



本科毕业设计(论文)

题目：基于自然语言处理的校园智能问答系统设计与实现

院（系）	计算机科学与工程学院
专 业	计算机科学与技术
班 级	20060723
姓 名	王文青
学 号	2020032612
导 师	胡秀华

2024 年 6 月

基于自然语言处理的校园智能问答系统设计与实现

摘要

随着智能时代来临，为了能使计算机更好的理解人类的语言并给出回应，自然语言处理技术应然而生。智能问答系统已广泛使用于各个领域，而其对现代教育环境也具有重要意义。本研究旨在设计并实现一款基于自然语言处理技术的校园智能问答系统，以提供学生、教师和其他校园成员有针对性的帮助和支持。该系统基于知识图谱构建，并使用余弦相似度匹配方法实现问答功能。同时，通过 Flask 框架部署网站作为系统的前端界面。

本研究首先构建了一个校园知识图谱，涵盖了涉及多个领域的信息，如课程内容、教学资源、校园生活等。接着，使用 AC 树的方法匹配问题，同时结合余弦相似度计算方法实现了问答系统的匹配模块，用于与用户提出的问题进行相似度匹配以获取相关答案。最后，采用 Flask 框架开发了网站前端，实现了系统的可视化展示和与用户的交互。

通过系统的实际应用和测试，本项目取得了显著的成果。校园智能问答系统在知识图谱的支持下，能够对用户提出的问题做出精准的回答，且具有良好的智能交互体验。同时，通过网站部署，系统能够直观地呈现给用户，并得到了用户的积极反馈。

本研究设计并实现了一款基于知识图谱和余弦相似度匹配的校园智能问答系统，并成功将其部署为网站前端。该系统为校园成员提供了高效、智能的信息获取方式，丰富了校园智能化建设的内涵，具有良好的推广和应用前景。

关键词：智能问答系统；自然语言处理；知识图谱；AC 树；余弦相似度匹配；Flask 框架

Spacecraft Docking System Based on Computer Vision

Abstract

With the advent of the era of artificial intelligence, natural language processing technology has emerged to enable computers to better understand human language and provide responses. Intelligent question-answering systems have been widely used in various fields, and they have significant implications for modern educational environments. This study aims to design and implement a campus intelligent question-answering system based on natural language processing technology to provide targeted assistance and support to students, teachers, and other campus members. The system is built on a knowledge graph and utilizes cosine similarity matching methods to implement the question-answering functionality. Additionally, a website frontend was deployed using the Flask framework.

The study first constructed a campus knowledge graph covering information from multiple domains such as course content, teaching resources, and campus life. Subsequently, the system employed the AC-tree method to match questions and combined it with cosine similarity calculation to implement the matching module for the question-answering system, enabling similarity matching with user queries to retrieve relevant answers. Finally, the website frontend was developed using the Flask framework to visualize the system and enable interaction with users.

Through the practical application and testing of the system, this project has achieved significant results. The campus intelligent question-answering system, supported by the knowledge graph, can provide precise answers to user queries and offers a good experience of intelligent interaction. Furthermore, through website deployment, the system can be intuitively presented to users and has received positive feedback from them.

This study designed and implemented a campus intelligent question-answering system based on a knowledge graph and cosine similarity matching, successfully deploying it as a website frontend. This system provides efficient and intelligent information retrieval for campus members, enriching the connotation of intelligent campus construction, and demonstrating good prospects for promotion and application.

Key Words: intelligent question-answering system; natural language processing; knowledge graph; AC-tree; cosine similarity matching; Flask framework

目录

中文摘要.....	1
英文摘要.....	2
1 绪论	1
1.1 研究背景	1
1.2 研究目的及意义	1
1.3 国内外研究现状	2
1.4 论文主要工作安排	3
2 智能问答系统相关技术理论及总体设计方案	4
2.1 知识图谱实现原理	4
2.2 文本预处理技术	5
2.2.1 中文分词技术	7
2.2.2 词向量	7
2.3 预训练模型	9
2.4 总体设计方案	9
2.5 本章小结	11
3 基于知识图谱的智能问答系统设计	12
3.1 校园问答语料库的构建及存储	12
3.2 在知识图谱中匹配用户提出的问题	14
3.2.1 使用 AC 树匹配关键词（实体）	14
3.2.2 使用模板匹配得到意图疑问词（属性）	15
4 基于余弦相似度匹配问题	17
4.1 数据预处理	17
4.2 模型训练结果	17
4.3 余弦相似度的计算	18
5 智能问答系统的实现与终端显示	19

5.1 使用 Flask 实现问答终端	19
5.2 智能问答系统前端设计	20
5.3 问答系统的前端实现	22
5.4 本章小结	25
6 系统测试	26
6.1 测试内容与测试环境	26
6.1.1 系统测试内容	26
6.1.2 系统测试环境	26
6.2 功能性测试	27
6.2.1 问题匹配问答模块	27
6.2.2 问答界面模块	28
6.3 非功能性测试	31
6.4 本章小结	32
7 总结与展望	33
7.1 本文总结	33
7.2 未来展望	33
参考文献	34
致谢	35

1 绪论

1.1 研究背景

随着人工智能技术的不断发展，自然语言处理（NLP）成为智能系统中的关键技术之一。校园内存在大量信息和资源，学生、教师以及其他校园相关人员常常需要获取各种信息，解决问题或获取支持。在这种背景下，一个基于自然语言处理的校园智能问答系统可以成为学习和工作中不可或缺的工具，有助于提高校园信息服务的智能化水平。

基于自然语言处理的校园智能问答系统设计与实现，是一个富有挑战性且备受期待的课题。随着信息技术的不断发展和普及，人们对信息获取的需求也日益增长。在高校校园中，学生、教职员工和访客经常需要获取各种信息，例如课程表、图书馆开放时间、活动通知、校园地点等。然而，传统的信息查询方式，如网页检索或人工咨询，往往存在效率低、查询过程繁琐等问题。因此，设计并实现一款能够智能回答校园相关问题的系统，将大大提高信息的获取效率和便捷程度。

当前，随着自然语言处理技术的迅猛发展，包括语音识别、信息抽取、文本分析等方面的技术已经取得巨大进展。这些技术的发展为基于自然语言处理的校园智能问答系统的设计与实现提供了有力的技术支撑。通过利用自然语言处理技术，可以使系统能够理解用户提出的问题，并以自然语言的形式准确地回答用户的疑问。

此外，当前人工智能和大数据技术的蓬勃发展，为构建更加智能、高效的校园智能问答系统提供了更多的可能性。应用机器学习、深度学习等技术，可以使系统不断优化自身的问答能力，更好地适应用户的需求。

1.2 研究目的及意义

首先，随着自然语言处理技术的飞速发展，构建校园智能问答系统可以为自然语言处理领域提供一个实证平台，探索自然语言理解 and 生成的新理论和方法。通过解决校园问答系统中存在的语义理解、信息抽取、语境识别等问题，可以拓展自然语言处理领域的理论边界，推动相关技术的发展。

其次，校园智能问答系统的设计与实现对于知识图谱构建和应用具有重要意义。在系统的运行过程中，需要构建校园相关的知识库，包括课程信息、人

员机构信息、校园地点等。这将促进知识图谱的扩建和优化，为知识图谱在教育领域的应用提供实际案例和支撑。

同时，此项研究有助于提升校园信息化建设的水平。作为高校的重要信息化应用，该系统可以为学生、教师、访客提供全面、便捷、及时的信息服务。学生、教师可以更加便捷地获取到资源和信息。比如，学生可以通过系统查询课程安排、教学资料等信息，教师可以通过系统了解校园活动、学术讲座等信息。这有利于学校资源的高效利用和共享，提升了教育教学的效率和质量。系统的实现有助于推动学校管理的智能化。通过系统收集和分析用户的查询行为，可以为学校领导提供决策支持和数据参考，有助于学校管理的科学决策和智能化管理。

此系统的实现还有助于对自己的专业技能培养。在研究和开发过程中，自己将接触到自然语言处理、人工智能等前沿技术，提升自己的技术能力和创新意识。这有助于自己将理论知识应用到实际项目中，培养实际动手能力和综合素质。

1.3 国内外研究现状

基于自然语言处理的校园智能问答系统设计与实现是当前人工智能和教育信息化领域的研究热点。在国内，随着自然语言处理技术的不断发展，相关研究取得了显著进展。许多高校研究机构及企业纷纷投入到这一领域的研究和实践中。在国外，也有很多相关研究正在进行，特别是在一些发达国家的大学和科研机构。

在自然语言处理技术方面，国内的研究者在语义理解、信息抽取、自然语言生成等领域取得了突破。例如，自然语言处理中的词嵌入模型如 Word2Vec 和 BERT 等的迅速发展，为学者们提供了更加丰富的语义表示方法，使得系统能够更好地理解自然语言。同时，中文自然语言处理技术也取得了长足的进步，为校园智能问答系统的设计与实现提供了有力支持。

在校园智能问答系统的应用实践方面，国内一些高校和科研机构已经开展了相关研究与实践。这些系统通常整合了自然语言处理技术和知识图谱，能够回答校园相关的问题，如课程查询、教学资源、校园活动通知等。然而，在实际应用中，仍然存在着对于复杂问题的处理能力不足、对上下文的理解和用户意图的识别等方面还有待提高。

