学号 2019302110045

密级 _____

武汉大学本科毕业论文

基于知识图谱的《环境科学》教案自动生成方法研究

院（系）名 称：计算机学院

专 业 名 称 ：软件工程

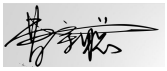
学 生 姓 名 ：曹宇聪

指 导 教 师 ：刘峰 教授

二〇二三年四月

郑 重 声 明

本人呈交的学位论文，是在导师的指导下，独立进行研究工作所取得的成果，所有数据、图片资料真实可靠。尽我所知，除文中已经注明引用的内容外，本学位论文的研究成果不包含他人享有著作权的内容。对本论文所涉及的研究工作做出贡献的其他个人和集体，均已在文中以明确的方式标明。本学位论文的知识产权归属于培养单位。

本人签名：  　　日期：2023 年 4 月 15 日

摘　要

随着信息技术的快速发展和人们理念的进步，教育在现代社会中变得越来越重要。教育自动化是一种新兴的教学模式，是指应用计算机、人工智能和其他现代技术，使教育过程变得更加自动化、高效化和个性化的一种教育方式。PPT 教案制作是教师教学工作中关键的一环，研究教案自动生成方法，将有助于提高教育效率、降低教育成本和提升教育质量。

针对上述问题的研究，本文提出基于知识图谱的《环境科学》教案自动生成方法，主要研究工作如下：

本文计划开发前端跨平台桌面应用，搭载基于知识图谱的《环境科学》教案自动生成方法。采用 vue3-typescript-electron 作为系统的核心框架，使用前端语言打造跨平台桌面应用。系统后端使用 node.js 封装 chatGPT 对话接口，通过结合设计的 prompt 和以《环境科学》数据为主的用户输入向 chatGPT 模型请求数据，以获取系统构建所需的知识图谱数据和环境科学 PPT 教案数据。在数据的知识图谱可视化方面，本文采用 Echarts.js 库中的力导向图，将各种知识元素以图形化的方式进行表示和连接。同时，由知识图谱提供知识脉络，本文根据 PPT 教案数据构建符合 Office Open XML 标准的文档对象，以此合并生成 PPT 教案文件 PPTX。本文研究将有助于进一步推进知识图谱在教学领域的应用和环境科学教育自动化，协助教师的教学工作，为教育自动化领域的教案自动生成方向提供实践案例。

关键字：教育自动化；PPT 教案；知识图谱；环境科学

ABSTRACT

With the rapid development of information technology and the advancement of people's ideas, education has become increasingly important in modern society. Educational automation is an emerging teaching model that applies computers, artificial intelligence, and other modern technologies to make the education process more automated, efficient, and personalized. PPT teaching materials production is a key part of teachers' teaching work, and research on the automatic generation of teaching materials will help improve education efficiency, reduce education costs, and enhance education quality.

To address these issues, this paper proposes a knowledge-graph-based automatic generation method for "*Environmental Science*" teaching materials. The main research work is as follows:

This paper plans to develop a front-end cross-platform desktop application that carries the knowledge graph-based automatic generation method for "*Environmental Science*" teaching materials. Vue3-typescript-electron is used as the core framework of the system, and front-end languages are used to create cross-platform desktop applications. The system backend uses Node.js to encapsulate the chatGPT dialogue interface, and combines the designed prompt and user input mainly based on "*Environmental Science*" data to request data from the chatGPT model, in order to obtain the knowledge graph data and PPT teaching materials data required for system construction. For the visualization of knowledge graph data, this paper uses the force-directed graph in the Echarts.js library to represent and connect various knowledge elements in a graphical way. In addition, by providing the knowledge context, this paper constructs a document object that conforms to the Office Open XML standard based on PPT teaching materials data, which is used to merge and generate PPT teaching materials file PPTX. This research will help further promote the application of knowledge graphs in the teaching field and the automation of environmental science education, assist teachers' teaching work, and provide practical examples for the direction of automatically generating

teaching materials in the field of educational automation.

Key words: Education automation; PPT teaching materials; Knowledge graph; Environmental science

目　录

1 绪论

1.1 研究背景

随着信息技术和教育技术的迅速发展，教育领域正经历着深刻的变革。在这个过程中，教案作为教学活动的重要组成部分，对于提高教学质量、引导学生学习具有至关重要的作用。然而，传统的教案制作过程往往繁琐、耗时，导致教师在教学准备阶段面临很大的工作压力。在此背景下，能提供直观知识脉络展示的知识图谱和基于此的教案自动生成方法研究显得尤为重要。

知识图谱作为一种直观的知识表达形式，可以帮助教师和学生更好地理解和掌握教学内容，提高教学效果。而教案自动生成方法旨在通过计算机技术和人工智能技术，为教师提供自动化、高效的教案生成方案，以此来节省教师的时间，进一步提高教案制作的效率。除此之外，自动生成的教案还可以实现个性化，根据教师不同的资料满足不同的教学需求。

近年来，知识图谱和教案自动生成方法的研究得到了广泛关注。许多学者通过对教学资源、知识结构、教学策略等方面的分析和挖掘，探讨了如何利用现有的数据和技术实现知识图谱的可视化表示和教案的自动生成[1]。然而，当前这两方面的研究仍然存在一些问题和挑战，如知识图谱可视化效果与实际教学需求的匹配、自动生成教案的质量和适用性、自动生成教案与教师个性化教学的结合等。因此，深入研究基于知识图谱的教案自动生成方法，既可以推动教育技术的发展，也有助于提高教育质量。

本文将围绕基于知识图谱的教案自动生成方法的研究展开，首先回顾相关领域的研究进展，然后探讨现有研究中的问题和挑战，并提出相应的知识图谱可视化技术和一种创新的教案自动生成方法。接着，将这两者有机结合，实现对教学内容的智能化生成与可视化表示，以满足不同教学场景的需求。最后，通过实际开发研究验证该方法和技术的有效性和可行性，为实际教学提供有益的参考。

本文研究采用的数据来源为《环境科学》一书，全名《环境科学与工程概论》[2]，是“环境保护概论”课程的教学参考书，基于近年来环境管理和技术发展的实际编写，着眼于培养学生解决工程问题的能力、培育可持续发展的理念，能为本文提供丰富契合研究数据。

本研究的目的是为教师提供一个高效、易用的教案制作工具，同时帮助学生更好地理解和掌握知识点。通过对知识图谱可视化技术和教案自动生成方法的深入研究，期望能够为教育技术领域贡献新的理论和实践经验，进一步推动教育领域的创新与发展。

1.2 国内外研究现状

1.2.1 国内研究现状

在知识图谱生成与可视化技术方面，国内关于知识图谱生成与可视化技术的研究逐渐受到重视[3]。知识图谱可视化方面，学者们通过研究知识图谱的构建方法和可视化呈现方式，提出了一系列可视化技术，例如力导向布局、分层布局、径向布局、聚类布局等等[4]。这些技术有助于呈现复杂知识结构，为教师和学生提供更直观的学习体验。然而，当前国内研究在知识图谱的可视化布局优化、交互设计等方面尚存在不足，需要进一步深化。

在教案自动生成方法方面，近年来，国内关于教案自动生成方法的研究取得了一定的进展。许多学者尝试将自然语言处理、知识表示和推理等技术应用于教案生成过程。例如，一些研究者尝试利用关键词提取[5]、文本分类[6]等自然语言处理技术从大量的教育资源中提取和分析教学内容，然后根据教学目标和策略生成相应的教案；也有学者尝试将教学内容和教学策略表示为形式化的知识结构（如本体、规则等），并利用知识推理技术对这些知识进行分析和推导，以生成个性化的教案[7]。此外，还有研究者探讨了利用人工智能技术，如深度学习和神经网络，对教学资源进行自动分析和生成[8]。不过尽管国内研究取得了一定成果，但在教案质量保证、教学资源推荐准确性等方面仍有待进一步研究。

1.2.2 国外研究现状

在知识图谱生成与可视化技术方面，在国际研究领域，知识图谱生成与可视化技术已经取得了丰富的成果[9]。学者们针对知识图谱的特点和需求，提出了多种方法和工具。例如，Cytoscape 是一款广泛应用于生物信息学领域的开源网络数据可视化工具，它为研究者提供了直观的可视化操作和丰富的功能。另一个典型的工具是 Gephi，它为用户提供强大的图算法和可视化功能，支持动态和分层网络，

并允许用户通过直观的操作进行交互式探索。这些工具在知识图谱的生成与可视化方面为研究人员提供了很好的支持。虽然国际研究在知识图谱可视化技术方面取得了较高水平的成果，但在处理大规模知识图谱、多维度呈现等方面仍有一定挑战。

在教案自动生成方法方面，在国际研究领域，教案自动生成方法已取得与国内研究相当的进展。众多国际研究者高度重视在教案生成过程中的个性化需求，并因此提出了多种适应性教学策略[10]。例如，根据学生知识水平、学习风格和学习兴趣构建学生模型，再依此推荐个性化教学资源或生成教案；设计个性化学习路径推荐算法，让老师根据学生个性选择适合的课程安排和教学资源，帮助他们更有效地达到教学目标。在国际研究中，学者们对于教案个性化和教学资源推荐方面投入了更多精力。然而，当前研究在普适性和资源深度方面仍有一定的不足，有待进一步加强和完善。

1.3 论文主要研究内容

目前，教师在知识脉络划分、文字教案编写和 PPT 教案创作过程中，往往依赖于自身经验。然而，这种方式可能会导致教学资料的时效性和丰富程度存在一定程度的不足。因此，以《环境科学》一书为主要数据来源，探讨如何基于纯文本资料实现知识图谱生成及可视化和 PPT 教案的自动生成方法是本文的主要研究内容。

本文将完成一个 Vue3-Typescript-Electron 的项目开发，并将知识图谱生成及可视化和 PPT 教案的自动生成方法嵌入项目中，实现方法的测试、展示和应用。

关于知识图谱生成与可视化，本文首先分析知识图谱的组成，编写对应 typescript 接口，定义知识图谱的数据结构。接着，构造 prompt 以从 chatGPT 对话接口获取知识图谱数据。然后，选用 Echarts 图形库制作自定义图表，将知识图谱中的结点与连接渲染到自定义的力导向图上。最后，将整个可视化方法封装在 vue3 自定义 hook 中供页面组件调用。

关于教案自动生成方法，在生成知识图谱的基础上，本文首先确定生成教案的结构，编写合适的 PPT 模板。其次，由知识图谱数据结合文本资料构造 prompt 以从 chatGPT 对话接口获取 PPT 教案数据。然后，根据 PPT 数据与模板，构建符合 Office Open XML 标准的文档对象，通过合并这些文档对象生成 PPT 教案。最

后，将教案自动生成方法封装在 vue3 自定义 hook 中，由页面组件调用。

关于系统开发，本项目采用 Vue3、Typescript 和 Electron 的技术栈，力在开发高效、稳定、可维护和跨平台的系统。

1.4 论文组织结构

本文正文共有六个章节，章节标题与章节主要内容安排如下所述：

第一章 绪论：介绍了研究背景、研究意义、国内外研究现状以及本文的研究内容和组织结构。

第二章 关键技术分析：介绍了系统开发技术栈和核心研究方法。系统开发技术栈为 Vue3、Typescript 和 Electron，后端为 node.js。核心研究方法分别为知识图谱生成及可视化方法和教案自动生成方法，包括 chatGPT、Echarts、PptxGenjs 等技术。

