一种高效的NLP Word2Vec使用方法

原创 LG 数据科学初学者 2019-01-06

综述

本文主要介绍一个在进行NLP领域深度学习任务时的词嵌入小技巧w2vembeddings。减少不必要的内存占用以及95%的加载时间。也大大加强了词嵌入库的管理能力及复用性。

问题

在进行NLP任务的时候，一般会利用Word2Vec进行嵌入。生成一个基于当前任务的词向量矩阵。正如下面这样。

```
def embedding_matrix_p(embeddings_index, word_index, max_features):
    embed_size = embeddings_index.get('a').shape[0]
    embedding_matrix = np.random.normal(0, 0, (max_features, embed_size))
    for word, i in word_index.items():
        if i >= max_features: continue
        embedding_vector = embeddings_index.get(word, np.zeros(embed_size))
        embedding_matrix[i] = embedding_vector
    return embedding_matrix

EMBEDDING_FILE = '../input/embeddings/glove.840B.300d/glove.840B.300d.txt'
def get_coefs(word,*arr): return word, np.asarray(arr, dtype='float32')
embeddings_index = dict(get_coefs(*o.split(" ")) for o in open(EMBEDDING_FILE))
embedding_matrix, embed_size = embedding_matrix_p(embeddings_index, word_index, max_fea
```

但是这样就会有一个问题，如果每次都从文件(txt/bin)中load的话需要解析，是比较费时间的，而且load进内存中向量又不都用得到。假如某次任务分词下来只有2万个，而词向量文件一般都是几百万的级别，所以真正发挥作用的其实只有百分之几。那么如何解决这部分不必要的时间浪费及内存占用呢?

解决方法

了解了一些部署word2vec的方法之后，发现大概是这么两种：

1、从本地load到内存。

2、起服务，调用REST API。

方案1也就上面讲的模式。每次load需要时间，且会占用运行内存。

方案2也就是为请求词向量单独起个服务，这个架构会比较清晰。但是如果需要使用到全局词向量的话也会比较麻烦，而且，真的有必要起个服务这么麻烦吗？

在闲暇时参考embeddings的实现重新封装了一个包w2vembeddings。

主要是将词向量内容迁移到SQLite中，本质上来讲也就是将word2vec换了一种存储格式。但是由于SQLite是数据库，且无服务器。所以方便高效，查询快速（也不需要用到写入功能，除第一次转换格式的时候）。

主要有以下功能：

词向量库管理

词向量即时调用，随取随用

主要解决内存占用和加载耗时的问题。还可以将多个词向量库放在一起管理。

使用方法：

```
from w2vembeddings.w2vemb import EMB
emb = EMB(name='tencent', dimensions=200)  # 加载词向量库，相当于原来从文件加载的步骤
# 按需提取词向量矩阵
def embedding_matrix_p(emb, word_index, max_features):
    embed_size = len(emb.get_vector('a'))
    embedding_matrix = np.random.normal(0, 0, (max_features, embed_size))
    for word, i in word_index.items():
        if i >= max_features: continue
        embedding_matrix[i] = np.array(emb.get_vector(word))
    return embedding_matrix
```

优势

可以节约掉几乎所有的不必要的内存占用，以及95%的时间。

问题

当然这个方案也有问题，跟REST API一样，想要使用词向量全局信息的时候也不方便。如果有这种需求，建议参考gensim。

这个方法主要面向那些经常使用word2vec做研究的人，以及小型的线上部署场景。

原文发表于

文章已于2019-01-06修改