
2020 届硕士专业学位论文

分类号: _____

学校代码: 10269

密 级: _____

学 号: 71174500104



華東師範大學

East China Normal University

硕士专业学位论文

MASTER' S DISSERTATION

论文题目：基于 Bot 框架和 Chrome 扩展的出行旅游 AI 辅助系统

院 系: 软件工程学院

专业学位类别: 工程硕士

专业学位领域: 软件工程

论文指导教师: 彭超 副教授

论 文 作 者: 蒲昱

2020 年 11 月

Dissertation for master degree in 2020

Student ID: 71174500104

University code:10269

East China Normal University

**Title: Intelligent Travel Recommendation System
Based on Chrome Extension and Bot Framework**

Department: Software Engineering Institute

Type: Master of Engineering

Domain: Software Engineering

Supervisor: Associate Prof. Chao Peng

Candidate: Yu Pu

November 2020

华东师范大学学位论文原创性声明

郑重声明：本人呈交的学位论文《基于 Bot 框架和 Chrome 扩展的出行旅游 AI 辅助系统》，是在华东师范大学攻读硕士/博士（请勾选）学位期间，在导师的指导下进行的研究工作及取得的研究成果。除文中已经注明引用的内容外，本论文不包含其他个人已经发表或撰写过的研究成果。对本文的研究做出重要贡献的个人和集体，均已在文中作了明确说明并表示谢意。

作者签名： 蒲昱

日期：2020 年 11 月 17 日

华东师范大学学位论文著作权使用声明

《基于 Bot 框架和 Chrome 扩展的出行旅游 AI 辅助系统》系本人在华东师范大学攻读学位期间在导师指导下完成的硕士/博士（请勾选）学位论文，本论文的研究成果归华东师范大学所有。本人同意华东师范大学根据相关规定保留和使用此学位论文，并向主管部门和相关机构如国家图书馆、中信所和“知网”送交学位论文的印刷版和电子版；允许学位论文进入华东师范大学图书馆及数据库被查阅、借阅；同意学校将学位论文加入全国博士、硕士学位论文共建单位数据库进行检索，将学位论文的标题和摘要汇编出版，采用影印、缩印或者其它方式合理复制学位论文。

本学位论文属于（请勾选）

（ ） 1. 经华东师范大学相关部门审查核定的“内部”或“涉密”学位论文*，
于 年 月 日解密，解密后适用上述授权。

☒ 2. 不保密，适用上述授权。

导师签名： 郭超

本人签名： 蒲昱

2020 年 11 月 17 日

* “涉密”学位论文应是已经华东师范大学学位评定委员会办公室或保密委员会审定过的学位论文（需附获批的《华东师范大学研究生申请学位论文“涉密”审批表》方为有效），未经上述部门审定的学位论文均为公开学位论文。此声明栏不填写的，默认为公开学位论文，均适用上述授权）。

蒲昱 硕士学位论文答辩委员会成员名单

姓名	职称	单位	备注
丁岳伟	教授	上海理工大学	主席
琚小明	副教授	华东师范大学	
陈伟婷	副教授	华东师范大学	

内容摘要

近年来，中国的旅游业发展迅猛，2019 年全国旅游总人数达到 60.06 亿人次。但是很多旅行者难以自主完成旅游产品预定，尤其是老年人，往往需要他人帮助。对于这些老年用户，采用对话交互模式可有效减少用户在使用产品过程中的学习流程，可最直观化的将产品展示给用户。为此，本文拟研究嵌入在浏览器中的旅行推荐聊天机器人。机器人通过“聊天对话”方式与用户进行多阶段问答，从中获取用户旅行方面需求，并根据需求向用户推荐合适的旅游服务或信息。

在机器人开发方面，首先使用 Bot 框架开发聊天逻辑框架，从初层次的用户意向获取对话，到深层次的信息提取对话，设计与用户聊天的逻辑结构。开发 LUIS 语言理解模块，在 Bot 框架中通过 LUIS 模块对用户每句话进行语义理解，理解用户需要的服务意向并提取语言中所包含的相关信息数据，进而分析出用户需求。使用 Python 爬虫获取目标旅游门户网站中用于服务信息生成的数据，并创建数据库保存，利用用户相关信息数据与数据库中查询到的数据生成用户所需信息或服务，通过聊天对话回复用户其所需的互联网旅游信息或服务，满足用户个性需求。然后将聊天机器人部署在 Azure 云服务器上，得到云端聊天对话机器人，设计并实现 Chrome 扩展应用，为用户提供对话聊天窗口页面，在扩展中通过 Web Chat 连接发布在 Azure 上面的聊天机器人，简化用户与聊天机器人交互。最终将得到的扩展应用安装在含有 Chromium 内核的浏览器当中，当用户点击扩展应用开关即可弹出聊天窗与聊天机器人聊天，通过聊天获取到符合用户需求的旅游相关信息及服务。

通过此项研究，我们研发出使用聊天方式获取需求并提供服务的聊天机器人软件，相较于以往的旅游互联网网站手工输入需求交互方式更为直观，减少了产品学习流程，优化了用户体验。同时人工智能聊天也减少了客服人力、财力成本，提升互联网旅游中服务方和消费者沟通的效率。

关键词：旅行推荐系统，聊天机器人，Bot 框架，Chrome 扩展应用，LUIS 语言理解

ABSTRACT

In recent years, Chinese tourism industry has developed rapidly. In 2019, the total number of national tourists reached 6 billion. But many travelers are difficult to complete travel product reservations on their own, especially for the elderly, who often need help from others. For these elder users, dialogue interaction mode can effectively reduce the user's learning process of using the product, and display the product to the user most intuitively. So, the main content of this paper is to develop a travel recommendation chat bot, which is embedded in browser. This bot can ask and answer questions with user in multiple stages. And get user's requirement via chat. Then provide travel services or information to user based on their requirements.

