**广东外语外贸大学信息科学与技术学院**

**《自然语言处理》课程**

**大作业**

**题目：基于三国演义的简单的搜索引擎系统**

**多人完成：**

**成员1： 卢彦竹 软工2301 20231003097**

**成员2： 曾佳佳 软工2303 20231003102**

**作业提交时间：2026.1.2**

**中文摘要**

在信息爆炸的数字时代，高效精准的信息检索工具尤为重要。本课题基于古典名著《三国演义》，设计并实现了一个专门的智能搜索引擎系统。系统对约60万字的原文进行了深度预处理（清洗、断句、分词），为后续分析奠定基础。

在文本表示层面，系统采用TF-IDF与Word2Vec双重方法。TF-IDF用于捕捉关键特征，Word2Vec则通过创新的双加权机制（IDF权重+词频权重）生成高质量语义向量。为提升检索精度，系统引入了带长度惩罚和词重叠奖励的优化相似度计算机制，有效解决了短查询匹配失真的问题。

系统通过Flask框架构建了友好的Web界面，支持TF-IDF（关键词匹配）、Word2Vec（语义相似度）及两者结合的混合搜索三种查询方式。实验表明，系统能有效处理不同复杂度的查询，尤其长句效果显著，优化后的相似度计算使结果更合理。本项目为古典文学文本的智能检索提供了有益的技术实践。

**英文摘要**

In the digital age of information explosion, efficient and precise information retrieval tools are particularly important. This project designs and implements an intelligent search engine system specifically for the classical masterpiece The Romance of the Three Kingdoms. The system performs deep preprocessing (cleaning, sentence segmentation, tokenization) on the approximately 600,000-character source text, laying the foundation for subsequent analysis.

At the text representation level, the system employs dual methods: TF-IDF and Word2Vec. TF-IDF captures key features, while Word2Vec generates high-quality semantic vectors through an innovative dual-weighting mechanism (IDF weights + word frequency weights). To improve retrieval accuracy, the system introduces an optimized similarity calculation mechanism with length penalties and word overlap rewards, effectively addressing the matching distortion problem with short queries.

The system features a user-friendly web interface built with the Flask framework, supporting three query modes: TF-IDF (keyword matching), Word2Vec (semantic similarity), and a hybrid search combining both. Experiments show that the system can effectively handle queries of varying complexity, with particularly outstanding performance on long sentences. The optimized similarity calculation yields more reasonable rankings. This project provides a valuable technical practice for the intelligent retrieval of classical literary texts.

**一、任务描述**

在信息爆炸的时代，搜索引擎已成为人们快速获取信息的关键工具。它能够帮助用户在海量数据中精准定位所需内容，极大地提高了信息检索的效率。想象一下，当我们想要深入了解《三国演义》中某一经典情节的详细描述、特定人物在不同场景下的言行表现，或是探究某场战役背后的策略与影响时，传统的手动翻阅书籍方式不仅耗时费力，还可能因遗漏关键信息而无法全面满足需求。此时，一个专门针对《三国演义》全文构建的搜索引擎系统就显得尤为必要，它能够为用户提供便捷、高效、精准的信息检索服务。

本选题旨在设计并实现一个基于《三国演义》全文的简单搜索引擎系统，并通过 WEB 端展示系统效果。该系统将围绕《三国演义》的文本内容，从数据采集到最终的系统展示，完成一系列关键任务，具体如下：

（一）数据采集

从网上下载《三国演义》全文约60万字文本作为数据。

（二）已有文档数据预处理

采集到的数据往往存在格式不规范、包含大量无关信息等问题，因此需要进行预处理。将文本进行清洗、断句、分词，然后收集统计信息，保存预处理结果供后续使用。比如原文中有一个句子是“ 及刘焉发榜招军时，玄德年已二十八岁矣。”，预处理后变成“及 刘焉 发榜 招军 时 ， 玄德年 二十八岁”，去除了开头的空格和“已”、“矣”两个停用词，然后进行了分词。

（三）文本表示

对于已有文档数据及用户查询语句或短语，系统采用TF-IDF和Word2vec两种方式表示。

（四）语义相似度计算

在完成文本表示后，系统将运用经典的余弦相似度算法，并在此基础上写了一个带惩罚机制的优化相似度计算器，计算用户查询句子与《三国演义》文档中各个句子之间的相似度。通过比较向量之间的夹角余弦值，衡量它们在语义上的相似程度。最终，系统将筛选出与查询句子最相似的10 个句子，并按照相似度从高到低的顺序输出，为用户呈现最符合需求的检索结果。

（五）系统展示

为了使用户能够方便地使用该搜索引擎系统，设计一个简洁、直观、易用的 WEB 页面展示系统。在页面上设置了查询输入框，用户可以在此输入想要查询的句子或短语，如“周瑜纵火”“分久必合，合久必分”等。查询输入框下方有三种不同的查询方式供用户选择。用户选择了查询方式、点击查询按钮后，系统将在后台进行数据处理和相似度计算，并将检索结果以清晰的方式展示在页面上，包括相似句子的原文内容和索引。

通过完成以上任务，本选题将构建一个功能完整、操作简便的基于《三国演义》全文的搜索引擎系统，为用户深入了解这部经典文学作品提供有力的工具支持。

**二、数据集描述**

（一）名称：sanguoyanyi.txt

（二）大小：约60万字

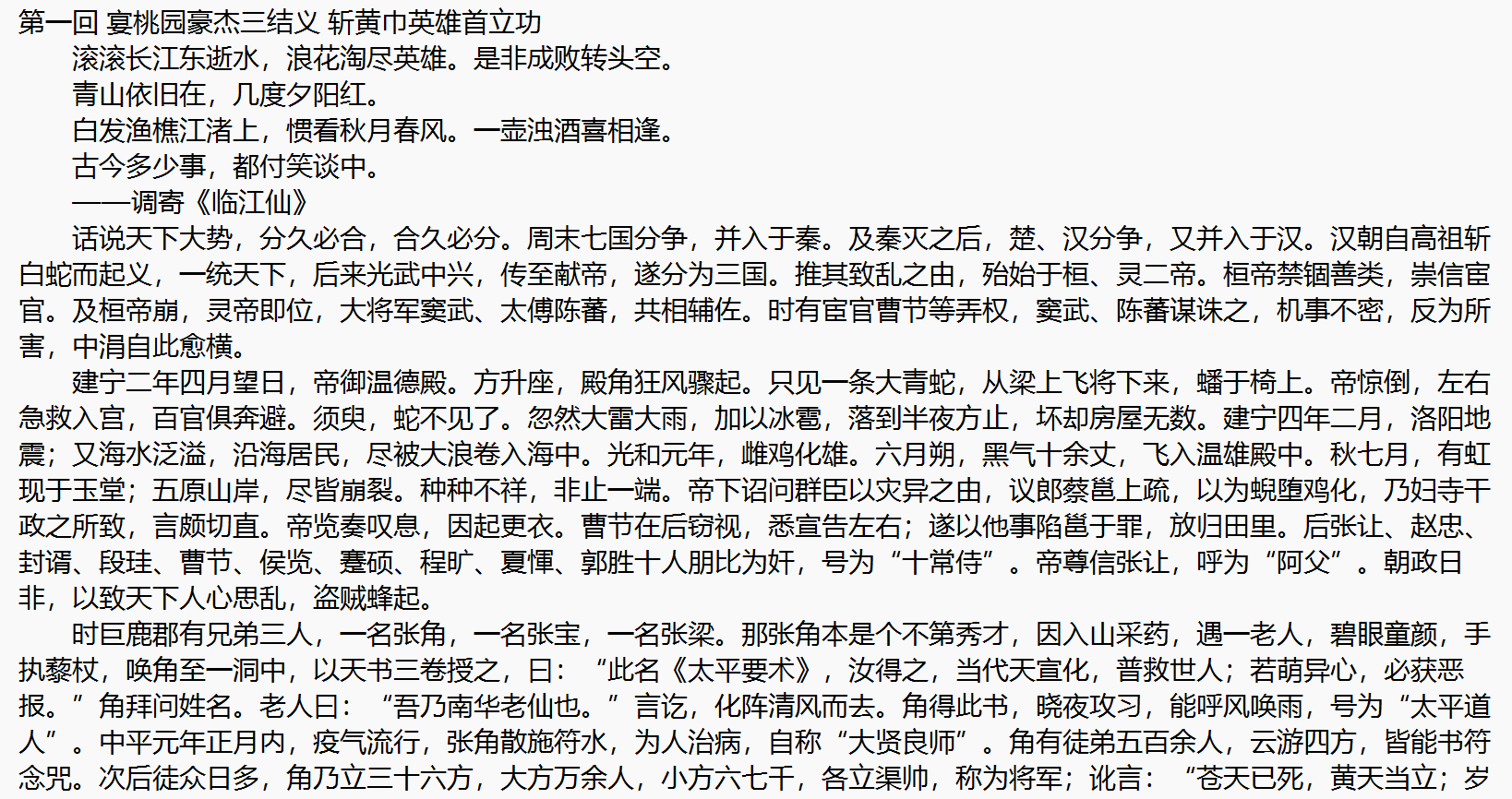
（三）下载链接：<https://github.com/flowers2023/lm-LSTM/blob/master/data/sanguoyanyi.txt>

