

介绍

前几天，某个公众号发文质疑马蜂窝网站，认为它搬运其它网站的旅游点评，对此，马蜂窝网站迅速地做出了回应。相信大多数关注时事的群众已经了解了整个事情的经过，在这里，我们且不论这件事的是是非非，也不关心它是否是通过爬虫等其他技术手段实现的。本文将会展示一种自动生成旅游点评的技术手段。我们用到的模型为LSTM模型。

LSTM模型是深度学习中一种重要的模型，全称为Long Short-Term Memory，中文译为长短期记忆网络，是RNN家族中的重要成员，它模拟了人的大脑，具有一定的记忆功能，适合于处理和预测时间序列中间隔和延迟相对较长的重要事件，在翻译语言、控制机器人、图像分析、文档摘要、语音识别图像识别、手写识别、控制聊天机器人、预测疾病、点击率和股票、合成音乐等方面有较多应用。

在本文中，你将会看到LSTM在自动生成文字（在这里就是旅游点评）方面的应用，如果你感到好奇的话，请继续阅读~

获取数据集

第一步，就是获取数据集，我们利用Python爬虫来实现。我们需要爬取的旅游评论来自于携程网站上的旅游评论，在本文中，我们以杭州西湖景点的旅游评论为例，页面如下：



Language

首页 · 酒店 · 旅游 · 跟团游 · 机票 · 火车 · 汽车票 · 用车 · 门票 · 团购 · 攻略 · 全球购 · 礼品卡 · 商旅 · 邮轮

名店购 | 退税 | 银联特惠 | 万千赏 | 外币兑换

旅游攻略社区 > 目的地 > 中国 > 浙江 > 杭州 > 西湖

西湖 West Lake

西湖点评 (23766 条点评)

写点评

4.7 /5分

景色 4.8

趣味 4.5

性价比 4.6

情侣出游 593条

家庭亲子 1122条

朋友出游 835条

商务旅行 208条

单独旅行 386条



通往霜冻...

5.0

你要是提出个想法，说浙江可代表江南，估计无人回应；若说杭城可代言中国，大概也是应者寥寥。你要反过来说，西湖很中国，想必无人质疑。山外青山，水映三月，西湖的灵秀透在一片雅致之中，心灵的温暖柔软，到了这里，份外缠绵。无奇峰怪石、无人工堆砌，风流自显；无急流险滩、无白浪震天，气自天成。安静从容的西湖是我们渴望的人生。父亲生前，钟爱西湖，钟爱龙井和西泠印社，他的日记本上还留着1979年和母亲同游西湖后写的一首诗：水依宁静显致远，山因淡泊得明志，此处灵秀千古间，天下无湖再名西。因西湖，纪念我的父亲母亲。



2018-05-26 有用 (104)

我们爬取这些评论中的第1至10页，采用concurrent.futures模块实现并发爬取。完整的Python代码如下：

```

import requests
import pandas as pd
from bs4 import BeautifulSoup
from concurrent.futures import ThreadPoolExecutor, wait, ALL_COMPLETED

# 评论列表
comments = []

# 提取评论, 传入参数为网址url
def get_comment(url):

    global comments

    try:
        # 发送HTTP请求
        headers = {'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 \
                    (KHTML, like Gecko) Chrome/67.0.3396.87 Safari/537.36'}
        r = requests.get(url=url, headers=headers)

        # 解析网页, 定位到评论部分
        soup = BeautifulSoup(r.text, 'lxml')
        main_content = soup.find_all('div', class_='comment_single')

        # 提取评论
        for para in main_content:
            comment = para.find('span', class_='heightbox')
            #print(comment.text)
            comments.append(comment.text.replace('&quot;', ''))

    except Exception as err:
        print(err)

def main():
    # 请求网址
    urls = ["http://you.ctrip.com/sight/hangzhou14/49894-dianping-p%d.html"%x for x in range(1,11)]
    urls[0] = urls[0].replace('-p1', '')

    # 利用多线程爬取景点评论
    executor = ThreadPoolExecutor(max_workers=10) # 可以自己调整max_workers,即线程的个数
    # submit()的参数: 第一个为函数, 之后为该函数的传入参数, 允许有多个
    future_tasks = [executor.submit(get_comment, url) for url in urls]
    # 等待所有的线程完成, 才进入后续的执行
    wait(future_tasks, return_when=ALL_COMPLETED)

    # 创建DataFrame并保存到csv文件
    comments_table = pd.DataFrame({'id': range(1, len(comments)+1),
                                   'comments': comments})

    print(comments_table)

    comments_table.to_csv(r"E://LSTM/hangzhou.csv", index=False)

main()

