_, cv2.COLOR_Lab2RGB)
plt.title("Predicted Image")
plt.imshow(img_)

plt.subplot(5,5,1+2)
plt.title("Ground truth")
plt.imshow(img_1)
```

Out[58]: <matplotlib.image.AxesImage at 0x7fe76d4e9a50>



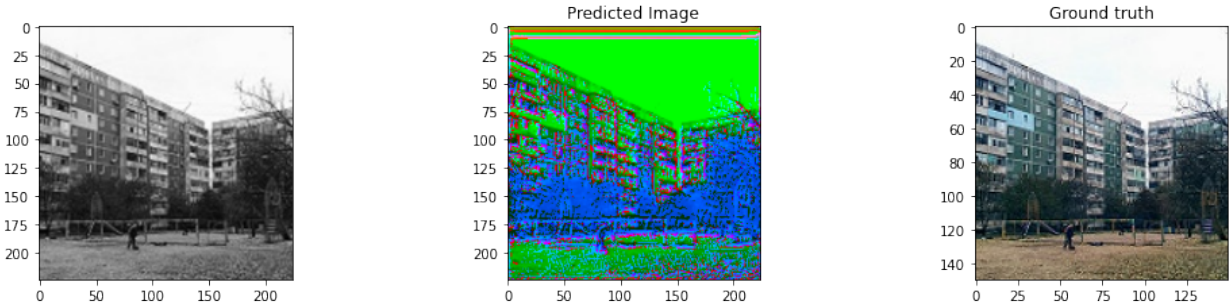

In [ ]:
```
TestImagePath="/content/drive/MyDrive/Deep Learning/21461.jpg"
image_for_test = ExtractTestInput(TestImagePath)
Prediction_3 = Model_Colourization.predict(image_for_test)
Prediction_3 = Prediction_3*128
Prediction_3=Prediction_3.reshape(224,224,2)
```

In [ ]:
```
plt.figure(figsize=(30,20))
plt.subplot(5,5,1)
img = cv2.imread(TestImagePath)
img_1 = cv2.cvtColor(img,cv2.COLOR_BGR2RGB)
img = cv2.cvtColor(img_1, cv2.COLOR_RGB2GRAY)
img = cv2.cvtColor(img, cv2.COLOR_GRAY2RGB)
img = cv2.resize(img, (224, 224))
plt.imshow(img)

plt.subplot(5,5,1+1)
img_ = cv2.cvtColor(img, cv2.COLOR_RGB2Lab)
img_[:,:,1:] = Prediction_3
img_ = cv2.cvtColor(img_, cv2.COLOR_Lab2RGB)
plt.title("Predicted Image")
plt.imshow(img_)

plt.subplot(5,5,1+2)
plt.title("Ground truth")
plt.imshow(img_1)
```

Out[60]: <matplotlib.image.AxesImage at 0x7fe76d3b1210>



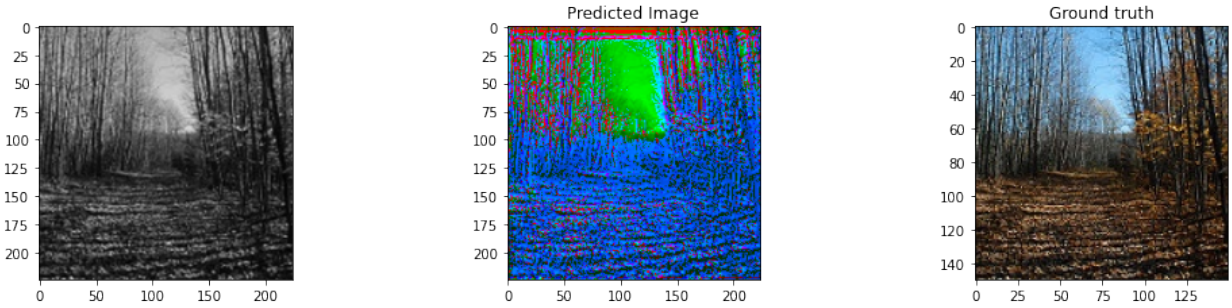

In [ ]:
```
TestImagePath="/content/drive/MyDrive/Deep Learning/21422.jpg"
image_for_test = ExtractTestInput(TestImagePath)
Prediction_4 = Model_Colourization.predict(image_for_test)
Prediction_4 = Prediction_4*128
Prediction_4=Prediction_4.reshape(224,224,2)
```

In [ ]:
```
plt.figure(figsize=(30,20))
plt.subplot(5,5,1)
img = cv2.imread(TestImagePath)
img_1 = cv2.cvtColor(img,cv2.COLOR_BGR2RGB)
img = cv2.cvtColor(img_1, cv2.COLOR_RGB2GRAY)
img = cv2.cvtColor(img, cv2.COLOR_GRAY2RGB)
img = cv2.resize(img, (224, 224))
plt.imshow(img)

plt.subplot(5,5,1+1)
img_ = cv2.cvtColor(img, cv2.COLOR_RGB2Lab)
img_[:,:,1:] = Prediction_4
img_ = cv2.cvtColor(img_, cv2.COLOR_Lab2RGB)
plt.title("Predicted Image")
plt.imshow(img_)

plt.subplot(5,5,1+2)
plt.title("Ground truth")
plt.imshow(img_1)
```

Out[62]: <matplotlib.image.AxesImage at 0x7fe76c3ad9d0>




In [ ]:
```
TestImagePath="/content/drive/MyDrive/Deep Learning/21486.jpg"
image_for_test = ExtractTestInput(TestImagePath)
Prediction_5 = Model_Colourization.predict(image_for_test)
Prediction_5 = Prediction_5*128
Prediction_5=Prediction_5.reshape(224,224,2)
```

In [ ]:
```
plt.figure(figsize=(30,20))
plt.subplot(5,5,1)
img = cv2.imread(TestImagePath)
img_1 = cv2.cvtColor(img,cv2.COLOR_BGR2RGB)
img = cv2.cvtColor(img_1, cv2.COLOR_RGB2GRAY)
img = cv2.cvtColor(img, cv2.COLOR_GRAY2RGB)
img = cv2.resize(img, (224, 224))
plt.imshow(img)

plt.subplot(5,5,1+1)
img_ = cv2.cvtColor(img, cv2.COLOR_RGB2Lab)
img_[:,:,1:] = Prediction_5
img_ = cv2.cvtColor(img_, cv2.COLOR_Lab2RGB)
plt.title("Predicted Image")
plt.imshow(img_)

plt.subplot(5,5,1+2)
plt.title("Ground truth")
plt.imshow(img_1)
```

Out[64]: <matplotlib.image.AxesImage at 0x7fe76b68de90>