国外在自然语言处理领域也取得了重要进展。例如，句法分析、语义理解等技术的进步为校园智能问答系统的设计与实现提供了基础。在校园智能问答系统的应用实践方面，国外的一些高校和研究机构也开展了相关研究和项目。这些系统通常整合了自然语言处理技术、知识图谱和机器学习算法，能够回答

校园相关的问题，如课程查询、学术资讯、校园活动通知等。

目前根据应用领域的不同，智能问答系统一般分为面向开放域的问答系统和面向限定域的问答系统。面向开放域的问答系统拥有丰富的知识库，能够回答多个领域的问题，但其在回答专业领域的问题时难以精准定位答案、表现较差。目前已有较为成熟的针对法律、医疗、金融等限定领域的智能问答系统的研究，但针对高校领域的问答系统研究仍处在起步阶段。

1.4 论文主要工作安排

本文主要的研究项目是基于自然语言处理的校园智能问答系统的设计与实现，通过结合知识图谱、余弦相似度和 Flask 技术，来实现校园生活中各种问题的回答。

本文用七章内容对该研究进行详细阐述，大体内容结构如下：

第一章，绪论。本章要介绍基于自然语言处理的校园智能问答系统的研究背景、目的以及意义，同时介绍当前该领域在国内外的发展现状。最后对论文大体组织架构进行阐述。

第二章，智能问答系统相关技术理论及总体设计方案。这一章主要大体介绍了本论文工作所用的理论知识，包括知识图谱的相关理论、AC 自动机的详细介绍。同时本文使用了词向量对文本进行预处理，本章也对文本预处理进行了详细介绍。最后介绍了整体的设计方案。

第三章，基于知识图谱的智能问答系统设计。本章介绍了如何构建一个校园智能问答系统的语料库，并且使用 neo4j 将其存储到知识图谱中。然后介绍了如何将用户的提问与知识图谱中存储的数据进行多模式匹配，以得到答案。

第四章，基于余弦相似度匹配问题。本章主要包括两部分，首先将用户提出的问题数据进行数据预处理，主要是去除停用词并将其转换为词向量形式。然后将词向量与语料库中各个问句的词向量进行余弦相似度计算，匹配出相似度最高的问句，返回答案。

第五章，智能问答系统的实现与终端显示。本章主要是介绍如何借助 Flask 和 HTML 为系统搭建前端界面，让用户有更好的交互体验。

第六章，系统测试。本章对前面所做的模块进行测试，主要为了展示系统的安全性、稳定性和可读性。

第七章，总结与展望。本章对全文进行总结概括，对本系统目前的不足和优势进行分析，并对该模型项目的后续改进工作进行展望。

2 智能问答系统相关技术理论及总体设计方案

2.1 知识图谱实现原理

知识图谱是一种利用图模型来呈现知识和建模世界各个事物之间关系的技术方法。这种技术由节点与边构成，节点可以代表实体，如个人、图书等，或者抽象概念，例如人工智能、知识表示等。而边则可以代表实体的属性，例如姓名、书籍名称，或者实体之间的关联，例如友谊、婚姻关系。知识图谱最初灵感源自 Semantic Web（语义网络），旨在将基于文本链接的万维网转变为基于实体链接的语义网络。通过知识图谱，自然语言处理任务能够获得丰富的语义信息和背景上下文。借助知识图谱，自然语言处理系统可以更有效地解析语义、实体链接和问答系统等任务，以提升文本数据处理的效率和准确性。

在本研究中，计划运用知识图谱的方法来开发智能问答系统。知识图谱本质上是结构化的语义知识库，能够将问答信息关联为关系，汇聚成知识，使得问答信息更易于理解和处理，实现快速响应。作为一个小规模专用系统，校园智能问答系统可以采用结构化数据库，并自顶向下构建知识图谱。另外，采用图数据库进行存储能够提供更直观的展示和更便捷的使用体验。因此，本研究选择利用 neo4j 来构建与校园问答相关的图数据库。

实现知识图谱的过程通常包括以下步骤：

1) 数据抽取与整合：从各种数据源中抽取信息，包括结构化数据（如数据库）、半结构化数据（如 XML、JSON）和非结构化数据（如文本、图片、音频等）。这些数据涵盖了各种领域的知识，需要经过整合和清洗，以建立统一的信息体系。

2) 实体识别与链接：识别数据中的实体并将其链接到已知的实体标识符（如维基百科页面、数据库记录等）。这有助于建立实体之间的关联。

3) 关系抽取：分析文本或结构化数据，抽取实体之间的关系信息。这可能涉及自然语言处理和机器学习技术，以提取关系并确定其类型。

4) 属性提取：从数据中获取实体和关系的属性信息，这有助于丰富知识图谱的内容。

5) 知识表示：将抽取的信息以图的形式进行表示，形成具有实体-关系-实体结构的图谱模型。

6) 图数据库存储：将构建的知识图谱存储在图数据库中，如 Neo4j、Amazon Neptune 等，以便于图查询和分析。

7) 查询与推理：利用知识图谱进行查询、推理和分析，支持自然语言处理、

智能问答等应用场景。

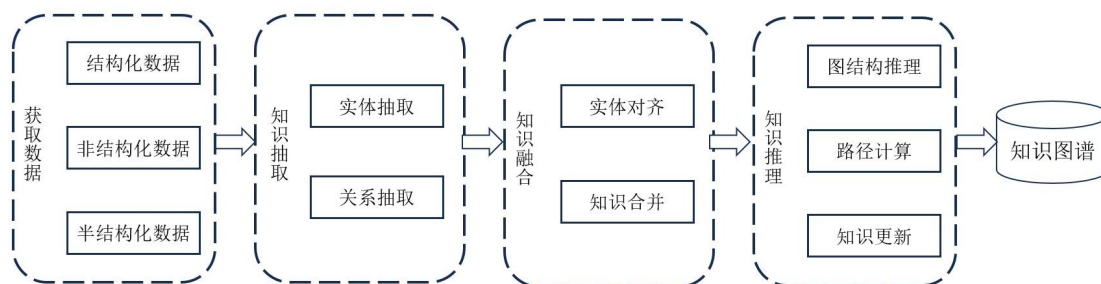


图 知识图谱的实现过程

2.2 AC 自动机

自动机，一般指“确定有限状态自动机”，是一个对信号序列进行判定的数学模型。“信号序列”指的是一连串有顺序的信号，例如字符串从前到后的每一个字符、数组从 1 到 n 的每一个数、数从高到低的每一位等。“判定”指的是针对某一个命题给出或真或假的回答。有时我们需要对一个信号序列进行判定。一个简单的例子就是判定一个二进制数是奇数还是偶数，较复杂的例子例如判定一个字符串是否回文，判定一个字符串是不是某个特定字符串的子序列等等。

而 AC 自动机，是以 Trie 树结构为基础，结合了 KMP 思想建立的自动机，用于解决多模式匹配的任务。

2.2.1 Trie 树

Trie 树，也称为前缀树或字典树，是一种有序树，用于保存关联数组，其中的键通常是字符串。不同于二叉查找树，Trie 树中键值并不直接保存在节点中，而是由节点在树中的位置决定。一个节点的所有子孙都有相同的前缀，也就是这个节点对应的字符串，而根节点对应空字符串。通常情况下，并非所有节点都有对应的值，只有叶子节点和部分内部节点所对应的键才有相关的值。

AC 自动机在初始时会将若干个模式串存放在 Trie 树中，然后在 Trie 树上建立 AC 自动机。Trie 中的结点表示的为某个模式串的前缀，在后文将其称作状态。一个结点表示一个状态，Trie 的边即为状态的转移。

图为一颗由字符串 $\{i, his, she, hers\}$ 组成的 Trie 树。

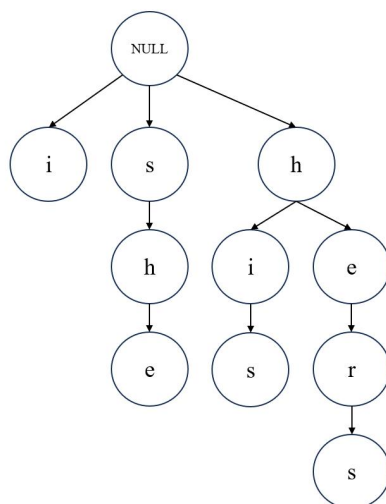


图 Trie 树示例

由上图可归纳出 Trie 树的基本性质：

- 1) 根节点为空不包含字符，除根节点外的每一个子节点都包含一个字符。
- 2) 从根节点到某一节点路径上经过的字符串起来，为对应的字符串。
- 3) 每个节点的所有子节点包含的字符互不相同。
- 4) 从第一字符开始，若有连续重复的字符，则其只占用一个节点，如 he 和 his 中重复单词 h 只占用了节点。

2.2.2AC 算法

AC（Aho-Corasick）算法是由 Alfred V.Aho 和 Margaret J.Corasick 于 1975 年提出的。该算法通过为模式集构造确定性有限状态自动机（DFA），然后将文本串中的字符逐一输入到自动机中进行状态转移，从而实现多模式匹配。

设有一个长度为 m 的文本串 $S = c_1c_2...c_m$ ，和由 n 个模式串组成的集合 $P = \{p_1, p_2...p_n\}$ ，使用 AC 算法可在 $O(m)$ 时间复杂度下查找到文本串 S 里面所有属于 P 的元素，查找效率与 n 无关。AC 算法由状态转移函数（goto 表）、失效转移函数（fail 表）以及输出函数（output 表）组成，它们的具体功能如下表。