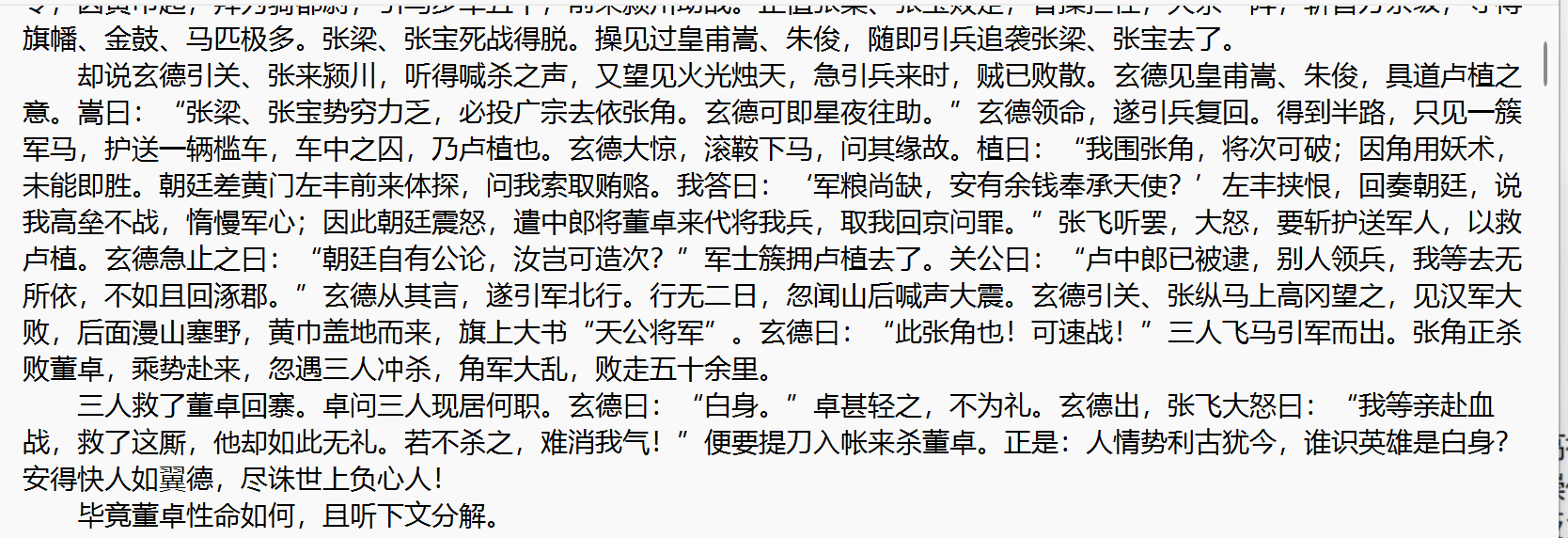
（四）内容：《三国演义》原文全文，共一百二十回。

（五）结构：每回以“第X回”开头，后接对仗式标题，如：“第六十回　张永年反难杨修　庞士元议取西蜀”，结尾常以“正是：……”“不知……且听下文分解”结束。

（六）语言特点：文言文，有大量的人名、地名等专有名词，如“诸葛亮”“刘备”“赤壁”等，还有大量的诗词、歌谣、赞词，比如“青山依旧在，几度夕阳红”。

（七）样例：第一回部分内容

****

****

**三、数据采集及预处理**

（一）数据集采集：在Github上下载。

（二）预处理：

1.文本清洗：

①去除多余的空格，将多个空格合并成一个空格，比如每段开头的两个空格只保留一个。

②去除异常标点符号。

③移除控制字符。

④移出网址和邮箱地址。

相关代码：



1. 断句：

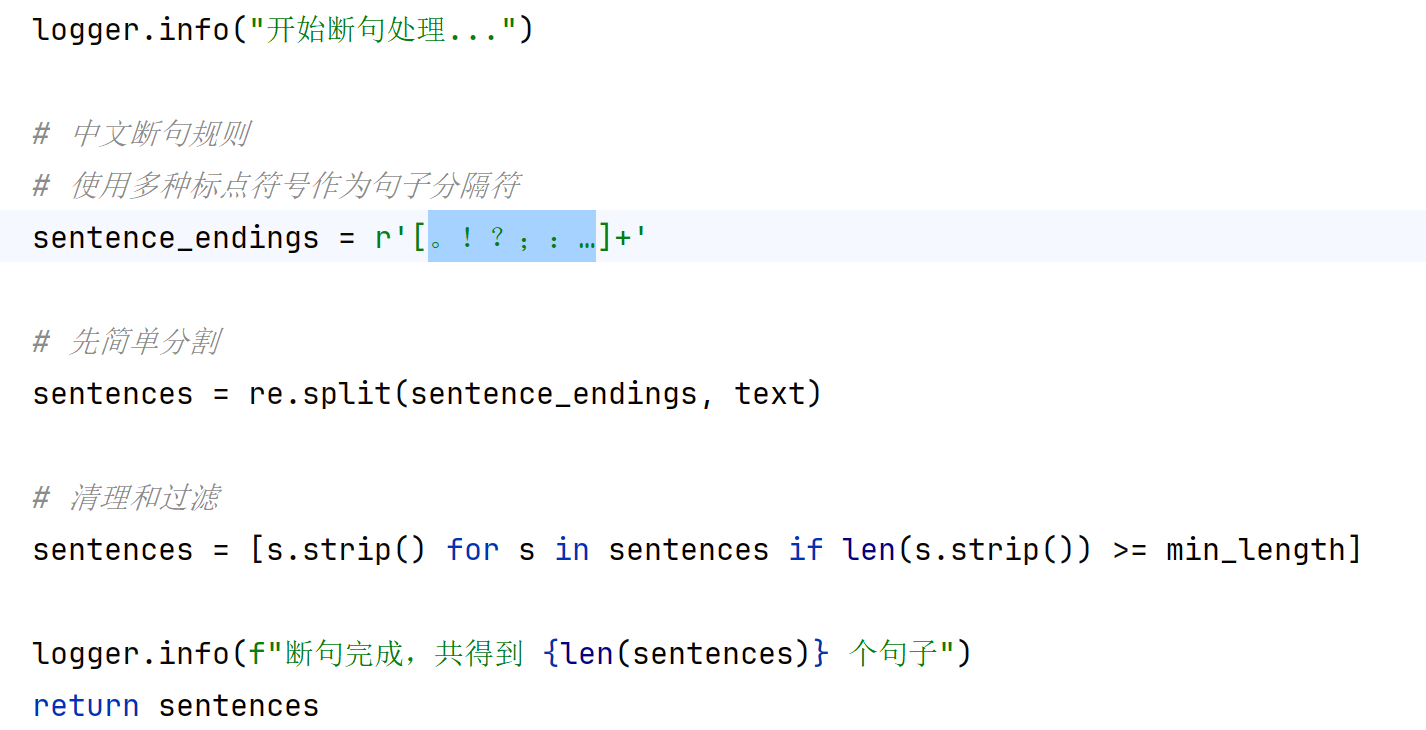
①使用中文标点符号(。！？；：…)作为分隔符。

②按分隔符分割成句子列表。

③过滤掉长度小于3的短句子。

④得到句子列表。

相关代码：



1. 分词：

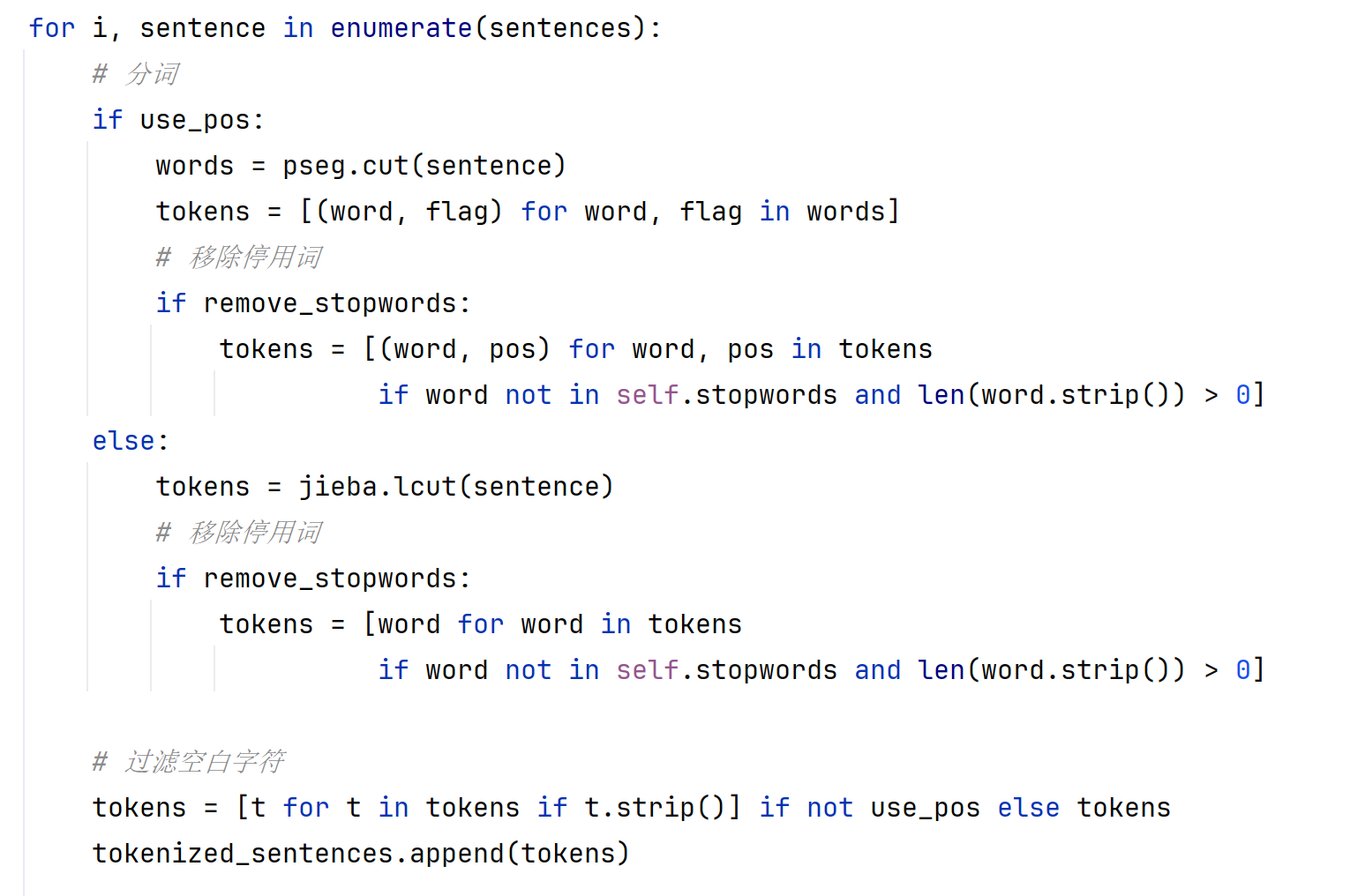
①对得到的句子列表中的每句话使用jieba进行分词；

②检查每个词是否在停用词表中，是的话去除；

③过滤空白字符。

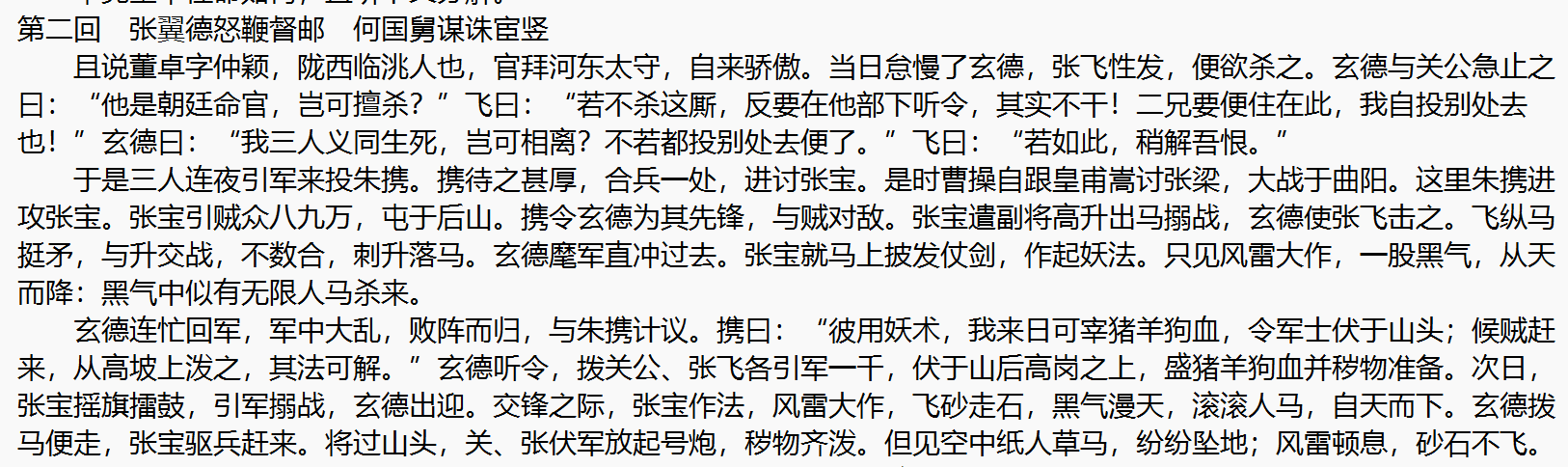
④保存分词后举子列表供后续流程使用。

相关代码：

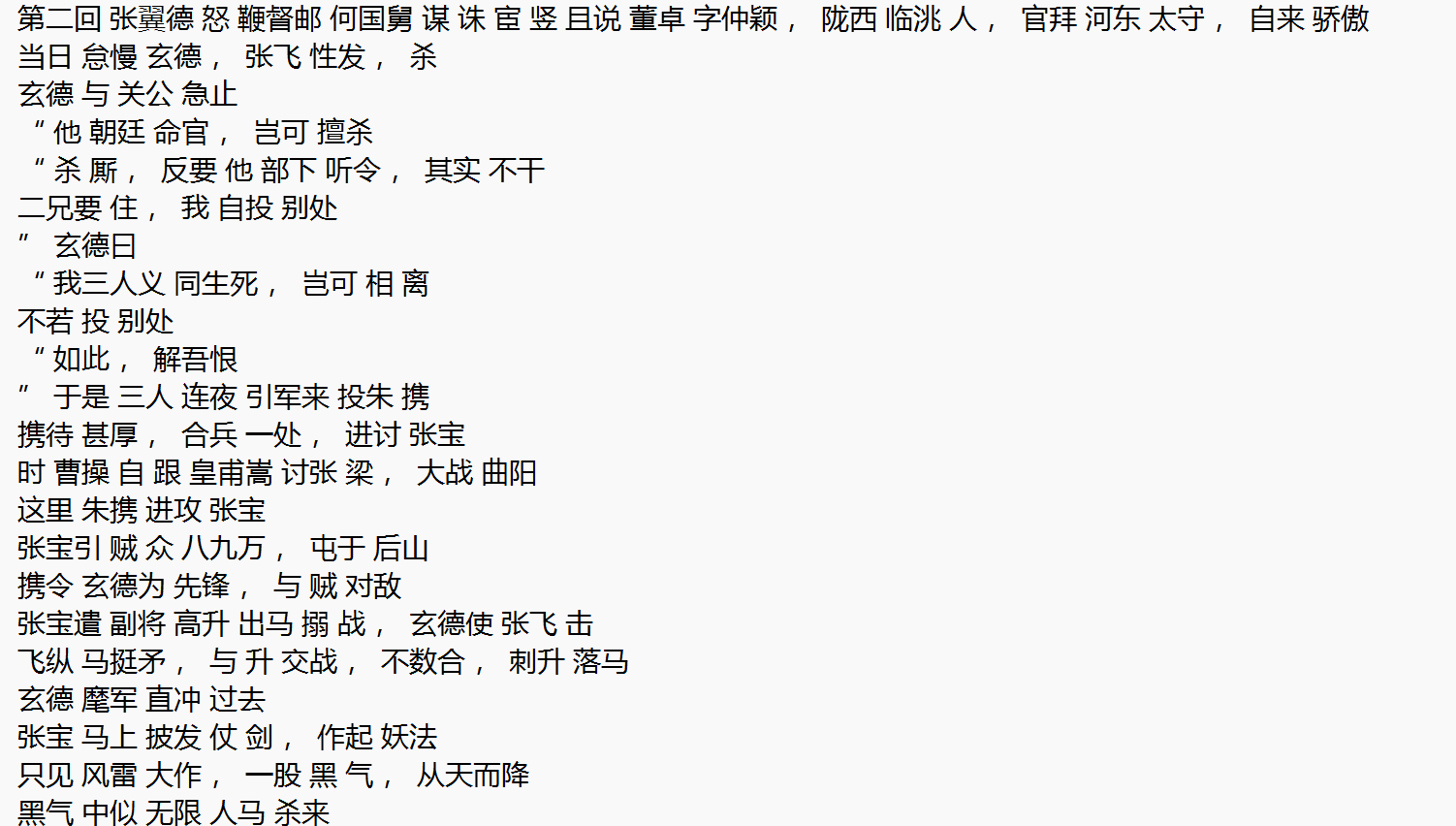


1. 处理样例：

①处理前：



②处理后：



**四、文本表示**

（一）TF-IDF文本表示方法

1.基本概念及计算公式

TF-IDF是一种用于信息检索与文本挖掘的常用加权技术，用于评估一个词对于一个文档集或语料库中某份文档的重要程度。

①TF（词频）：

TF(t,d) = 词t在文档d中出现的次数 / 文档d的总词数

一个词在文档中出现的次数越多，即TF越高，说明它对该文档越重要

②IDF（逆文档频率）：

