Image_to_dense_vector

February 5, 2019

```
In [4]: from PIL import Image
        import itertools
        import requests
        from io import BytesIO
        import matplotlib.pyplot as plt
        import numpy as np
        import pandas as pd
        import warnings
        from bs4 import BeautifulSoup
        from nltk.corpus import stopwords
        from nltk.tokenize import word_tokenize
        import nltk
        import math
        import time
        import re
        import os
        import seaborn as sns
        from collections import Counter
        from sklearn.feature_extraction.text import CountVectorizer
        from sklearn.feature_extraction.text import TfidfVectorizer
        from sklearn.metrics.pairwise import cosine_similarity
        from sklearn.metrics import pairwise_distances
        from matplotlib import gridspec
        from scipy.sparse import hstack
        from sklearn.feature_extraction.text import CountVectorizer
        import pickle
        #Deep-Learning Library
        from keras.preprocessing.image import ImageDataGenerator
        from keras.models import Sequential
        from keras.layers import Dropout, Flatten, Dense
        from keras import applications
        import plotly
        import plotly.figure_factory as ff
        from plotly.graph_objs import Scatter, Layout

        plotly.offline.init_notebook_mode(connected=True)
        warnings.filterwarnings("ignore")
```

```
In [15]: start_time = time.clock()
         # dimensions of our images.
         img_width, img_height = 224, 224

         top_model_weights_path = 'bottleneck_fc_model.h5'
         train_data_dir = '16k_images/'
         nb_train_samples = 1000
         epochs = 50
         batch_size = 1
         print("Started")

         def save_bottlebeck_features():

             #Function to compute VGG-16 CNN for image feature extraction.

             asins = []
             datagen = ImageDataGenerator(rescale=1. / 255)
             print("asins and datagen done")
             # build the VGG16 network
             model = applications.VGG16(include_top=False, weights='imagenet')
             print("Model settled")
             generator = datagen.flow_from_directory(
                 train_data_dir,
                 target_size=(img_width, img_height),
                 batch_size=batch_size,
                 class_mode=None,
                 shuffle=False)
             print("Genrator work finish")
             for i in generator.filenames:
                 asins.append(i[2:-5])

             print("For loop ended")
             print(generator)
             print("training start")
             bottleneck_features_train = model.predict_generator(generator, nb_train_samples //
             print("training finised")
             print("reshape the trained vector")
             bottleneck_features_train = bottleneck_features_train.reshape((1000,25088))
             print("Trainning done")
             np.save(open('data_1_cnn_features.npy', 'wb'), bottleneck_features_train)
             np.save(open('data_1_cnn_feature_asins.npy', 'wb'), np.array(asins))
             print("npy file saved")

         save_bottlebeck_features()

         print(time.clock() - start_time, "seconds")
```

```
Started
```

```
asins and datagen done
Model settled
Found 1000 images belonging to 1 classes.
Genrator work finish
For loop ended
<keras.preprocessing.image.DirectoryIterator object at 0x000000471B681CC0>
training start
training finised
reshape the trained vector
Trainning done
npy file saved
2305.7393515661142 seconds
```