表 AC 算法所使用的函数（表）及其功能

函数（表）名	功能
状态转移函数（goto 表）	由模式串集合中所有的元素构成的状态转移自动机。输入文本串字符时，可根据 goto 表来找到将要转移到的状态。
失效转移函数（fail 表）	goto 表中匹配失败后状态跳转的依据。当 goto 表的下一个状态无效时，可根据 fail 表来找到应该退回的状态。

输出函数（output 表）	抵达终止状态时将成功匹配的模式串输出。
----------------	---------------------

2.3 文本预处理技术

文本预处理在自然语言处理任务中占据首要地位，因为机器语言和自然语言之间存在巨大差异，为了使计算机能够理解自然语言，必须对输入的文本进行一系列处理，以将自然语言转换为计算机可理解的形式。

2.3.1 中文分词技术

分词是文本预处理的初始步骤，也是关键的一环。英文分词技术主要基于单词间的空格分隔，在目前已相当成熟。相较于英文分词，中文分词更为复杂。由于中文的多样性及复杂性，在不同语境下，同一句话可能产生不同的分词结果，这对计算机的理解提出了挑战。当前使用的中文分词工具包括 jieba 分词和 Hanlp 分词。在本文中，主要采用 jieba 分词技术。这一分词方法主要利用加载词典和正则化处理文本，将其分隔成语句，并通过字符串匹配生成所有可能的分词情形的有向无环图（DAG）。最后计算每个汉字节点到语句末尾的路径概率，并记录最高概率路径的有向无环图，以确定分词结果。未包含在词典中的词语将依赖隐式马尔可夫模型进行处理。jieba 分词的显著优点在于能够添加自定义词典，适用于特定领域的分词需求。

jieba 分词提供三种分词模式：精确模式可精准切分句子而不产生冗余单词；全模式扫描句子中所有可能的词语，可能存在冗余；搜索引擎模式在基础精确模式上对长词语再次进行切分。

2.3.2 词向量

为了让计算机能够理解自然语言，我们将其转换为向量形式。词向量在文本表征中扮演着关键角色，适当的词向量转换对于后续的文本分类至关重要。最初常用的词向量方法是独热编码（one-hot 编码），其原理是将词语表示为长向量，其中向量的维度为句子中包含的词语数量，其中仅有所表示的词所在维度为 1，其他维度则填充为 0。尽管独热编码较好地表示了词语，但由于未考虑词语顺序，因此无法表达词语之间的关系。此外，对于一些长语句或文本来说，独热编码可能导致特征空间过大，出现维度灾难。鉴于独热编码的局限性，对于长度较大的文本需要的维度过大，且利用率不高，会造成计算资源浪费。为了解决这些问题，提出了 Word2Vec。

Word2Vec 是一种通用的词向量技术，是一种通过无监督学习语义知识的模型，其学习过程涉及大量文本语料。Word2Vec 采用分布式表示方法，其思想是通过对词语上下文进行表示，因为上下文相似则语义也相似。相对于独热编码这种分布式表示方法，Word2Vec 可以实现更低维度的表示。它包括了 CBOW（Continuous Bag-of-words，连续词袋模型）和 Skip-gram（连续跳字模型）两种结构。在 CBOW 模型中，通过已知目标词的上下文来预测目标词。

对于 CBOW 模型，若给定语料组 $\{t_1, t_2, t_3, \dots, t_R\}$ ，目标词语为 t_r ，设上下文窗口为 1，则其后验概率为 $P(t_r|t_{r-1}, t_{r+1})$ ，损失函数为：

$$L_{CBOW} = -\frac{1}{R} \sum_{r=1}^R \log P(t_r|t_{r-1}, t_{r+1}) \quad (2-1)$$

Skip-gram 与 CBOW 不同，是通过目标词来预测其周围多个词（即上下文）。Skip-gram 的输入层只有一个，输出层数量与上下文的词语相同。若将 Skip-gram 模型建模为 $P(t_{r-1}, t_{r+1}|t_r)$ ，则它的损失函数为：

由于 Skip-gram 模型的预测次数与上下文词语数量相同，其损失函数对应于上下文损失的总和，而 CBOW 模型只需要计算目标词的损失。因此，CBOW 模型在速度上比 Skip-gram 模型更快，但就词语的分布式表示准确度而言，Skip-gram 模型的效果要优于 CBOW 模型。特别是在语料库内容增加的情况下，Skip-gram 模型通常比 CBOW 模型表现更出色。

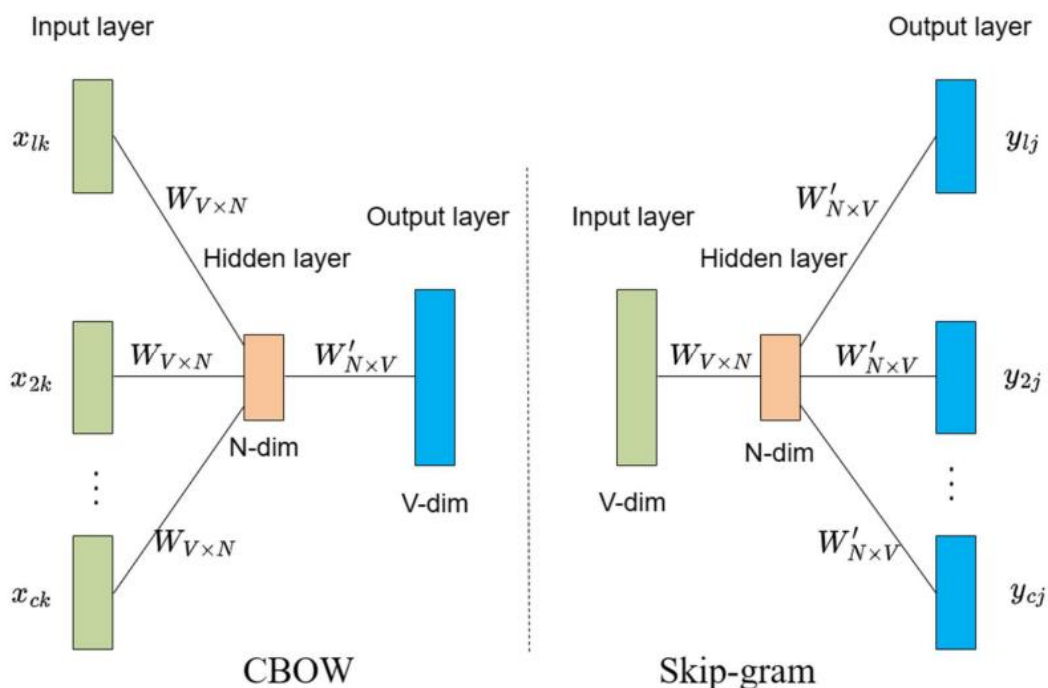


图 Skip-gram 模型和 CBOW 模型的结构图

2.4 预训练模型

由于自然语言处理（NLP）相关任务每个任务所需的数据和模型各不相同，对数据的需求也各不相同，大多数 NLP 任务需要大规模的标记数据，这是 NLP 任务面临的一大挑战。然而，自 2013 年提出 Word2Vec 以来，自然语言处理预训练的时代已经到来，Transformer、BERT、GPT 等预训练模型相继问世，为自然语言处理任务带来了显著的效果。

预训练模型的核心思想是首先在大型数据集上进行训练，得到一套模型参数，然后通过微调将其应用到特定的下游任务，这样即使在使用小型数据集进行训练时，也能取得不错的效果。例如，目前广受欢迎的 ChatGPT 模型采用了 GPT-3.5 大规模语言模型引入强化学习对预训练语言模型进行微调，在日常对话、信息检索等方面具有强大的功能，甚至可以编写代码，几乎可以完全理解人类语言，展示了预训练模型在 NLP 领域的良好应用。

2.5 总体设计方案

本课题旨在设计实现一个校园智能问答系统，主要包括：

（1）收集数据建立校园问答语料库，对收集到的数据通过分词处理、词向量转换等方法进行处理并存储；

（2）建立合适的模型，对数据进行处理。训练模型，在模型建立过程中多次调整参数和测试集训练集，以找到最佳相似度匹配的模型。让用户的问题能够最高准确度的与语料库中的问题匹配，更准确的输出答案；

（3）用户界面前端的设计，本课题拟将问答系统部署在网页上进行交互流程，最后实现完整系统，并对系统进行测试。

本系统作为一个面向限定域的问答系统，语料库数据量较小，但需要更快的响应时间。故提出了使用知识图谱查询和余弦相似度匹配两种方法结合的设计，更高效准确的给出回应。同时由于是小型的项目，选择使用 Flask 来构建问答系统前端。