IDF(t) = log(语料库中文档总数 / 包含词t的文档数 + 1)

一个词在越少的文档中出现，即IDF越高，说明它越能区分文档，重要性越高

③综合值：TF-IDF(t,d) = TF(t,d) × IDF(t)

一个词既在当前文档频繁出现，又在其他文档中少见，即高TF-IDF，说明这个词比较关键

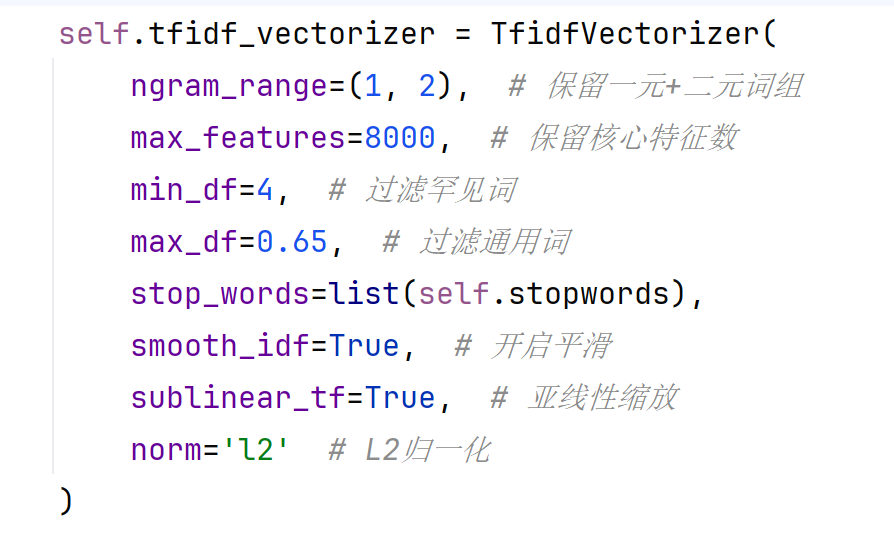
2.代码实现分析

2.1训练流程：

①准备数据

②向量器初始化

参数设置如图：



保留一元/二元词组是为了更好地识别专有名词；

IDF平滑处理是为了避免零除错误；

亚线性词频缩放是为了降低高频词影响；

L2归一化是为了统一向量模长，提高相似度计算精度。

③训练

具体流程都封装在函数内部，具体过程如下：

n-gram生成

构建词汇表

构建词频矩阵

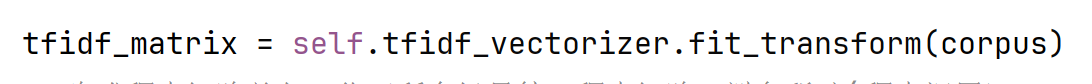
计算DF

计算IDF权重

应用TF变换、IDF变换

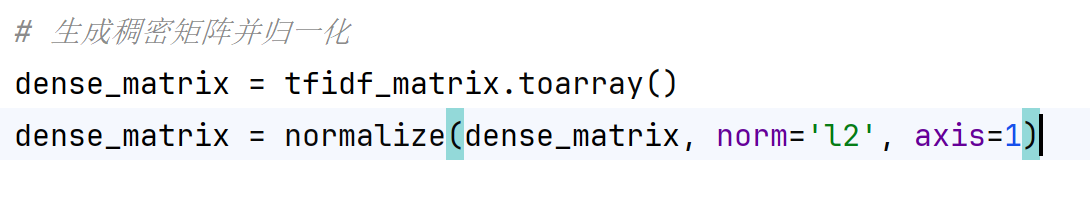
L2归一化

生成TF-IDF稀疏矩阵

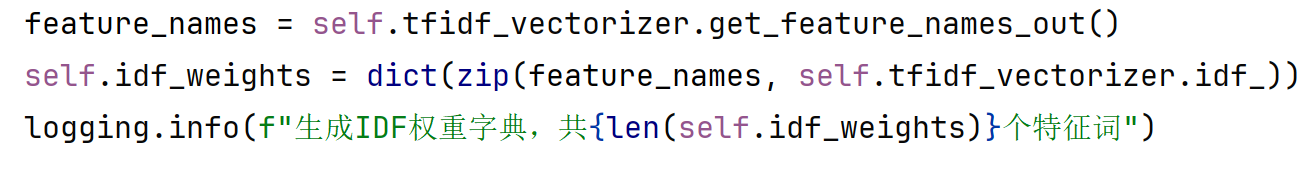


④转换成稠密矩阵

转换后再次归一化是为了确保向量模长严格为1，不影响后续计算的精度。



⑤保存IDF权重供Word2Vec使用



⑥保存模型

2.2查询流程

①输入查询句

②分词预处理（采用和模型训练一样的预处理）

③生成TF-IDF向量 （和文档句生成方式一样）

④计算余弦相似度（带惩罚机制）

⑤结果排序

⑥返回TOP-10



1. Word2Vec文本表示方法
2. 基本概念

Word2Vec是一种基于神经网络的词嵌入模型，通过训练将词汇映射到低维连续向量空间，使得语义相似的词语在向量空间中距离相近。主要包含CBOW和Skip-gram两种架构，本系统采用的是后者，通过中心词预测上下文词。

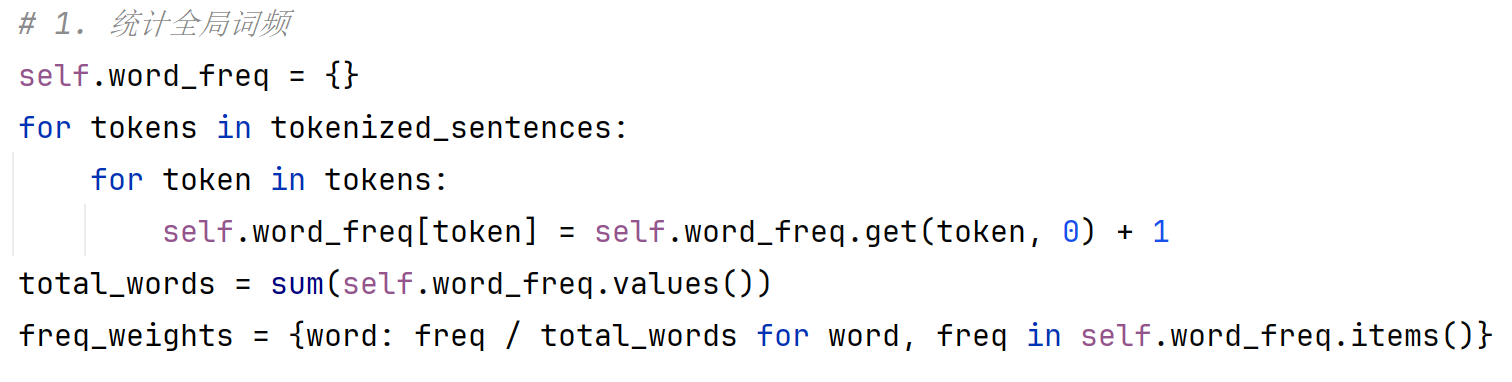
Skip-gram:输入中心词的向量，输出周围词的概率分布，通常对低频词效果更好，适合古籍。