第三章 知识图谱生成及可视化：详细阐述了知识图谱数据结构构建、获取数据的方法以及采用 Echarts 的力导向图实现可视化的具体流程。

第四章 教案自动生成方法：描述了 PPT 教案模板设计、基于知识图谱的 PPT 教案数据获取以及构建符合 Office Open XML 标准的 PPT 教案的方法。

第五章 系统设计与实现：介绍了基于 Vue3、TypeScript 和 Electron 技术栈的系统设计和开发过程，包括需求分析、总体架构、功能设计以及关键技术实现细节。之后针对本文实现的系统进行功能测试、性能评估，并通过案例分析展示系统的有效性和实用性。

第六章 结论与展望：对全文进行总结，展示本文的研究内容完成情况和研究结果，同时反思、分析并指出本文研究中存在的问题，并对后续工作进行展望。

2 关键技术分析

本文开发的基于知识图谱的教案自动生成系统采用前后端分离架构，使用VScode开发环境，Vue3-typescript-electron的前端技术栈，以及node.js搭建的后端。后端负责chatGPT对话接口的搭建，前端完成对知识图谱生成、知识图谱可视化和教案自动生成方法的封装和应用，并提供页面展示模块，实现系统的主要功能。本章主要对实现系统方法使用到的相关技术进行介绍。

2.1 node.js

在JavaScript开发领域，随着Web技术的不断发展，开发人员对于高效、简洁的开发框架的需求也愈发迫切。Node.js应运而生，致力于简化JavaScript服务器端开发，为JavaScript开发者提供了一个快速、高效的轻量级平台。它基于Google的V8引擎，以事件驱动、非阻塞I/O模型为核心特性，使得JavaScript能够高效地处理大量并发连接[11]。

Node.js继承了JavaScript的灵活性，并进一步简化了开发流程和配置。开发者只需通过包管理器（如npm）引入所需的依赖，无需繁琐的配置文件，即可开始使用。Node.js已成为许多现代Web应用和API服务的首选技术栈，为开发者提供了一站式解决方案。通过使用Node.js，JavaScript开发者可以实现端到端的全栈开发，统一前后端技术，提高开发效率和维护便捷性。

2.2 Vue3-Typescript-Vite

作为一款用于构建用户界面的渐进式JavaScript框架，Vue3在前端开发中展现出诸多优势。核心库专注于视图层，实现在不同场景下的轻松集成。性能方面，Vue3采用更快的虚拟DOM、响应式系统和编译时优化，提高了运行速度和性能。Composition API则为开发者提供灵活、模块化的代码组织方式，强化了代码的可维护性和可读性。

Vue3对TypeScript支持友好，内置定义文件简化了TypeScript在组件中的应用。TypeScript是一种可选的静态类型系统，允许开发者在JavaScript中使用类型标注和类型检查，以提高代码质量、可维护性和健壮性。TypeScript的类型

推断和编辑器集成为开发者提供了强大的智能提示和错误检测功能，有助于加速开发流程。

在体积优化上，Vue3 利用 Tree-shaking 和按需引入降低构建大小，实现轻量化和快速加载。实用内置特性如 teleport 和 suspense 也简化了前端开发过程。在项目构建上，本项目采用 Vite 作为构建工具和开发服务器。Vite 是一款基于浏览器原生 ES 模块导入功能的下一代前端构建工具，具有快速冷启动、按需编译和模块热更新等特点，大幅提高了开发者的工作效率。 综上，Vue3 结合 TypeScript 和 Vite，凭借丰富功能和易用性成为前端开发的显著优势。本项目将充分利用这些关键技术打造高效、稳定的前端界面。

2.3 Electron

Electron 作为一款跨平台桌面应用开发框架，整合了 Chromium 浏览器引擎和 Node.js 运行时环境，让开发者能够利用 HTML、CSS 和 JavaScript 构建具有高性能和原生体验的桌面应用[12]。其核心优势包括跨平台能力，使得开发者只需编写一套代码便可部署到多个操作系统平台，降低开发、维护和测试成本。并且 Electron 与主流前端框架如 React、Angular 和 Vue 兼容，有利于开发者构建高效、稳定、易于维护的桌面应用程序。得益于其背后活跃的开源社区，开发者能获取丰富的开源库和插件，加快开发进度。本项目将综合运用 Electron 与 Vue3 等技术框架，实现跨平台桌面应用程序。

2.4 openAI API

OpenAI API 提供了一个高度强大的自然语言处理（NLP）服务，它能够与用户进行高效且智能的对话互动，执行各种文本生成和分析任务。这一服务基于先进的 chatGPT 模型，该模型不仅在语法、语义和世界知识方面表现出深入理解，而且具备极高的智能水平和强大能力。借助大量的训练数据和精心设计的架构，chatGPT 成为了世界最好的自然语言处理模型之一，能够提供高质量、准确且实用的生成内容。用户可以通过设置引导语（prompt）和请求头来灵活地调整生成内容的长度、多样性和创造性，以满足各种需求。OpenAI API 在处理请求时具备高度的实时性能，使得快速自然的对话交互成为可能。这使得开发者能够轻松地将该 API 集成到各种应用程序、服务或工具中，进而为用户带来智能化且实用的

交互体验。

本项目将充分利用 OpenAI API 中 chatGPT 模型的对话接口能力，通过运用多段对话和上下文连接，在获取知识图谱数据的基础上，根据用户的具体需求生成高质量的 PPT 教案数据。

2.5 PptxGenJs

PptxGenJS 是一个强大且易于使用的 JavaScript 库，适用于浏览器和 Node.js 环境，用于创建和导出 PowerPoint 演示文稿（PPTX 文件）。它支持多种格式，如文本、图片、图表、形状和媒体等，并提供直观的 API，使开发者能够在不依赖 Microsoft PowerPoint 的情况下轻松制作演示文稿。它可以简化从 XML 文件合并 PPTX 文件的过程。本文将以该库为基础，优化和拓展其构建 PPT 模板文件的方法。

3 知识图谱生成及可视化

3.1 知识图谱实体分析

在设计知识图谱数据结构的过程中，本文首先对多个知名知识图谱进行了深入的观察和研究，例如：Google Knowledge Graph[14]、Wikidata[15]和 ConceptNet[16]等，以便更好地了解这些知识图谱的主要特点和组成部分。在深入分析之后，发现知识图谱主要包括两个核心组成部分：节点（Node）和连接（Link）。

本文通过对这些知识图谱的特点进行总结，确定了节点和连接数组作为数据结构的基础。节点用于表示知识图谱中的实体，每个节点都具有一个唯一的 ID 和一个名称。连接则用于表示实体之间的关系，每个连接包含源节点、目标节点以及表示它们之间关系的字符串。

为了进一步优化知识图谱的数据结构，本文深入分析了节点和连接的属性，以确定它们最佳的表示类型。本文采用简洁的数据类型——string 和 number，提高了知识图谱数据结构的表示效率和可处理性。同时，本文还考虑了知识图谱的可扩展性，以便未来方便地满足不同的需求。

在设计完数据结构之后，本文将其整合为一个名为 KnowledgeGraph 的接口，该接口包含节点和连接数组。这种设计配合 Typescript 类型检查使得处理和操作知识图谱变得简单直接，同时保持了良好的灵活性，以支持各种知识图谱应用和分析任务。综上，通过研究现有知识图谱的特点，优化节点和连接的属性表示，本文设计了一个简洁、易于使用且具有可扩展性的数据结构（如图 3.1 所示），以便更好地支持各种知识图谱应用和分析任务。

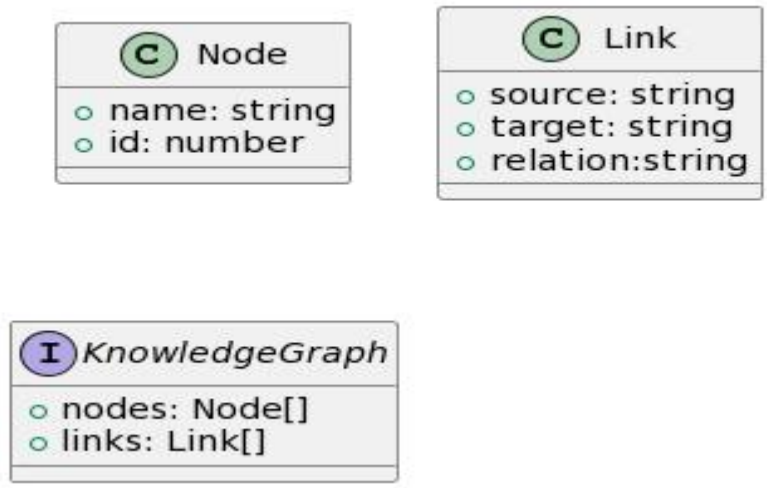


图 3.1 知识图谱类图

3.2 知识图谱数据获取方法

一般来说，知识图谱的生成需要经过多个步骤，如信息抽取、实体识别、关系抽取、数据融合等，涉及的技术较为复杂，而且需要耗费大量的时间和精力。

本文采用 OpenAI API 的对话接口来获取知识图谱数据，省略了对纯文字抽丝剥茧的步骤。OpenAI API 提供目前世界上最先进、最强大的自然语言处理模型 ChatGPT，可以直接处理自然语言输入，并将其转换为结构化的知识图谱数据。本文依据其响应习惯，编写了稳定、简练的引导语 prompt（如图 3.2 所示），通过字符串拼接和对话接口，将用户输入的纯文字生成知识图谱数据（如图 3.3 所示）。

```
const kgPrompt = `帮我将纯文字填入以下数据结构,这个数据结构代表的是一个知识图谱，它包含结点数组和连接数组。
interface KnowLedgeGraph{
  nodes:Node[]
  links:Link[]
}
type Node = {
  name: string
  id: number
}
type Link = {
  source: string
  target: string
  relation:string
}

请以json对象返回你的结果要求符合以上结构

纯文字如下：`
```

