浙江大学
ZHEJIANG UNIVERSITY

人工智能基础实验报告

作业名称 实验报告七

姓　　名 叶畅飞

学　　号 3240103132

电子邮箱 yeelysia@qq.com

联系电话 19557031080

指导老师 钟先平

2025 年 6 月 3 日

目　　录

1　计算余弦相似度和广义 Jaccard 相似度 1
1.1　方法一 1
1.2　方法二 2
2　词袋模型实践 2
2.1　构建两句话的词袋模型并计算相似度 2
2.2　构件两句话以上的模型 3
3　使用 word2Vec 计算文本相似度 5
3.1　代码以及可视化截图 5
4　实验感受 8

1 计算余弦相似度和广义 Jaccard 相似度

1.1 方法一

令苹果的词向量 $A = [0.96, 0.77, 0.85, 0.15]$,

大米的词向量 $B = [0.18, 0.22, 0.05, 0.93]$。

$$\begin{aligned} \|A\| &= \sqrt{0.96^2 + 0.77^2 + 0.85^2 + 0.15^2} \\ &= \sqrt{0.9216 + 0.5929 + 0.7225 + 0.0225} \\ &= \sqrt{2.2595} \approx 1.5032 \end{aligned} \tag{1-1}$$

$$\begin{aligned} \|B\| &= \sqrt{0.18^2 + 0.22^2 + 0.05^2 + 0.93^2} \\ &= \sqrt{0.0324 + 0.0484 + 0.0025 + 0.8649} \\ &= \sqrt{0.9482} \approx 0.9738 \end{aligned} \tag{1-2}$$

$$\begin{aligned} \text{余弦相似度} &= \frac{A \cdot B}{\|A\| \cdot \|B\|} \\ &= \frac{0.96 \times 0.18 + 0.77 \times 0.22 + 0.85 \times 0.05 + 0.15 \times 0.93}{1.5032 \times 0.9738} \\ &= \frac{0.1728 + 0.1694 + 0.0425 + 0.1395}{1.4638} \\ &= \frac{0.5242}{1.4638} \approx 0.3581 \end{aligned} \tag{1-3}$$

$$\begin{aligned} \text{广义 Jaccard 相似度} &= \frac{A \cdot B}{\|A\|^2 + \|B\|^2 - A \cdot B} \\ &= \frac{0.5242}{1.5032^2 + 0.9744^2 - 0.5242} \\ &= \frac{0.5242}{2.2595 + 0.9482 - 0.5242} \\ &= \frac{0.5242}{2.6835} \approx 0.1953 \end{aligned} \tag{1-4}$$

1.2 方法二

```
import numpy as np

A = np.array([0.96, 0.77, 0.85, 0.15])
B = np.array([0.18, 0.22, 0.05, 0.93])

cosine_similarity = np.dot(A, B) / (np.linalg.norm(A) * np.linalg.norm(B))
jaccard_similarity = np.dot(A, B) / (np.linalg.norm(A)**2 + np.linalg.norm(B)**2 -
np.dot(A, B))

print(f"余弦相似度: {cosine_similarity}")
print(f"广义Jaccard相似度: {jaccard_similarity}")
```

```
› python "/home/yee/elysiam/python/try.py"
余弦相似度： 0.3581301365315878
广义Jaccard相似度： 0.19534190422955097
```

2 词袋模型实践

2.1 构建两句话的词袋模型并计算相似度

2.1.1 准备工作

```
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics.pairwise import cosine_similarity

# 输入的两个句子
sentence1 = "这颗星球上的一些沙漠曾是海洋"
sentence2 = "被黑夜笼罩的地方，也会迎来光明"
```

2.1.2 分词

```
# 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)

print("句子1:", sentence1_cut)
print("句子2:", sentence2_cut)
```

```
句子1: 这颗 星球 上 的 一些 沙漠 曾 是 海洋
句子2: 被 黑夜 笼罩 的 地方 ， 也 会 迎来 光明
```

2.1.3 构建词袋模型

```
# 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])

print(f"词袋向量{X}")
```

```
词袋向量<Compressed Sparse Row sparse matrix of dtype 'int64'
        with 10 stored elements and shape (2, 10)>
  Coords        Values
  (0, 8)        1
  (0, 3)        1
  (0, 0)        1
  (0, 4)        1
  (0, 5)        1
  (1, 9)        1
  (1, 6)        1
  (1, 2)        1
  (1, 7)        1
  (1, 1)        1
```

2.1.4 计算余弦相似度

```
# 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}")
```

```
两个句子的余弦相似度为: 0.0000
```