```

运行完该代码, 就会得到hangzhou.csv文件, 在这个文件中, 我们需要把旅游评论中的文字做一些修改, 比如去掉特殊字符, 添加掉电, 去掉换行, 修改个别错别字等。修改完后的csv文件(部分)如下:

A	B
id	comments
1	G20峰会后，西湖周边的马路更干净整洁，看得出来环卫工人的辛苦，马路上人行道上基本看不到脏物，机动车司机礼让行人做的很好，很远就将车停下等待行人过马路，听不到鸣笛声。在杭州没有身份证寸步难行，住宿一人一证且要刷脸对比身份证照片，如果差异较大会影响住店，非住宿者到酒店探望客人要出示证件，坐机场大巴也要凭证件。杭州公交2元起步，出租车11元起步，打的建议上车前拍下车号或者使用打的软件，不然绕路或给你拉到茶厂、丝绸店而影响行程。吃，老字号“知味观”价格很高，口味一般，“外婆家”几个分店到了吃饭时间都要等上至少40分钟，位于湖滨路上的“老头油爆虾”很有特色，值得推荐。住，湖滨路或者南宋御街一带很方便，南宋御街一条街上好多快捷连锁，古街比较有特色，晚上可以逛逛。整体来说杭州物价偏高，但西湖，去几次都不厌。
2	晚上看音乐喷泉，特别壮观，七点半和八点各一场，每场大概十分钟，可以提前到岸边等着，也可以坐船在湖上观看，顺便游湖，就是坐船的话，不能完整观看，为了照顾两边的游客，船会在湖上打转的。白天坐小船游湖，可以看到雷峰塔，和三潭映月，都很清楚，我们现在去是淡季，游船的师傅也会给你多讲一些，湖边有一个商店，说是唯一一家天然珍珠店，百度查了下，说不可信，也不知道啦，不过国营的应该有保障吧，赶上g20珍珠半价，挺好的
3	山水秀美，风光旖旎的西湖。曾经去过的美丽地方，一定要品的西湖龙井，让人印象深刻的是各种曲径通幽的小巷。美景[心]西湖风景区，湘湖，湘湖的老虎洞是勾践卧薪尝胆之地，千岛湖，钱塘江，最好的观景时间八月十八，或者每月的初三或十八。
4	冬天去的，感觉不如春天去来的生意盎然~ 整个西湖适合情侣游玩，对亲子感觉一般，主要是老人慕名想去~ 整体感觉西湖还是挺不错的，水清景远，周边设施齐全，城市文明度比较高！值得回味吧~
5	西湖，在落日余晖映照下，还是很美的。如果不坐船游玩，是不花钱的。坐船游玩，可以二十人包一艘船，可以看到两岸的堤岸，还有远处的雷峰塔。
6	杭州西湖天下闻名，西湖一年四季特色纷呈，西湖有湖光山色交相辉映，既可看湖又可看山，西湖上的桥也是兼具南北特色，长桥不长，断桥不断。
7	西湖古迹遍布，山水秀丽，景色宜人，西湖处处有胜景。最著名的西湖十景，再加上北宋大诗人的一句“浓妆淡抹总相宜”，更是把西湖赞颂成了人间仙境。