图 3.2 知识图谱 prompt

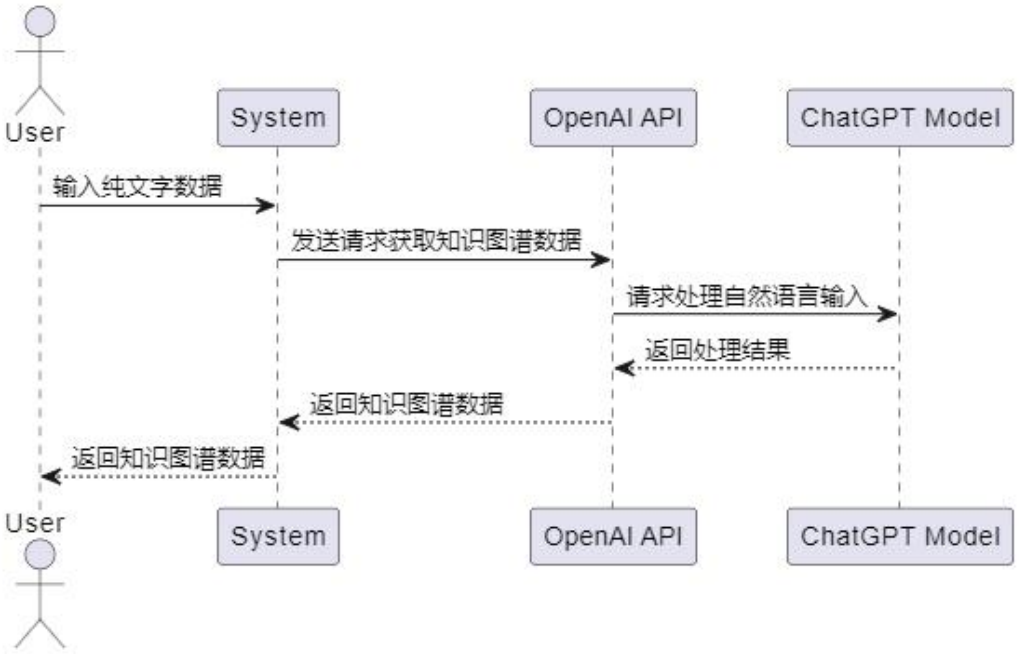


图 3.3 知识图谱相关序列图

3.3 基于 Echarts 的可视化方法

在前端社区日益繁荣的今天，有关力导向图的生成，市面上存在多种可供选择的方案，其中包括 D3.js[17]、Three.js[18]以及 Echarts.js[19]。由于 Three.js 主要用于 3D 场景的渲染，且在处理稍大规模的力导向图时性能存在问题，该库在本文研究场景下并不是首选方案。在 D3.js 和 Echarts.js 之间的选择中，通过对比官方示例和 API 的便利性，Echarts 所提供的示例交互性、可定制性都更加优异。Echarts.js 具有易于使用和扩展的特性，并且该库具有强大的性能，对 Typescript 也提供了更好的支持，在处理大规模的力导向图时也有良好的表现。因此本文最终选择了 Echarts.js 作为知识图谱生成的主要库。

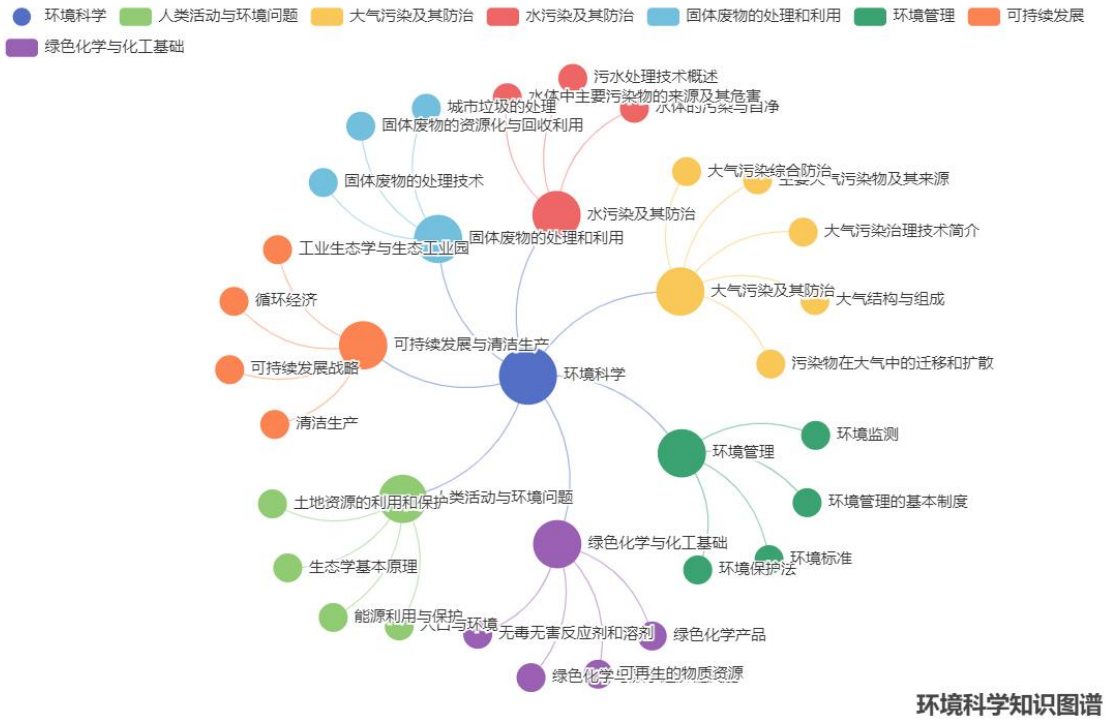


图 3.4 Echarts 力导向图示例

Echarts 为开发者提供 option 对象，开发者可根据自身需求对 option 对象属性进行定义和赋值。在本文研究中，知识图谱可视化依赖于力导向图，下面描述使用 Echarts 进行力导向图构建的过程。

首先，创建一个名为 generateOption 的函数，它接受两个参数，分别为节点数组和边数组。然后，定义一个名为 option 的对象，用于存储 ECharts 的配置选项。这些选项包括提示框，动画更新的持续时间，动画缓动效果，以及系列中的各种设置。在系列设置中，指定图表类型为力导向图，并启用力导向布局。在这基础上，还可以配置布局大小、边界、节点大小、漫游功能、节点标签、边的符号、边符号的大小和边标签等。最后，定义力导向布局的参数，例如斥力、引力和阻尼系数等。

在 generateOption 函数内部，使用 nodes.map()和 links.map()方法将原始数据集（前文类图中的 Node 和 Link 类型）转换为 ECharts 所需的数据格式。对于节点数组，将每个节点的名称和交互属性绑定到 ECharts 的 data 配置中。对于边数组，将每条边的源节点和目标节点绑定到 ECharts 的 links 配置中，并设置边的符号大小、标签、线条样式和高亮样式等。在边标签的 formatter 属性中，绑定 Link 中的 relation 属性，使其显示在力导向图上。最后，将 option 对象返回，供 ECharts 的 setOption()函数使用。

3.4 方法封装与调用

在前文中，generateOption 函数接收知识图谱的节点数组和边数组，返回适用于 Echarts 的配置选项 option。然而，在实际应用中，复杂的处理过程需要进行逻辑封装。Vue 3 提供了一种逻辑提取方式——自定义 hook，这是 Vue 3 中的一种高级功能。通过使用自定义 hook，开发者可以将复杂的逻辑处理过程封装在单独的 hook 文件中，以此提高代码的可维护性和复用性。

本文定义了一个名为 useOption 的自定义 hook，该 hook 返回一个包含 generateOption 函数的对象。通过这种方式，外部组件可以使用 ES6 的模块化指令 import，轻松调用 useOption 自定义 hook，实现知识图谱可视化的逻辑解耦和代码复用。在整个封装过程中，本文充分利用 Vue 3 的响应式特性和自定义 hook 的高级功能，实现了知识图谱生成及可视化方法。

4 教案自动生成方法

4.1 PPT 教案结构设计

本文参考众多本科课程 PPT 教案，详细研究与分析其常见形式和结构，总结归纳出一种适用于多数应用场景的 PPT 结构。这一结构主要包括以下四个部分：封面页、目录页、内容页以及结束页。

封面页：封面页作为 PPT 的第一页，具有极高的展示价值，其内容可能包括课程主标题、课程副标题、讲解人名字、学校或单位名称等。封面页设计以简洁明了为主。

目录页：目录页作为导航，提供整个 PPT 的大纲结构，帮助学生了解教案内容的安排和分布。目录页通常列出各个内容页的标题，可以采用简洁的列表或分级目录的形式。

内容页：内容页是 PPT 的核心部分，负责呈现课程的具体知识点、案例、活动等。内容页应保持一致的版式和风格，通过合理的排版、配色，使信息易于理解和记忆。在设计内容页时，要注意避免过多的文字堆砌，合理安排内容密度，做好分页。

结束页：结束页作为 PPT 的收尾，通常以致谢收尾，教师们在这一页做出总结陈述或与学生们进行交流。

4.2 基于知识图谱的 PPT 数据获取与处理

在当前研究领域中，关于从纯文字资料自动化生成 PPT 教案的研究十分匮乏。现有市场上的 PPT 生成解决方案，主要依赖于用户根据预定义的格式自行填充 PPT 内容，并根据已经加工处理过的文字资料相应地生成 PPT。这种方法存在局限性，可能无法满足教学场景中对自动化和智能化生成 PPT 教案的需求。

本文采用 OpenAI API 提供的对话模型 GPT 进行数据获取，通过第三章中知识图谱的生成，给予对话模型上下文联系。再以稳定且简洁的引导语 prompt，实现基于知识图谱从纯文字中提取符合 PPT 教案结构的数据。利用 OpenAI API 的高度智能化和强大的自然语言处理能力，可以更有效地分析和提炼出有价值的信息，