In chat bot development, the system's chat logic framework is developed in Bot Framework project. Design the chat logic structure from the basic level of user intention acquisition dialogue to the deeper level of information extraction dialogue. At the same time, develop a customized LUIS (Language Understanding Intelligence Services) module to analysis user language. In Bot Framework project, we use LUIS module to understand the user services intent and get related data information from each sentence user said, so that we can analysis user's requirement from what user said. After that, use the Python Spider in the target travel website to obtain the data used for service information generation, and create a database to save these data. Then generate tourism services and information user want based on the data in database and from user dialog. And response the services and information to user via chat response. When we finish that, we publish chat bot to Azure Cloud Server, in order to generate an Azure Chat Bot. Design and develop Chrome Extension application which can provide chat box in Chrome Extension and connect Azure Chat Bot via Web Chat, so that we can simplify communication between user and chat bot. Finally, install extension application to browser which has Chromium kernel. When user turn on the extension application in extension application bar, it will pop up a chat box to user, and user can chat with Azure

Chat Bot via chat box to get the travel related information and services which user want. Through this research, we provide a chat bot, which can get requirement and provide services via chat. Compared with traditional travel website which need get requirement by user manually input. Our interaction is more intuitive. And it can reduce the product learning process, optimize the user experience. Also, Artificial Intelligence Chat Bot can reduce human cost and financial cost from customer service. And improve the efficiency of communication between the servicers and customers in the internet tourism.

Keywords: Travel Recommendation System, Chat Bot, Bot Framework, Chrome Extension, LUIS (Language Understanding Intelligence Services)

目 录

第一章 引言	1
1.1 研究背景	1
1.2 国内外研究现状	2
1.3 本文研究内容	3
1.4 研究目的与意义	4
1.5 论文创新	4
1.6 论文结构	4
第二章 相关技术介绍	6
2.1 爬虫相关技术	6
2.1.1 Urllib 库	6
2.1.2 Beautiful Soup 库	6
2.1.3 广度优先遍历算法（BFS 算法）	7
2.1.4 HttpWebRequest 类	7
2.2 聊天机器人开发相关技术	7
2.2.1 Bot Framework	7
2.2.2 LUIS（Language Understanding Intelligence Services）	8
2.2.3 Bot Framework Emulator	9
2.2.4 Azure 云服务	9
2.3 Google Chrome Extension	10
2.4 本章小结	10
第三章 系统需求分析	11
3.1 系统整体需求分析	11
3.2 可行性分析	12
3.2.1 技术可行性	12
3.2.2 经济可行性	12
3.2.3 运行可行性	12
3.3 功能性需求分析	13
3.3.1 浏览器界面层需求	13
3.3.2 聊天逻辑层需求	14
3.4 非功能性需求分析	15
3.4.1 软件环境需求	15

3.4.2 软件质量需求	15
3.5 本章小结	16
第四章 系统概要设计	17
4.1 总体结构设计	17
4.2 系统部署	18
4.3 功能模块概要设计	18
4.3.1 导航模块	19
4.3.2 航班预订推荐模块	19
4.3.3 酒店预订推荐模块	19
4.3.4 旅行推荐模块	20
4.3.5 铁路预订推荐模块	20
4.4 数据库概要设计	20
4.4.1 数据表设计	20
4.4.2 数据库的创建与连接	22
4.5 接口概要设计	23
4.5.1 聊天逻辑 Bot Framework 接口	23
4.5.2 LUIS 语言理解接口	24
4.6 本章小结	26
第五章 系统详细设计	27
5.1 数据库模块的设计与实现	28
5.1.1 数据需求	28
5.1.2 离线数据爬取	28
5.2 LUIS 语言理解模块的设计与实现	32
5.2.1 LUIS 模块概述	32
5.2.2 LUIS 模块的意向与实体定义	33
5.2.3 LUIS 模块语料标注与训练	35
5.2.4 LUIS 模块测试	35
5.2.5 LUIS 模块发布	37
5.3 后台聊天逻辑模块的设计与实现	37
5.3.1 后台聊天模块开发概述	37
5.3.2 创建新的 Bot 工程	40
5.3.3 主对话开发	40

5.3.4 航班预定对话开发	41
5.3.5 酒店预定对话开发	45
5.3.6 旅行推荐对话开发	48
5.3.7 火车票预定对话开发	51
5.3.8 后台聊天逻辑模块整体测试	54
5.3.9 模型 Azure 发布	56
5.3.10 Azure 线上聊天机器人服务测试	57
5.4 前端模块的设计与实现	58
5.4.1 开发扩展框架	58
5.4.2 引入 Web Chat 嵌套网页	60
5.4.3 安装扩展至 Chromium 内核浏览器	61
5.5 本章小结	62
第六章 系统测试	63
6.1 测试目的	63
6.2 测试步骤	63
6.3 测试结果分析	63
6.3.1 航班预订对话测试	63
6.3.2 酒店预订对话测试	65
6.3.3 旅行推荐对话测试	66
6.3.4 铁路预订对话测试	67
6.3.5 问题对话测试	68
6.4 压力测试	69
6.5 本章小结	70
第七章 总结与展望	71
7.1 本文工作总结	71
7.2 未来工作展望	72
参考文献	73
致谢	75