得到该csv文件后，我们需要将这些评论（只包含评论）转移到txt文件，以便后续的操作，利用下面的Python代码即可完成：

```
import pandas as pd

# 读取csv文件
df = pd.read_csv('E://LSTM/hangzhou.csv')['comments']
# 将pandas中的评论修改后，写入txt文件
for item in df:
    comments = item.replace('\n', '').replace('"', '') \
        .replace(r' ', '').replace(r'#', '').replace(r'&', '') \
        .replace(r'<', '《').replace(r'>', '》') \
        .replace(r'↑', '').replace(r'[', '').replace(r']', '') \
        .replace(r'❤', '')
    with open('E://LSTM/comments.txt', 'a', encoding='utf-8') as f:
        f.write(comments)
        f.write('\n')
    #print(comments)
```

txt文件部分如下：

G20峰会后，西湖周边的马路更干净整洁，看得出来环卫工人的辛苦，马路上人行道上基本看不到脏物，机动车司机礼让行人做的很好，很远就将车停下等待行人过马路，听不到鸣笛声。在杭州没有身份证寸步难行，住宿一人一证且要刷脸对比身份证照片，如果差异较大会影响住店，非住宿者到酒店探望客人要出示证件，坐机场大巴也要凭证件。杭州公交2元起步，出租车11元起步，打的建议上车前拍下车号或者使用打的软件，不然绕路或给你拉到茶厂、丝绸店而影响行程。吃，老字号“知味观”价格很高，口味一般，“外婆家”几个分店到了吃饭时间都要等上至少40分钟，位于湖滨路上的“老头油爆虾”很有特色，值得推荐。住，湖滨路或者南宋御街一带很方便，南宋御街一条街上好多快捷连锁，古街比较有特色，晚上可以逛逛。整体来说杭州物价偏高，但西湖，去几次都不厌。晚上看音乐喷泉，特别壮观，七点半和八点各一场，每场大概十分钟，可以提前到岸边等着，也可以坐船在湖上观看，顺便游湖，就是坐船的话，不能完整观看，为了照顾两边的游客，船会在湖上打转的。白天坐小船游湖，可以看到雷峰塔，和三潭映月，都很清楚，我们现在去是淡季，游船的师傅也会给你多讲一些，湖边有一个商店，说是唯一一家天然珍珠店，百度查了下，说不可信，也不知道啦，不过国营的应该有保障吧，赶上g20珍珠半价，挺好的

山水秀美，风光旖旎的西湖。曾经去过的美丽地方，一定要品的西湖龙井，让人印象深刻的是各种曲径通幽的小巷。美景心西湖风景区，湘湖，湘湖的老虎洞是勾践卧薪尝胆之地，千岛湖，钱塘江，最好的观景时间八月十八，或者每月的初三或十八。

冬天去的，感觉不如春天去来的生意盎然~整个西湖适合情侣游玩，对亲子感觉一般，主要是老人慕名想去~整体感觉西湖还是挺不错的，水清景远，周边设施齐全，城市文明度比较高！值得回味吧~

西湖，在落日余晖映照下，还是很美的。如果不坐船游玩，是不花钱的。坐船游玩，可以二十人包一艘船，可以看到两岸的堤岸，还有远处的雷峰塔。

该txt文件一共含有41412个文字。我们将会以这些评论为数据集，在此基础上利用Keras建立LSTM模型，训练完模型后，能自动生成其他的旅游点评。

LSTM模型

Keras是一个高级的神经网络API，利用它能够轻松地搭建一些复杂的神经网络模型，是一个不错的深度学习框架。对于刚才得到的旅游点评，为了能够生成其他的旅游点评（人类可读），我们将会用到LSTM模型，之所以使用这个模型，是因为LSTM具有长短时记忆功能，能够很好地处理文本中的文字之间的联系，而不是将文字看成是独立的个体。