1. 代码实现分析

2.1训练流程：

①准备数据

②词汇表构建：统计词频、过滤低频词、词汇表编码、构建霍夫曼树



③模型初始化：

模型参数初始化，参数设置如图：



窗口设置为15适配古文长句，捕捉长距离语义关系；

训练60次保证充分收敛；

启用层级softmax解决古文中人物/事件词训练困难；

关闭负采样避免与hs冲突；

下采样降低高频词干扰。

④训练循环：

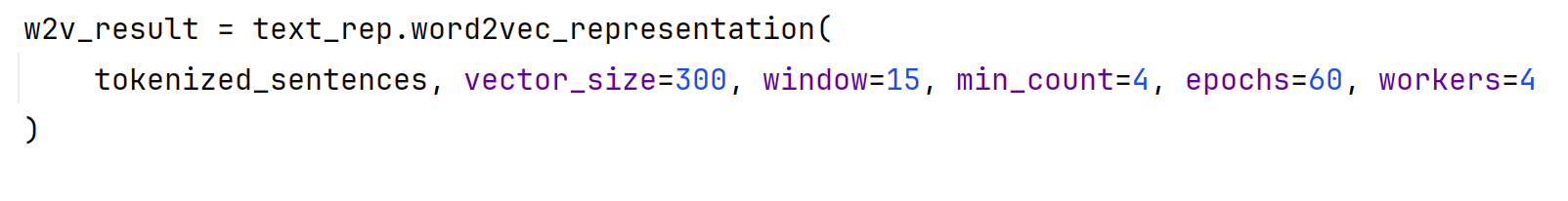
调用gensim的接口，将训练循环封装在一个函数内部。实际循环流程：

外层循环：Epoch迭代

内层训练：Skip-gram前向传播

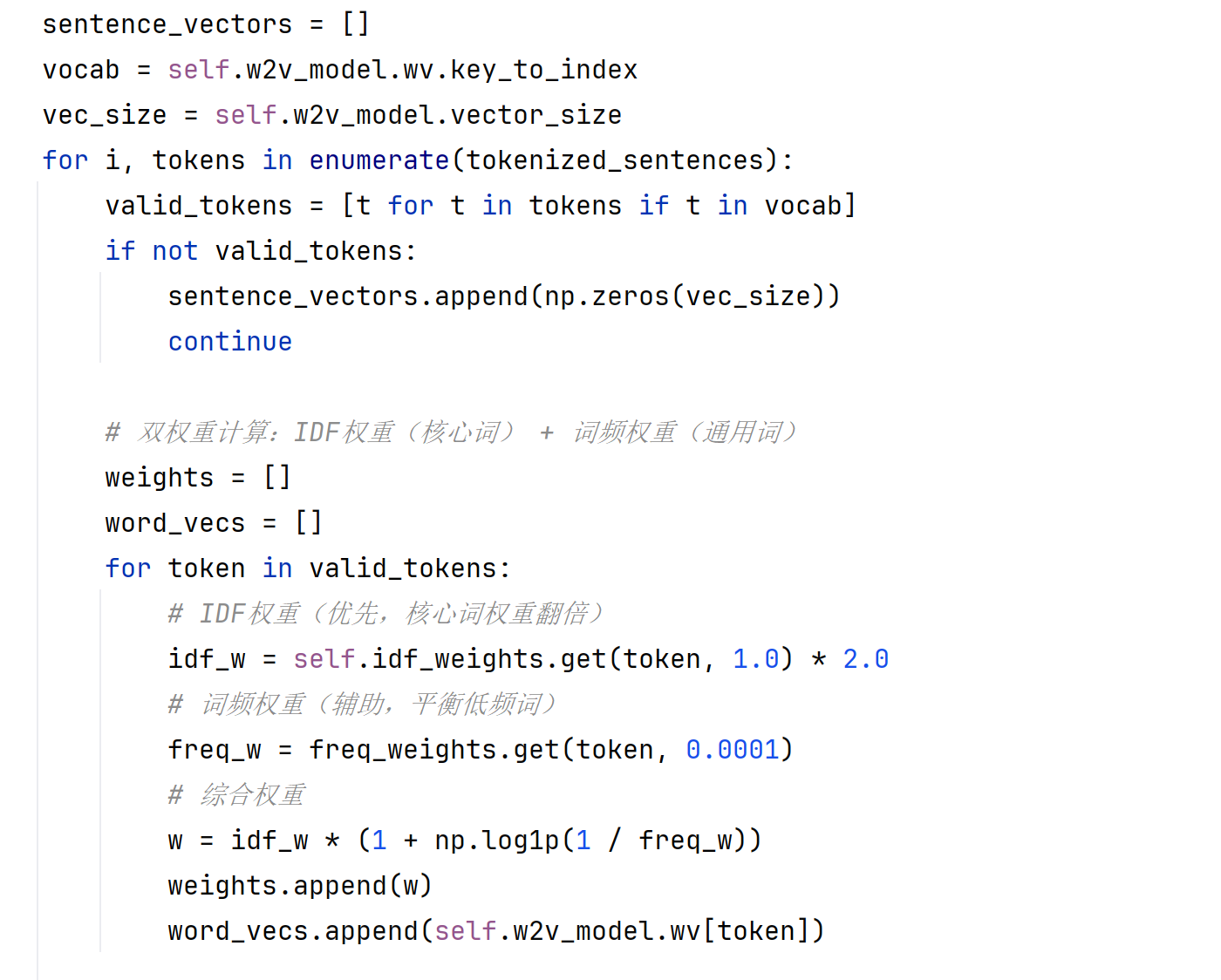
 反向传播与参数更新

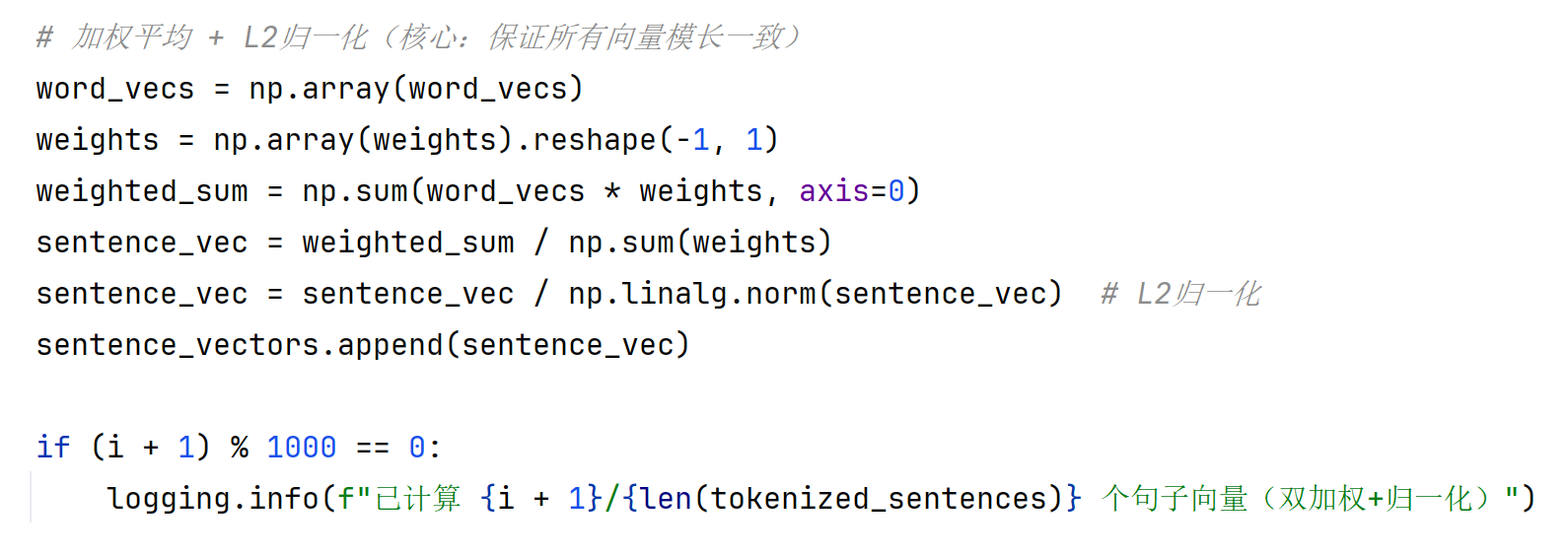
学习率衰减



⑤句子向量生成

采用双加权机制，对句子中每个词的词向量采用IDF和词频这****两种不同的权重进行加权平均****，而不是简单的算术平均，突出句子的核心词，提高语义相似度计算的准确性。