第一章 引言

1.1 研究背景

根据文化和旅游部公布的数据,2019 年全国旅游总人数达到 60.06 亿人次,全年旅游收入总计 6.63 万亿元人民币^[1]。旅游成为居民主要的休闲娱乐生活方式,其中老年群体是旅游市场中一个重要部分。根据携程旅行网发布的《老年群体旅游行为报告》数据显示,“65%的老年群体年旅行次数达 3 次以上,老年人旅游出行需求日益庞大。但在老年人旅行预定方式上,找导游、旅行顾问的比例为 13.7%,找子女预定比例为 19.9%,找亲朋帮忙预定比例为 8.9%,一起去的人帮忙预定比例为 12.6%,自主预定比例仅为 17.8%”^[2]。言下之意老年人的旅行需求庞大,造就老年人庞大旅行预定市场,但是同时老年人旅游预定时寻求他人帮助预定比例为 55.1%,是自主预定比例的 3 倍以上,说明大多数老年人难以自主完成旅游产品预定。经过进一步分析,老年人大多寻求他人帮助预定的主要原因有以下两个方面:

在当前互联网大环境中,不同类型旅游互联网产品分类较为分散,如机票酒店预订市场份额较大的携程旅行网、飞猪网、去哪儿网等,旅行信息汇总为主的马蜂窝网,火车票预订的 12306 网站等等^[3],旅游相关信息较为杂乱的分布在多个不同的网站,用户无法通过一个特定网站,一次性的综合获取所有旅游资讯及预订服务。即使相同旅行服务,也存在不同网站价格不同的情况,用户无法快速获取全网最便宜的报价。在线互联网旅行服务中,来回交通预定、酒店预订、旅行服务预定、旅行情报获取等各种不同的信息获取和服务预定需要进入不同的网站平台,这种模式对于老年用户来说是较难独立自主完成,所以需要寻求他人帮助。

旅行相关产品预订与其他互联网产品不一样的是,旅行产品预订存在大量不同信息的检索,面临大量信息交互,例如购买机票需要了解起点、终点、时间等信息。造成当前主要出行互联网网站的预订服务步骤都较为繁琐,需要用户经过很多步骤,才能获取到用户需求信息后,再进行搜索,容易造成因信息填写错误降低了用户体验。同时老年用户对于不同交互模型学习速度较慢,在线互联网

预定产生错误的概率更高。但增加线下服务则需要更多的人工客服辅助用户检索，人力与物力投入增加。

综上所述，当前旅行互联网服务查询与预定由于其信息填写过多过于复杂的自身特性，无法满足老年用户在旅行方面查询与预定的庞大需求。同时旅行产品分布繁杂，缺乏门户导航服务引导用户在网站中寻找自己所需的信息服务。对于一些网络接触较少的用户，很难独立自主地在互联网上获取其所需的旅游出行信息或服务。

1.2 国内外研究现状

伴随互联网技术的快速发展，互联网旅行信息服务查询变得愈加便捷，用户足不出户就可以完成所有旅行信息的获取与服务的预定。但由于旅行产品本身的复杂性和旅行服务的分散性，当前互联网中并没有一个旅行信息汇总平台方便用户查询所需旅行服务，更没有利用聊天机器人来辅助用户搜索获取全网旅行信息服务的软件。

1966 年，世界上最早的聊天机器人 Eliza 诞生。自此开启了人类对于聊天机器人的研究和发展。在当下，随着机器学习和深度神经网络等算法的出现，聊天机器人利用这些算法变得更加智能，聊天机器人也得到了更大的发展空间，更推动了对聊天机器人的研究更上一层楼，很多互联网软件公司推出了属于自己的聊天机器人，例如苹果的 Siri、亚马逊的 Alexa 和微软的 Cortana 等。但在国内，由于中文语言处理相较于英文更为复杂的特点，对聊天机器人的研究起步较晚^[4]。但伴随着聊天机器人技术的快速发展，国内很多互联网企业也开始投入对聊天机器人的研究，并推出了一批属于自己的聊天机器人产品。例如阿里巴巴集团于 2017 年发布其聊天机器人天猫精灵，用于解决用户一些简单基本的需求，类似音乐播放、讲笑话、查天气等服务^[5]。除此以外，还有百度的度秘、微软针对国内市场的微软小冰以及科大讯飞的讯飞语音，都是当前国内场所涌现出来的聊天机器人产品。但是，国内仍然缺少对于可以处理旅行预定服务的聊天机器人的研究。

1.3 本文研究内容

在当前互联网环境下,假定能够让用户通过聊天对话问答的形式提出自己的需要,来获取用户需求,汇总满足用户需求的的不同网站在线旅行服务,再通过聊天回复的方式,向用户提供出其所需要的旅行信息资讯或者服务链接,使用户更加直观地了解和使用相关旅游互联网产品。对于老年用户来说,采用对话交互模式可有效减少用户在产品使用过程中的学习流程,最直观地将产品展示给用户,方便老年用户使用。使用在线聊天机器人实现这样的功能,则又能够省下一定的人力资源与物力成本^[6]。

为解决以上问题,本文主要内容为开发出一种能够通过与用户对话聊天获取用户需求并向用户推荐满足其需求的的不同网站旅游产品以及旅行信息的智能线上 AI 机器人软件。它通过与用户的“聊天对话”,与用户相互“提出问题”与“回答问题”,获取用户不同方面的在线旅行需求,并基于需求向用户推荐合适的旅游出行产品或者帮助用户查询到所需的旅游信息。使用户仅通过对话便可获取到其想要的信息。

本研究将尽可能满足用户在旅游方面的以下几个需求:

1. 火车票预订
2. 机票预订
3. 酒店预订
4. 旅行目的地推荐

为方便用户在浏览过程中随时可以打开并使用本系统,本论文将聊天窗口设计支持在 Chromium 内核浏览器中,方便用户通过浏览器直接使用。Chrome, Edge, Opera 等多种用户常用的浏览器均支持扩展应用的安装使用。同时,设计主流即时通信聊天软件接口,方便用户通过聊天软件获取需求的过程。