从而自动生成具备一定教学价值和结构完整性的 PPT 教案。这种方法在一定程度上解决了传统方式中的局限性，为提高教学资源的开发效率和质量奠定了坚实的基础。

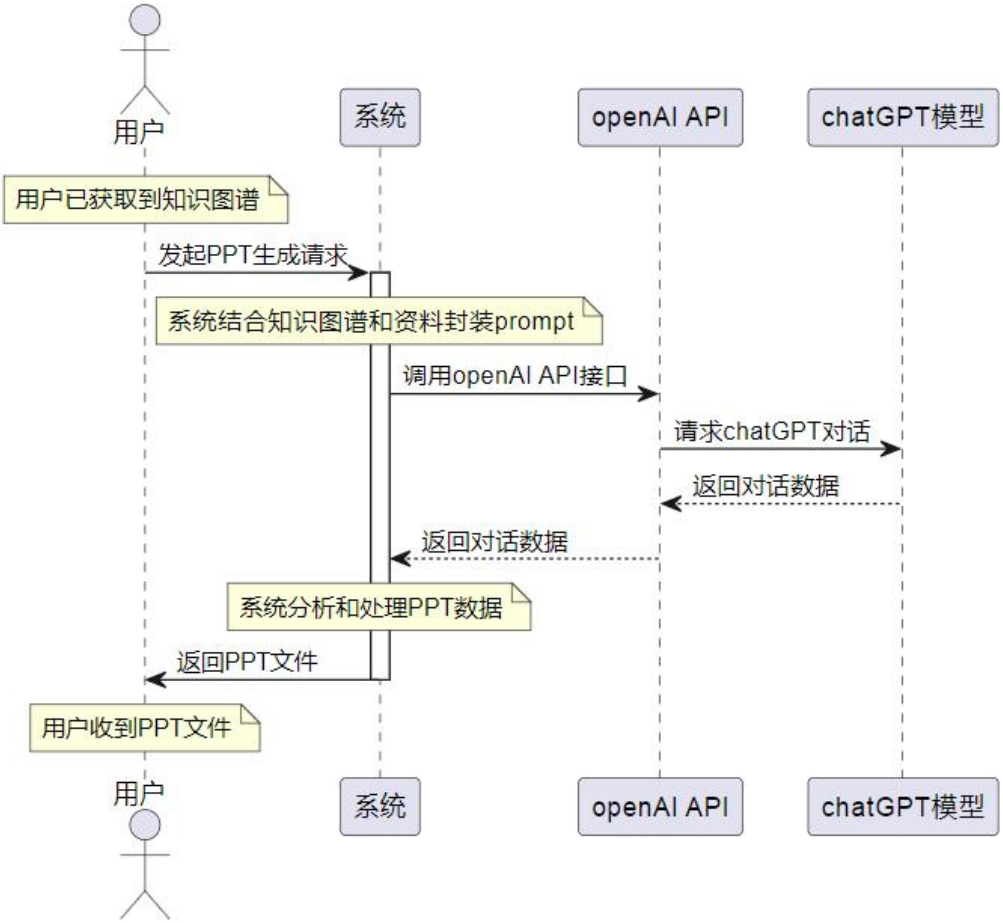


图 4.1 PPT 生成序列图

生成 PPT 教案的过程中需要处理大量的数据，若以数据结构的形式进行存储和处理，如图 4.2 所显示的结构，对获取到的数据质量有很大的要求，容易出现分页类型错误、内容混杂等问题。在处理纯文本数据时，由于信息缺乏明确的结构化标识，对自然语言处理模型的要求相对较高。在进行语义识别和文本分析的过程中，非结构化数据可能导致诸多挑战。并且纯文本数据之间可能存在显著差异，生成的 PPT 数据质量的稳定性通常难以保证。本研究以数据结构进行 PPT 数据存储出现了多种异常情况，例如出现多个主标题页和目录页，以及目录页与内容页不匹配等现象。

```
interface PPTPresentation {
  title: string
  slides: Slide[]
}

type Slide = {
  title: string
  type: '主标题' | '目录' | '内容页' | '结束页'
  content: string
}
```

图 4.2 拟 PPT 数据结构

本文针对上述问题，提出使用 ES6 模板字符串进行 PPT 教案数据存储和基于知识图谱的 PPT 教案数据获取与处理。在调用自然语言处理模型生成 PPT 数据前，首先使用模型分析、处理资料生成对应的知识图谱，为纯文本资料构建知识脉络体系和结构。接着，利用引导语 prompt 对响应内容加以约束限制，使结果保持稳定、不混乱。引导语中规定了一个特定结构的 PPT 教案模板，要求目录页与内容页必须严格对应。为了确保自然语言处理模型能够准确理解这个结构，引入文本的知识图谱生成对话，形成上下文联系，并在引导语中添加了注释、分隔符，指导模型正确解析和生成 PPT 教案数据。同时，也规定了响应结果的格式，以便于后续数据处理和使用。

```
const pptData = `
地球的形成和演化
=====
1.星云假说
2.地球的形成和演化过程
3.地球上的原始生命
=====
根据“星云假说”的解释，距今大约60亿年以前，地球的轮廓尚未形成，是一团没有凝聚在一起的混杂着大量宇宙尘埃的星云状气尘物质，到距今大约45亿～20亿年前这段时期，地球星体和原始的地理环境逐渐形成。随着地球物质分异过程的持续进行，多种原始火山气体（如CO2、CH4、NH3、H2S等）上升至地表形成了原始的大气圈。
=====
随着地球内部温度的升高，原来存在于地球内部的结晶水大量蒸发进入地球表面的原始大气层，冷凝后形成大气降水，在地球表面逐渐形成了河流、湖泊和海洋等水体。
=====
在地球的原始地理环境刚刚形成的时候，地球上没有生物（当然更没有人类），只有原子、分子的化学及物理运动。地球形成的初期，大气中没有氧和臭氧，辐射到地球表面的紫外线十分强烈，而海洋中由于水层能挡住紫外线和宇宙射线，加上海水具有流动性、恒温性、不恒压性以及巨大的气-液界面和固-液界面等特点，有利于物质的积累、浓缩、迁移和充分利用太阳能，并有利于发生化学、物理和生物的变化，于是，海洋成为原始生命的诞生地、保护伞和储存库。
`;
```

图 4.3 PPT 数据示例

获取到模板字符串，本文需要对其进行分割，提取出所需的部分。通过String.prototype.trim()方法以及正则表达式实现非换行符的空格消除。

```
const pptData = result.trim().replace(/[ \t]+/g, "")
```

图 4.4 模板字符串处理

拿到 PPT 数据后，首先，使用 content 字符串变量接收 PPT 教案数据，通过String.prototype.split()方法，将原字符串按分隔符`=====`划分成不同字符串数组，对应不同的 PPT 页面。

```
const sections = content.split('\n=====\n')
```

图 4.5 PPT 数据按页分割

经过对响应模板字符串进行分割后，得到的 sections 数组中，sections[0]表示 PPT 教案的封面页，sections[1]表示 PPT 教案的目录页，sections.slice(2 , sections.length)表示 PPT 教案的所有内容页，Array.prototype.slice()方法按照参数索

引截取数组中的内容生成新数组，实现所有内容页提取。在获取目录页部分后，再使用`\n`分隔符对目录页进行一次按行分割，获取目录列表和各个内容页标题。以上过程实现了纯文字到能够生成 PPT 教案的数据的转换。

4.3 PptxGenJs 使用与拓展

从 PPT 教案数据生成 PPT 教案文件需要进行复杂的 XML 文件编写，在 Office Open XML 格式中[20]，一个 PPTX 文件是由多个 XML 文件组成的压缩包，这些 XML 文件包含着 PPT 教案的文本、图形、排版等内容，通过开放打包约定 Open Packaging Conventions[21]进行规范管理和组织。

在前端生成 PPTX 文件的方法中，PptxGenJs 是一种常用的前端库，通过在代码中创建 PptxGenJs 对象，可以使用其提供的简单 API。尽管 PptxGenJs 库提供了一些基础的 PPT 元素和模板，但它对 XML 文件操作的封装程度较基础，对于复杂的 PPT 模板设计，它的支持相对较少，该库的设计重点在于快速生成简单的 PPT 文件，而对于复杂的 PPT 设计和布局，开发者需要自己编写定制的代码实现。

本文使用 PptxGenJs 库的基础 API，拓展封装了可以根据自定义 PPT 模板和 PPT 教案数据生成美观的 PPT 教案文件的方法。下面以武汉大学主题 PPT 模板为例，PPT 数据由《环境科学》书中第一章部分内容生成。该主题模板首先实例化一个 PptxGenJs 对象，该对象在底层对应一个 PPTX 文件，可以在此规定 PPT 的作者、布局等。接着使用 defineSlideMaster 方法定义了一个背景模板，规定背景颜色、每页索引样式等。

```
//创建ppt教案文件对象
const pptx = new PptxGenJS()
// 定义一个幻灯片背景模板
pptx.defineSlideMaster({
  title: 'whu',
  background: { fill: 'ffffff' },
  margin: [0.5, 0.5, 0.5, 0.5],
  slideNumber: { x: '95%', y: '95%', color: '1D6295', fontFace: 'Microsoft YaHei UI', fontSize: 12 },
})
```

图 4.6 pptx 对象定义

然后通过 addSlide()方法在 PPTX 文件中添加新的一页幻灯片，并使用 slide.addShape(),slide.addImage(),slide.addText()等方法定义好背景模板，这页 PPT 为主标题页，加上自定义主题设计为如下样式：

图 4.7 whu 模板标题页

在前文模板字符串的处理中，本文保留了换行符的存在，目的是通过字符串分割的方式获取到目录页数据。

```
const tocContent = sections[1].split('\n')
```

图 4.8 获取目录页数据

目录页同样以 addSlide()方法创建出来添加进 PPTX 文件中，根据目录页内容设计不同的样式，并使用等分算法，目录列表数量不同的情况下，计算出每个列表项所占据的纵向空间大小。

```
for (let i = 0; i < tocContent.length; i++) {
  tocSlide.addShape(pptx.ShapeType.rect, {
    x: 5.5,
    y: 2.1 - ((tocContent.length / 2 - i) * 2) / 7.5 + 0.5 * i,
    w: 4.5,
    h: 0.5,
    fill: { color: '1D6295' },
  })
  tocSlide.addText(tocContent[i], {
    x: 5.7,
    y: 2.3 - ((tocContent.length / 2 - i) * 2) / 7.5 + 0.5 * i,
    fontFace: 'Microsoft YaHei UI',
    fontSize: 18,
    color: 'FFFFFF',
    bold: true,
  })
}
```

图 4.9 目录页等分算法

图 4.10 whu 模板目录页

内容页则利用 Array.prototype.forEach()方法批量生成，同时将目录列表各项作为内容页小标题添加到内容页设计中。

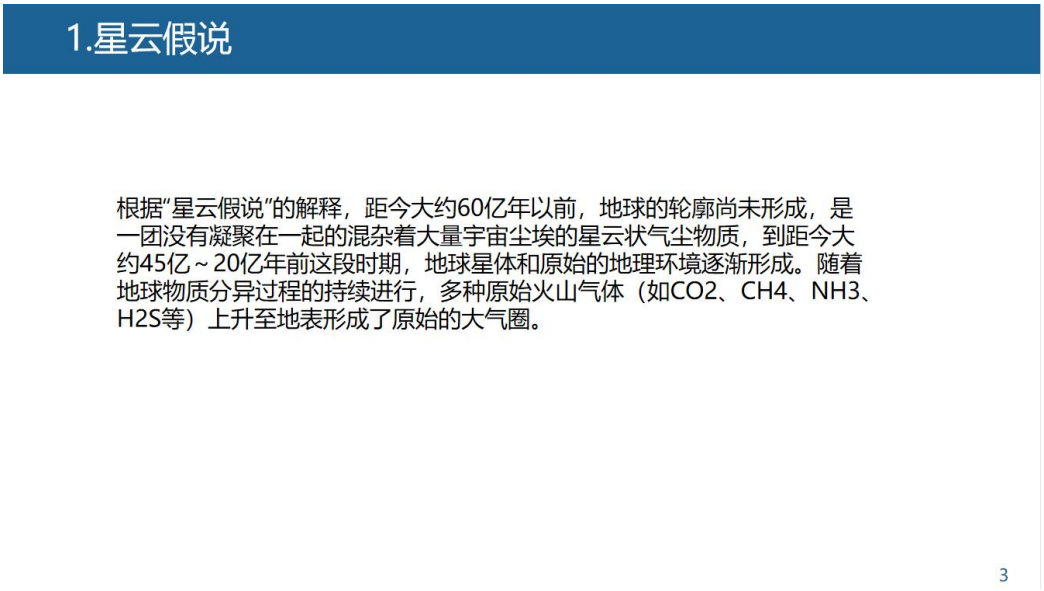


图 4.11 whu 模板内容页

最后结束页采用默认数据，格式参考标题页。

图 4.12 whu 模板结束页

汇总以上各页，调用 pptx.writeFile()方法时，它会将生成的 PPT 文档对象转换为对应的 XML 格式。此后，该方法会将这些 XML 格式的数据和图片打包成一个 Zip 文件，写入磁盘。这些 XML 文件的结构遵循 PresentationML 标准，即每个 XML 文件都包含在一个 PPTX 文件夹中。最后，所有这些文件被打包在一个 Zip 文件中，并使用扩展名.pptx，完成 PPT 教案文件的生成。