在搭建LSTM模型之前，我们需要做一些准备工作。首先我们需要将每个文字对应到一个数字，该模型的输入特征向量为前10个文字对应的数字组成的向量，目标变量为该10个文字的下一个文字对应的数字。该txt文件中一共有1949个文字（包括汉子和标点符号），按照我们的处理模式，共有41402个样本，将这些样本传入到LSTM模型中。我们建立的模型很简单，先是一个LSTM层，利用含有256个LSTM结构，然后是一个Dropout层，能有效防止模型发生过拟合，最后是Softmax层，将它转化为多分类的问题，采用交叉熵作为模型的损失函数。

训练模型的Python代码如下：

```
# 搭建简单的LSTM模型用于生成旅游评论
import numpy
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Dropout
from keras.layers import LSTM
from keras.callbacks import ModelCheckpoint
from keras.utils import np_utils

# 读取txt文件
```

```
filename = "E://LSTM/comments.txt"
with open(filename, 'r', encoding='utf-8') as f:
    raw_text = f.read().lower()

# 创建文字和对应数字的字典
chars = sorted(list(set(raw_text)))
#print(chars)
char_to_int = dict((c, i) for i, c in enumerate(chars))
int_to_char = dict((i, c) for i, c in enumerate(chars))
# 对加载数据做总结
n_chars = len(raw_text)
n_vocab = len(chars)
print("总的文字数: ", n_chars)
print("总的文字类别: ", n_vocab)

# 解析数据集，转化为输入向量和输出向量
seq_length = 10
dataX = []
dataY = []
for i in range(0, n_chars-seq_length, 1):
    seq_in = raw_text[i:i + seq_length]
    seq_out = raw_text[i + seq_length]
    dataX.append([char_to_int[char] for char in seq_in])
    dataY.append(char_to_int[seq_out])
n_patterns = len(dataX)
print("Total Patterns: ", n_patterns)
# 将x重新转化为[samples, time steps, features]的形状
X = numpy.reshape(dataX, (n_patterns, seq_length, 1))
# 正则化
X = X/n_vocab
# 对输出变量做one-hot编码
y = np_utils.to_categorical(dataY)

# 定义LSTM模型
model = Sequential()
model.add(LSTM(256, input_shape=(X.shape[1], X.shape[2])))
model.add(Dropout(0.2))
model.add(Dense(y.shape[1], activation='softmax'))
model.compile(loss='categorical_crossentropy', optimizer='adam')
#print(model.summary())






# 定义checkpoint
filepath="E://LSTM/weights-improvement-{epoch:02d}-{loss:.4f}.hdf5"
checkpoint = ModelCheckpoint(filepath, monitor='loss', verbose=1, save_best_only=True, mode='min')
callbacks_list = [checkpoint]
# 训练模型，每批训练128个样本，总共训练1000次
epochs = 1000
model.fit(X, y, epochs=epochs, batch_size=128, callbacks=callbacks_list)
```

首先让我们看一下模型的结构及参数情况， 使用代码中的print(model.summary())即可，输出如下：

Layer (type)	Output Shape	Param #
=====		
lstm_1 (LSTM)	(None, 256)	264192
<hr/>		
dropout_1 (Dropout)	(None, 256)	0
<hr/>		
dense_1 (Dense)	(None, 1949)	500893
=====		
Total params: 765,085		
Trainable params: 765,085		

虽然是一个很简单的LSTM模型，但也有76万多个参数，深度学习的参数的个数可见一斑~

运行代码，训练该模型，在训练了漫长的4,5个小时后，在613次的时候，损失值为0.3040，我们就以这个文件作为模型训练的结果，而不是1000次，因为1000次太费时了。文件如下：

 weights-improvement-568-0.3149	2018/10/27 20:09	HDF5 File	8,992 KB
 weights-improvement-585-0.3128	2018/10/27 20:15	HDF5 File	8,992 KB
 weights-improvement-591-0.3108	2018/10/27 20:17	HDF5 File	8,992 KB
 weights-improvement-608-0.3079	2018/10/27 20:23	HDF5 File	8,992 KB
 weights-improvement-613-0.3040	2018/10/27 20:25	HDF5 File	8,992 KB