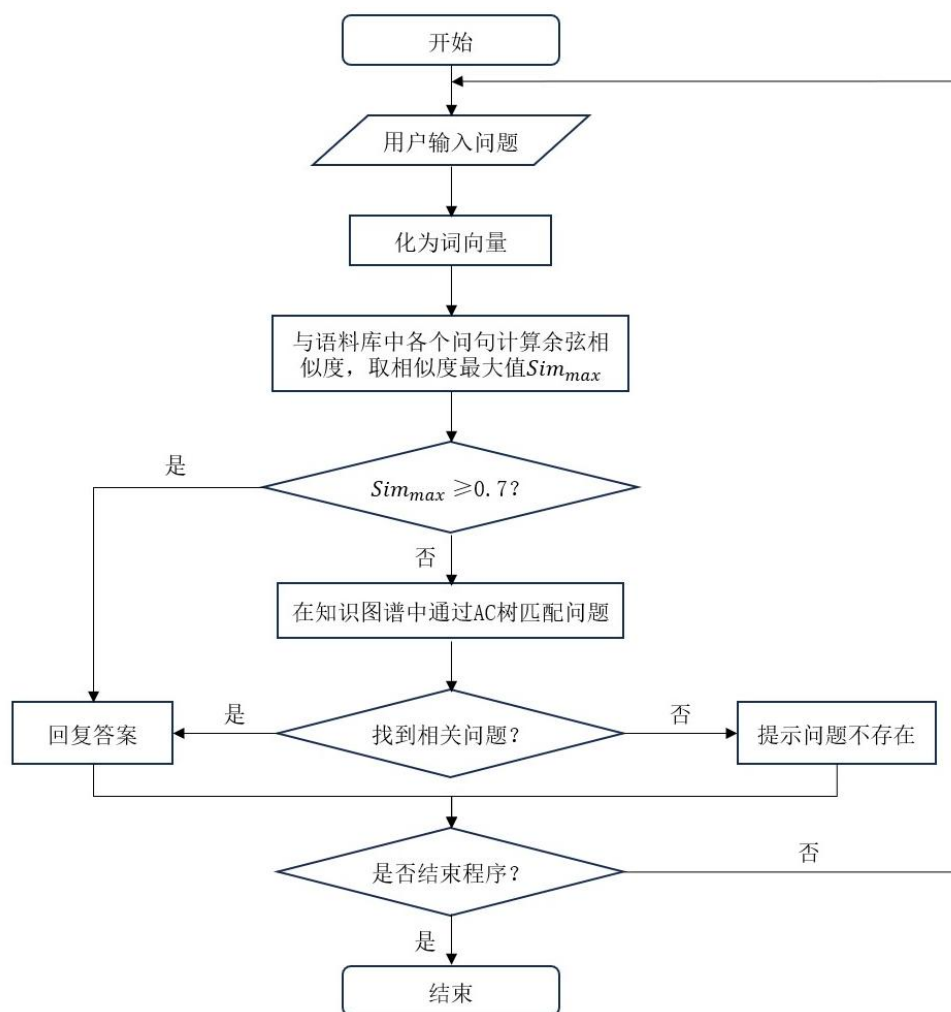


图 1 校园智能问答系统项目流程图

本项目已实现基本的校园问答系统，流程图如上。基本流程为：根据用户所输入的问题，首先进行余弦相似度匹配计算，若能匹配到的最大相似度大于等于 0.7，则直接输出所匹配问题的答案；若小于 0.7，则通过知识图谱的方法来多进行多模式匹配，找到对应问题并输出答案。若仍找不到匹配的问题，则返回提示，提示用户问题不存在。

本项目已实现基本的校园问答系统，项目总体框架如上。基本处理流程为：

1) 通过收集西安工业大学校园相关常见问题和答案，来构建语料库。同时使用 neo4j 来建立知识图谱。

2) 用户通过前端界面提出问题，传入后端进行响应。对于用户所输入的问题，先将自然语言处理成机器能理解的语言，把问题去除停用词后转换为词向量。首先进行余弦相似度匹配计算，若能匹配到的最大相似度值大于等于 0.7，则直接输出所匹配问题的答案；若小于 0.7，则通过使用知识图谱的方法，借助 AC 树来多进行多模式匹配，找到对应问题并输出答案。若仍找不到匹配的问题，

则返回提示，提示用户问题不存在。

3) 系统后端得到答案后，返回到前端界面显示。同时用户可以继续进行提问、获取答案的过程，直至用户关闭浏览器。

2.6 本章小结

本章是对整个项目的技术框架的介绍。首先介绍了自然语言处理的一些重要技术，包括知识图谱构建和 AC 自动机的构建。在文本预处理方面，对中文分词、词向量技术做了详尽阐述。其次，对将要用到的预训练模型的构建和原理进行了深入分析。同时本章节大体介绍了项目的整体框架流程，为后续讨论搭建问答系统的实现奠定了基础。

3 基于知识图谱的智能问答系统设计

3.1 校园问答语料库的构建及存储

本项目通过本人及身边同学的日常问题，结合西安工业大学学生手册，初步构建了一个小型语料库，保存为 csv 格式。

要构建知识图谱，首先要对其逻辑架构进行分析。知识图谱通常分为两个层次：模式层及数据层。模式层位于数据层之上，存储经过提炼的数据；而数据层即存储着真实数据。

以下为例例：

表 3.1 “图书馆进入方式”问题的逻辑架构

问题及回答	怎么进图书馆？ ----刷校园卡进入		
模式层	实体	属性	性值
数据层	图书馆	进入方式	刷校园卡进入

将所有问答按此逻辑进行知识图谱构建，实体需要人工分辨，并赋予属性，性值即为对应问题答案。

（1）信息抽取和存储。首先将各个问题和回答按逻辑架构进行划分，对于提出的问题，主要由实体（关键词）和属性（意图疑问词）组成，本项目目前主要将问题划分成八个实体及其若干属性。

表 3.2 八个实体及其对应若干属性

实体（关键词）	属性（意图疑问词）
校园卡	丢失、充值、损坏、用处.....
图书馆	开门时间、借书、找书、还书.....
毕业	要求、退学、学业警告.....
课程	主修、重修、免听、免修
考核	考核方式、不允许考试、缓考、作弊、绩点.....
餐厅	数量、位置、开放时间、支付方式.....
教学事务	转专业、体测、购买教材、量化评教.....
选课	选课方法、选错、班级课表、冲突、上限.....

至此，每个问题及答案逻辑架构大致分为：问题（实体+属性）+答案（性值）。

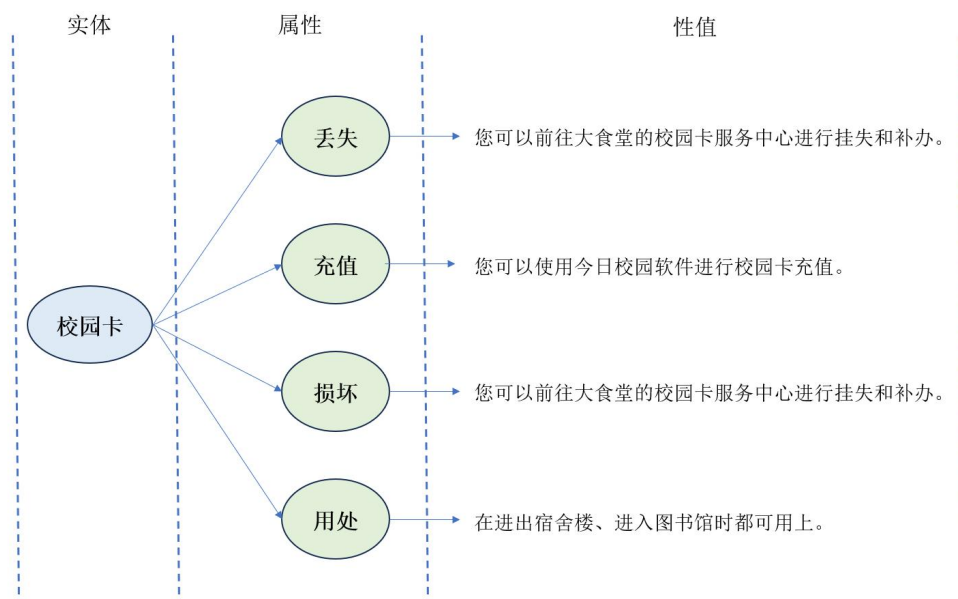


图 3.2 划分为“校园卡”实体的具体逻辑架构

将每个实体及其所有属性、性值，各放入一个表格中暂存，在 neo4j 中使用 CQL 语句进行数据的导入。如在导入校园卡全部信息时运行：

```
LOAD CSV WITH HEADERS FROM 'file:///1CampusCard.csv' AS row
MERGE(n:校园卡{name:row.name,丢失:row.lost,充值:row.recharge,损坏:row.damaged,用处:row.w_to_use})
```