4.4 方法封装与调用

本文定义了一个名为 usePptx 的自定义 hook，该 hook 返回一个包含构建各种 PPT 模板方法的对象，该对象包括 writeFileByWhu、writeFileBySunny、writeFileByRainbow，分别提供 Whu、Sunny、Rainbow 三种主题模板。通过这种方式，外部组件可以根据自身需求使用 import 导入 usePptx，调用该 hook 提取对应 PPT 模板方法，例如使用下拉选择框将三个方法绑定至对应的选项，实现个性化 PPT 教案自动生成。

5 系统设计与实现

5.1 需求分析

5.1.1 系统需求分析

本文研究开发的系统主要功能是根据《环境科学》数据实现知识图谱的生成与可视化以及基于知识图谱的教案自动生成。用户从《环境科学》一书中选取纯文字数据，上传到系统，系统调用后端接口与 OpenAI API 的语言模型构建对话联系，获取到知识图谱和 PPT 教案数据。随后前端解析并处理获取的数据，完成知识图谱可视化并提供 PPT 教案生成按钮以供用户下载到本地。

系统需求分析需要考虑到多个方面，除了实现主要功能外，性能、可维护性、可扩展性和用户的使用体验等也是优秀系统该有的特性。在设计系统架构和选取技术栈时，需要考虑这些因素。本系统采取客户端－服务端架构（C/S），前端开发选取的是 Vue-typescript-electron 的技术栈，后端采用 node.js，前后端分离开发，满足系统需求。

5.1.2 功能性需求

本系统主要功能需求包括两项：

（1） 用户通过系统将纯文字可视化为知识图谱

用户将纯文字资料上传到系统中，系统根据资料生成知识图谱。该知识图谱应当涵盖纯文字中的实体与实体之间的关系，图表可以拖拽、放大、自由调整等。

（2） 基于知识图谱的 PPT 教案自动生成

在生成知识图谱形成纯文字的知识脉络体系后，用户可以将纯文字生成为可保存在本地的 PPT 教案文件。系统提供主题选择，用户在生成 PPT 教案文件之前，可以自定义所需要的 PPT 模板主题。保存到本地时，用户可以自行决定存放路径、文件命名。

系统使用者只有一类普通用户，用例图如下所示：

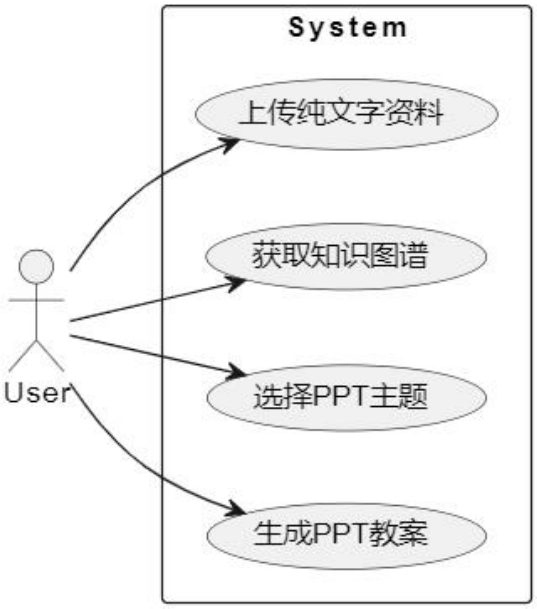


图 5.1 系统用例图

5.1.3 非功能性需求

非功能性需求主要围绕性能、可维护性、可扩展性和用户体验展开说明。一个系统首先需要保证其系统的性能和稳定性，然后也需要有足够的扩展性，从而在投入使用和未来开发与维护中让用户和开发者体验舒适。

性能方面，系统在接收用户提供的文字资料、更新知识图谱状态以及生成 PPT 教案时需要尽可能降低响应时间，过长的等待会使用户体验不佳。

可维护性上，主要页面组件中需要减少逻辑处理，核心技术以导入使用为先。开发过程中遵守项目管理规范和变量、函数命名规范，在复杂操作部分可以加以注释。

可扩展性上，针对复杂逻辑的处理需要封装成自定义 hook，常用模块封装成组件，减少冗余代码。常用数据结构需要进行类型定义，严格进行类型检查。保证优秀的扩展性，可以使系统在未来开发和迭代中更有优势。

用户体验上，系统需要在客户端中提供使用说明，为用户讲解系统功能以及使用方法。在较长时间的异步操作时，系统应当提供 loading 动画，让用户使用时得到反馈。

5.2 系统设计

5.2.1 系统架构设计

本文使用客户端－服务端架构，将系统拆分为客户端和服务器两部分，客户端负责展示、处理用户输入以及核心功能实现，而服务器负责调用接口获取数据。客户端和服务器通过网络通信进行交互，实现数据传输和处理。

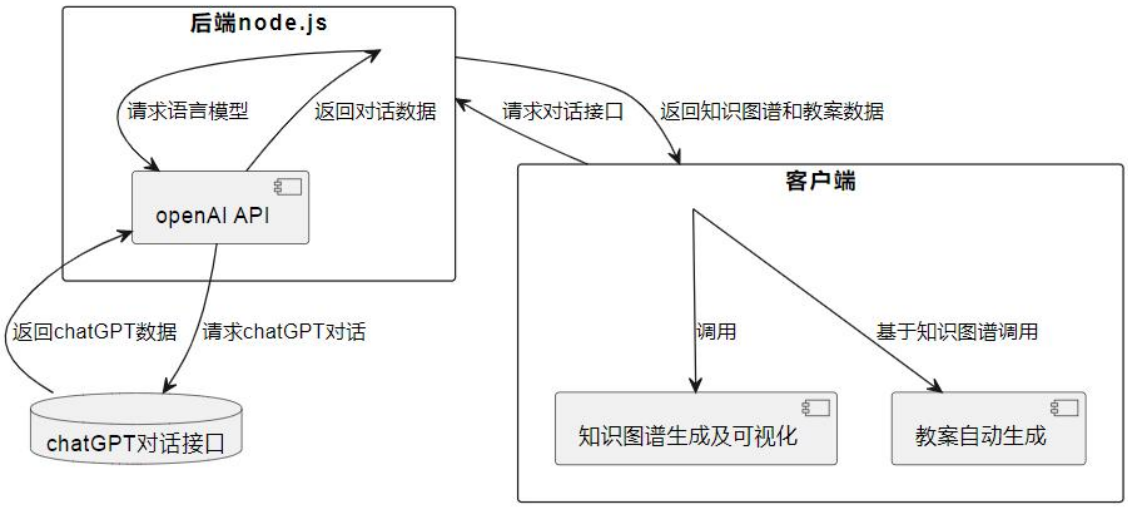


图 5.2 系统架构

客户端开发使用 Vue3-Typescript 框架结合 Electron 框架，并使用 Vite 进行项目管理。Vue3 采用组件化的方式构建应用程序，其组合式 API 能使代码结构清晰、易于维护。而 TypeScript 是一个强类型的 JavaScript 超集，它提供了严格的类型检查和代码提示，避免了一些常见的类型错误，提高了代码的可维护性和可扩展性。同时，Electron 框架提供了一种可编写桌面应用程序的方式，使得开发者可以通过 web 技术构建跨平台的桌面应用程序，提高了应用程序的可扩展性和适用范围。

服务端开发使用 node.js，引入 OpenAI API，搭建对话接口供前端请求数据。

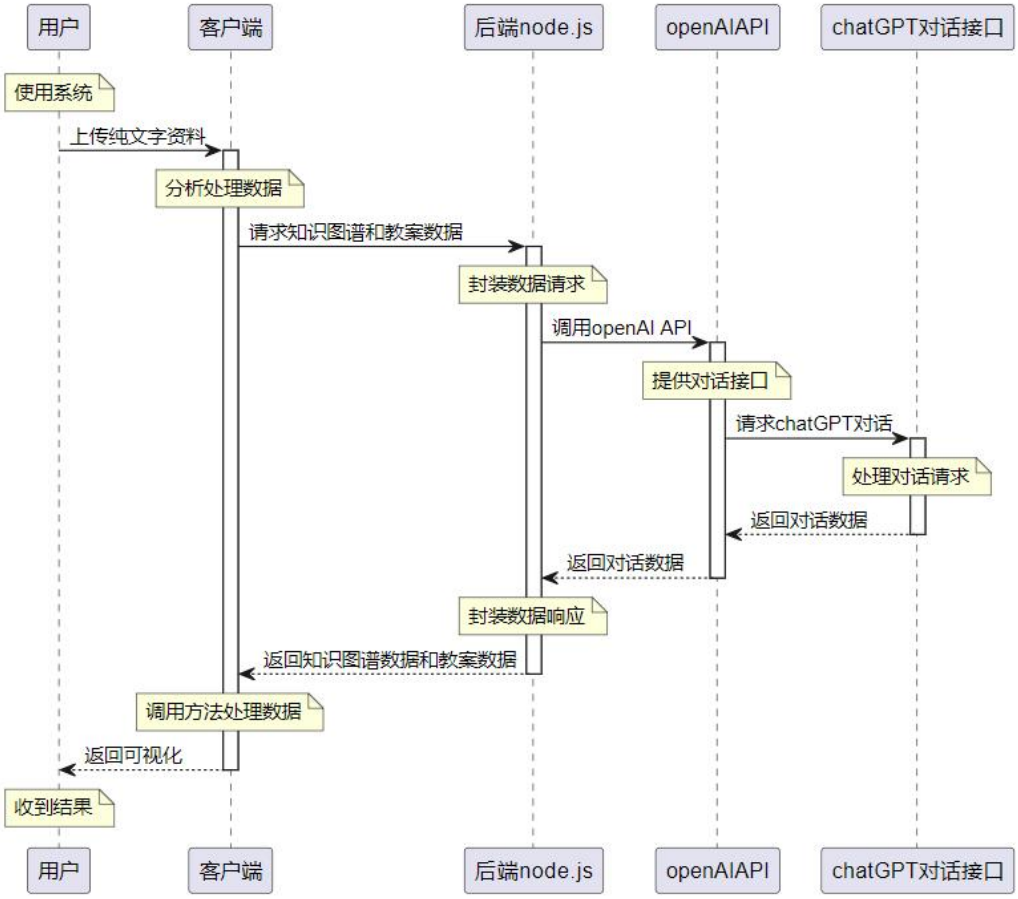


图 5.3 系统时序图

5.2.2 系统模块设计

当设计系统时，需要将系统的各个功能拆分为不同的模块，并对每个模块进行设计和实现。在本系统中，主要包括页面布局模块、知识图谱可视化模块、PPT 生成模块和状态管理模块。

