1. 构建知识图谱

1.1 数据采集

1.1.1 非结构化数据采集

1.1.1.1书籍等

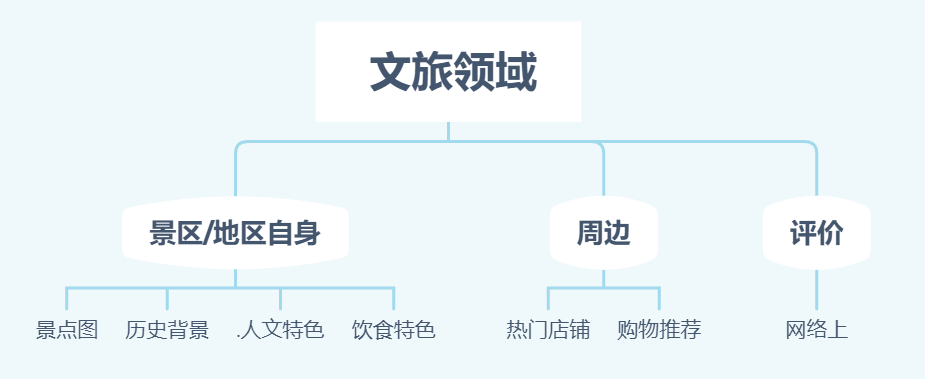
1.1.2 半结构化数据采集

1.1.2.1基于beautifulsoup+requests+Selenium的爬虫系统,再结合正则表达式和Xpath的配合解析得到.本文选择xpath表达式定位所需数据元素。首先我们依据Selenium模仿用户操作行为,进行网页获取,之后通过xpath解析网页数据,对文旅各项信息进行匹配捕捉,并进行爬取。 本文选择xpath表达式定位所需数据元素。首先我们依据Selenium模仿用户操作行为,进行网页获取,之后通过xpath解析网页数据,对汽车各项信息进行匹配捕捉,并进行爬取。 爬取下来的数据存储为CSV格式,方便后续的数据处理

1.1.2.2 后以较专业的百度百科数据作为Scrapy爬取的来源。Scrapy是一种以Python为基础的爬虫引擎框架

1.2 模式层设计

1.2.1 在知识图谱的抽象层上，由不同的概念模型组成的知识图谱范例形成知识本体。本文在充分了解某地区文旅领域知识体系后，基于病文旅模型定义了所需要的知识结构及关系、属性类型，设定对应的约束和清晰的知识提取界限。 文旅领域一般包括一个景区/地区自身，周边以及评价这几个点。根据这几点还能向下进行分类。具体概念层次图如下图

 1.3数据层构建，建立三元组数据

采用规则的方式提取半结构性的文本数据，并运用深度学习的方式对无结构的文本数据进行抽取。根据获得的数据，将数据分类成多个实体，并且构建实体-关系-实体这样的三元组数据 据语料库的特点,基于BIOES序列标注方法对多文本进行标注,并采用BiLSTM-CRF模型进行文本特征提取,能够有效实现长文本提取多个三元组数据。

1.4数据层构建

将经过处理的汽车领域数据以RDF文件形式存放到Neo4j数据库中。Neo4j是属于NoSQL类型的图形化数据库 ，Neo4j的数据导入方式有多种,包括直接载入CSV数据格式、编写Cypher语句进行导入、使用官方的Neo4j-import工具进行导入。编写Cypher导入速度较慢,查漏补缺可以使用。所以本文使用直接结合Python导入CSV数据。再查漏补缺时自行编写Cypher进行导入

1. 问答系统算法框架

2.1问答系统算法框架

2.1.1

BBA-CRF实体识别算法 （BBA-CRF模型的实现+基于字典匹配的实体识别 +实体链接 ）+B-CNN关系识别算法（B-CNN模型的实现 +相似度计算 ）

2.1.2

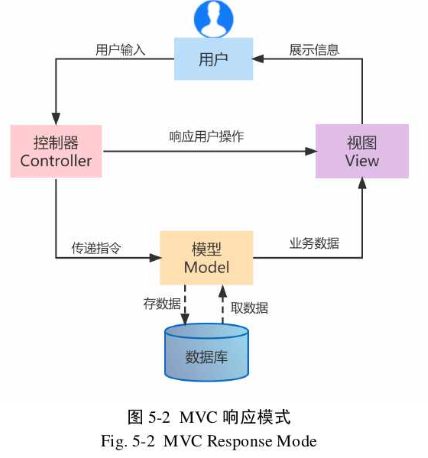
问句意图分类算法（基于Word2Vec的词向量模型+意图分类数据集的构建+基于TextCNN的意图分类）+问句命名实体识别（基于AC自动机的苹果问句实体识别+基于BiLSTM-CRF模型的苹果问句实体识别）+基于苹果病虫害知识图谱的答案检索和评估（基于Cypher语句的答案检索） 使用基于Word2Vec词向量的TextCNN模型进行问句意图的分类,使用AC自动机匹配和BiLSTM+CRF的形式对问句实体进行提取,最后使用Cypher构建查询语句对问句三元组检索答案,通过构建的答案模板组合成自然语言陈述句输出。最后对实体类、实体关系类、实体属性类问句分别进行试验评估