综上所述,本文主要工作为从无到有的设计实现旅行推荐聊天机器人软件,并搭载一定接口方便用户快速访问,使用户简洁直观地获取其所需的互联网旅行信息与服务。

1.4 研究目的与意义

根据本文研究内容，本文设计实现了一款通过与用户聊天，获取用户旅行方面需求并向用户推荐符合其需求的旅行相关信息服务聊天机器人软件，主要目的是解决在当前互联网大环境下，用户搜索互联网旅行信息或服务困难的问题，使其能够尽可能简化、直面化、优化地搜索到其想要的旅行信息服务。

研究旅行推荐 AI 聊天机器人主要意义如下：

其一，更加直观地向用户提供服务。通过聊天方式获取信息服务对于用户更加容易接受。

其二，它一定程度上减轻了人工客服的工作负荷以及企业对人工客服的依赖性，用户通过与 AI 机器人对话确定基础需求，把拥有同一类需求的用户带入对应需求聊天分析对话，更加迅速有效的满足用户需求。

其三，AI 智能客服的可塑造性是无穷的，它可以 7*24 小时连续工作，并且还能拥有无限大的“知识库”。

其四，AI 聊天机器人智能客服具有很强的包容性，不会情绪化，会根据指令帮助用户解决问题，凸显了其专业性的特征。

1.4 论文创新

本文主要的创新点有以下两个方面：

1. 设计实现了可以处理旅游推荐问题的复杂逻辑聊天对话机器人。
2. 将聊天机器人嵌入在浏览器扩展应用当中，让用户通过浏览器与聊天机器人交互。

1.5 论文结构

本文共分七章，内容安排如下：

第一章 引言 介绍本文研究背景、国内外研究现状、研究意义以及研究内容等。

第二章 相关技术介绍 通过系统实现所使用的爬虫相关技术，Bot 框架相关

技术, Chrome 扩展相关技术三个方面来介绍本论文所涉及的相关技术与算法。

第三章 系统需求分析 分析模型技术、经济、运行方面的可行性, 需满足的功能需求以及非功能需求。

第四章 系统概要设计 从总体框架结构设计、系统部署设计、功能与逻辑模块设计、数据库结构设计、接口设计这五个方面介绍模型框架的概要设计。

第五章 系统详细设计 主要从数据库模块设计与实现, 后台聊天逻辑模块设计与实现, LUIS 语言理解模块的设计与实现, 前端浏览器扩展应用模块设计与实现和系统整体测试等几方面来介绍本论文的具体实现过程。

第六章 系统测试 对模型进行功能测试和压力测试。

第七章 总结与展望 概述全文完成工作以及研究的意义 本论文相关附录, 感谢语。

第二章 相关技术介绍

2.1 爬虫相关技术

在本系统中，我们需要一定数据支持，从而在之后的聊天中可以基于用户的需求来生成用户所需的旅行信息与服务。为获取这些信息，我们可以通过爬虫技术，批量化请求一些网站网页，并通过解析其页面信息方法来获取，爬虫技术即通过分析单一网页，得到页面架构，并基于页面结构判断数据在页面中的具体位置，我们基于网页结构设置一定规则，使得程序可以根据规则自动查找网页中我们所需数据在网页中的位置，同时通过批量化地请求页面 URL，并且根据我们所设置的规则自动从网页中寻找目标数据出现的位置，获取我们所需数据，实现自动化、批量化地向目标网站获取目标信息的程序或脚本。在本系统中我们所需要使用到的爬虫技术以及相关算法如下。

2.1.1 Urllib 库

Urllib 库是 Python 中用于网络请求的 HTTP/HTTPS 请求库，主要作用是操作 URL 的模块，包含对 URL 发送请求的模块，还有对 URL 的异常做出处理模块，以及对请求 URL 后，获取反馈信息模块^[7]。调用库中的 URL 请求模块可以指定需要爬取网站页面的 URL，即可向目标 URL 发送请求，当成功请求到目标 URL 时候，则可以返回目标网页的 HTML 代码，通过解析页面 HTML 代码即可获取到所需要的数据内容。

2.1.2 Beautiful Soup 库

Beautiful Soup 库是用于解析 HTML 或 XML 格式数据的一种数据解析库，它可以将 HTML 或 XML 格式的数据构建成 Struct Dom 树结构框架，并从 Struct Dom 树中通过一些特定的标签名或者类名对目标数据在树中位置进行定位，并根据定位，从树中提取对应目标数据^[8]。通过 Urllib 库请求目标 URL 获取到的 HTML 代码形式的响应，通过使用 Beautiful Soup 库对目标数据进行解析，在解析中查找特定位置从而提取到网页中的目标数据。

2.1.3 广度优先遍历算法（BFS 算法）

广度优先遍历算法是个基础且常用的图形搜索演算算法。具体的广度优先，我们可以理解成搜索从根节点开始，沿着目标树的层次由浅至深依次遍历树中每个节点，如果在遍历过程中找到了目标节点，则查找终止^[9]。在我们对目标网站批量爬虫抓取网页的过程中使用广度优先遍历算法，从目标网站中选取某一网页作为起点，开始解析该网页的同时获取该目标网页中的相关数据。在获取相关数据时候，继续从网页中搜寻并获取从该网页出发可到达的下一节点网页相关信息，比如下一节点名称，URL 等数据。并将新的可到达节点网页的 URL 及其名称依次加入遍历队列等待算法遍历，以此类推不断地遍历队列中节点从而实现广度优先遍历。当遍历队列为空时，则标志已经完整获取目标网站中所有节点网页的需求信息。