页面布局模块是整个系统的基础，负责页面上下两部分的布局以及文本框、按钮等 UI 组件的展示和交互逻辑。在这个模块中，需要对页面元素的尺寸、位置、样式和交互逻辑进行设计，保证页面的美观性和易用性。同时，需要将不同的 UI 组件与其他模块进行连接，实现功能的协同作用。

知识图谱可视化模块能将用户输入的纯文本数据传入服务端接口，获取知识图谱数据并将其可视化。该模块需要实现数据解析和可视化方法，完成服务端返回知识图谱数据的分析和处理。在前端展示方面，解析出知识图谱数据后需要保证视图的动态更新，并提供拖拽、放大缩小、移动等功能。

PPT 生成模块在知识图谱视图更新后可用。实现知识图谱可视化后，客户端再向服务端接口发起请求，在知识图谱数据的基础上获取 PPT 教案数据。PPT 生成模块完成对 PPT 教案数据的解析，并存储到状态管理模块中。在前端展示方面，当 PPT 教案数据处理完毕后，需要提供 PPT 教案保存本地功能。

状态管理模块负责控制获取知识图谱和生成 PPT 的按钮状态，以及用户输入控制、系统重置按钮、PPT 主题选择 Modal 和数据临时存储。根据用户使用系统的不同阶段，该模块需要实现交互提示和对系统流程的控制。状态管理模块使用 Pinia 工具，它是一个比 Vuex 更轻量级、性能更好的状态管理工具，专门为 Vue3 设计的状态管理库。Pinia 的状态管理包括 state、getter 和 action，通过这些 API 可以轻松实现各种状态的管理，并且能适应响应式系统。

综上所述，对于这个系统，模块设计应包括页面布局模块、知识图谱可视化模块、PPT 生成模块和状态管理模块。每个模块对应特定的功能和作用，这些模块协同工作，实现整个系统的顺利运行。

5.2.3 系统路由

系统路由采用 Vue-Router，它是 Vue.js 官方提供的路由管理器，用于构建 SPA（Single Page Application）单页应用程序。它通过动态切换组件，可以实现不同页面的跳转。使用前端路由，系统不需要每次访问新页面时重新请求服务器，可以提升用户体验。本文开发的系统中，引入 Vue-Router，设计了一级路由，能够协助用户在使用帮助和主页面间进行切换。

5.3 系统实现

本文研发的系统整个项目结构如下所示：

main 文件夹下为 Electron 框架相关，renderer 文件夹下为 Vue3 框架相关。

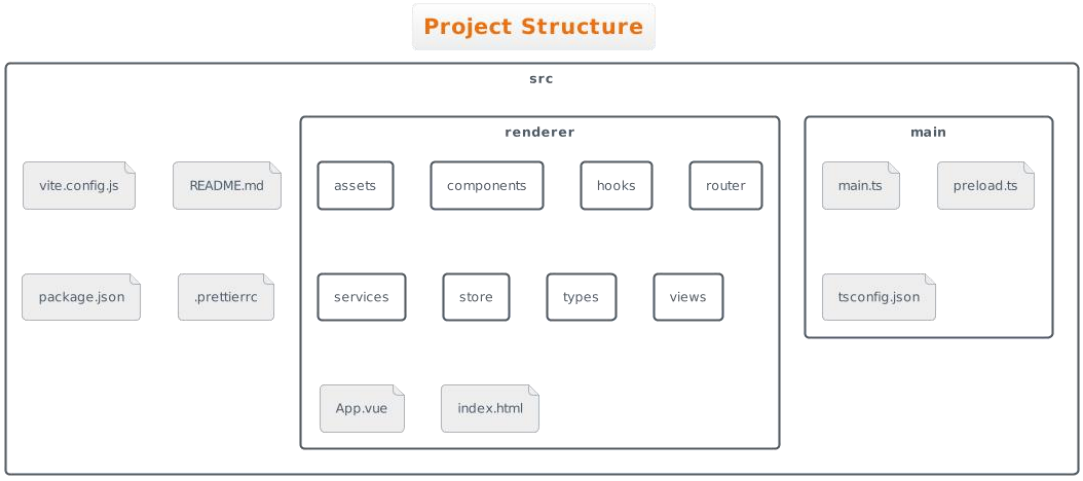


图 5.4 项目结构

5.3.1 主页面实现

主页面 MainWindow 采用双栏布局，使用 naive-ui 进行 UI 设计。页面上方左侧为文本框，接收用户输入，通过回调函数，将其存储在状态管理 store 中的 message 中，它的 disabled 属性由响应式变量 loading 和 showKg 一同控制，在点击获取知识图谱后进入不可编辑状态。

```
<n-input
  :disabled="loading||!showKg"
  maxlength="1024"
  show-count
  clearable
  v-model:value="originText"
  type="textarea"
  placeholder="请输入文本"
  :autosize="{ minRows: 5, maxRows: 7 }"
  class="default-content"
  @change="handleInputChange"
>
</n-input>
```

图 5.5 input 文本框

右侧有四个按钮，分别为获取知识图谱、选择主题和重置按钮以及一个隐藏按钮生成 PPT。

获取知识图谱按钮负责核心功能的实现。按钮文字由响应式变量 buttonLable 控制，其加载状态即由响应式变量 loading 控制。除此之外，通过设置 v-if 属性，实现“获取知识图谱”按钮与“生成 PPT”按钮切换，在本按钮中由响应式变量 showKg 控制。

隐藏按钮生成 PPT，它在 showKg 为 false 时显示，其主要负责实现 PPT 生成与保存操作，因此安排在知识图谱可视化后、已完成 PPT 数据获取时再显示。

选择主题按钮控制一个 Modal 框的开关，该 Modal 提供 PPT 主题的选择与预览，用户可根据自身需求选择对应 PPT 主题。主题的选择将决定生成 PPT 时所用 PPT 模板方法。

```
<n-modal v-model:show="themeModalVisible">
  <n-card
    class="theme-modal-container"
    title="选择主题"
    :bordered="false"
    size="huge"
    role="dialog"
    aria-modal="true"
  >
    <n-select v-model:value="currentTheme" :options="themeOptions"  clearable></n-select>
    <div class="preview-container">
      <n-image :src="currentTheme ? imageSrcs[currentTheme] : ''" width="300" />
    </div>
  </n-card>
</n-modal>
```

图 5.6 ThemeModal

重置按钮负责恢复系统至初始状态，将所有响应式变量、状态管理 store 重置回原始值。用户希望重新填写数据或者对当前结果不够满意时可以点击此按钮进行重置。本按钮能保障系统的稳定性，提升用户体验。

页面下方为封装的组件 KnowledgeGraphByEcharts，负责容纳知识图谱，实现知识图谱可视化，该图谱能够进行拖拽、放大缩小、移动等操作，开发者也可以根据用户选择打开 linkLable，展示连接关系名称。

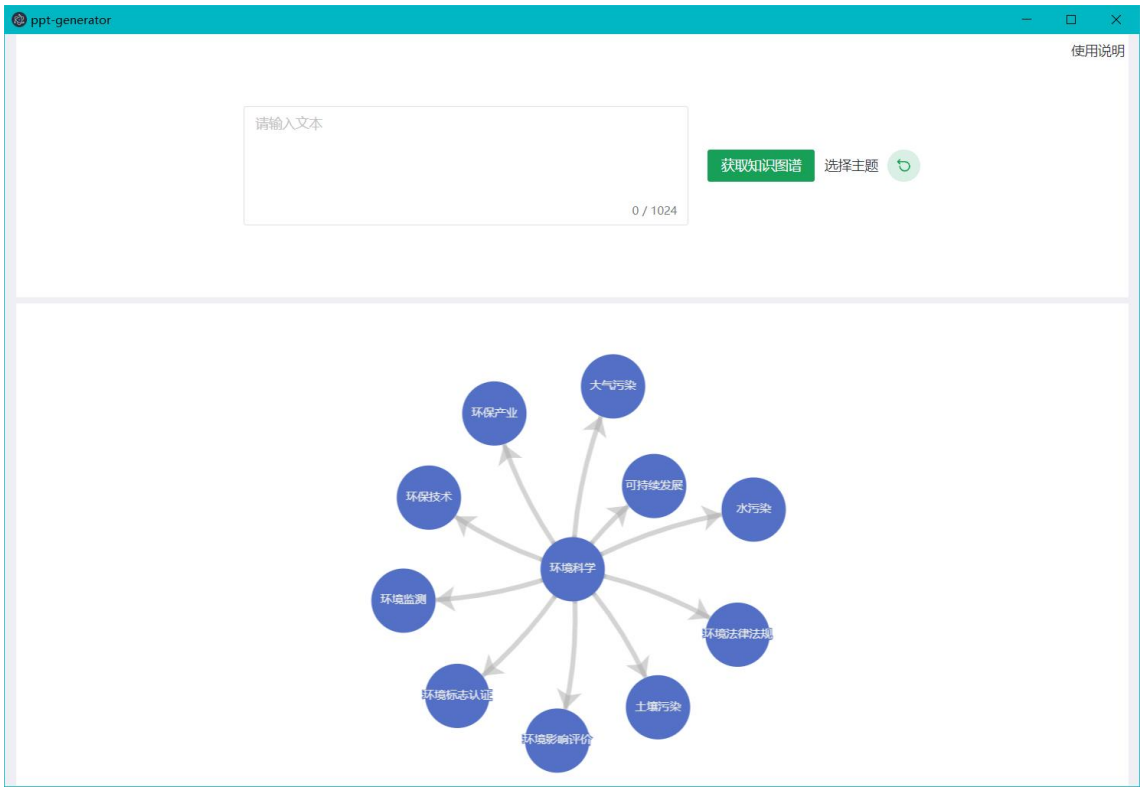


图 5.7 主页面

5.3.2 知识图谱可视化实现

在本文第三章已经详细说明了知识图谱生成及可视化方法实现过程，在本章中将着重于展示方法的使用过程与页面效果。

首先在 KnowledgeGraphByEcharts 组件中申明一个 div 标签，其绑定响应式对象 chart。接着，在 script 代码块中使用 mounted，完成默认数据的知识图谱可视化。然后，使用组合式 API 中的 watch 监控状态管理中的 kgData，根据知识图谱数据的更新，实时刷新组件状态。

```
mounted() {
  // 获取 Echarts 实例
  const store = useStore()
  const chart = echarts.init(this.$refs.chart)
  const nodes = store.kgData.nodes.length>0? store.kgData.nodes:defaultKg.nodes
  const links = store.kgData.links.length>0? store.kgData.links:defaultKg.links
  const {generateOption} = useOption()
  const option = generateOption(nodes,links)
  // 渲染图表
  chart.setOption(option)
  watch(() => store.kgData, (newVal) => {
    const nodes = newVal.nodes.length > 0 ? newVal.nodes : defaultKg.nodes
    const links = newVal.links.length > 0 ? newVal.links : defaultKg.links
    const option = generateOption(nodes, links)
    chart.setOption(option)
  })
}
```

