**《人工智能基础A》实验报告七**

**自然语言处理之语义相似度分析**

## 实验目的

通过本次实验，了解并实现文本语义相似度计算的常见方法，具体目标要求如下：

1. 理解文本语义相似度计算的基本原理。
2. 实现常见的语义相似度计算方法（如Word2Vec）等。
3. 对比不同算法在文本相似度计算中的表现。

## 实验内容及要求

**实验步骤（仔细阅读，按照步骤完成实验）**

### 完成下列题目（形式任意）

根据下面的表格，计算苹果和大米的余弦相似度和广义Jaccard相似度。



### 词袋模型实践（注意截图）

1. **构建两句话的词袋模型并计算相似度**
   1. **准备工作**

首先导入所需的模块，然后准备需要计算相似度的句子。在这个例子中，我们将使用 CountVectorizer 来构建词袋模型，并通过 cosine\_similarity 来计算句子之间的相似度。大家可以自行输入两个相似但不相同的两个中文句子。

from sklearn.feature\_extraction.text import CountVectorizer

from sklearn.metrics.pairwise import cosine\_similarity

# 输入的两个句子

sentence1 = "……"

sentence2 = "……"

* 1. **分词**

下面对中文句子进行分词。

import jieba

# 使用jieba进行分词

def jieba\_cut(sentence):

return " ".join(jieba.cut(sentence))

sentence1\_cut = jieba\_cut(sentence1)

sentence2\_cut = jieba\_cut(sentence2)

* 1. **构建词袋模型**

首先需要把句子转化为词袋模型，每个句子的词袋模型是一个向量，向量的每个维度对应词汇表中的一个词，值是该词在句子中的出现频率。

# 使用CountVectorizer构建词袋模型

vectorizer = CountVectorizer()

# 将两个句子转换为词袋向量

X = vectorizer.fit\_transform([sentence1\_cut, sentence2\_cut])

* 1. **计算余弦相似度**

通过计算两个向量之间的余弦相似度，我们可以衡量两个句子在词袋空间中的相似度。

# 计算余弦相似度

cosine\_sim = cosine\_similarity(X[0:1], X[1:2])

# 输出相似度

print(f"两个句子的余弦相似度为: {cosine\_sim[0][0]:.4f}")

1. **参考上述代码，尝试构建两句话以上的词袋模型并计算相似度**

### 使用Word2Vec计算文本相似度（注意截图）

1. **准备工作**

首先导入所需库。我们利用jieba 库来对中文句子进行分词，它能够将句子分解成一个个词语；gensim.models.Word2Vec 用来训练 Word2Vec 模型；cosine\_similarity 用于计算句子之间的余弦相似度。

import jieba

from gensim.models import Word2Vec

from sklearn.metrics.pairwise import cosine\_similarity

接下来准备句子数据，可以用下面提供的，也可以自行定义。

sentences = [

"我喜欢在清晨喝一杯热茶，享受安静的时光。",

"今天是我生日，我准备和家人一起庆祝。",

"科技改变了我们的生活，尤其是人工智能的发展。",

"早晨的阳光洒在大地上，给一切带来了温暖。",

"我喜欢在每天下午来一杯咖啡，它能让我更加集中精神。",

"每当下雨时，我喜欢在窗前静静地看着雨滴落下。",

"昨晚我和朋友一起去看了一场电影，感觉很放松。",

"电脑和手机已经成为现代社会不可或缺的一部分。",

"他每天都在健身房锻炼，已经养成了好习惯。",

"今天是我的生日，但我与朋友约好了一起在海底捞庆祝。",

"我喜欢读科幻小说，因为它们带我进入了另一个未知的世界。"

]

1. **分词**

将输入的句子列表进行中文分词，并输出每个句子的分词结果。

# 使用jieba对句子进行分词

def cut\_sentences(sentences):

return [list(jieba.cut(sentence)) for sentence in sentences]

tokenized\_sentences = cut\_sentences(sentences)

# 打印分词后的结果

for sentence in tokenized\_sentences:

print(" ".join(sentence))

1. **模型训练**

Word2Vec 是一个将单词转换为向量的模型，训练时会从文本中学习词与词之间的语义关系。

# 训练Word2Vec模型

model = Word2Vec(tokenized\_sentences, size=100, window=5, min\_count=1, sg=0)

1. **计算句子向量**

计算每个句子的词向量平均值，作为句子的向量表示。

import numpy as np

# 计算句子向量

def sentence\_vector(tokens):

vectors = [model.wv[word] for word in tokens if word in model.wv]

return np.mean(vectors, axis=0) if vectors else np.zeros(100)

# 获取每个句子的词向量

sentence\_vectors = [sentence\_vector(sentence) for sentence in tokenized\_sentences]

1. **计算相似度**

利用cosine\_similarity 来计算句子之间的余弦相似度。

# 计算余弦相似度

from sklearn.metrics.pairwise import cosine\_similarity

cosine\_sim = cosine\_similarity(sentence\_vectors)

1. **可视化结果**

可以用热图来可视化结果，热图中的每个格子表示两个句子之间的余弦相似度。

import seaborn as sns

import matplotlib.pyplot as plt

plt.figure(figsize=(10, 8))

sns.heatmap(cosine\_sim, annot=True, cmap="coolwarm", xticklabels=[f"句子{i+1}" for i in range(len(sentences))], yticklabels=[f"句子{i+1}" for i in range(len(sentences))], fmt=".2f")

plt.title("句子余弦相似度热图")

plt.show()

**作业上交内容与事项：**

1. 按照要求完成实验并将关键步骤实验结果进行截图记录，注意文档工整。
2. 请在截止日期内提交，若逾期提交，成绩会被适当打折。

**本次作业上交内容：**

* 实验报告文档(.docx)

## 三、实验感受与记录

**3-1、实验感受（总结实验过程中的收获或疑问）**

**3-2、实验记录（截取一些操作界面放置于如下）**