2.1.4 HttpRequest 类

HttpRequest 类是 C#语言中同样用于网络请求 HTTP/HTTPS 请求类，用于向目标 URL 发送请求，接收从目标 URL 得到的反馈，并得到目标网页 HTML 形式的响应^[10]。与 Python 中的 Urllib 类作用基本相同。在本模型中，除一些城市主要信息的离线爬取，我们将使用 Python 的 Urllib 库来完成。但在后台模型中，为了保证页面时效性和减少储存内容，所以我们对于一些当前数据规模较大且无法有效遍历获取的数据内容进行在线爬取。同时由于我们后台 Bot 框架主要为 C#语言构成，我们更需要一种直接可以嵌合的爬虫语言嵌入我们的 Bot 框架系统内，所以我们选用 HttpRequest 类来实现我们在线数据爬取模块。

2.2 聊天机器人开发相关技术

2.2.1 Bot Framework

Bot Framework 是在 Build 2016 大会中，微软公司发布的一款便捷的聊天机器人开发的框架工程。通过 Bot 框架，我们可以实现理解自然语言，并且设计聊天的逻辑框架等，初期已经开放了 22 个 API 供开发者在设计开发聊天机器人

的过程中使用^[11]。开发者则可以通过开发自定义的 Bot Framework 框架工程并按照个性需求搭建符合自己需要的聊天机器人，同时在开发过程中，创建不同类型的对话，实现机器人的聊天逻辑，简化开发人员工作量，丰富了聊天机器人功能。搭建完成后，开发者可轻松地将 Bot 框架开发出来的聊天对话机器人，发布在 Azure 云服务器上，并轻松搭载在 Skype, Teams 等社交媒体上，方便开发者直接调用^[12]。

在我们的模块中，将 Bot 框架作为模型的基础逻辑框架，建立 Bot 框架工程，通过在 Bot 框架工程中创建不同类型的对话，同时编写不同逻辑实现不同对话切换，运用不同的瀑布流模型来设计用户对话框架，同时通过调用 LUIS 语言理解模型等手段解析用户交流内容，提取相关意向与数据，基于用户数据与数据库中的数据生成最终用户所需要的旅行服务和信息，再次通过聊天回复的方式将服务与信息反馈给用户。所以，Bot 框架是我们设计的整个聊天系统的语言逻辑框架。

2.2.2 LUIS (Language Understanding Intelligence Services)

智能语言理解服务 (LUIS) 是一种用于语言智能理解并基于语言判断语言意向同时提取语言数据的 API 接口服务，可在用户对话的自然语言文本中使用一些机器学习和自然语言处理相关技术，对开发者提供的语料进行训练，进而训练出可以通过语料分析意图并提取出目标信息的能力，即模型可以从自然语言中判断开发者自定义的一些意向，同时提取出开发者自定义的一些实体^[13]。开发者可以在 LUIS 平台中可视化地创建语言的相关特征，同时标注自然语言语料，并对语料进行训练生成语言理解训练模型。在 LUIS 模型中，我们可以定义的特征主要为以下两种语言特征：

- 意向 (Intent): 即语言的意图，即为整句话我们需要表达的意思。例如用户提问“苏州的穹窿山开车怎么走？”则表明用户在话语中想要问路意图。
- 实体 (Entity): 即语言中的内容，即整句话中我们可以提取到的部分信

息。例如“今天上班路上有点堵”，我们则可以从用户的语言中提取到时间实体“今天”。

模型训练完成后，我们就可以将自定义的语言理解模型发布到线上。模型发布后，开发者可以在程序中轻松的调用语言理解的模型 API 来使用模型，便捷化语言理解开发的过程。

在模型中，我们需要开发一套 LUIS 模型，使得模型可以判断用户所需旅游服务具体需求意向的种类，同时能够从语言中提取用户语言中需求所对应的相关数据。

2.2.3 Bot Framework Emulator

Bot Framework Emulator 是用于 Bot 框架机器人测试的辅助开发本地测试应用，通过该应用，能够在远程或本地检测和调试聊天对话机器人内部逻辑^[14]。使用此模拟器，开发者可以在 Bot 框架的开发过程中和机器人聊天并检查机器人发送和接收的消息是否符合预期，同时对于一些设计问题可以逐步调试，找到问题。在将 Bot 框架部署到 Azure 云服务之前，可以使用该应用在本机运行和测试。即使还没有将聊天逻辑框架发布至 Azure 服务器，也不影响在开发过程中对逻辑进行测试。减少系统发布次数，简化开发流程。

2.2.4 Azure 云服务

Azure 云服务平台是一款稳定、灵活的云服务平台，为用户提供了数据库、CDN、云服务、云存储、人工智能互联网等稳定、高效、可扩展的云端服务^[15]。Azure 可以作为多种不同应用开发的线上后台，其中包括 Bot Framework 框架和 LUIS 语言理解的服务应用。

在 Bot Framework 开发过程中，将模型发布在 Azure 云平台上，以方便模型的前端模块能够通过连接 Azure 端的应用程序从而连接到后台逻辑模块，以方便不同模块通过接口对模块的调用。同时将 LUIS 语言理解模型也发布在 Azure 云平台上，使得 Bot 框架语言解析相关工作可以直接通过调用模块 API 接口实现，