图 5.8 动态更新知识图谱代码

下面展示功能过程，样例数据来自《环境科学》书中内容。

（1）用户输入样例数据，并点击“获取知识图谱”按钮：

图 5.9 知识图谱可视化 step1

（2）客户端包装好请求 prompt 传递给服务端接口/dialogue，服务端接收到数据后调用 OpenAI API，并将响应结果返回客户端。

```
app.post("/dialogue", async (req, res) => {
  try {
    const messages = req.body.prompt;

    const response = await axios({
      method: "POST",
      url: "https://api.openai.com/v1/chat/completions",
      headers: {
        "Content-Type": "application/json",
        Authorization: `Bearer ${OPENAI_API_KEY}`,
      },
      data: {
        model:"gpt-3.5-turbo",
        messages: [{"role": "user", "content": messages}],
        max_tokens: 2048,
        n: 1,
        stop: ["\n"]
      }
    });

    const text = response.data.choices[0].text.trim();

    res.json({ response: text });
  } catch (error) {
    console.log(error);
    res.json({ response: "An error occurred." });
  }
});
```

图 5.10 后端接口

（3）客户端接收到返回结果后，进行 JSON.parse 操作，并存储入 store.kgData 中。watch 方法检测到 kgData 变更，重新挂载 KnowledgeGraphByEcharts 组件，实现知识图谱视图更新。

图 5.11 知识图谱可视化 step2

5.3.3 PPT 教案自动生成实现

在本文第四章已经详细说明了 PPT 教案自动生成实现过程，在本章中将着重于展示方法的使用过程与页面效果。

如上图 5.10 所示，此时，buttonLabel 已更换为“获取 PPT 数据中”，在底层代码中，最初的点击回调函数已经进行到第二个异步请求——请求 PPT 数据。

```
const handleText = async () => {
  loading.value = true
  const requestPptPrompt = pptPrompt + originText.value
  const requestKgPrompt = kgPrompt + originText.value
  const kgData = await getKgData(requestKgPrompt)
  store.setKgData(kgData)
  buttonLabel.value = '获取PPT数据中'
  const pptData = await getPPTData(requestPptPrompt)
  store.setPptData(pptData)
  loading.value = false
  showKg.value = false
}
```

图 5.12 按钮回调函数

（1）知识图谱可视化后，客户端以知识图谱作为纯文字知识脉络和体系结构构建 PPT 数据请求 prompt，向服务端发起请求，收到服务端响应数据后，将数据处理成为 PPT 模板字符串。showKg 设置为 false，“生成 PPT”按钮显示。

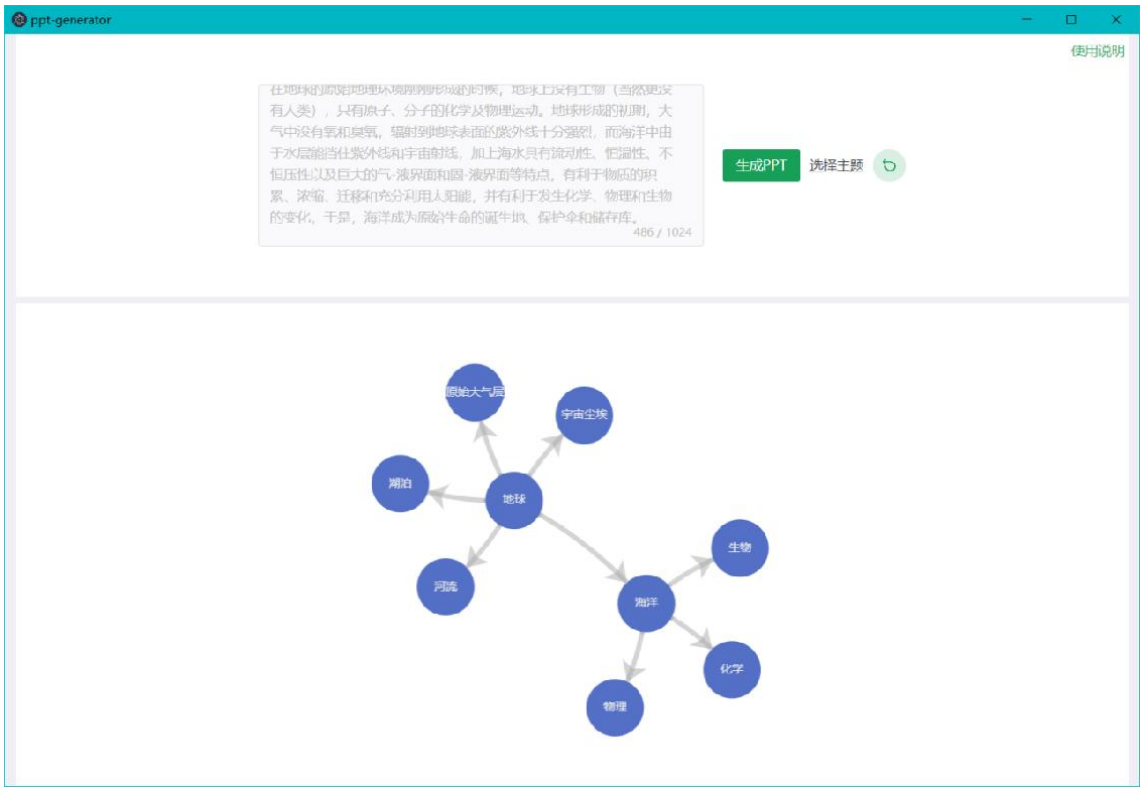


图 5.13 PPT 教案生成 step1

（2）用户点击选择主题按钮，跳出 ThemeModal 后，用户再通过下拉选择框结合预览主题图片挑选期望的 PPT 模板。该 Modal 可通过点击遮罩关闭。

图 5.14 ThemeModalView

（3）点击生成 PPT 按钮，弹出资源管理器，用户自行决定 pptx 文件保存的路径和命名。生成的 PPT 如下所示：

图 5.15 生成的 PPT

5.4 系统测试

系统测试是软件开发过程中的关键步骤，旨在确保软件能够按预期运行。通过这一过程，可以验证系统的运行状况和性能是否符合预期。本章节主要内容为对本文研究开发的系统的测试，通过 Vite 打包构建可执行的桌面应用程序，服务端在 node.js 上运行，对客户端－服务端的功能和性能进行测试。

5.4.1 测试环境

客户端硬件：DELL 灵越 7591，CPU：英特尔酷睿 i7-9750H，内存：16GB，硬盘：256GB SSD

客户端操作系统： 64 位 Windows 10 专业版。

Node.js 版本： v16.13.0。

Vue 版本： 3.2.40。

Electron 版本：22.0.2

5.4.2 系统功能性测试

系统功能测试具有重要意义，它是对各个组件进行联合测试，旨在确保每个组件的功能按预期运行并保证组件间的连通性。通过对整个系统的功能实现进行全面验证，系统功能测试能够确保系统在实际运行中满足用户需求和业务目标。此外，功能测试也有助于及时发现潜在的问题和缺陷，以此降低未来的维护成本和风险。通过对系统功能测试的重视和执行，开发团队可以提高软件质量，提升用户满意度，为项目的成功奠定基础。

本系统测试主要内容包括：用户上传纯文字资料、知识图谱生成及可视化任务、PPT 教案自动生成任务。

（1）用户上传纯文字资料是进行后两项任务的必要前提，只有系统能正确解析、包装用户提供的资料并调动服务端接口，后续任务才能使用正确的数据进行执行。开发者可以在客户端中通过 devtools，在控制台处观察数据的处理结果，本功能测试用例均来自《环境科学》一书，随机选取书中每章核心内容进行测试。

表 5.1 用户上传资料功能测试

序号	测试用例	测试内容	预期目标	结果
1	空文本资料	用户上传纯文字资料	客户端程序识别为 null，弹出提示要求用户重新输入。	达成预期
2	《环境科学》第一章绪论第三节环境污染与人类健康内容	用户上传纯文字资料	客户端程序成功读取用户上传的资料。	达成预期
3	《环境科学》第二章第二节人口与环境内容	用户上传纯文字资料	客户端程序成功读取用户上传的资料。	达成预期
4	《环境科学》第三章第一节大气结构与组成内容	用户上传纯文字资料	客户端程序成功读取用户上传的资料。	达成预期
5	《环境科学》第六章第二节环境保护法内容	用户上传纯文字资料	客户端程序成功读取用户上传的资料。	达成预期
6	《环境科学》第七章可持续发展内容（字数远超系统预期）	用户上传纯文字资料	客户端检测文本超出，限制为 1024 字。	达成预期

测试步骤如下：从《环境科学》书籍中选取每章首节内容作为输入数据，将选取的章节文本依次输入到系统中的文本获取模块中。调用数据获取及处理相关方法，检查系统是否能正确接收和处理输入的文本数据，最后验证系统对输入文本的处理结果是否符合预期功能需求。

测试结果：关于《环境科学》书籍内容的知识图谱数据获取以及 PPT 教案数据获取，系统响应数据能够做到结构稳定、内容联系紧密，符合核心功能模块所需数据要求。

（2）知识图谱生成及可视化任务是系统的核心功能之一，对于这项功能的测试，本系统对测试用例的选择需要做到高覆盖率、边界条件、异常情况和错误处理等，例如空数据、大量数据、无结构文字和特殊字符文字等均需作为本功能测试用例。

表 5.2 知识图谱生成功能测试

序号	测试用例	测试内容	预期目标	结果

序号	测试用例	测试内容	预期目标	结果
1	空数据	知识图谱生成及可视化	客户端不接收数据，不进行知识图谱视图更新	达成预期
2	《环境科学》第一章绪论第三节环境污染与人类健康内容	知识图谱生成及可视化	客户端生成对应内容知识图谱并动态更新视图	达成预期
3	《环境科学》第二章第二节人口与环境内容	知识图谱生成及可视化	客户端生成对应内容知识图谱并动态更新视图	达成预期
4	《环境科学》第三章第一节大气结构与组成内容（涉及大量化学式和特殊符号）	知识图谱生成及可视化	客户端生成对应内容知识图谱并动态更新视图	基本达成预期，部分未编码字符异常
5	《环境科学》第六章第二节环境保护法内容（抽象内容）	知识图谱生成及可视化	客户端生成对应内容知识图谱并动态更新视图	达成预期
6	《环境科学》第七章可持续发展内容（字数远超系统预期）	知识图谱生成及可视化	客户端检测文本溢出，不更新知识图谱视图	达成预期

测试步骤如下：知识图谱生成及可视化主要依赖知识图谱数据结构中的结点数组和边数组，通过控制用以获取知识图谱的文本资料中的内容，完成不同结点数组和边数组的生成即测试用例构建。使用构建的测试用例调用知识图谱生成及可视化方法，观察知识图谱视图更新结果以及控制台可能的提示信息，判断系统对知识图谱生成及可视化的效果是否符合预期。

测试结果：在空数据、合理数据、大量数据的情况下，系统均能较好地完成知识图谱生成及可视化功能；无结构文字或者使知识图谱数据结构异常的文字，系统能将存在的结点和边进行可视化；特殊字符数据，系统有概率存在编码异常，文字显示问题。

