13z311-Exercise 3- Movie Poster

October 17, 2018

1 Exercise 3

1.1 MOVIE POSTER

```
In [ ]: import numpy as np
        import pandas as pd
```

```
In [ ]: path = 'SampleMoviePosters'
        import glob
        import scipy.misc
        image_glob = glob.glob(path+"/"+"*.jpg")
        img_dict = {}
        def get_id(filename):
            index_s = filename.rfind("/")+1
            index_f = filename.rfind(".jpg")
            return filename[index_s:index_f]
        _ = [img_dict.update({get_id(fn):scipy.misc.imread(fn)}) for fn in image_glob]
```

```
In [ ]: df = pd.read_csv("MovieGenre.csv",encoding="ISO-8859-1")
        genres = []
        length = len(df)
        for n in range(len(df)):
            g = str(df.loc[n]["Genre"])
            genres.append(g)

        classes = list(set(genres))
        classes.sort()
        num_classes = len(classes)

        def get_classes_from_movie(movie_id):
            y = np.zeros(num_classes)
            g = str(df[df['imdbId']==movie_id]['Genre'].values[0])
            y[classes.index(g)] = 1
            return y
```

```
In [ ]: import random
```

```
def preprocess(img,size=32):
    img = scipy.misc.imresize(img,(size,size))
    img = img.astype(np.float32)
    img = (img / 127.5) - 1.
    return img


def get_dataset(train_size,img_size=32):

        indices = random.sample(range(len(list(img_dict.keys()))),train_size)
        x = []
        y = []
        x_test = []
        y_test = []
        for i in range(len(list(img_dict.keys()))):
            id_key = int(list(img_dict.keys())[i])
            if i in indices:
                x.append(preprocess(img_dict[list(img_dict.keys())[i]],size=img_size))
                y.append(get_classes_from_movie(id_key))
            else:
                x_test.append(preprocess(img_dict[list(img_dict.keys())[i]],size=img_s
                y_test.append(get_classes_from_movie(id_key))
        return x,y,x_test,y_test

SIZE = 128
x,y,x_test,y_test = get_dataset(900,img_size=SIZE)
x = np.asarray(x)
y = np.asarray(y)
x_test = np.asarray(x_test)
y_test = np.asarray(y_test)
```

```
In [ ]: from keras import backend as K
        import keras
        from keras.models import Sequential
        from keras.layers import Dense, Dropout, Flatten
        from keras.layers import Conv2D, MaxPooling2D

        model = Sequential()
        model.add(Conv2D(32,kernel_size=(3,3),activation='relu',input_shape=(SIZE,SIZE,3)))
        model.add(Conv2D(64, (3, 3), activation='relu'))
        model.add(Conv2D(64, (3, 3), activation='relu'))
        model.add(Conv2D(64, (3, 3), activation='relu'))
        model.add(MaxPooling2D(pool_size=(2, 2)))
        model.add(Dropout(0.25))
        model.add(Flatten())
        model.add(Dense(128, activation='relu'))
        model.add(Dropout(0.5))
        model.add(Dense(num_classes, activation='softmax'))
        model.compile(loss='categorical_crossentropy',
                      optimizer=keras.optimizers.Adam(),
                      metrics=['accuracy'])


        model.fit(x, y,
                  batch_size=50,
                  epochs=5,
                  verbose=1,
                  validation_data=(x_test, y_test))
        score = model.evaluate(x_test, y_test, verbose=0)
        print('Test loss:', score[0])
        print('Test accuracy:', score[1])

In [ ]: pred = model.predict(np.asarray([x_test[5]]))
        print(pred)
        print(np.argmax(pred))
        print(np.argmax(y_test[5]))

In [ ]:
```