使系统更加的高内聚、低耦合。

2.3 Google Chrome Extension

Chrome Extension 是一款由 Google 公司开发的允许用户在浏览器上自定义其 Chrome Web 浏览器的应用附加组件。通过开发相关组件，将开发完成的附加扩展应用上传至 Chrome 网上应用商店，可以使更多用户通过 Chrome 网上商店下载扩展应用并在浏览器中体验。扩展应用通常由 HTML、CSS 和 JavaScript 三种前端语言开发并由一定的扩展应用管理框架进行架构。自 2008 年 9 月 2 日发布的第一版本 Chrome 浏览器中加入 Chrome Extension，Chrome 浏览器中大量的浏览器组件大多使用 Chrome 扩展应用开发完成，使得 Chromium 内核浏览器的功能超越了浏览器功能的边界，成为一种开源应用平台^[16]，从而提高了 Chromium 内核浏览器的用户体验并树立其在浏览器内核的主导地位，使之成为当下最主流的浏览器内核。当前主流的 Chrome, Firefox, Opera, Edge 等浏览器均可支持 Chrome 扩展应用。

在我们的模型中，设计开发一款新的 Chrome 扩展应用程序当作系统前端模块，在前端页面中调用我们发布在 Azure 端的聊天逻辑模块即可实现前端与后台的连接，使用户打开浏览器后即可通过页面聊天窗口与我们所设计的应用聊天。

2.4 本章小结

本章通过分析并介绍爬虫相关技术及算法，Bot Framework 相关技术，Chrome 扩展技术，展示实现了本文模型从数据提取到后台逻辑，再到前端界面实现所需要使用的技术手段，接下来的内容会则是介绍模型具有应该满足的需求以及模型的可行性。

第三章 系统需求分析

3.1 系统整体需求分析

软件系统的开发，必须准确了解用户需求。需求分析是整个软件系统开发与实现过程的基础，直接决定了应用程序研发效率与质量。在软件工程整个生命周期中，需求分析是指在新建或更改现有软件系统的时候，制定系统的目标、范围、定义及功能时需要完成的工作。在对系统开发进行全方位地分析后，才能得到符合预期的软件功能^[17]。本系统主要需求为开发一款能够嵌入在浏览器中的聊天机器人，用户通过与聊天机器人进行不同层次对话，可以获取符合用户需求的旅行相关信息或服务。对系统的分析则需要从可行性、功能性需求和非功能性需求三个方面进行。

基于问答对话类的聊天机器人对话设计，则要在对话中全面考虑用户可能的意向分支，并且聊天逻辑尽可能符合人为对话交流习惯^[18]。依据这一原则。本系统设计为向用户提供聊天窗口，利用与用户聊天的方式向用户逐步提出基于意向判断的初层次问题和基于信息提取的深层次问题，同时对用户的回复内容进行解析，从解析结果中获取到用户旅行需求，并生成相应的旅行信息或服务，通过聊天回复的方式向用户提供相关信息或服务的推荐系统。对本系统分析后应有如下要求：

1. 在浏览器中，提供聊天机器人启动按钮，用户点击后提供聊天窗口，通过聊天窗口与用户聊天。
2. 向用户提供系统介绍和功能导航，向用户概述本系统功能。同时向用户提供导航按钮，使用户通过按钮到达其想要的推荐服务。然后分析用户回复表达服务意向。并根据用户意向，代入对应推荐模块进行深层次提问。
3. 从用户回答中，提取关键信息，基于未提取到的信息继续对用户进行进一步的信息提取提问，并根据用户的信息生成符合其需求的信息或服务。
4. 根据用户信息与后台数据，生成满足用户需求的相关旅行信息或服务，通过聊天回复的方式反馈给聊天窗口，聊天窗口将结果展示给用户。

3.2 可行性分析

3.2.1 技术可行性

本系统需要开发一款能够让用户在浏览器中通过对话方式获取其所需服务的聊天机器人。聊天框架构造方面，我们通过 Bot Framework 框架项目中设计实现聊天逻辑，使模型能够按照设计的逻辑执行。同时设计并实现 LUIS 语言理解模型，从而实现判断用户语言意向，从语言中提取相应用户需求数据，使模型可以通过语言理解结果进行下一步策略的决策。通过爬虫获取旅行网站相关数据，使用该数据生成满足用户需求的信息或服务，通过设计 Chrome Extension 扩展应用，将实现的聊天机器人嵌入在浏览器扩展应用中，方便用户在浏览器中打开扩展应用后便可与其聊天。模型开发过程中各项技术成熟，所以本系统在技术方面可行性成立。

3.2.2 经济可行性

本系统开发主要成本主要来自于搭载在 Azure 上的数据库与应用程序的搭载成本与人员维护成本，开发成本不高。由于在模型前期数据量及用户流量还较小的时候，可以选择 Azure 平台提供的小带宽服务器，所以后台聊天逻辑模块、LUIS 语言理解模块和数据库模块前期搭载成本较低。后期可根据用户流量变化更新至高性能、高带宽服务器。同时用户在流量提升后可以通过广告推荐等方式带来可观的经济收益，所以本系统具有经济方面的可行性。

3.2.3 运行可行性

在浏览器中，增加聊天对话框，聊天机器人通过与用户对话的方式得到用户旅行相关需求，根据获取到的用户需求，向用户推荐旅行服务，旅游博客或旅行推荐，并为用户提供出行相关的预订服务链接，使得用户能够以最简单的、最直面的窗口聊天方式在互联网上获取到需要的旅行信息或服务。用户上手容易，交互简单，大幅度简化了用户获取信息服务的步骤。故本系统在运行方面可行性成立。

3.3 功能性需求分析

本系统需要在浏览器中向用户提供对话框，在对话框中使用户通过对话与聊天机器人进行交互。功能上在与用户对话交互的过程中，能够分析并获取用户旅行方面需求，提供符合用户旅行需求的信息服务，包括航班预定需求、铁路服务预定需求、酒店服务预定需求和旅行推荐需求，最终将符合用户需求的信息反馈给用户。