图 3.3 使用 CQL 语句向 neo4j 中导入数据

依此，将所有信息写入图数据库，即可得到有全部实体和其属性、性值的图数据库：

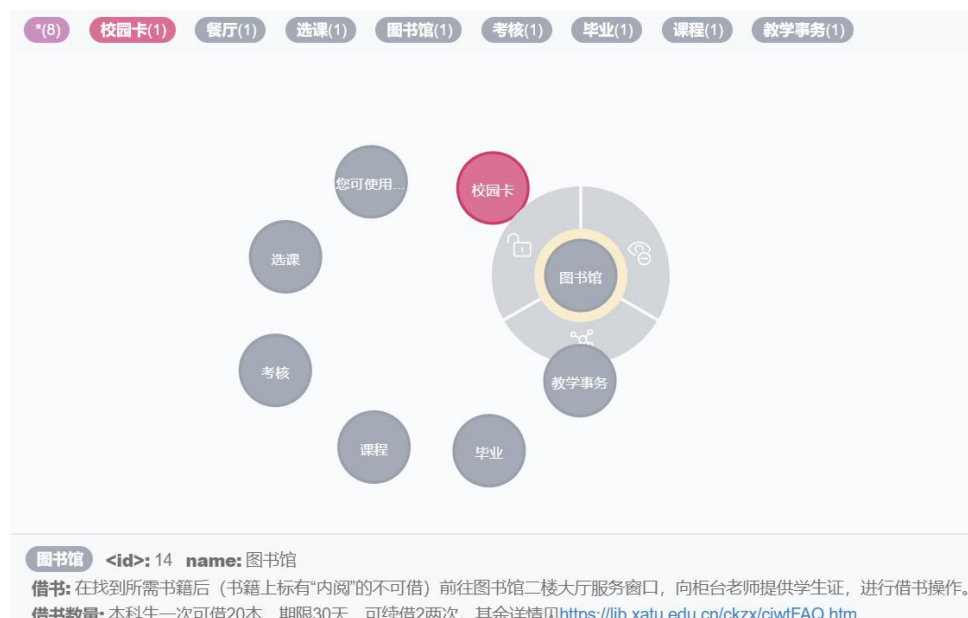


图 4 存储好的图数据库展示

（2）知识融合。在将知识全部存入数据库后，需要对其进行整合，来消除矛盾和歧义。比如，某些实体可能会有多种表达（如“校园卡”也可称其为

“学生卡”、“一卡通”、“水卡”等等)；某些属性也会有多种表示（如“损坏”也可表达成“坏掉”、“污损”等）。

实现知识融合的方法，是在建立实体节点和属性特征疑问词时将可能出现的同义词、同义疑问词包含进去。

```
self.campuscard = ['校园卡', '饭卡', '学生卡']#学生卡
self.canteen=['餐厅', '食堂']#食堂
self.select_lesson=['选课', '选错课', '']#选课
self.library=['图书馆', '阅览室']#图书馆
self.assessment=['考核', '考试', '成绩', '缓考', '旷考', '作弊']#考核
self.graduation=['毕业', '学业警告', '退学']#毕业
self.lessons=['主修', '重修', '免听', '免修']#课程
self.affairs=['转专业', '教务系统', '评教', '学生评教', '成绩单', '在读证明', '在校证明', '体测']#教学事务

#学生卡
self.campuscard_lost=['丢失', '丢了', '找不到']
self.campuscard_recharge=['充值', '存钱', '充钱']
self.campuscard_damaged=['损坏', '坏掉', '折损']
self.campuscard_wtouse=['用处', '用在哪里']
```

(3) python 与 neo4j 链接。上述工作已完成了数据在图数据库中的初步存储和知识融合。接下来是要将图数据库中的数据取出放入 python 中进行系统的下一步实现，只需使用 py2neo 包，初始化端口号、IP 地址、用户名、密码与建立的 neo4j 数据库相同，即可成功链接。

3.2 在知识图谱中匹配用户提出的问题

在构建知识图谱的过程中，将问题划分为了关键词（实体）和意图疑问词（属性）两部分。接下来就是要使用算法匹配出用户提出的问题对应的关键词和意图。

3.2.1 使用 AC 树匹配关键词（实体）

AC 树（Aho-Corasick Algorithm），是一种基于 AC 自动机的原理，利用 Trie 树的结构来存储模式集合，并在此基础上构建自动机来实现高效的多模式匹配算法。此项目在构建完知识图谱后，利用 AC 自动机算法，匹配问句中是否存在特征词，即查询问句中是否存在知识图谱实体名字，来提取实体。

AC 自动机用于在输入的一串字符串中匹配有限组“字典”中的子串。它与普通字符串匹配的不同点在于同时与所有字典串进行匹配。鉴于此，在使用 AC 树前要建立将关键词转换为问题类型的字典，形如{"重修": [课程]}的字典。

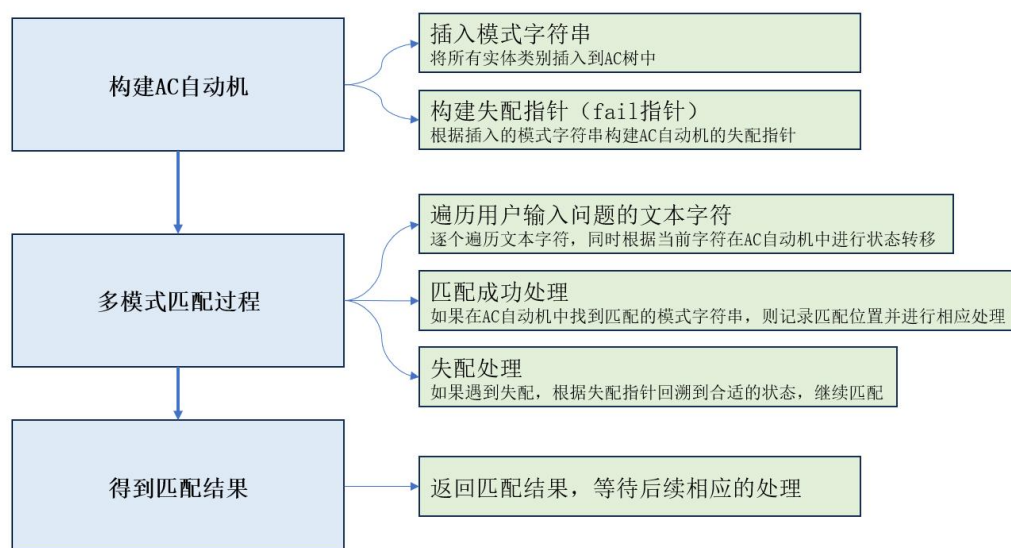


图 7 ACTree 实现多模式匹配流程

借助于 AC 树，可以提取出用户所提出问题的关键词（实体）。同时在提取出关键词后要进行过滤去重，防止用户多次输入相同实体名称导致的异常。将关键词返回，用于后续操作。

3.2.2 使用模板匹配得到意图疑问词（属性）

在匹配意图关键词时，主要使用了循环逐一匹配的方式。

如，在用户提出问题“食堂在哪里”时，根据前面 AC 树的匹配，已知此问题属于“餐厅”实体。对于餐厅实体，其对应的全部意图特征词列表如下：

```

#食堂
self.carteen_amount=['数量','多少个','几个']
self.carteen_location=['位置','哪里','在哪儿']
self.carteen_openingtime=['时间','开门','关门','开放时间']
self.carteen_payway=['支付方式','付款','支付','刷卡']
  
```

图 8 “餐厅”实体对应的全部特征词列表

将上述列表依次与用户提出的问题模板匹配。

输入： words={意图特征词列表};

question= ‘用户提出的问题’

过程： 函数 check_words(words,question)

1: **for** words 中的每一个词语

2: **if** 词语在 question 中 **then**

3: **return** True

4: **else return** False

5: **end if**

6: **end for**

输出： 逻辑 T 或 F

图 9 模板匹配函数基本算法

得到问题中的“哪里”与“location”意图列表相匹配，故返回{"位置":[餐厅]}所对应的答案。

3.3 本章小结

4 基于余弦相似度匹配问题

对于和数据库中的问题几乎相同的用户输入，借助知识图谱查询较为复杂，效率较低。故加入了余弦相似度匹配算法，优先处理相似度大的问题。

4.1 数据预处理

（1）分词处理：词是汉语中最基本的语义单位，分词主要是将原先没有分割符的中文语句（例如“校园卡丢失怎么办”）按照规定的划分原则拆分其中的字或词（“校园卡/丢失/怎么办”）的过程。项目使用了 jieba 对数据进行中文分词。

（2）去除停用词：停用词是指在自然语言处理中被认为对文本分析无实际意义或对文本特征提取过程无贡献的词语。这些词通常是高频出现的、在不同文本中普遍存在的词语，如“的”、“是”、“在”、“有”等。在文本分析中，去除停用词有助于减小特征空间、提高模型效能、和改善文本的可读性。本项目使用了百度停用词表和中文（CN）停用词表结合的停用词表。

（3）接着基于 Word2vec 模型对知识库中的问题数据构建词向量集。在训练 Word2vec 模型时，本项目将词向量维数定为 128，训练轮数为 10000。训练后得到的词向量保存于 bin 文件内便于后续使用。

4.2 模型训练结果

模型使用了 gensim 依赖包中的 Word2vec。训练所使用的参数中 size 为特征向量的维度，默认值 100 维，为了避免词映射冲突导致影响结果，将其设为 128；sg 用来选择 CBOW 算法还是 Skip-gram 算法，sg=1 对应 Skip-gram 算法，sg=0 对应 CBOW 算法，这里使用 CBOW 算法；window 为窗口大小，表示当前词与预测词在句子中的最大距离；min_count 表示对字典进行截断，当词频小于设置的 min_count 值时则丢弃该词语；workers 是训练的并行数；iter 指随机梯度下降法中迭代的次数。训练词向量使用的训练参数如表所示。

表 词向量训练参数

参数	取值
min_count	1
workers	8
size	128

window	3
iter	5

最后将训练好的结果存入 word2vec_XATU.bin 文件中。至此，模型训练结束，已得出结果数据。

word2vec_XATU.bin	
1	486 128
2	学 -0.6174554 0.09271708 -0.6769255 -0.8507471 0.23571786 -2.330217
3	课 -0.32476947 -0.30345938 0.06037415 2.5949614 1.5634483 -0.985290
4	程 -1.5818284 -1.44329 -1.8144706 0.7018224 0.370088 -0.6026433 -1.
5	生 1.4709319 -1.0464717 1.2146672 -2.6021445 2.4028661 -1.4250637 0
6	考 -0.21316826 -3.2734406 -0.5965273 0.23021406 -0.42298877 0.29808
7	教 0.5613308 -3.4136472 -0.31103474 -1.0256546 1.580775 1.3010883 0
8	业 3.9500923 -0.989989 -0.12881821 -0.26001006 -1.1148667 0.1578259
9	绩 -0.89728 0.6168787 -0.374489 0.38787583 2.1418164 -3.6674976 1.9
10	书 -0.9804251 0.7799979 1.3461083 -1.860065 0.16821383 -1.3589975 -
11	校 4.2535596 -2.160329 -4.9734387 0.080484636 -2.6133015 0.13987252
12	成 0.088268004 -0.428589 -1.0398328 -2.135145 1.2699932 -0.9217348
13	选 2.1149294 -2.5775723 1.1234779 0.9573757 -0.82393867 -0.60170394
14	分 -0.37772632 1.0780634 -0.2626458 0.41355136 -0.029877586 -2.0168