⑥模型保存

2.2查询流程

①输入查询句

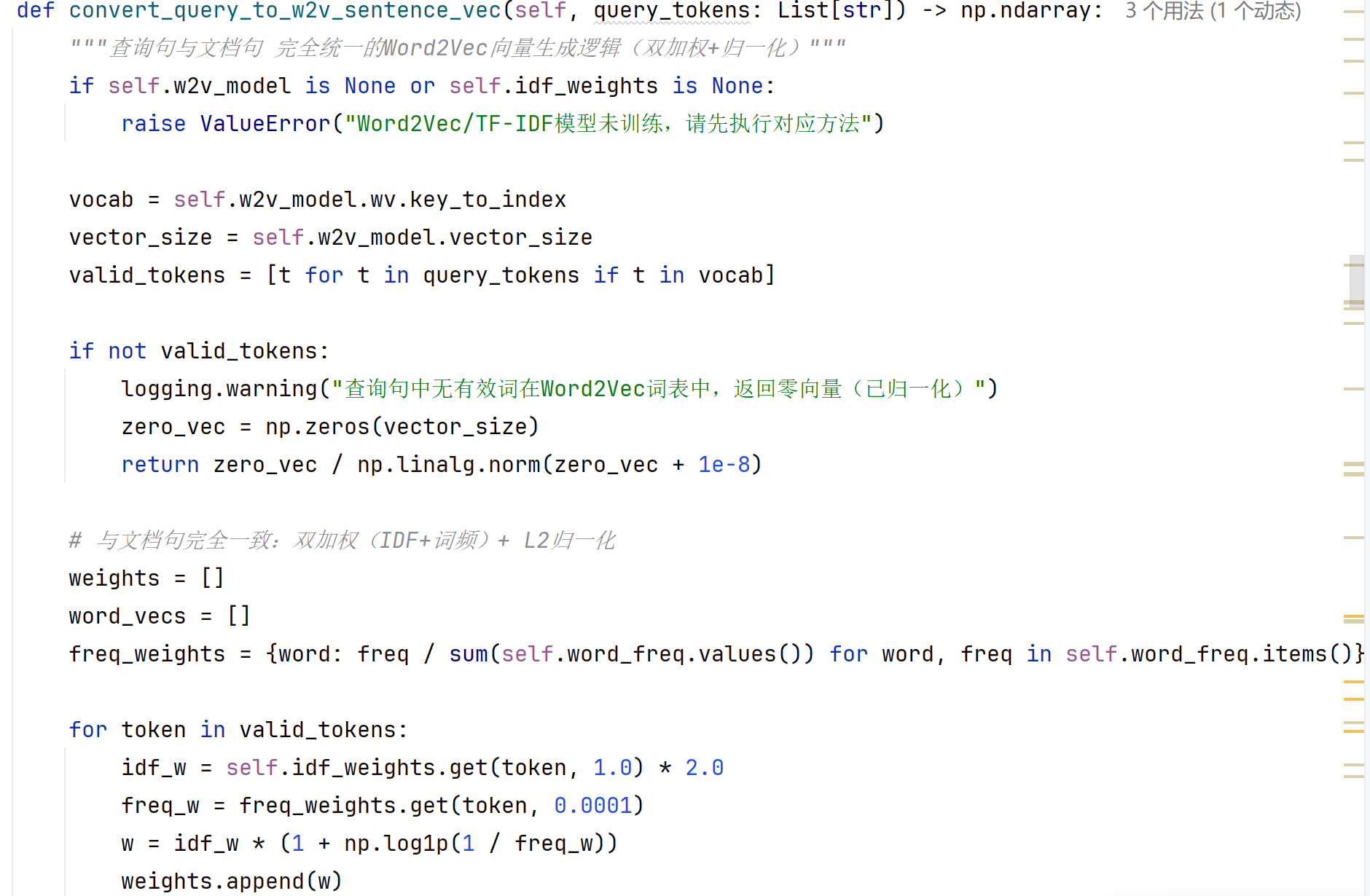
②分词预处理（采用和模型训练一样的预处理）

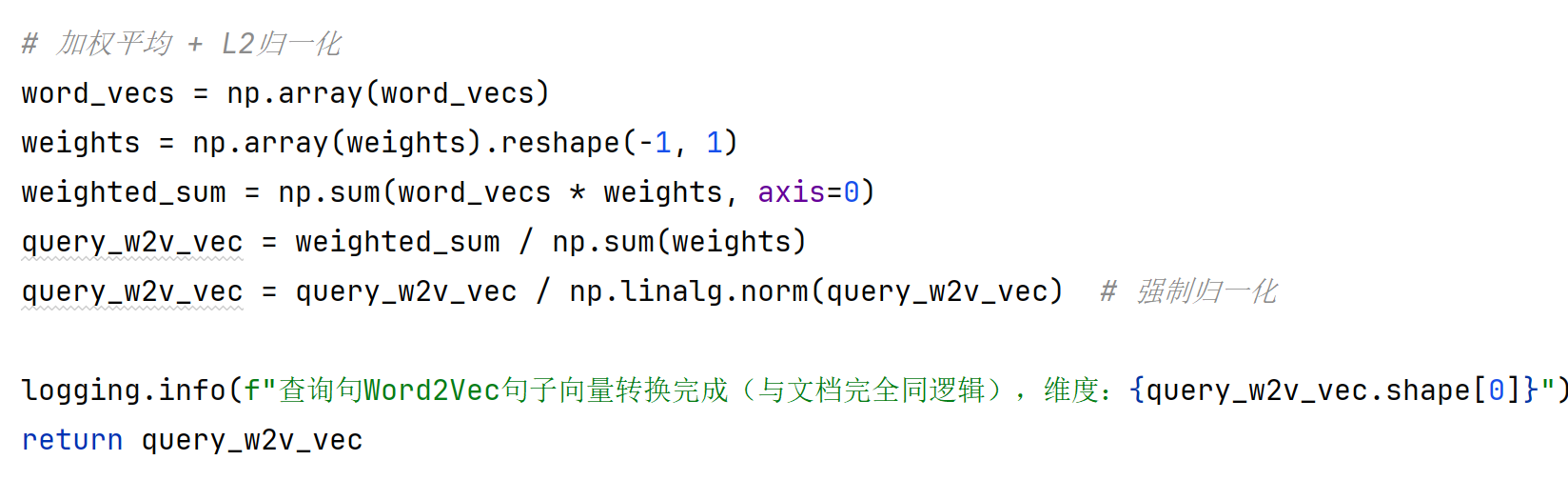
③生成Word2Vec向量 （和文档句生成方式一样）

④计算余弦相似度（带惩罚机制）

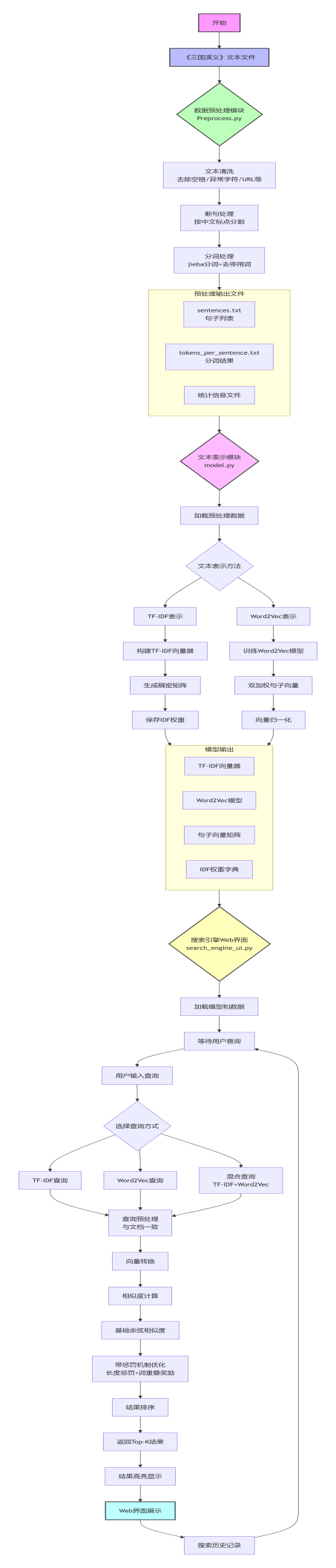
⑤结果排序

⑥返回TOP-10





**五、实验总体框架描述**

****

1. **系统展示与分析**
2. 系统页面展示

这是一个基于Flask框架的Web应用，专门用于对《三国演义》文本进行智能搜索。系统支持Word2Vec、TF-IDF以及混合搜索三种搜索方式，并提供了友好的用户界面。用户点进加载按钮后，系统会加载训练好的Word2Vec模型和TF-IDF模型，用户可以自主选择查询方式，然后自行输入或点击下方的搜索示例进行搜索，系统会返回相关的10个结果。其中，混合搜索是指综合两种模型搜索的结果，Word2Vec占0.6的权重，TF-IDF占0.4的权重。