3.3.1 浏览器界面层需求

浏览器界面层主要需求为向用户提供聊天机器人与用户对话交互模块，使用户能够便捷的打开聊天窗口，让用户直观地与我们后台的聊天机器人进行交互。即主要作用为向用户提供聊天窗口，完成用户与线上聊天机器人，即后台聊天逻辑的交互。

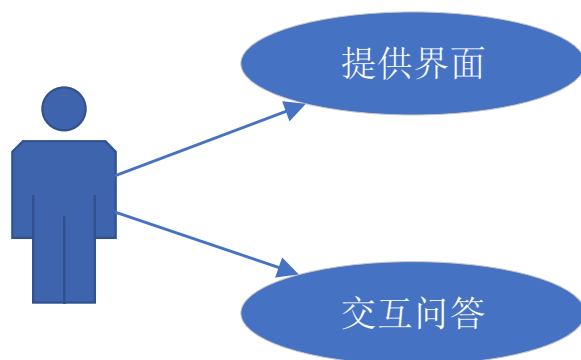


图 3.1 浏览器界面层 UML 用例图

如图 3.1，浏览器界面层主要需要满足的功能需求如下：

- 在浏览器中提供扩展应用，使用户在浏览器扩展应用菜单栏中可以方便快捷地打开扩展应用。
- 在浏览器扩展应用打开窗口中提供聊天窗口，提供用户与后台聊天逻辑模块的交互平台，发送用户指令至聊天逻辑层，并将聊天逻辑层的反馈展现给用户。
- 通过聊天窗口接收用户向聊天机器人发送对话，并将其反馈给后台聊天逻辑层。接收从聊天逻辑层中发送出来的信息反馈，在聊天窗口中展示

给用户。

3.3.2 聊天逻辑层需求

聊天逻辑层主要需求为处理用户对话的逻辑模块，当用户进入模型，向用户提供关于聊天机器人功能介绍，并且进行功能导航，方便用户能够快速了解并使用聊天机器人软件，分析用户语言，判断并处理用户在机票预定、火车票预定、酒店预订、旅行信息推荐方面的旅行预定需求，并根据用户需求，推荐满足用户需求的服务与信息，即主要作用是处理与用户对话逻辑，理解用户语言，并基于与用户聊天得到的意向与信息，根据设定逻辑对下一步对话流进行决策，进入下一步的问答过程。

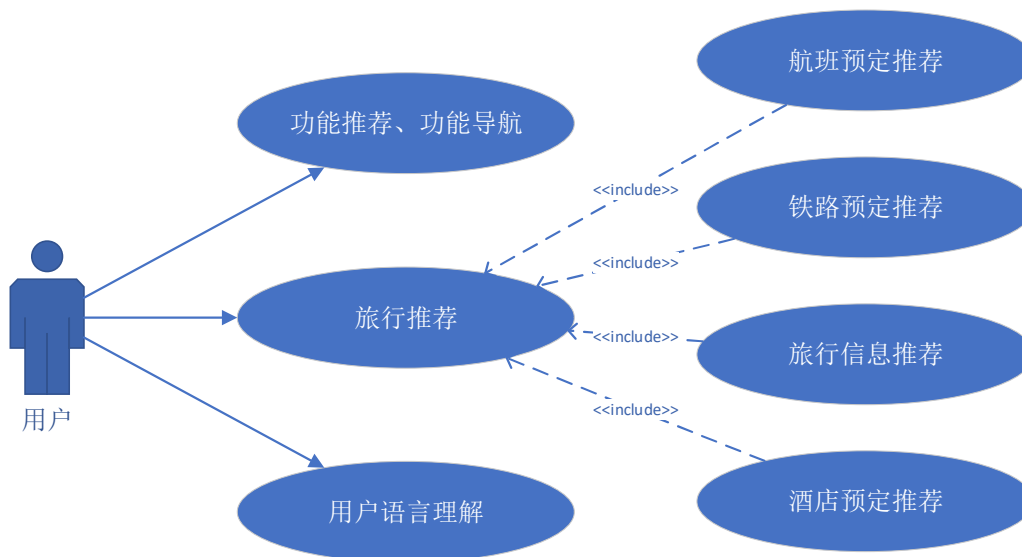


图 3.2 聊天逻辑层 UML 用例图

如图 3.2，在聊天逻辑层主要需要满足以下功能需求：

- 在用户进入后，向用户介绍模型功能，并向用户提供一定导航，引导用户使用。
- 根据用户语言内容，理解用户所需服务意向，基于用户机票预订、酒店预订、旅行推荐或铁路预订方面的旅行意向，进一步向用户询问相关问题，并基于数据库层中的数据以及用户信息，生成旅行信息及服务，并通过聊天向用户提供其想要的信息或服务。

- 理解用户语言，对于用户发送的语言文字内容，分析其意向从中提取出相关数据。

3.4 非功能性需求分析

3.4.1 软件环境需求

由于浏览器 Chrome 扩展应用运行环境的特点。本系统主要设计在 Chromium 为内核的浏览器当中使用，其中包括 Chrome、Edge、Opera、UC 浏览器、360 浏览器等当下主流浏览器。具有广泛的使用基础。