2.2 构件两句话以上的模型

```
import jieba
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics.pairwise import cosine_similarity
```

```
# 输入的两个句子
sentence1 = "置身十字路口正中央 混入熙来熙往的人群 唯独我漫无目的 有如漂流者一般 那些流行的首首歌曲总是唱不出我的心思 请不要在幻象中 对着我露出微笑 今天又语不成声 将感情吞如心底 书页写满了草稿 啊啊 就这么度过了数千个夜晚 为了我自己 只是如此 就仅仅是如此而已 为从心头满溢而出的话语找寻出口 若能传达至你的心中 稍稍填补虚无的空隙 即使这样的我 也将大喊我在这里 唱着迷途之星的歌 不知为何每每问起 谈论的总都是未来 但讲到眼前的我 却总是敷衍了事 无视于 我无法融入亮丽世界的心绪 请不要推荐我走向散发光辉的明天 夜空中闪烁光芒 无依无靠的星尘 在踌躇之中失散 啊啊 那颗彷徨的星儿就是我啊 成为我自己 只有如此 我能做的只有如此 模仿他人这种事 我也没办法做得好 为何要将这种痛苦的日子 用一句无趣带过 就算步伐多踉跄 我也是在挣扎着啊 唱着迷途之星的歌 为了我自己 只是如此 就仅仅是如此而已 在流下泪水之后 才终于诞生的话语 彼此似乎有些许相似 怀抱着隐隐作痛的胸口 面对在颤抖的你 大喊着我与你同在 唱着迷途之星的歌"
sentence2 = "内心满是憔悴，眼神游动不止。我在这世界孤身一人，这不断凋零的春季中，每年都只感受到冰冷。在一片黑暗中，单向往前走着，我只能不断胡乱写着。明知期待也是一场空，却依然不断寻求救赎（令人揪心却又叫人心爱）。此刻感觉好像能了解（幸福却又让人心乱神迷）。照耀着无法哭泣的我，光芒温柔地携我同行，穿过层层云朵，变得闪闪发光。内心的思绪满溢而出，脸颊回过神来，也正闪闪闪光。热泪沾湿了我的脸庞，为什么你的手是如此地温暖？拜托你，请你从此再也不要放手。人与人的缘分，总是断断续续。人们在喜悦及悲伤中，细数一个又一个的爱。为了确认内心的跳动（令人喜悦却又叫人寂寞）。此刻感觉好像能了解（重要却又让人感到害怕）。照耀着无法哭泣的我，是光芒温柔地将我拥入怀中。在这阳光普照的世界，骄傲绽放的重要之人，知晓何谓温暖的春天。因为你我，而留下泪水。啊啊，多么地耀眼，啊啊，多么地美丽。穿过层层云朵，变得闪闪发光。内心的思绪满溢而出，脸颊回过神来，也正闪闪闪光。热泪沾湿了我的脸庞，为什么你的手是如此地温暖？呐，拜托你，请你从此再也不要放手。永远，永远，再也不要放手。"

# 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)

# 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])
# 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}")
```

```
两个句子的余弦相似度为: 0.1297
```

3 使用 word2Vec 计算文本相似度

3.1 代码以及可视化截图

3.1.1 准备工作

```
import jieba
from gensim.models import Word2Vec
from sklearn.metrics.pairwise import cosine_similarity
```