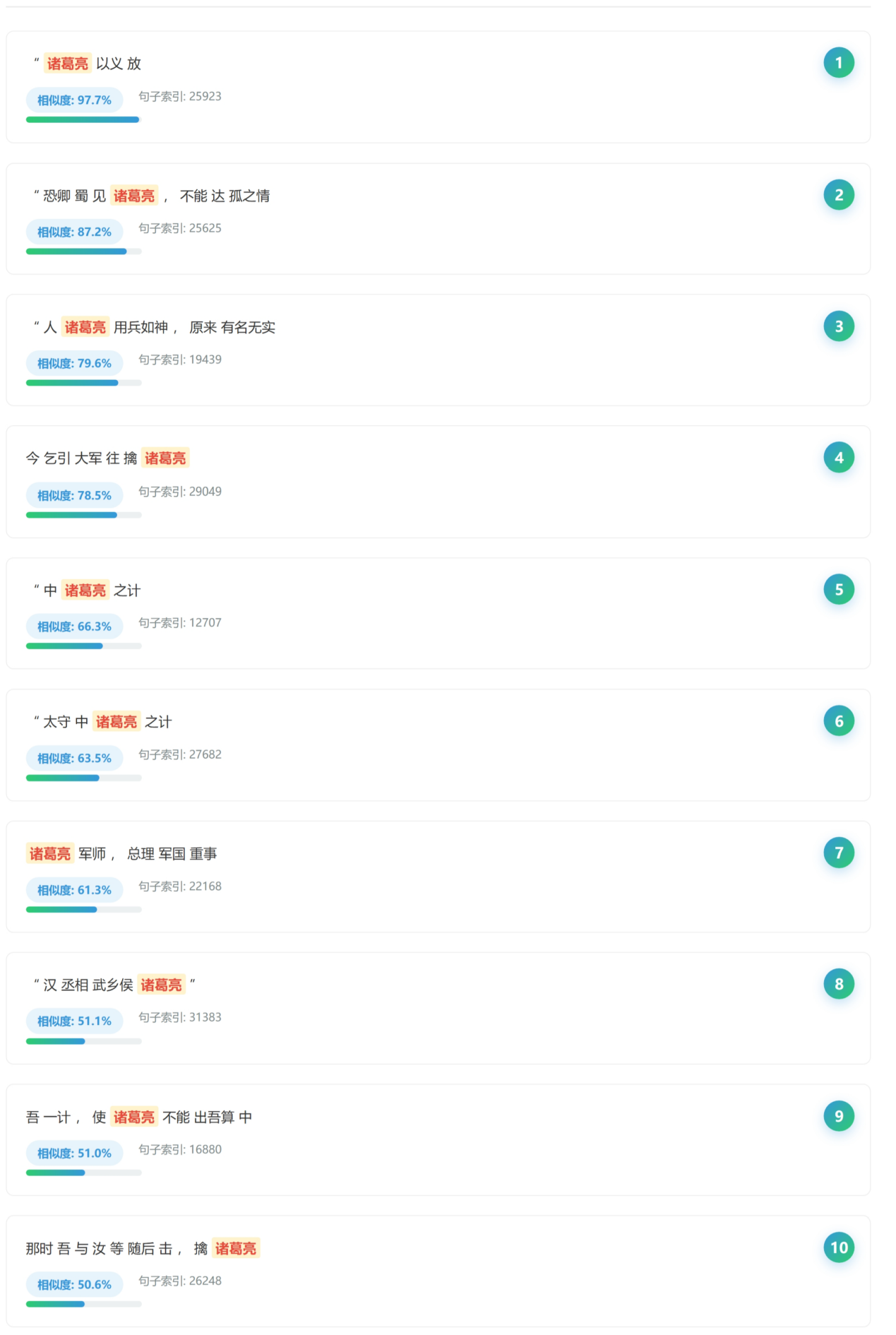


1. 展示结果分析
2. 短查询

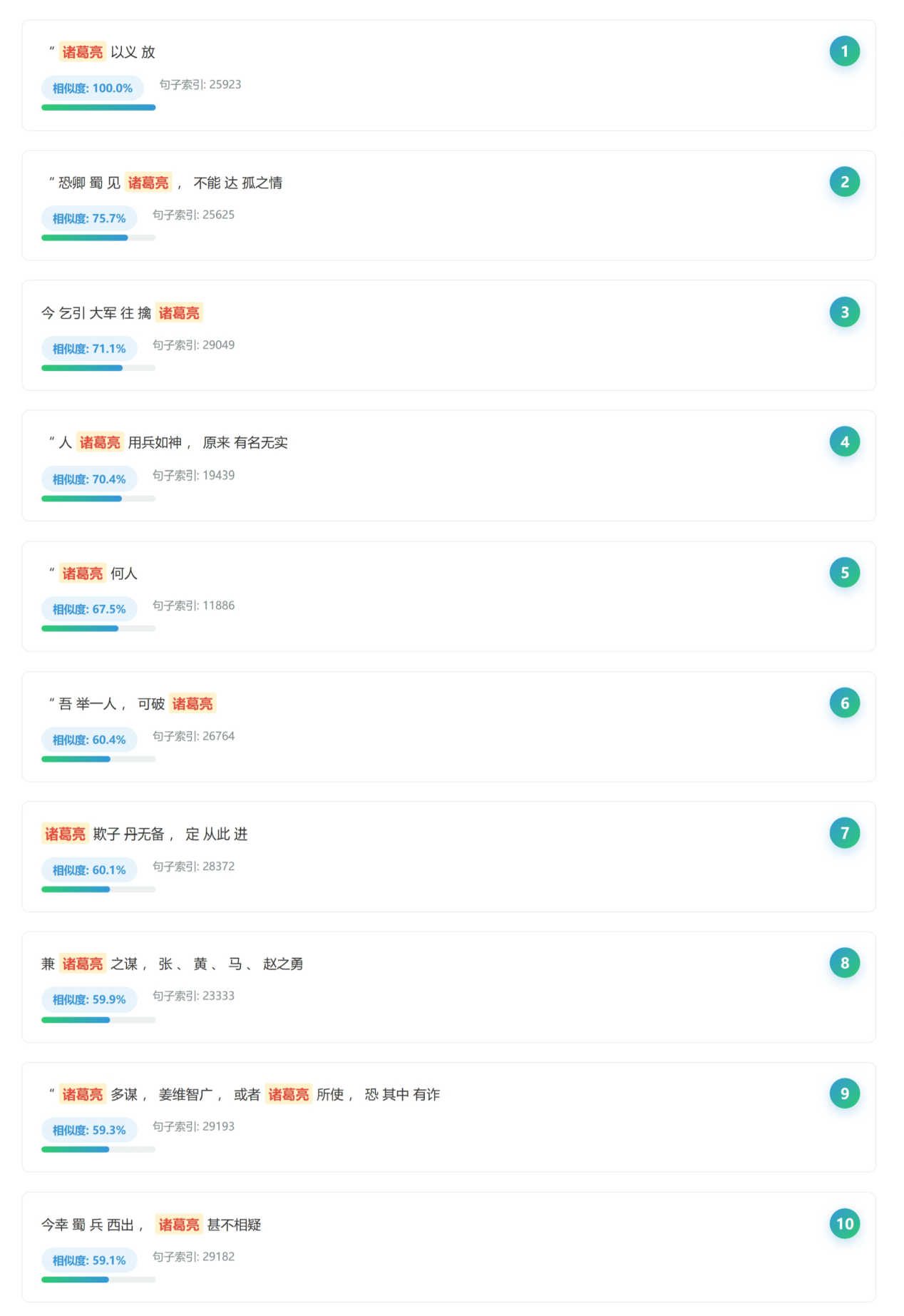
1.1单个名词

示例：诸葛亮

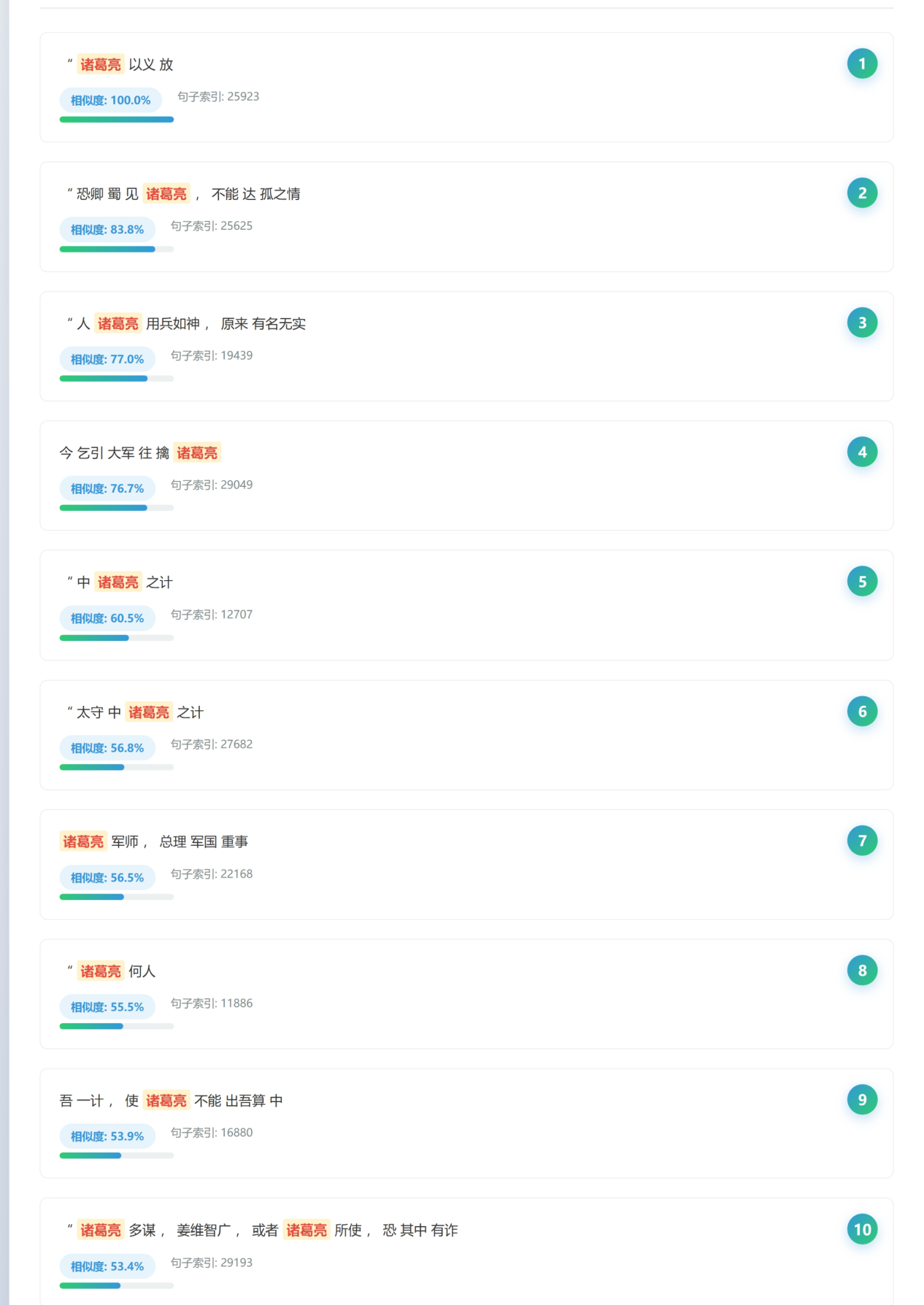
①Word2Vec搜索结果



②TF-IDF搜索结果



③混合搜索结果

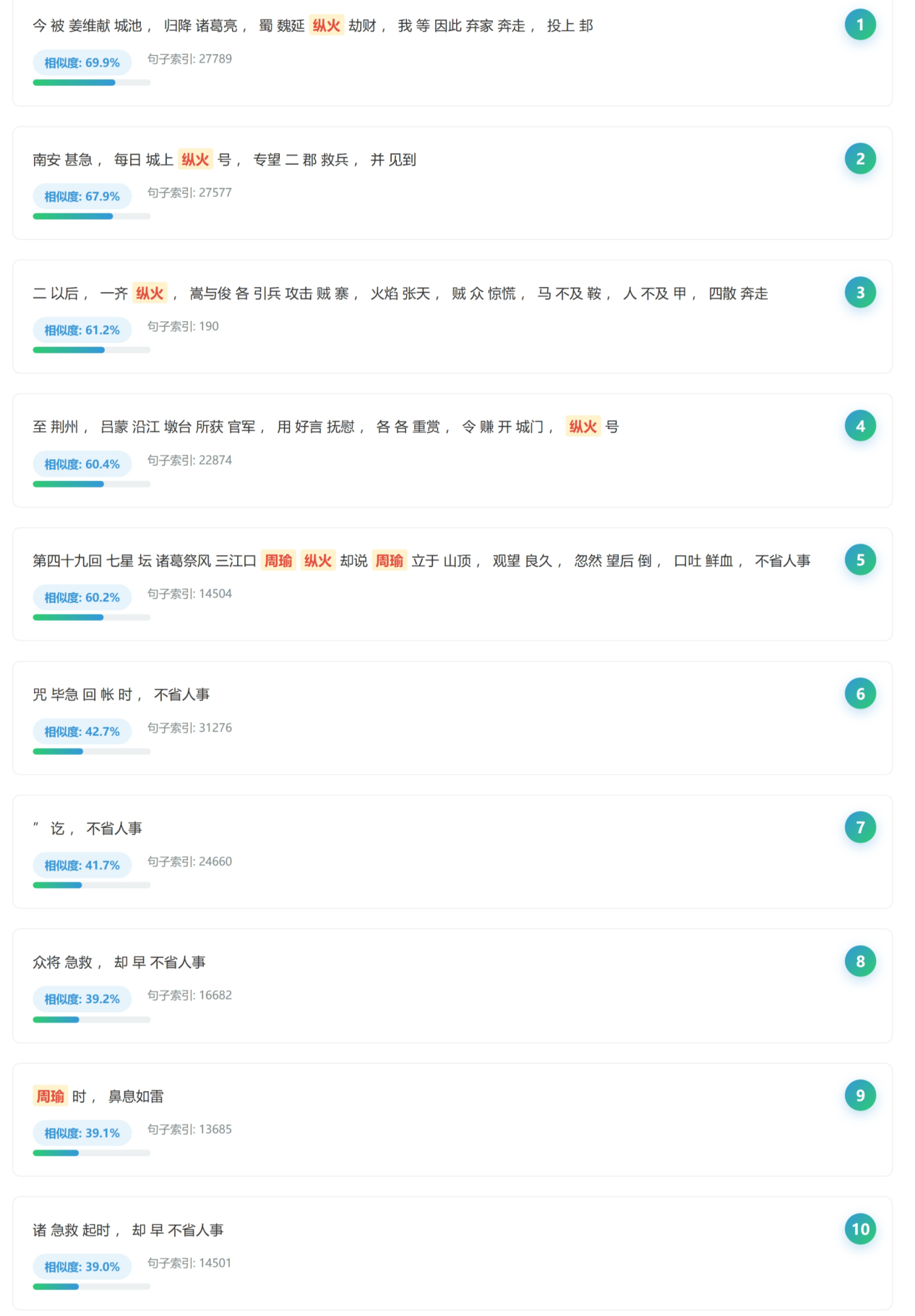


分析：三种方法的效果都差不多，返回结果都和诸葛亮有关，但是相似度的计算有点问题，前面几个的相似度不应该那么高。