图 查看词向量

4.3 余弦相似度的计算

将句子向量定义为字符向量求平均。对于每个文本组合(q_{user}, q_{sys})，其中 q_{user} 为用户提出的问题的句子向量， q_{sys} 为校园语料库库中的候选问题的句子向量，词向量余弦相似度计算公式如式(1)所示。

$$Similar_{text}(q_{user}, q_{sys}) = \frac{vector(q_{user}) \cdot vector(q_{sys})}{|vector(q_{user})| \times |vector(q_{sys})|}$$

通过和语料库中每个问句进行相似度计算，得到相似度最大的问句。若相似度计算值大于 0.7，则直接返回该问句的答案，否则进入知识图谱查询阶段。

4.4 本章小结

5 智能问答系统的实现与终端显示

5.1 使用 Flask 实现问答终端

本文在完成问答系统整体算法实现后，使用了 Flask 技术将其部署在网页上。Flask 是一个轻量级的 Web 应用程序框架，基于 Python 编程语言。Flask 被设计为简单、灵活的框架，使得开发 Web 应用变得更加容易。Flask 使用了模块化的设计，允许开发人员根据需求选择并整合不同的功能。同时其使用 Jinja2 模板引擎，Werkzeug 作 WSGI 工具箱。

Flask 使用的 Jinja2 是很流行的模板引擎，用来生成 HTML 内容，使得页面呈现更加灵活且可定制。同时为了让 Web 界面更美观流畅，使用了 Javascript 和 CSS 来实现界面样式的设计。Werkzeug 作为 WSGI 中的一个工具包，用于实现请求并响应。WSGI 是 Python 语言下定义的 Web 服务器和 Web 应用程序之间的一种简单而通用的接口，它将 Web 服务划分为服务端和应用程序两部分。WSGI 服务器只承担与网络相关的两个任务：接收来自浏览器的 HTTP 请求和向浏览器发送 HTTP 响应；而具体处理 HTTP 请求的逻辑则通过调用 WSGI 应用程序来实现。WSGI 的运行方式如下图所展示。

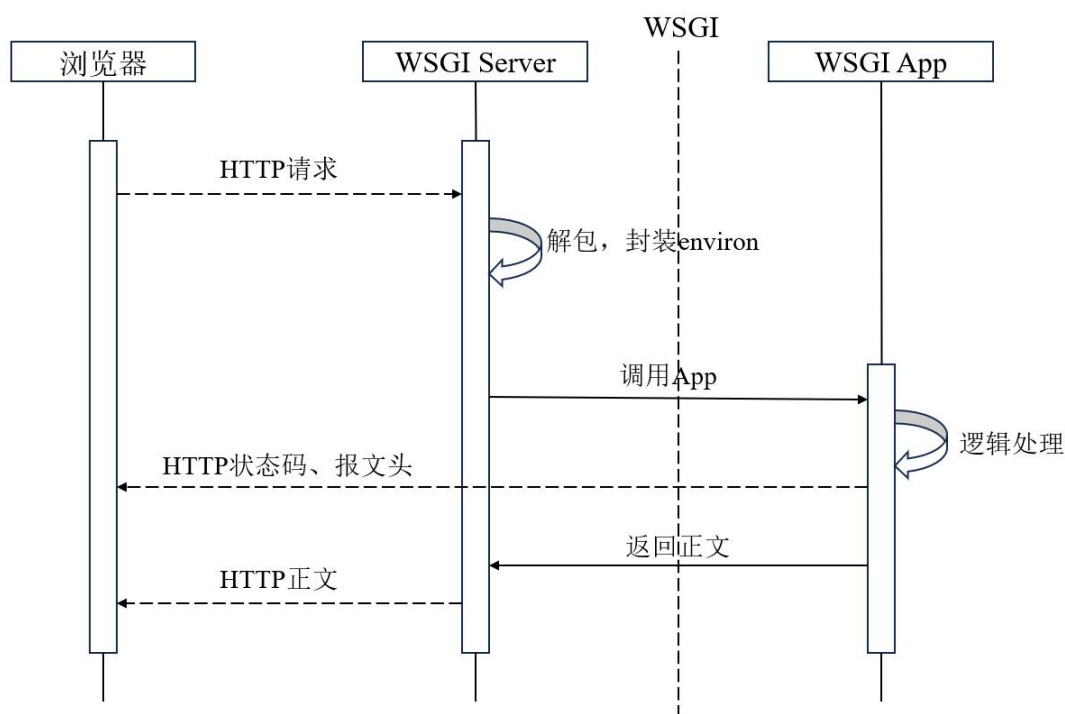


图 WSGI 运行方式

Flask 构建前端的流程具体如下图所示。

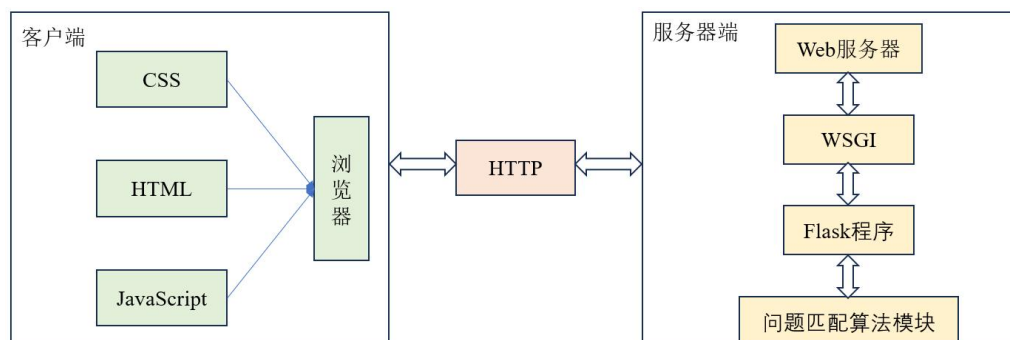


图 Flask 工作流程图

在 Flask 中，我们处理一个请求的流程为：

- 1) 页面前端输入问题后朝向后台发送 http 请求；
- 2) 首先经过 WSGI Server 对其进行解析和封装，调用 WSGI App；
- 3) 通过 WSGI App 对请求进行逻辑处理，主要利用 `dispatch_request()` 和 `make_response()` 函数将 http 请求转化为 http 响应；
- 4) 最终将响应返回给 WSGI Server，然后由其发送至浏览器。

对于小型项目或需求较简单的应用，Flask 提供了一个简单而有效的解决方案。因此本项目选择使用 Flask 来构建问答系统前端。

5.2 智能问答系统前端设计

本文的智能问答系统界面参照常见的聊天问答界面设计，包括大聊天框、问答系统名称与图标展示、系统头像、系统聊天框、用户聊天框、用户输入框、提交按钮组成。同时为了增加交互性，在用户打开问答系统后，系统聊天框会有“欢迎使用智能问答系统，请输入您的问题”的提示语。前端设计主要使用 HTML 完成，包括静态文件和程序模块。

首先在项目目录下新建 `templates` 文件夹，用于存放程序模块。其中包括实现网页的特殊效果，如上下滚动、提交按钮、系统获取并显示答案等功能；同时新建 `static` 文件夹，用于存放静态文件，包括 `css` 文件和网页所用到的图片文件。其中 `css` 文件用于设计网页中各个元素的位置大小和形状颜色，图片文件用于丰富网页设计。

最后在程序模块中使用 JavaScript 编写功能函数，同时使用写好的 `css` 文件来部署网页。主要的功能函数与作用如下表。

表 5.2.1 主要功能函数及其详细信息

函数	详细信息
UserMessage(message)	<p>功能：用于将用户消息追加到聊天框中。</p> <p>步骤：</p> <ol style="list-style-type: none"> 1、获取 ID 为 chat-box 的聊天框元素。 2、创建一个新的 div 元素来包含用户消息。 3、添加 user-message 类到新创建的 div 元素中。 4、将消息内容添加到 div 元素中，使用字符串内嵌表达式 <code>\${message}</code>。 5、将包含用户消息的 div 添加到聊天框中。 6、将聊天框的滚动条位置设置为最底部。
SystemMessage(message)	<p>功能：用于将系统消息追加到聊天框中。</p> <p>步骤：</p> <ol style="list-style-type: none"> 1、获取具有 ID chat-box 的聊天框元素。 2、创建一个新的 div 元素来包含系统消息。 3、添加 system-message 类到新创建的 div 元素中。 4、在 div 元素中插入系统的图像和消息内容。 5、将包含系统消息的 div 添加到聊天框中。 6、将聊天框的滚动条位置设置为最底部。
askQuestion()	<p>功能：启动问答过程，发送用户提出的问题并显示系统返回的回复。</p> <p>步骤：</p> <ol style="list-style-type: none"> 1、获取具有 ID question 的输入框中的问题内容，并去除首尾空格。 2、如果问题为空，则不执行后续步骤。 3、将用户提出的问题追加到聊天框中，同时清空问题输入框内容。 4、使用 fetch 方法向服务器端发送 POST 请求，携带问题内容。 5、处理服务器返回的文本类型响应，将响应添加为系统消息显示在聊天框中。 6、捕获可能发生的错误并在控制台打印错误信息。