请注意删除没用的文件，因为这些生成的文件都很大。

生成旅游点评

好不容易训练完模型后，下一步，我们将要利用这个模型来生成旅游点评。怎么样，是不是有点期待？生成旅游点评的完整Python如下（我们以输入的句子“杭州西湖天下闻名，西”为例，请注意，每次输入正好10个文字，因为模型训练的输入向量为含10个元素的向量）：

```
# 搭建简单的LSTM模型用于生成旅游评论
import numpy
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Dropout
from keras.layers import LSTM
from keras.callbacks import ModelCheckpoint
from keras.utils import np_utils

# 读取txt文件
filename = "E://LSTM/comments.txt"
with open(filename, 'r', encoding='utf-8') as f:
    raw_text = f.read().lower()

# 创建文字和对应数字的字典
chars = sorted(list(set(raw_text)))
#print(chars)
char_to_int = dict((c, i) for i, c in enumerate(chars))
int_to_char = dict((i, c) for i, c in enumerate(chars))
# 对加载数据做总结
n_chars = len(raw_text)
n_vocab = len(chars)
print("总的文字数: ", n_chars)
print("总的文字类别: ", n_vocab)

# 解析数据集，转化为输入向量和输出向量
seq_length = 10
dataX = []
dataY = []
for i in range(0, n_chars-seq_length, 1):
    seq_in = raw_text[i:i + seq_length]
    seq_out = raw_text[i + seq_length]
    dataX.append([char_to_int[char] for char in seq_in])
    dataY.append(char_to_int[seq_out])
n_patterns = len(dataX)
print("Total Patterns: ", n_patterns)
# 将x重新转化为[samples, time steps, features]的形状
X = numpy.reshape(dataX, (n_patterns, seq_length, 1))
# 正则化
```

```
X = X/n_vocab
# 对输出变量做one-hot编码
y = np_utils.to_categorical(dataY)

# 定义LSTM模型
model = Sequential()
model.add(LSTM(256, input_shape=(X.shape[1], X.shape[2])))
model.add(Dropout(0.2))
model.add(Dense(y.shape[1], activation='softmax'))
model.compile(loss='categorical_crossentropy', optimizer='adam')

# 导入训练完后的文件
filename = "E://LSTM/weights-improvement-613-0.3040.hdf5"
model.load_weights(filename)

# 示例的输入句子
input = '杭州西湖天下闻名, 西'
pattern = [char_to_int[value] for value in input]
print("输入：")
print(''.join([int_to_char[value] for value in pattern]))
print("输出：")
# 生成1000个文字
for i in range(1000):
    x = numpy.reshape(pattern, (1, len(pattern), 1))
    x = x / float(n_vocab)
    prediction = model.predict(x, verbose=0)
    index = numpy.argmax(prediction)
    result = int_to_char[index]
    print(result, end='')
    seq_in = [int_to_char[value] for value in pattern]
    pattern.append(index)
    pattern = pattern[1:len(pattern)]
print("\n生成完毕。")
```