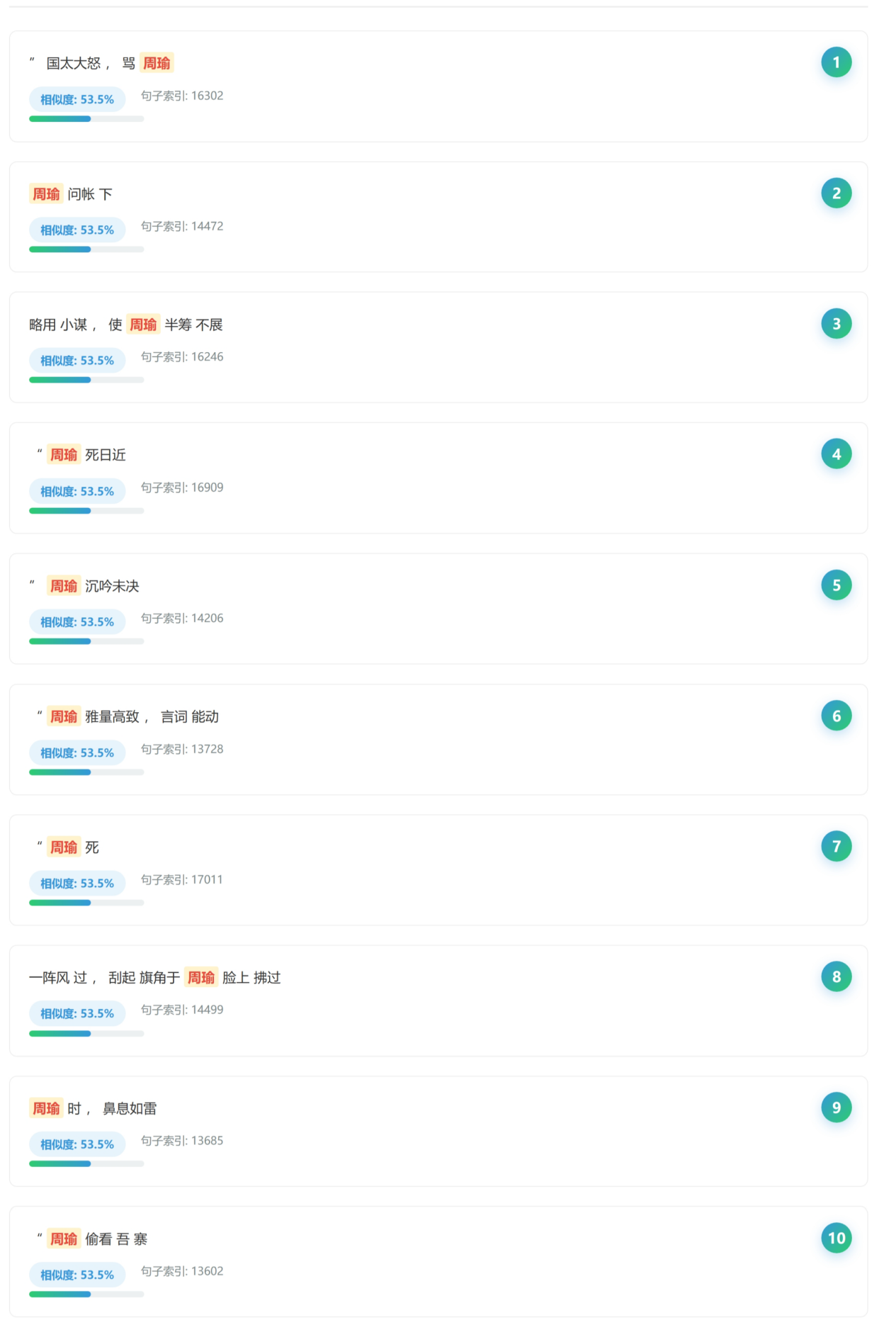
1.2短语

示例：周瑜纵火

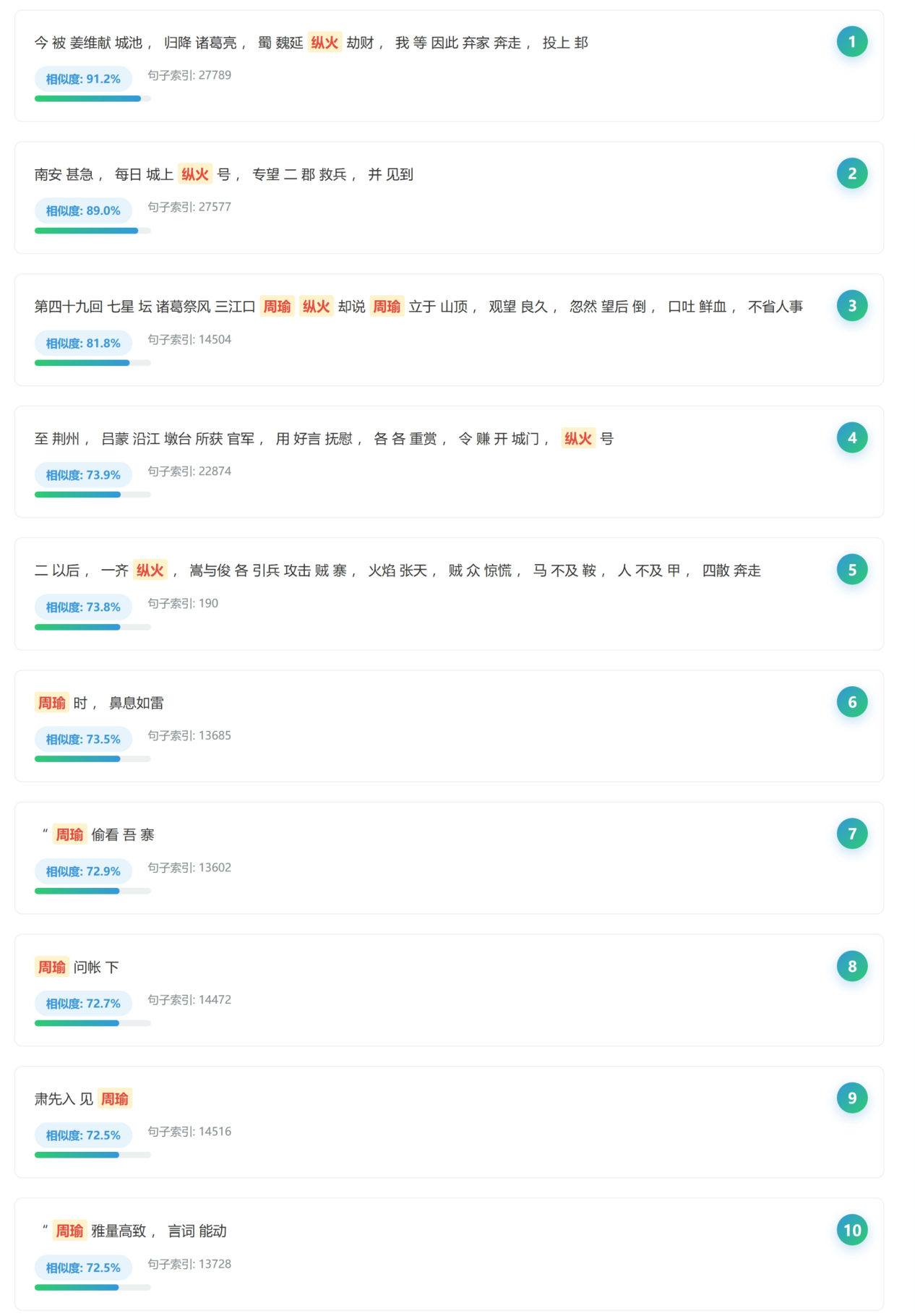
①Word2Vec搜索结果



②TF-IDF搜索结果



③混合搜索结果

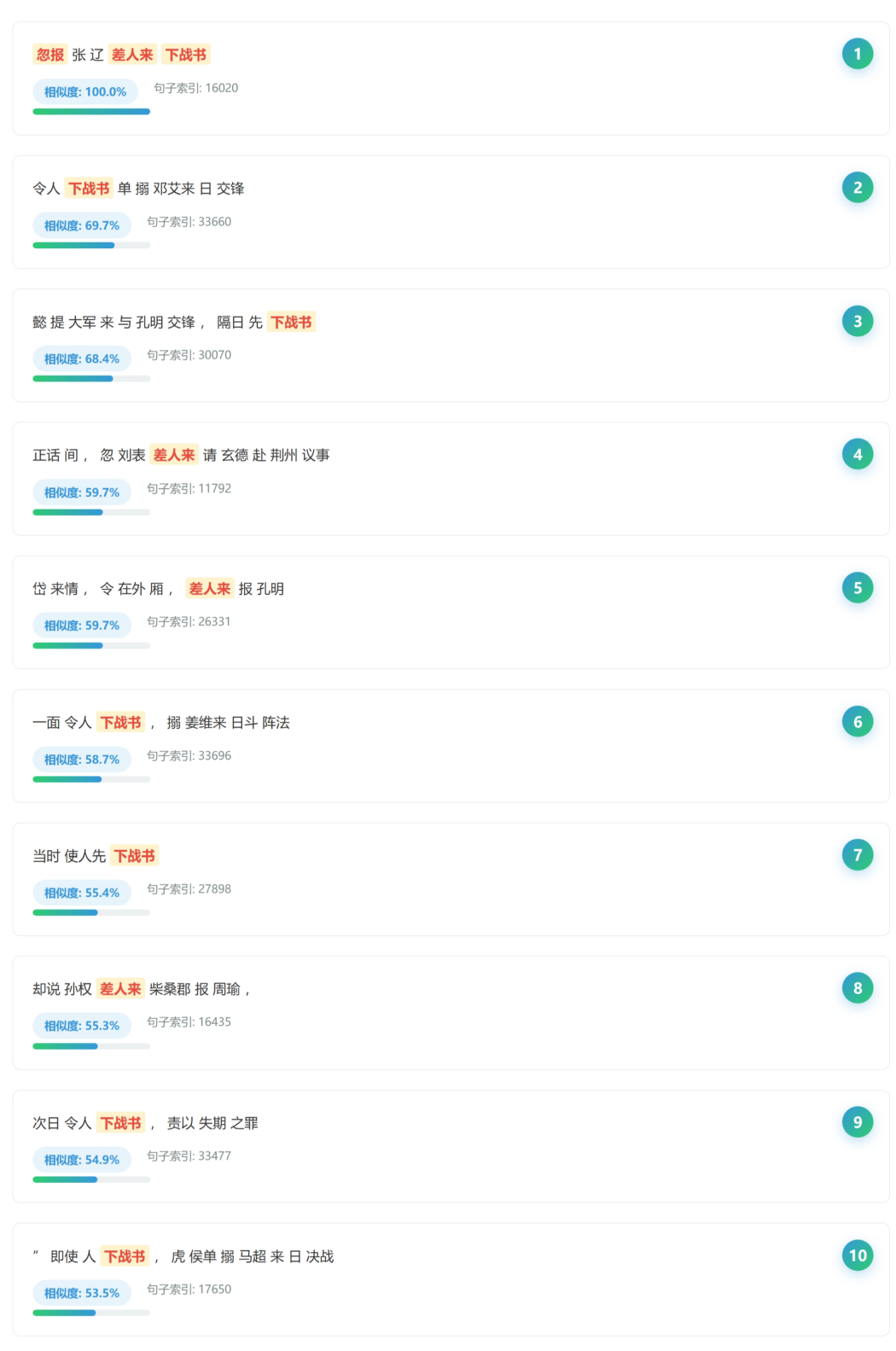


分析：只有TF-IDF没有返回同时包含了“周瑜”和“纵火”的句子，但是另外两个返回的结果也有一点问题，同时包含了这两个词的句子的相似度不是最高的，并且Word2Vec里面还返回了两个词都不包含的句子，综合来看在这里混合所搜的效果最好。

