Predicting-whale-Keras-CNN

May 22, 2018

1 Image Classification using Keras using Convolution Neural Network

```
* This code helps in classifying whether the given image is a whale or not
* Download the dataset from kaggle
* link : https://www.kaggle.com/c/noaa-right-whale-recognition
```

```
In [ ]: '''
            After Downloading the dataset from the given link
            The following code contains the directory named "whale_images"

            The basic structure is

            whale_images/
                    images_train/
                            whales/
                                w_01.jpg
                                ....
                            notWhales/
                                images1.jpg
                                ...
                    images_test/
                            whales/
                                w_08.jpg
                                ....
                            notWhales/
                                images9.jpg
                                ...
                    prediction/
                            w_01.jpg
                            ...


        '''
```

```
In [ ]: from keras.preprocessing.image import ImageDataGenerator
        from keras.models import Sequential
        from keras.layers import Conv2D, MaxPooling2D
```

```
from keras.layers import Activation, Dropout, Flatten, Dense
from keras import backend as K


# dimensions of our images.
img_width, img_height = 150, 150

train_data_dir = 'whale_images/images_train'
validation_data_dir = 'whale_images/images_test'
nb_train_samples = 512
nb_validation_samples = 80
epochs = 1
batch_size = 16
#input shape-> (150,150,3)
input_shape = (img_width, img_height, 3)

# Build a Simple Sequential model
model = Sequential()

# First Convolutional Layer
# number of features = 32
# filter Size = (3,3)
model.add(Conv2D(32, (3, 3), input_shape=input_shape)) # input_shape is used for the f
model.add(Activation('relu'))                          # activation Function (relu is
model.add(MaxPooling2D(pool_size=(2, 2)))              # Pooling applied on the layer

# Second Convolutional Layer
model.add(Conv2D(32, (3, 3)))   # note here  input_shape is not used
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))

# Third Convolutional layer
model.add(Conv2D(64, (3, 3)))  # number of features = 64
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))

# Flattenning the Layers
model.add(Flatten())
model.add(Dense(64))
model.add(Activation('relu'))
model.add(Dropout(0.5))
model.add(Dense(1))                    # to predict only one value
model.add(Activation('sigmoid'))

# Compiling our model
model.compile(loss='binary_crossentropy',
              optimizer='rmsprop',
              metrics=['accuracy'])
```

```
        # Data Preprocessing
        # (Refer to my Data Augmentation for more detailed approach for data augmentation)
        train_datagen = ImageDataGenerator(
            rescale=1. / 255,
            shear_range=0.2,
            zoom_range=0.2,
            horizontal_flip=True)
        # we only use rescaling for test_images
        test_datagen = ImageDataGenerator(rescale=1. / 255)

        train_generator = train_datagen.flow_from_directory(
            train_data_dir,
            target_size=(img_width, img_height),
            batch_size=batch_size,
            class_mode='binary')

        validation_generator = test_datagen.flow_from_directory(
            validation_data_dir,
            target_size=(img_width, img_height),
            batch_size=batch_size,
            class_mode='binary')

        # fitting our model
        model.fit_generator(
            train_generator,
            steps_per_epoch=nb_train_samples // batch_size,
            epochs=epochs,
            validation_data=validation_generator,
            validation_steps=nb_validation_samples // batch_size)
```

In [ ]:
```
        # importing the Required modules for predicting
        import numpy as np
        from keras.preprocessing import image
```

In [ ]:
```
        # Loading the image for prediction
        test_image =image.load_img('whale_images/prediction/w_0.jpg',target_size=(150,150))
```

In [ ]:
```
        test_image = image.img_to_array(test_image)
        test_image = np.expand_dims(test_image,axis=0)
        pr = model.predict(test_image)
```

In [ ]:
```
        # 0 for not whale
        # 1 for whale
        np.round(pr[0][0])
```

2 Done