1. 系统功能实现

3.1 表示层实现

3.1.1 使用Python设计了一个B/S结构的汽车领域智能问答系统,整个系统主要基于Flask+vue进行开发。前端主要基于Vue进行开发,前端主要的语言以js、html为主。

3.1.2采用 Python编译语言+ Flask框架+Neo4j数据存储的方式,在web端进行交互。Python在开发过程中得到了很好的扩展和可移植性,它拥有很多强大的库,其中Flask是 Python中常见的一个轻量级框架,可用于中小规模网站的搭建。系统采用 MVC模式进行实现,其结构如图5-2所示。其中M代表模型层(Model),它用于对系统的业务逻辑进行处理;V表示可视化层(View),主要工作是显示网页和结果,让用户能够进行交互;C表示控制器(Controller),它作为不同层次之间的连接,主要通过对模型和视图的调用来实现用户的要求。架构采用四层主体层+基础技术层相结合的设计,利用Ajax技术将数据传输到后端,实现了用户在 Web上进行人机信息交互



3.2前后端分工

3.2.1前端开发人员专注于 UI 交互和展示逻辑的设计，后端开发人员专注于业务逻辑和数

据存储等，前后端通过接口进行数据的交换。

前后端可以并行开发，提升开发效率 前后端系统独立部署，提升性能

3.2.2 MVVM模式

实现模型层和视图层的分离，具备有低耦合、高 复用性、独立开发和方便测试的优点。 MVVM模式包括三个部分，模型层（Model） 存放着程序所需要的数据，视图层（View）是系统的 UI 界 面部分，视图模型层（ViewModel）是 MVVM 模式的核心 部分，实现了视图层与模型层的解耦

3.3 前后端分离开发框架

Vue 是一个关注于视图层，用于构建用户界面的渐进式轻量级 JavaScript 框架。Vue 最核心的功能包括组件化的系 统和响应式的双向数据绑定。组件是可复用的 Vue 实例， 是 Vue 最强大的功能之一，可以扩展 HTML 元素，封装可重用的代码。Vue 的组件化系统可以通过使用小型的、独立可复用的组件来构建一个大型复杂的Web 应用程序。

3.4 Spring Boot

Spring Boot 是目前在 Java 中比较热门的一个微服务框架 前端系统采用 vue+Element UI 组件实现，后端系统使用 Spring Boot 框架开发，前后端之间通过 RESTful 架构的接口进行通信。

3.5 CROS 跨域访问

当前后端系统分别部署在不同服务器上，通过 API 接口 通信时，会面临跨域的问题，浏览器会组织Ajax的访问请求，CORS（跨域资源共享）是跨域的的一种解决方案，可以通过设置 CorsFilter 过滤器解决跨域问题。

3.6前端系统开发：

系统前端部分采用基于 vue+element UI 框架实现，首先通过 npm 安装 vue-cli 脚手架，通过脚手架可以快速生成Vue 项目基础的架构。然后在项目中引入 Vue Router 前端路由器和 element UI 组件库，Vue Router 是 Vue 的官方路由器，通过配置 Vue Router，在前端根据不同的 URL 渲染不用的

Vue 组件展示，而无需后台服务器的参与，因此让 Vue.js 构建单页应用程序（SAP）变得更加容易。

在系统中通过安装和配置 axios 库来实现 AJAX 请求，axios 是基于 promise 的 HTTP 网络请求库。因为采用前后端分离模式开发系统，用户的认证通过 web token 完成。因此在后端提供的 api 接口中，除了登录接口之外，每个接口都需要 token 的验证。当用户第一次登陆成功后，web 端会接收到服务器返回的 token，保存到浏览器的 sessionStorage 中。在 axios 上添加一个全局的请求拦截器，保证除了登陆以外的 api 请求，都会将 token 添加到 HTTP 的头部的Authorization 对象中。当 token 丢失或失效的时候，设置路由守卫，返回登陆页重新登陆。