运行该代码，就能看到生成的文字如下：

```
输入：
杭州西湖天下闻名, 西
输出：
湖一年四季特色纷呈，西湖有湖光山色交相辉映，既可见湖又可见山，西湖上的桥也是兼具南北特色，长桥不长，断桥不断。
西湖古迹遍布，山水秀丽，景色宜人，西湖处处有胜景。最著名的西湖十景，西湖十景”是指浙江省杭州市著名旅游景点西湖上的十处特色风景，分别是苏堤春晓、曲苑风荷、平湖秋月、断桥残雪、柳浪闻莺、花港观鱼、雷峰夕照、双峰插云、南屏晚钟、三潭印月等十个景点是杭州游览的热点，不用按照目前的时髦话说什么“非看不可”、“非去不可”，但是，如果去了杭州不看这些景点，不看这些景点的碑刻，就有点可惜了，特别是有康熙爷、乾隆爷的亲笔题词，不去看看，多少会对老祖宗的“大不敬”。
如果天气晴朗下午五点以后去西湖，如果拍视频你就会拍到天蓝蓝水蓝蓝的西湖和夕阳西下渐黄昏的完美，西湖边上有小凳子可以坐下来静静的欣赏西湖美景也可和同行的伙伴聊聊接下来的行程，目的是等待晚上更值得期待的音乐喷泉，得早早坐下来否则就看不到了，看喷泉不允许站着凳子坐满其余人站凳子后面。
西湖游船是游西湖必不可少的。另外早晨可以早点去，人少，而且凉爽。花港观鱼的景色也是不错。雷峰塔就看你需求了，可以俯瞰整个西湖（其实去雷峰塔也是为了看西湖吧，雷峰塔本身貌似没什么看的）吃饭的话，别在景区吃，强烈推荐去弄堂里。好吃又便宜。
8月份出差路过杭州，可以轻松的偷闲半天，顺便预约了离西湖比较近便的酒店入住。8月底的清晨已经有了丝丝的凉意，顺着西湖边开始溜达。早上人还是比较少，本地起来锻炼的大爷大妈比较多，有蘸水在地上练字的，有在小公园练嗓子的，还有坐在长条凳上练二胡的。9点多点，游客开始多了，各种游船排队等候，暑假期间带小孩匆匆而过的人也不少，基本都是到此一游，很少有仔细的欣赏和解说景点的出处。乘凉的地方还是不少，但是蚊子也多的吓人。苏堤的景色不错，特别是两边的高树遮挡了大多数的阳光，即使在这里溜达也不会觉得热，如果时间允许的话，建议溜达过去，步行约30分钟左右，南边的出口还有苏东坡博物馆，比较小，免费开放。
西湖的范围蛮大的，从北山路（北山路上的老建筑，浙江博物馆，断桥残雪，锦带桥，保俶塔，楼外楼，孤山，西冷印社，平湖秋月等。）到南线景区（南山路，西湖天地，御码头，钱王祠，柳浪闻莺，万松书院，长桥，南屏晚钟，净慈寺，雷锋塔太子湾等）以及三堤，外西湖，西里湖等，小南湖，小瀛洲，从白堤_苏堤_曲院风荷_杨公堤_鹄鸪湾_茅家埠。飞来峰景
生成完毕。
```

让我们来看一下这段生成的文字，首先，这段文字的可读性还是很高的，基本上人类能够理解，其次，与原文相对比，这段文字并不是一味地抄袭原文，而是有选择地将原文件中的旅游点评组合起来，虽然每部分的旅游点评与原先的相差不多，但重新组合后，是能够生成新的旅游点评的，并不算真正意义上的抄袭。

用LSTM训练出来的模型生成的文本，是能够作为新的旅游点评的，并不是完全的抄袭，但是对于未在原文中出现的输入向量，可能训练出来的效果就不会太理想，这是需要改进的地方。

总结

本文纯属自娱自乐，如果不当之处，还望大家批评指正~~

当然，对于该项目，还有一些值得改进的地方，比如数据集不够大，这个可以爬取更多的评论；数据预处理过于简单，仅仅做了文字与向量的一一对应以及输入向量的正则化；模型过于简单，读者可以尝试搭建多个LSTM层或其他结构；模型训练过于耗时，可以尝试GPU或改进模型结构或数据预处理方式，等等等等。希望读者在阅读完本文后，能对LSTM模型在文字生成方面有一定的了解，欢迎拥抱LSTM~~

参考文献

Text Generation With LSTM Recurrent Neural Networks in Python with

Keras: <https://machinelearningmastery.com/text-generation-lstm-recurrent-neural-networks-python-keras/>

注意：本人现已开通微信公众号： Python爬虫与算法（微信号为：easy_web_scrape）， 欢迎大家关注哦~~