至此，一个完整的网页设计就已完成，后面只需使用 Flask 链接客户端和

服务器端即可使用完整网页。

5.3 问答系统的前端实现

Flask 大体部署流程如下图。

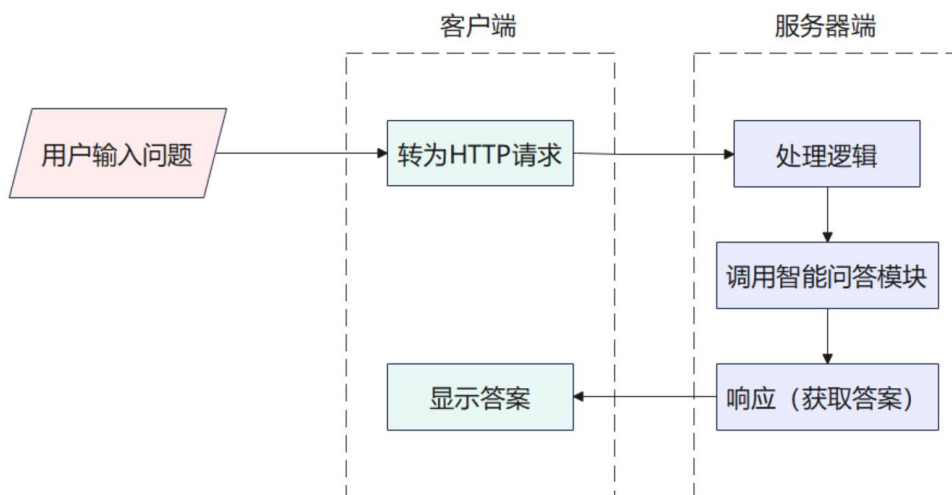


图 5. Flask 答题部署流程

使用 Flask 部署网站时要对端口地址等进行配置，下表为基本的参数设置。

表 5.3.1 Flask 参数设置

参数	描述
host	127.0.0.1
port	5000（默认端口）
debug	false（不提供调试信息）

明确参数设置后，开始创建 Flask 应用程序。首先创建一个 Flask 应用实例，然后创建路由。这里需要创建两个路由，一个用于渲染前端界面，另一个用于处理用户输入的问题。

在用于渲染前端界面的路由中，使用 `render_template()` 函数和模板来生成 HTML 页面。这样生成的好处，一是可以分离前端后端逻辑，使得代码更易于维护和管理；二是通过使用模板，可以轻松重用网页的结构和布局，减少了重复的 HTML 代码，维护和更新页面内容就会更加高效。使用 `render_template()` 函数直接调用设计好的 html 文件，即可显示网页界面。

另一个用于处理用户输入问题的路由需要完成的任务时获取用户输入的问题，然后调用智能问答系统的算法函数来获得答案。

最后调用 `run()` 函数来启动程序，在运行终端会得到一个 URL 地址，将该地址复制到浏览器打开即可得到对话系统界面，如下图。



用户刚刚进入系统还未提问时界面会显示提示语“欢迎使用智能问答系统，请输入您的问题”，问题输入框会提示用户“请输入您的问题”。用户输入自己的问题后点击绿色提交按钮，问题即会显示在浅绿色的用户聊天框中。等待 3s 左右系统会给出回复，显示在带有头像的系统聊天框中。最终的问答过程如下图所示。



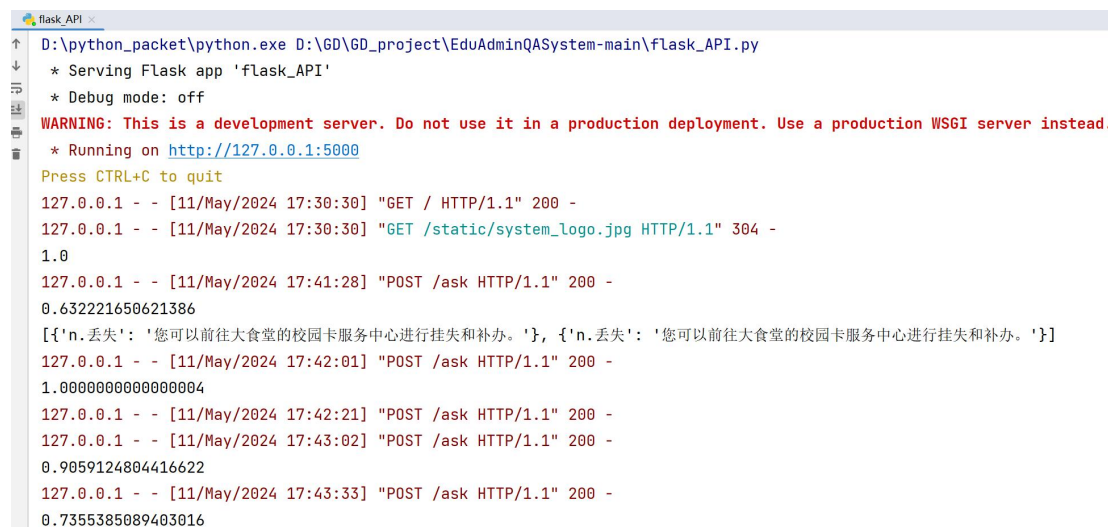
图 问答过程（1）



图 问答过程（2）

对于用户输入的每个问题，后端会通过 GET 的方法来接收问题，然后使用问题匹配算法模寻找答案并发送至前端，前端即会显示答案。本例同时展示了

答案匹配过程，显示出了余弦相似度的值与匹配到的知识图谱的数据。后端解析过程如下图。



```
flask_API x
D:\python_packet\python.exe D:\GD\GD_project\EduAdminQASystem-main\flask_API.py
* Serving Flask app 'flask_API'
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
127.0.0.1 - - [11/May/2024 17:30:30] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [11/May/2024 17:30:30] "GET /static/system_logo.jpg HTTP/1.1" 304 -
1.0
127.0.0.1 - - [11/May/2024 17:41:28] "POST /ask HTTP/1.1" 200 -
0.632221650621386
[{'n.丢失': '您可以前往大食堂的校园卡服务中心进行挂失和补办。'}, {'n.丢失': '您可以前往大食堂的校园卡服务中心进行挂失和补办。'}]
127.0.0.1 - - [11/May/2024 17:42:01] "POST /ask HTTP/1.1" 200 -
1.0000000000000004
127.0.0.1 - - [11/May/2024 17:42:21] "POST /ask HTTP/1.1" 200 -
127.0.0.1 - - [11/May/2024 17:43:02] "POST /ask HTTP/1.1" 200 -
0.9059124804416622
127.0.0.1 - - [11/May/2024 17:43:33] "POST /ask HTTP/1.1" 200 -
0.7355385089403016
```

图 Flask 后端解析信息

5.4 本章小结

本章节重点阐述了问答系统的前端开发过程，详细说明了采用的 Flask 框架以及其具体执行步骤，概括了利用 Flask 技术建立问答系统的基本原理。该系统利用前面所使用的答案匹配算法，将其部署在 web 上来实现校园相关信息的问答功能，可有效辅助学生与学校教职工的学习生活。

6 系统测试

本章节将对基于自然语言处理的校园智能问答系统的系统测试进行阐述。首先说明系统的测试内容和测试环境，随后介绍系统的主要测试要点和某些测试数据，大致包括功能测试和非功能测试两方面。一方面，验证已实现的功能服务是否符合最初需求，另一方面在上线前针对系统整体性能进行响应速度、可扩展性等方面的分析评估。最终，对系统所达成的功能及其运行效果进行简要展示。

6.1 测试内容与测试环境

6.1.1 系统测试内容

基于自然语言处理的校园智能问答系统的测试采用了主要以黑盒测试为策略的方法，对所有模块进行全面、规范、系统的测试，包括问题匹配问答模块、问答界面模块。本轮测试根据前述的功能性需求和非功能性需求来制定详细的测试环节和计划。在功能性测试过程中，针对各功能服务模块的内容设计不同的测试案例进行验证。测试案例执行完毕后，会发现其中的缺陷并根据其优先级进行修复，以最大程度地避免系统漏洞可能带来的损失。在非功能性测试过程中，利用相关工具和统计数据对整个软件系统进行性能测试、可靠性测试等环节，并进行统计分析与对比。最终完成对智能问答系统整体测试流程详细阐述。

6.1.2 系统测试环境

为模拟一般用户在食记使用问答系统时的情景，在系统测试的软件环境层面，使用了市面上较为常用的操作系统 windows10，数据库使用 Neo4j，客户端浏览器使用 Edge、chrome。同时硬件环境使用八核 CPU、i5 处理器、16G 内存以及 500G SSD 硬盘的配置。

6.2 功能性测试

6.2.1 问题匹配问答模块

本系统主要使用了知识图谱和余弦相似度值来匹配问题，从而得到答案，因此问题匹配模块为系统最重要的模块。该测试没有加入前端显示，更直观的显示出后端匹配问题的过程。此模块的部分测试用例如下表所示。