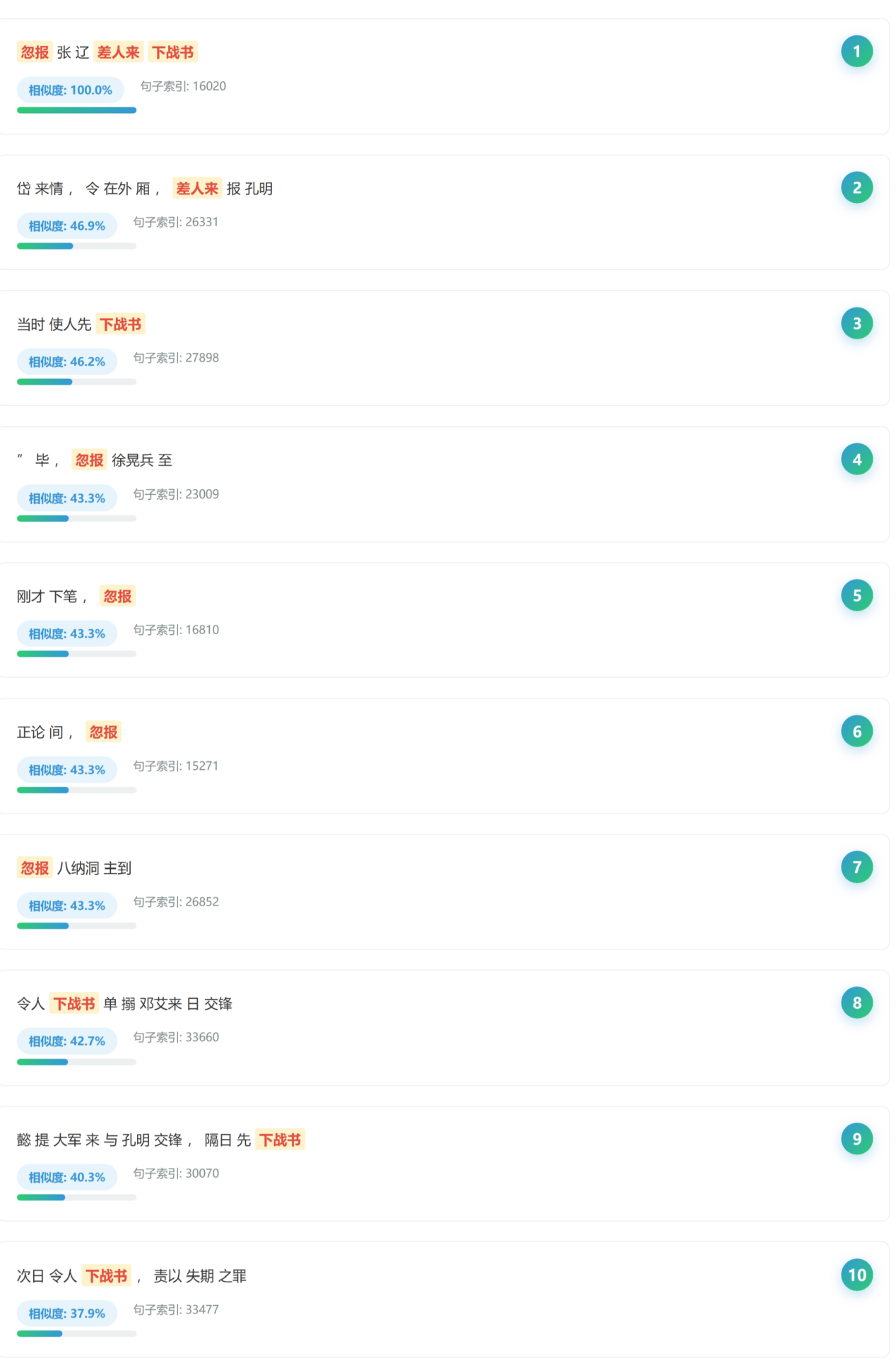
1. 长查询
   1. 中长句

示例：忽报张辽差人来下战书。

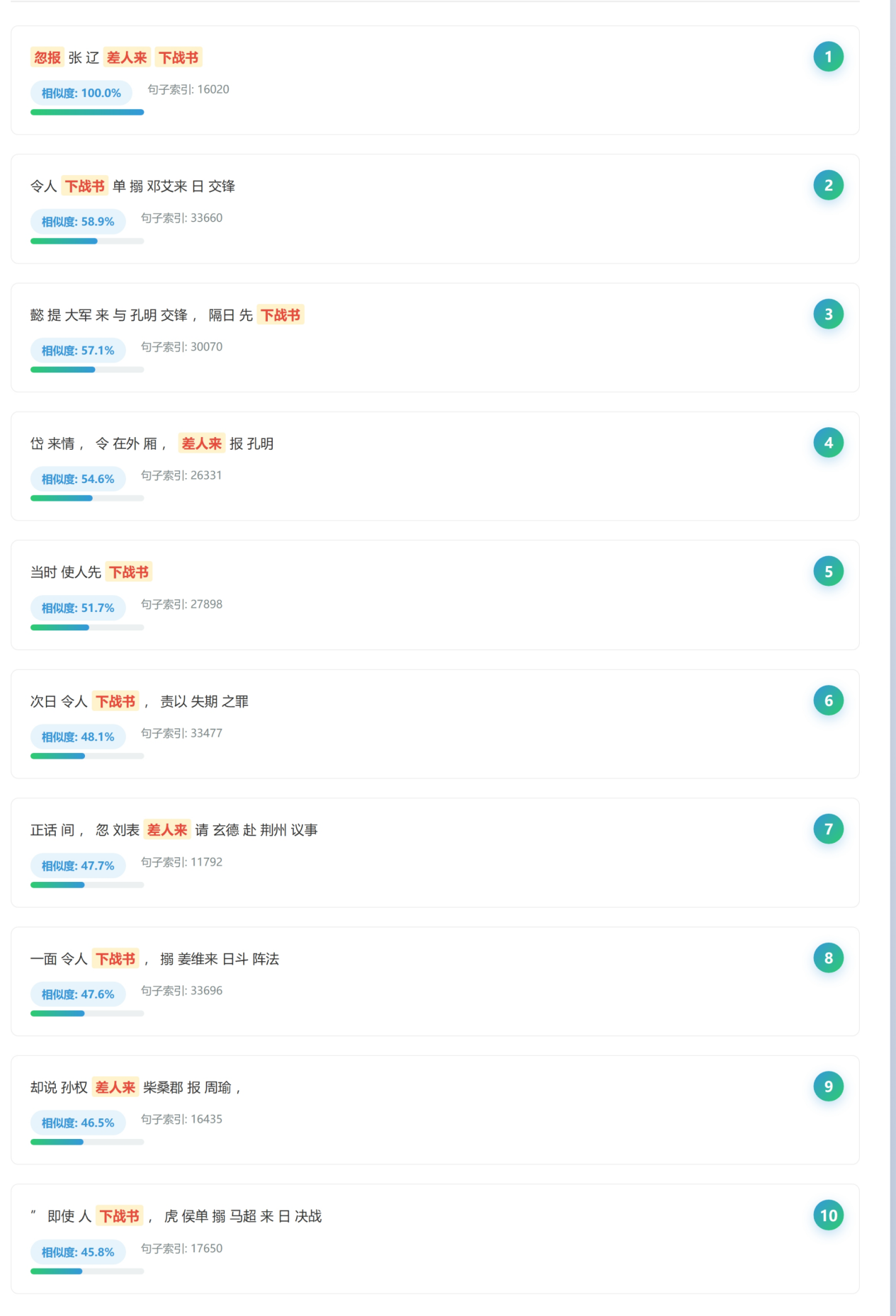
①Word2Vec搜索结果



②TF-IDF搜索结果



③混合搜索结果

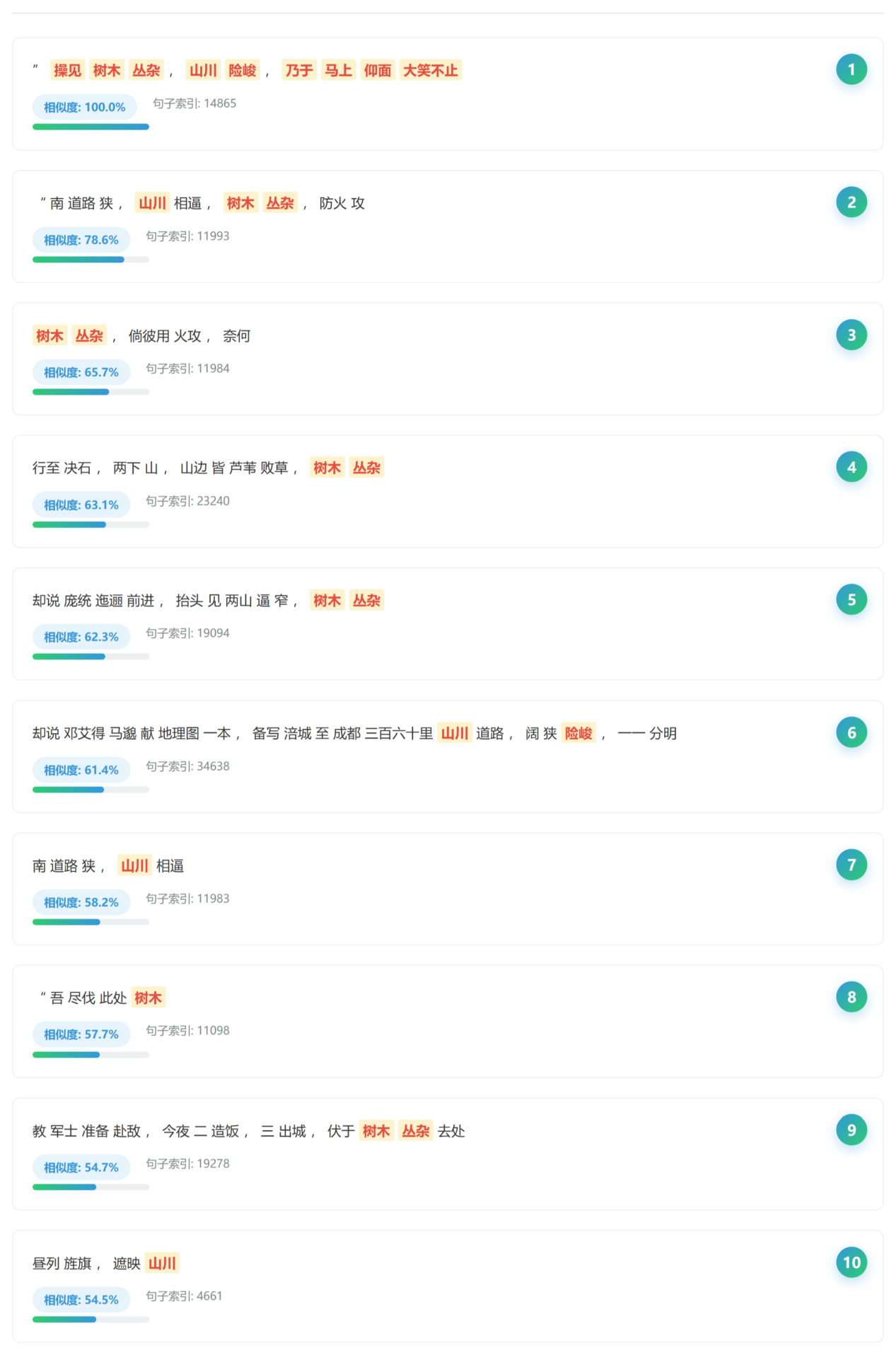


分析：三种方法的效果都差不多，返回的结果中第一句都是原句，并且相似度是100%，而其他句子相似度的计算结果也算合理。

2.2长句

示例：操见树木丛杂，山川险峻，乃于马上仰面大笑不止。

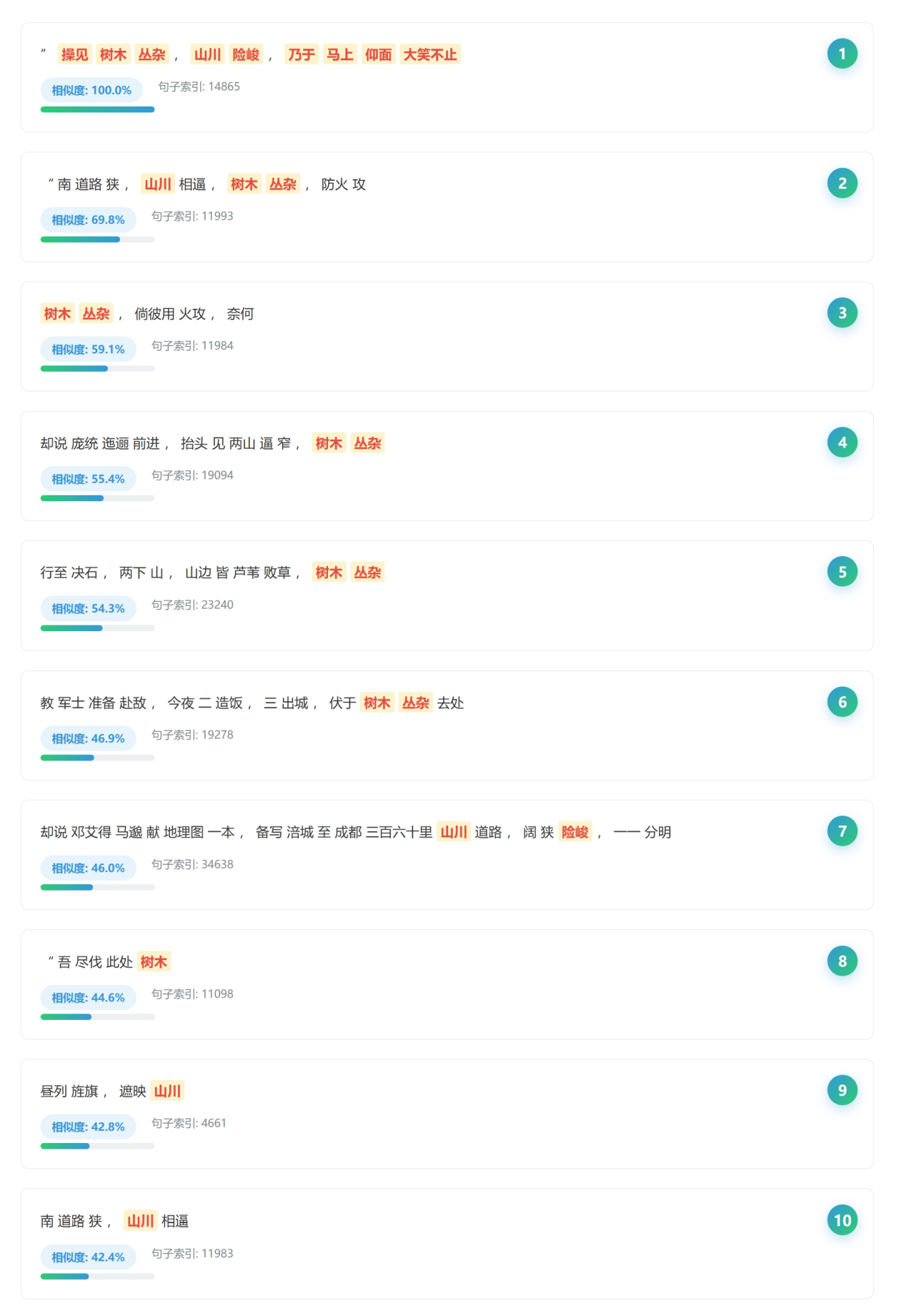
①Word2Vec搜索结果



②TF-IDF搜索结果



③混合搜索结果



分析：三种方法的效果都差不多，返回的结果中第一句都是原句，并且相似度是100%，TF-IDF在其他句子相似度的计算结果上较合理一点，另外两种方法有点偏高。

1. 结果总结：

①对比发现长查询的结果会比短查询的结果要好，可能因为短查询包含的信息较少，生成的向量维度低；

②在基本余弦相似度的基础上加上惩罚机制有效提高了相似度计算的精准度，改之前有些无关句子的相似度很高。

**七、总结**

本项目成功设计并实现了一个基于《三国演义》全文的智能搜索引擎系统，完成了从数据采集、预处理、文本表示、相似度计算到系统展示的完整流程。项目核心成果与经验总结如下：

（一）主要成果

1. 完整的系统构建：实现了包含预处理模块、表示模型、相似度计算引擎及Web交互界面的完整系统，形成了端到端的解决方案。

2. 双重文本表示方法：有效结合了TF-IDF（侧重关键词匹配）和Word2Vec（侧重语义相似度）两种技术，并通过混合搜索机制综合其优势。

3. 优化与创新：

预处理一致性：确保了用户查询与文档库采用完全相同的清洗、分词流程，避免了特征空间不匹配问题。

双加权向量生成：Word2Vec句子向量融合了IDF权重与词频权重，增强了核心词的表达能力。

惩罚机制：在余弦相似度基础上引入长度惩罚与词重叠奖励，有效缓解了短查询句相似度失真的技术难题。

友好的系统展示：开发了基于Flask的Web应用，提供了简洁直观的交互界面，支持用户进行多种模式的探索性检索。

（二）实验发现  
1. 系统测试表明：

长句查询效果显著：对于原文中存在的完整或较长句子，三种搜索方式均能精准定位，相似度计算合理。

短查询面临挑战：对于“诸葛亮”、“周瑜纵火”等短短语，尽管优化机制有所改善，但语义匹配的精准度和相似度分数的合理性仍有提升空间，这揭示了基于词袋和静态词向量的方法在捕捉简短语境信息上的固有局限。

混合搜索的综合优势：在多数场景下，混合搜索（TF-IDF 0.4 + Word2Vec 0.6）能够平衡关键词匹配与语义关联，返回更全面、可靠的结果。

（三）不足与展望  
尽管系统实现了既定目标，但仍存在可改进之处：

1. 模型局限性：当前的Word2Vec模型为静态词向量，无法解决一词多义问题。未来可引入BERT等基于上下文动态生成词向量的预训练模型，以更好地理解古文语义。

2. 交互功能深化：可增加高级搜索功能（如按人物、章节过滤）、可视化人物关系图谱或事件脉络图，提升系统的分析能力。

3. 算法持续优化：相似度惩罚机制的参数可基于更大规模的评估数据进行自动化调优，并探索更先进的排序学习（Learning to Rank）算法。

综上所述，本项目不仅构建了一个可用的《三国演义》专用搜索引擎，更通过实践深入理解了传统与现代NLP技术在具体应用中的结合点与挑战，为处理类似古典文献的智能化检索任务提供了有价值的参考范本和技术经验。