（3）PPT 教案自动生成是系统的另一核心功能，对于这项功能的测试，本系统需要专注于考察产物——即 PPT 教案文件的质量。评判一份 PPT 教案文件的质量需要考虑内容的合理性、PPT 样式布局、PPT 页面划分等因素，测试用例则采用《环境科学》书籍中的数据，具体而言，尽量从该书的每一章中挑选出讲述相近主题的段落作为输入数据，可以在验证 PPT 教案文件质量时更全面。

表 5.3 PPT 生成功能测试

序号	测试用例	测试内容	预期目标	结果
1	空数据	基于知识图谱的 PPT 教案自动生成	客户端不接收数据，不提供生成 PPT 方法	达成预期

序号	测试用例	测试内容	预期目标	结果
2	《环境科学》第一章绪论第三节环境污染与人类健康内容	基于知识图谱的 PPT 教案自动生成	客户端能够生成符合用例的 PPT 教案	达成预期
3	《环境科学》第二章第二节人口与环境内容	基于知识图谱的 PPT 教案自动生成	客户端能够生成符合用例的 PPT 教案	达成预期
4	《环境科学》第三章第一节大气结构与组成内容	基于知识图谱的 PPT 教案自动生成	客户端能够生成符合用例的 PPT 教案	达成预期
5	《环境科学》第六章第二节环境保护法内容（抽象内容）	基于知识图谱的 PPT 教案自动生成	客户端能够生成符合用例的 PPT 教案	达成预期，但与知识图谱联系较小。
6	《环境科学》第七章可持续发展内容（字数远超系统预期）	基于知识图谱的 PPT 教案自动生成	客户端检测文本溢出，不提供生成 PPT 方法	达成预期

测试步骤：输入测试用例，生成对应的 PPT 教案文件至本地，打开文件根据预期目标评估文件质量。

测试结果：本系统能稳定产出符合用户所设模板的 PPT 教案文件。

5.4.2 系统非功能性测试

系统性能测试的重要意义在于全面评估整个系统或某个组件的性能表现。通过实施一系列有针对性的测试策略深入检测系统的响应时间、运行效率、资源利用率等关键性能指标。性能测试的目的是确保系统在各种工作负载和环境条件下能够稳定运行，并满足用户的性能需求。就本系统而言，性能测试的主要内容是应用的交互速度、知识图谱可视化效率、知识图谱视图操作流畅性、PPT 教案生成效率。

依据系统实际应用场景和需求，本文设计了对应测试用例和测试目标，测试结果如表 5.4 所示。

表 5.4 非功能性测试

序号	测试内容	测试用例	目标	结果
1	交互速度	上传文本，点击获取知识图谱按钮	按钮能正确进入 loading 状态，并在响应结果后结束 loading。	达成预期
2	交互速度	点击生成 PPT 按钮	1s 内生成 PPT，并由用户自由决定文件生成路径和命名	达成预期

序号	测试内容	测试用例	目标	结果
3	知识图谱可视化效率	输入《环境科学》最大字数文本，获取结点数组和边数组，渲染知识图谱。例如：《环境科学》第一章第三节环境污染	数据响应后，在 500ms 内完成知识图谱视图更新。	达成预期
4	知识图谱视图操作流畅性	对知识图谱视图进行各种操作，例如缩放、拉拽、拖动和选中节点等。	视图能灵活响应用户的操作，延迟应当小于 50ms	达成预期
5	PPT 教案生成效率	使用《环境科学》最大字数文本数据生成 PPT 教案。例如：《环境科学》第一章第四节环境科学	1s 内生成 PPT，并由用户自由决定文件生成路径和命名	达成预期

6 总结与展望

本章节主要内容是对知识图谱生成及可视化与 PPT 教案自动生成方法研究与系统设计及实现过程进行总结概述，同时对未来的发展可能以及前景方向进行展望。

6.1 结论

在第六章中，系统的功能测试和非功能测试结果很好地达成了项目预期。本系统实现了纯文字生成知识图谱并可视化、纯文字自动生成 PPT 教案，为教育自动化领域提供了一种创新的解决方案，有助于提高教学质量、减轻教师负担，并为课堂内容呈现方式带来新的可能性。本文依次从研究背景及意义、关键技术分析、知识图谱生成及可视化、教案自动生成方法、系统设计与实现以及系统测试等方面进行了深入讨论。

在研究背景和意义以及关键技术分析章节。本文详细阐述了知识图谱生成及可视化与教案自动生成系统的产生背景和意义，介绍了国内外相关研究现状，并说明了本系统存在的意义及解决的问题，同时概述了文章的主要内容。在本文系统研究开发的过程中，使用了许多先进的开发框架和技术，如 Vue3、Electron、Typescript、Node.js、OpenAI API 的 GPT 模型、Echarts 力导向图、PptxGenjs 等。本文对这些主要技术进行了详细介绍，并对系统核心方法，进行了深入的探讨。

在知识图谱生成及可视化章节，本文首先对知识图谱的结构设计进行了详细分析，梳理了知识图谱中的关键元素和关系。在这之后讨论了知识图谱数据获取的方法，研究了从纯文字转化知识图谱的方法。在本章第三部分重点介绍了知识图谱可视化的核心方式，对可视化技术进行了深入的探讨。最后，对知识图谱相关方法的封装和调用进行了描述，让方法便于组件使用，降低系统耦合。

在教案自动生成方法章节，本章第一部分详细分析了 PPT 教案结构的设计，明确了 PPT 教案中的页面和布局设计。第二部分深入探讨了 PPT 数据获取与处理的方法，阐述了将纯文字转化为 PPT 模板文字的方式。在第三部分介绍了 PptxGenJs

库的使用和拓展，对其在实际应用中的应用方法进行了详细说明。最后，对本方法的封装和调用进行了综述，以便于在实际项目中使用和扩展。

在系统设计、实现和测试章节，从需求分析出发，明确系统的开发需求。接着对系统的整体架构和功能进行设计。设计完成之后，根据系统的设计进行编码实现，完成对各个模块的开发。在完成系统实现后，设计了不同的测试用例对系统进行各项测试。测试结果表示本系统已经符合在需求分析阶段提出各项需求，达到预期开发效果。

6.2 展望

本系统根据用户上传的纯文字资料，为用户提供资料的知识图谱和 PPT 教案文件，协助其梳理知识体系，有效地减少了用户分析总结教案的时间，提升教学效率。但是系统还存在一些未来可以提升的方向，因此在本节对未来发展提出如下展望。

第一，更好的自然语言处理模型，2023 年由 OpenAI 研发的 GPT 模型横空出世，其在理解、生成和处理自然语言方面取得了显著的进展，为各种应用场景提供了更为精准和高效的语言处理能力。而随着技术的持续迭代，自然语言处理模型也会更加智能、强大，本系统获取到的数据也会更加稳定、精确。

第二，更精细的数据分析和处理，可以设计出更详细的数据结构和 PPT 模板结构。通过精细、丰富的数据，可以生成出细节更好的知识图谱与 PPT 教案。

第三，可以将系统的核心功能进行封装，上传到 npm 包中，开源给广大开发者们使用。

参考文献

[1] Wang, Qi, and Shengquan Yu.Investigating the mechanism for automatic generation of online learning resources[J].Interactive Learning Environments (2022): 1-21.

[2] 龙湘犁, 何美琴. 环境科学与工程概论[M]. 华东理工大学出版社, 2007.

[3] 邱均平,方国平. 基于知识图谱的中外自然语言处理研究的对比分析[J]. 现代图书情报技术,2014(12):51-61.

[4] 宋梦雪. 基于自然语言处理的教育领域知识图谱的构建[D]. 山东:青岛理工大学,2020.

[5] 姜永清, 赵宪佳. 基于文本的关键词提取方法研究与实现[J]. 信息与电脑, 2020, 32(5):4.

[6] 李全. 基于多层类别主题图模型的教育文本分类方法[J]. 计算机与现代化,2016(7):55-59,67. DOI:10.3969/j.issn.1006-2475.2016.07.011.

[7] 赵蔚, 姜强, 王朋娇,等. 本体驱动的 e-Learning 知识资源个性化推荐研究[J]. 中国电化教育, 2015(5):6.

[8] 马健. 基于人工智能技术的智能教学系统设计与应用研究[J]. 电脑编程技巧与维护,2019(9):135-137. DOI:10.3969/j.issn.1006-4052.2019.09.049.

[9] 张吉祥,张祥森,武长旭,等. 知识图谱构建技术综述[J]. 计算机工程,2022,48(3):23-37. DOI:10.19678/j.issn.1000-3428.0061803.

[10] Bhutoria, Aditi. "Personalized education and artificial intelligence in United States, China, and India: A systematic Review using a Human-In-The-Loop model."[J] Computers and Education: Artificial Intelligence (2022): 100068.

[11] TILKOV, STEFAN, VINOSKI, STEVE. Node.js: Using JavaScript to Build High-Performance Network Programs[J].IEEE internet computing,2010,14(6):80-83.

[12] 褚孔统, 朱勇. 开发跨平台桌面应用的探讨[J]. 机电信息, 2019(33):2.

[13] Noack, Andreas. "Modularity clustering is force-directed layout."[J] Physical review E 79.2 (2009): 026102.

[14] Singhal, Amit. "Introducing the knowledge graph: things, not strings."[N] Official google blog 5.16 (2012): 3.

[15] VRANDECIC, DENNY, KRTOETZSCH, MARKUS. Wikidata: A Free Collaborative Knowledgebase[J]. Communications of the ACM,2014,57(10):78-85. DOI:10.1145/2629489.

[16] ROBERT SPEER, CATHERINE HAVASI. Representing General Relational Knowledge in ConceptNet 5[C]. //8th International Conference on Language Resources and Evaluation 2012 (LREC-2012) Volume 5 of 5.:Association for Computational Linguistics, 2012:3679-3686.

[17] 赵聪. 可视化库 D3.js 的应用研究[J]. 信息技术与信息化,2015(2):107-109. DOI:10.3969/j.issn.1672-9528.2015.2.47.

[18] 权西瑞,王凯,王小飞,等. 基于 Three.js 的全景漫游产品设计与实现[J]. 地理空间信息,2022,20(7):71-73,119. DOI:10.3969/j.issn.1672-4623.2022.07.015.

[19] 刘雪瑶. 药学知识图谱可视化分析系统的设计与实现[D]. 山东:中国海洋大学,2020.

[20] TED PATTISON. Office Space: Building Office Open XML Files[J]. MSDN magazine,2007,22(2):29-30,32-33,35-36-0.

[21] Information technology - Document description and processing languages - Office Open XML File Formats - Part 2: Open Packaging Conventions:UNI CEI ISO/IEC 29500-2-2013[S]. 2013.

致谢

时间荏苒，白驹过隙，在这毕设的尾声，四年的大学生活也即将画上句号。回首这四年来的点点滴滴，我深感自己的成长与变化，我要感谢自己四年来的努力；感谢武汉大学，为我提供了优良的学习与生活环境；对于那些不辞辛劳、耐心指导的老师们，我倍感敬意，感谢他们传道授业、解惑启迪；同时，我也要感激那些陪伴我共度美好时光的同学们，以及给予我无私帮助的家人们。