表 问题匹配问答模块部分测试用例

项目模块	问题匹配问答模块
测试类型	功能测试
测试者	管理员
测试目的	验证系统是否可以成功接收用户输入的问题，是否可以依据用户问题及时给出回复
测试步骤	(1) 进入问答系统界面，展示简单的输入提示 (2) 输入用户想要查询的问题 (3) 回车快速给出回复
预期结果	(1) 用户可以正常输入问题 (2) 能正确的算出匹配出的最大余弦相似度值并返回 (3) 若进入知识图谱查询，能返回查询到的问题的字典 (4) 若问题不在语料库中，返回失配提醒 (5) 一轮问答结束接着进入下一轮问答，循环进行
测试结果	测试结果成功通过

(1) 测试一：

欢迎使用智慧教务系统！

用户：毕业要达到哪些条件

0.6105300875165389

[{'n.毕业要求': '学生需在修业年限内修完专业培养方案规定课程，获得总学分不低于所学专业培养方案要求修读的最低学分，且同时达到以下'}]

系统回复：

学生需在修业年限内修完专业培养方案规定课程，获得总学分不低于所学专业培养方案要求修读的最低学分，且同时达到以下三个条件，方可毕业：

- 1) 获得第二课堂学分不低于7学分。
- 2) 参加国家学生体质健康测试且成绩合格。
- 3) 通过《Python语言程序设计》课程考核。

图 测试示例（1）

余弦相似度计算数值小于 0.7，故在知识图谱中进行匹配，得到最终结果与所需结果一致，通过。

（2）测试二：

用户：图书馆开门时间？
0.7749085700758748
系统回复 周一至周五：8：00--22：00；周六周日及法定节假日：9：00--21：00。

图 测试示例（2）

余弦相似度计算值大于 0.7，直接返回匹配的问题答案，得到最终结果与所需结果一致，通过。

（3）测试三：

用户：中期答辩在什么时候
0.4324983743204221
您的问题我们还要再向相关部门请教呢，请您过两天再来试试吧

图 测试示例（3）

此问题不在收集的语料库中，相似度值小于 0.7，且在知识图谱中匹配不到，故返回提示信息，与预期一致，通过

（4）测试四：

欢迎使用智慧教务系统！
用户：学生卡丢了怎么办
0.632221650621386
[{'n.丢失': '您可以前往大食堂的校园卡服务中心进行挂失和补办。'}, {'n.丢失': '您可以前往大食堂的校园卡服务中心进行挂失和补办。'}]
系统回复：
您可以前往大食堂的校园卡服务中心进行挂失和补办。
用户：可以转专业吗？
0.48230334556821874
[{'n.转专业': '1.修完第一学年课程的在读一年级全日制本科生，注册手续齐备，课程考核合格，符合拟转入专业当年公布的接受条件，可申请转专业。\\n2.学生有下列情况之一的，\\n'}]
系统回复：
1.修完第一学年课程的在读一年级全日制本科生，注册手续齐备，课程考核合格，符合拟转入专业当年公布的接受条件，可申请转专业。
2.学生有下列情况之一的，不予考虑转专业：
1)入学未满一学期者；
2)定向、委托培养、保送、体育特长生或其它特殊招生形式录取的学生；
3)因各种原因受到学校记过及以上处分，转专业申请尚未解除者；
4)上级主管部门及学校相关文件规定不予转专业的学生。
3.转专业的限制条件：
1)转专业原则上应在同一招生科类内进行，并且艺术、体育类转专业原则上应在同一专业类别内进行(且为加试科目相同的专业内)；
2)原则上不能由招生时所在地的下一批次录取专业转入上一批次录取专业，高考录取成绩低分转入高分专业；
3)上年度没有招生的专业不得接收学生转入；
4)学生在校期间只能转一次专业。
用户：怎么办理缓考？
1.0000000000000004
系统回复 1)因故不能按时参加考试时，必须在考试前1-3天办理缓考手续。课程开考后提交的缓考申请无效。缓考申请未批准或未申请缓考而擅自不参加考试的学生，一律按缺考处理。
2)缓考需在教务系统中办理，申请中应写明原因、考试时间并上传相关证明材料。
3)学生如因疾病本人无法办理的，可委托辅导员或同学代为办理。
4)学院审批、备案，本科生院(教务处)审核通过后，方可认为缓考手续办理成功，否则应按原计划参加考试。

图 测试示例（4）

在完成一轮问答后，可以正常进入下一轮问答，并重复循环进行，与预期结果一致，通过。

6.2.2 问答界面模块

问答界面模块面向用户，主要过程为用户在问答界面发出提问，系统相应

查询匹配到问题后，将相应答案显示给提问者。此模块的部分测试用例如下表。

表 问答界面模块部分测试用例

项目模块	问答界面模块
测试类型	功能测试
测试者	用户
测试目的	验证系统是否可以正常显示问答系统界面，用户是否可以正常输入问题，系统是否可以正确给出回复
测试步骤	(1) 进入问答系统界面，展示简单的输入提示 (2) 用户输入想要查询的问题 (3) 点击“提交”按钮将问题发送至后端进行解析 (4) 后端进行问题匹配，返回正确答案 (5) 返回的答案正确的显示在界面上 (6) 进行下一轮重复进行上述问答过程
预期结果	(1) 界面正常展示，美观简洁 (2) 用户可以正常输入问题并点击提交按钮 (3) 系统能正常完成问题匹配过程，且正确解决用户问题 (4) 用户提出新一轮问题成功 (5) 一轮问答结束接着进入下一轮问答，成功循环进行
测试结果	测试结果成功通过

(1) 测试五：



图 测试示例（5）

运行后打开浏览器界面显示正常，有提问的提示和输入问题的提示，系统 logo 和系统头像显现正常，通过。

（2）测试六：



图 测试示例（6）

用户可以正常在聊天框输入问题，并点击提交按钮发送问题。问题和系统给出的答案可正确显现在大聊天框中，且美观整洁、答案正确。通过。

（3）测试七：



图 测试示例（7）

用户可在辩论问答结束后顺利进入下一轮问答，并重复问答过程，直至结束关闭浏览器。整个过程界面显示正确、答案反馈正确，通过。

6.3 非功能性测试

非功能性测试涉及对校园智能问答系统进行性能、稳定性、保密性、可适应性等多个方面的测试。

系统性能测试：举例来看，问答界面模块作为系统主要对外显示界面，需要在后台查询后能够在较短实践内完成对用户提问的响应，一般在两到三秒。同时，以一百条以上的问答作为数据支持，达到 90%左右的准确率。另外，问题匹配问答模块的相应速度作为统计软件系统的性能测试指标，系统一般能够在 100 毫秒内完成响应，没有出现任何卡顿现象，同时需要完成聊天框、提示语、提交按钮等元素的显示。

稳定性测试：软件系统的稳定性主要从以下几个方面进行测试，包括系统连续运行时间、长时间停留在同一界面后进行跳转等操作、长时间停留在同一

界面且不执行任何操作。

保密性测试：校园智能问答系统，系统安全测试主要着眼于用户问答信息在传输过程中的保密性。因为用户对话的问答信息采用了多种加密方式进行传输，从而保证系统的安全性。

可适应性测试：此项测试主要用于检验系统是否能够在当前流行的多种浏览器中使用。通过使用 chrome，Firefox，IE 等终端进入系统，以及使用系统相关模块功能进行管理配置和用户问答，可以评判系统的可适应性。

6.4 本章小结

本章节首先概要介绍了智能问答系统的系统测试内容和测试平台。随后，详细说明了智能问答系统各个主要功能模块的测试案例，例如问答界面模块等。同时，完成了对该系统的非功能性测试，以确保软件能够正常运行。最后，简要呈现了智能问答系统的运行效果，特别着眼于系统的常见问答模式。

7 总结与展望

7.1 本文总结

本毕设项目致力于开发基于自然语言处理的校园智能问答系统，旨在为用户提供便捷、准确的问题解答，从而提升校园信息查询和服务体验。在项目实施过程中，深入研究和应用了知识图谱的方法，以及余弦相似度算法。通过此系统，用户可以以自然语言形式提出问题，系统能够将问题与知识图谱进行匹配，并提供相应的准确答案。同时，结合了余弦相似度算法，进一步提高了系统对复杂问题的回答准确性和智能化程度。此外，通过 Flask 框架连接 HTML，实现了用户友好的前端设计，使得用户可以以直观简单的界面进行交互。

7.2 未来展望

未来，随着人工智能和自然语言处理技术的不断发展，校园智能问答系统有着广阔的应用前景和发展空间。本项目只录入了一些简单问答，搭建了简单框架，仍有更多改进空间。

首先，可以对系统的知识图谱不断进行扩充和完善，包括各类学科知识和校园相关信息，以及招生、宿舍等信息。同时可以将其改进为通用的问答系统模板，管理员可以任意添加自己的问答条例，也可随时更新问答信息，以使系统对多领域知识的理解和应用能力更加全面。

其次，可以结合深度学习等前沿技术，不断提升系统的智能化水平，使其能够更好地理解用户意图，并给出更加精准和全面的答案。

此外，可以进一步优化系统的交互界面和用户体验，提供更加个性化、智能化的服务。同时，还可以考虑将系统扩展到移动端平台，以便用户能够随时随地进行信息查询和交流。

如上言，通过不断的技术创新和系统优化，校园智能问答系统有望成为一种普遍且不可或缺的信息服务手段，为校园信息管理和个性化服务提供更为便捷、高效的解决方案。

参考文献

致谢

感谢朋友们的鼓励，学校的栽培。感谢一直未放弃过的自己。