**冬奥会领域问答机器人实验报告**

吴俊亮 贾志杰 孙严顺 陈天宇

一、实验任务

本实验要求实现一个冬奥会领域的问答机器人：当输入一个有关冬奥会的问题时，机器人会给出相应的答案。按照实验要求，输入和输出的文件均为json格式；除去初始加载模型的时间，单次响应时间应在500ms以内。我们将提供的标注数据1.xlsx和标注数据2.xlsx转换成了json格式的训练集，读者可以参考train\_set.json。（测试集的处理）。

二、实验设计

（一）实验原理

我们的总体设计思路是使用python的gensim库，采用TF-IDF模型+余弦相似度的方法构建一个检索式问答系统。通过判断测试问题与训练集中问题的相似度，选择相似度最高的训练集中问题的答案作为相应测试问题的答案。

1. gensim库

gensim用于从原始的非结构化的文本中，无监督地学习到文本隐层的向量表达。gensim有三大宏观概念：语料→向量→模型。

语料：语料是指一组语句的集合，这个集合是gensim的输入。在本次实验中语料即为训练集的全部问题。收集语料之后，需要做一些预处理，对于英文就是根据空格将原来的句子划分成单词。我们这里的预处理是使用jieba库把中文语句切分成词组。

向量：如果要对语句的隐含结构进行推断，就需要使用适当的数学模型：我们在这里使用的方法是doc2bow，也就是将语句转化为bag-of-words（词袋）。在词袋向量中，每个句子被表示成一个向量，代表字典中每个词出现的次数。例如，给定一个包含[‘2020’,‘冬奥会’,‘举办地’, ‘金牌’]的字典，语句[‘2020’, ‘冬奥会’, ‘2020’]字可以表示成向量[(0,2),(1,1)]，表示‘2020’（编号0）出现了2次、 ‘冬奥会’（编号1）出现了1次。词袋模型的一个重要特点是，它完全忽略了单词在句子中出现的顺序，这也就是“词袋”这个名字的由来。

模型：现在我们已经向量化了语料，接下来可以使用各种模型了。我们可以认为模型转换是在两个向量空间进行转换。以本次实验为例，我们通过TF-IDF模型把词袋向量转换到另一个向量空间再进行比较，关于TF-IDF模型将在下面详细介绍。

1. TF-IDF模型

TF-IDF是一种针对关键词的统计分析方法，用于评估一个词对一个语料库的重要程度。一个词的重要程度跟它在测试语句（一个问题）中出现的次数成正比，跟它在语料库（训练集中的全部问题）出现的次数成反比。这种计算方式能有效避免常用词对关键词的影响，提高了关键词与测试语句之间的相关性。

TF（Term Frequency）指的是某词在测试语句中出现的总次数：

*TF = 某词在测试语句中出现的次数/测试语句的总词量*

IDF（Inverse Document Frequency）指的是逆向文档频率。语料库包含某词语的训练语句越少，IDF值越大，说明该词语具有很强的区分能力：

*IDF = ln(语料库中语句总数/包含该词的语句数 + 1)*

知道了“词频”（TF）和“逆文档频率”（IDF）以后，将这两个值相乘，就得到了一个词的TF-IDF值。某个词对测试语句的重要性越高，它的TF-IDF值就越大。

对于给定的语料库，我们可以计算出每一个词的IDF；给定一个测试语句，我们可以计算出该测试语句中每个词的TF。形象地来说，IDF是“全局变量”，表示一个词的区分能力；TF是“局部变量”，表示一个词在这个问题中的重要程度。举一个例子：问题“1924年冬奥会的举办地是什么？”中“冬奥会”和“？”的频率是一样的，即它们的TF相同；但是“冬奥会”的重要程度显然高于“？”，即它们的IDF不同。

1. 余弦相似度

对于训练集中的每一个问题，我们可以得到一个TF-IDF向量，表示为

*[(词语1的id, 词语1的tfidf值), (词语2的id, 词语2的tfidf值), …]*

用户输入的一个问题也可以得到相应的TF-IDF向量。我们通过比较向量之间的余弦相似度来进行训练语句和测试语句的相似度的比较。余弦相似度是通过计算两个向量的夹角余弦值来评估他们的相似度。余弦值的范围在[-1,1]之间，值越趋近于1，代表两个向量的方向越接近，也就是相似度越高。