3.1.2 分词

```
sentences = [
    "内心满是憔悴 眼神游动不止",
    "我在这世界孤身一人",
    "这不断凋零的春季中",
    "每年都只感受到冰冷",
    "在一片黑暗中 单向往前走着",
    "我只能不断胡乱写着",
    "明知期待也是一场空",
    "却依然不断寻求救赎",
    "（令人揪心却又叫人心爱）",
    "此刻感觉好像能了解",
    "（幸福却又让人心乱神迷）",
    "照耀着无法哭泣的我",
    "光芒温柔地携我同行",
    "穿过层层云朵 变得闪闪发光",
    "内心的思绪满溢而出",
    "脸颊回过神来 也正闪闪闪光",
    "热泪沾湿了我的脸庞",
    "为什么你的手是如此地温暖",
    "拜托你",
    "请你从此再也不要放手",
    "人与人的缘分 总是断断续续",
    "人们在喜悦及悲伤中",
    "细数一个又一个的爱",
    "为了确认内心的跳动",
    "（令人喜悦却又叫人寂寞）",
    "此刻感觉好像能了解",
    "（重要却又让人感到害怕）",
    "照耀着无法哭泣的我",
    "是光芒温柔地将我拥入怀中",
    "在这阳光普照的世界 骄傲绽放的重要之人",
    "知晓何谓温暖的春天",
    "因为你我 而留下泪水",
    "啊啊 多么地耀眼",
    "啊啊 多么地美丽",
    "穿过层层云朵 变得闪闪发光",
    "内心的思绪满溢而出",
```

```
    "脸颊回过神来 也正闪闪发光",
    "热泪沾湿了我的脸庞",
    "为什么你的手是如此地温暖",
    "呐，拜托你",
    "请你从此再也不要放手",
    "永远 永远 再也不要放手",
]

# 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))
```

```
内心 满是 憔悴  眼神 游动 不止
我 在 这 世界 孤身一人
这 不断 凋零 的 春季 中
每年 都 只 感受 到 冰冷
在 一片 黑暗 中  单向 往前走 着
我 只能 不断 胡乱 写 着
明知 期待 也 是 一场空
却 依然 不断 寻求 救赎
（ 令人 揪心 却 又 叫 人 心爱 ）
此刻 感觉 好像 能 了解
（ 幸福 却 又 让 人心 乱神 迷 ）
照耀 着 无法 哭泣 的 我
光芒 温柔 地 携 我 同行
穿过 层层 云朵  变得 闪闪发光
内心 的 思绪 满溢 而出
脸颊 回过 神来  也 正 闪闪 闪光
热泪 沾湿 了 我 的 脸庞
为什么 你 的 手 是 如此 地 温暖
拜托 你
请 你 从此 再也 不要 放手
人 与 人 的 缘分  总是 断断续续
人们 在 喜悦 及 悲伤 中
细数 一个 又 一个 的 爱
为了 确认 内心 的 跳动
（ 令人 喜悦 却 又 叫 人 寂寞 ）
此刻 感觉 好像 能 了解
（ 重要 却 又 让 人 感到 害怕 ）
照耀 着 无法 哭泣 的 我
是 光芒 温柔 地 将 我 拥入 怀中
在 这 阳光普照 的 世界  骄傲 绽放 的 重要 之 人
知晓 何谓 温暖 的 春天
因为 你 我  而 留下 泪水
啊 啊  多么 地 耀眼
啊 啊  多么 地 美丽
穿过 层层 云朵  变得 闪闪发光
内心 的 思绪 满溢 而出
脸颊 回过 神来  也 正 闪闪发光
热泪 沾湿 了 我 的 脸庞
为什么 你 的 手 是 如此 地 温暖
呐 ， 拜托 你
请 你 从此 再也 不要 放手
永远  永远  再也 不要 放手
```

3.1.3 模型训练

```
# 训练Word2Vec模型
model = Word2Vec(tokenized_sentences, vector_size=100, window=5, min_count=1, sg=0)
```

3.1.4 计算句子向量

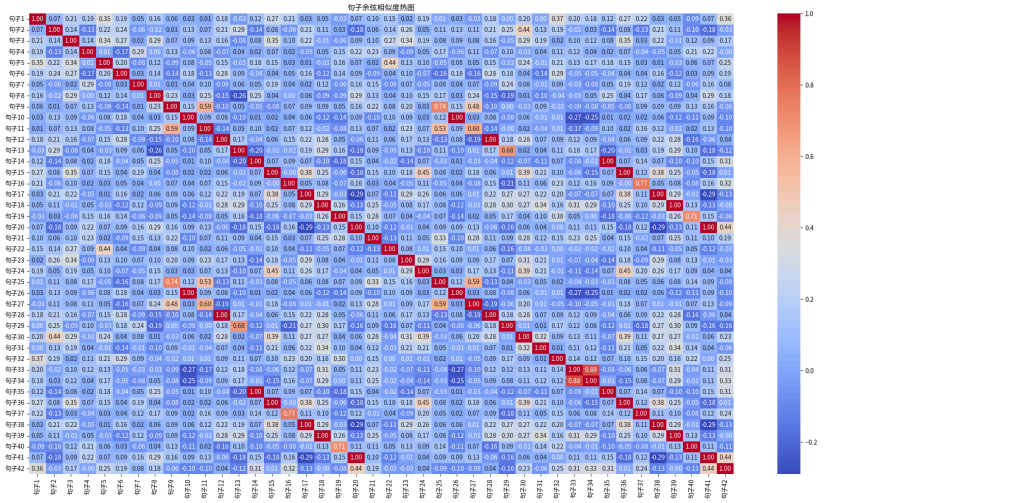
```
# 计算句子向量
def sentence_vector(tokens):
    vectors = [model.wv[word] for word in tokens if word in model.wv]
    return np.mean(vectors, axis=0) if vectors else np.zeros(100)

# 获取每个句子的词向量
sentence_vectors = [sentence_vector(sentence) for sentence in tokenized_sentences]
```

3.1.5 计算相似度

```
# 计算余弦相似度
cosine_sim = cosine_similarity(sentence_vectors)
```

3.1.6 可视化结果

```
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()
```



4 实验感受

学会了利用人工智能的工具来进行文本相似度的计算，了解了余弦相似度和广义 Jaccard 相似度的计算方法，并通过实践掌握了词袋模型和 Word2Vec 模型的使用。通过这些实践，我对自然语言处理有了更深入的理解，也感受到了人工智能在文本处理方面的强大能力。