（二）重要模块分析

1、生成分词结果

我们通过json.load方法从json格式的训练集中读取全部问答对，存在变量data中。

with open(train\_filepath, encoding='utf-8') as train\_file:

    data = json.load(train\_file)

如果分词结果已经存在，则读取分词结果；否则进行分词并保存分词结果：对于data中的每一个问答对，将其中的问题通过split\_word函数调用jieba库进行分词，加入到列表content中。

content = []

if os.path.exists(splitdata\_filepath):

    with open(splitdata\_filepath, encoding='utf-8') as f:

        content = json.load(f)

else:

    for value in data:

        question = value['question']

        content.append(split\_word(question))

    with open(splitdata\_filepath, 'w', encoding='utf-8') as f:

        f.write(json.dumps(content, ensure\_ascii=False))

2、生成gensim字典

如果gensim字典已存在，则读取gensim字典；否则生成gensim字典。上一步中训练集的所有问题已经转换成了分词后的列表content，现在对这个列表使用gensim中的corpora.Dictionary方法生成字典并且保存在文件dictionary中。字典存储的是词语和id之间的映射。此处使用num\_features变量保存字典长度，并将训练集中的每一个问题转换为一个词袋。所有的问题对应的词袋形成了语料库（corpus）。

if os.path.exists(dictionary\_filepath):

    dictionary = gensim.corpora.Dictionary.load(dictionary\_filepath)

else:

    dictionary = gensim.corpora.Dictionary(content)

    dictionary.save(dictionary\_filepath)

num\_features = len(dictionary)

corpus = [dictionary.doc2bow(line) for line in content]

3、生成TF-IDF模型

如果TF-IDF模型已存在，则读取TF-IDF模型；否则生成TF-IDF模型。根据语料库，我们可以计算出TF-IDF模型的总词语数、总语句数以及每一个词语对应的IDF值。

if os.path.exists(model\_filepath):

    tfidf = gensim.models.TfidfModel.load(model\_filepath)

else:

    tfidf = gensim.models.TfidfModel(corpus)

    tfidf.save(model\_filepath)

1. 生成TF-IDF相似度比较序列

如果TF-IDF相似度比较序列已存在，则读取TF-IDF相似度比较序列；否则生成TF-IDF相似度比较序列。此步通过gensim中的similarities.Similarity方法生成相似度比较序列，即对训练集中的每一个问题计算出一个TF-IDF向量。

if os.path.exists(index\_filepath):

    index = gensim.similarities.Similarity.load(index\_filepath)

else:

    index = gensim.similarities.Similarity(index\_filepath, tfidf[corpus], num\_features)

    index.save(index\_filepath)

1. 检索过程

用户提出的问题存储在变量sentences中。首先对用户输入问题进行分词、转词袋表示。相似度比较的过程将词袋vec传入TF-IDF模型中，得到一个TF-IDF向量，该向量与训练集中每个问题的对应向量计算余弦相似度，形成相似度比较序列。之后将相似度比较序列排序，排序后列表的第一位即为训练集中与输入问题相似度最高的问题，系统输出该问题对应答案。

sentences = split\_word(sentences)    *# 分词*

vec = dictionary.doc2bow(sentences)  *# 转词袋表示*

sims = index[tfidf[vec]]             *# 相似度比较*

sorted\_sims = sorted(enumerate(sims), key=lambda x: x[1], reverse=True)

i = sorted\_sims[0][0]                *# 最相似的问题的序号*

print(data[i]['answer'])

1. 实验结果
2. 组内分工

|  |  |
| --- | --- |
| 成员 | 分工 |
| 吴俊亮 | 完成代码，完善实验报告 |
| 贾志杰 |  |
| 孙严顺 |  |
| 陈天宇 | 进行环境配置，搜索相关资料，初步完成实验报告 |

1. 参考文献

1. 【gensim中文教程】开始使用gensim, <https://blog.csdn.net/duinodu/article/details/76618638>, 2017. [Online; accessed 2021-01-04].

2. TF-IDF算法原理及其使用详解, <https://zhuanlan.zhihu.com/p/94446764>, 2019. [Online; accessed 2021-01-04].

3. 检索式问答机器人, <https://github.com/vba34520/Retrieval-Bot>, 2020. [Online; accessed 2021-01-04].