3.4.2 软件质量需求

软件质量的需求需要满足以下几个方面的要求：

1. 系统可靠性：系统对用户语言的意向分析与信息提取准确，最终向用户提供的信息服务无误。尽可能减少语义理解错误、提供的信息及服务不符合用户要求、提供的信息及服务无效的情况。
2. 系统易用性：系统界面合理、操作清晰、聊天方便、容易上手，提供功能导航方便用户快速了解并使用系统。在浏览器中，进入界面操作足够便捷，界面交互一目了然。软件操作简单、使用容易，大大简化用户上手操作的流程。
3. 系统高效性：系统逻辑执行高效，尽可能减少在模型运行中由于逻辑过于复杂造成的时延。在网络正常情况下，消息回复不卡顿，每条回复用时控制在 3 秒范围以内。
4. 系统健壮性：对于用户一些错误表达，能够识别用户回复中的错误，并针对不同类型错误包含相应错误情况处理逻辑，例如向用户提示表明错误并重新提问相同问题或者忽略问题，使模型不会因用户表达错误而停止运行或崩溃。
5. 系统可扩展性：系统各模块之间高内聚、低耦合，各个模块内的更新都较为容易。不同需求功能对话之间相对独立，方便扩展新的功能对话。Bot

Framework 更新版本在完全发布至线上之前可以发布内测版本，方便产品扩展迭代。

3.5 本章小结

通过需求分析，分析了软件所要满足的设计要求，软件实现上的可行性以及技术上需要解决的问题。在接下来的内容中，我们则需要依据软件需求来设计聊天对话模型。

第四章 系统概要设计

4.1 总体结构设计

根据上一章的软件需求，我们对软件进行整体设计。如图 4.1 系统架构图，按照不同功能层级，系统体系可以划分为浏览器界面层、聊天逻辑层、数据库层三层 B/S 体系结构模型。

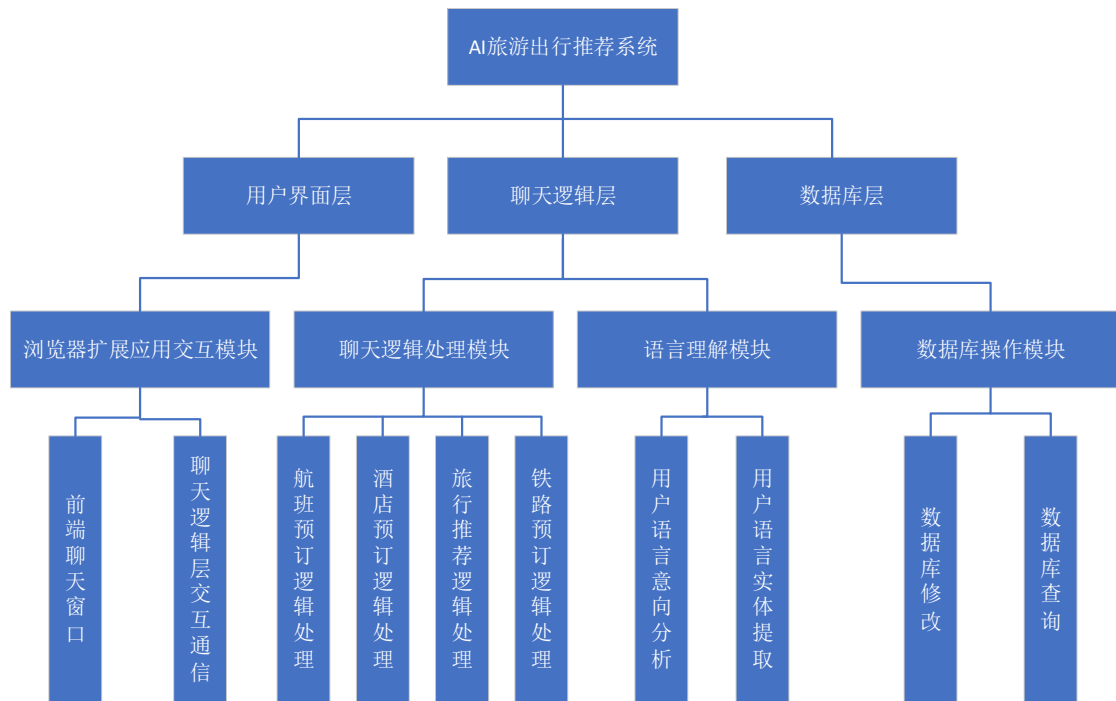


图 4.1 系统架构图

聊天逻辑层是系统核心部分，主要负责逻辑处理。首先接收并处理从接口获取到的用户聊天数据 HTTP 请求，经语言理解模块分析处理，判断语言意向并提取实体信息后，进入聊天对话逻辑处理模块进行相应决策，根据航班预订、酒店预订、旅行推荐、铁路预订等不同意向类型带入对应逻辑进行深层次提问。当获取完整用户数据后，通过逻辑层与数据库层之间通信，向数据库层发送请求查找对应需求数据，并将数据库层查找到的相关数据返回给逻辑对话模块以便生成相关信息服务。将处理后得到的反馈通过 JSON 格式数据返回给浏览器界面层。

数据库层储存了用于信息服务生成的相关数据，为聊天逻辑层提供数据库相关操作方法，方便了聊天逻辑层对数据库数据进行使用。

用户界面层设计实现扩展应用交互模块，并嵌入在浏览器扩展应用当中。在扩展应用中为用户提供聊天窗口，实现用户与聊天逻辑层交互。具体方法为将用户输入到聊天框中的语言通过 HTTP 请求发送给聊天逻辑层，将聊天逻辑层返回的结果解析后通过聊天窗回复展示给用户。

4.2 系统部署

如图 4.2，按照 B/S 体系结构模型，将聊天逻辑层上的语言理解模块和后台逻辑处理模块分别部署在 Azure 云服务器上，并生成云端应用供客户端调取，使用 Azure SQL Server 构建数据库层存储系统数据。同时设计实现浏览器扩展应用提供聊天窗口并与云端应用进行通信，部署在用户端 Chromium 内核浏览器内，用户通过将扩展应用下载安装至浏览器后，在浏览器中打开扩展应用，或在 Skype 聊天软件中加入聊天机器人，即可使用本系统。

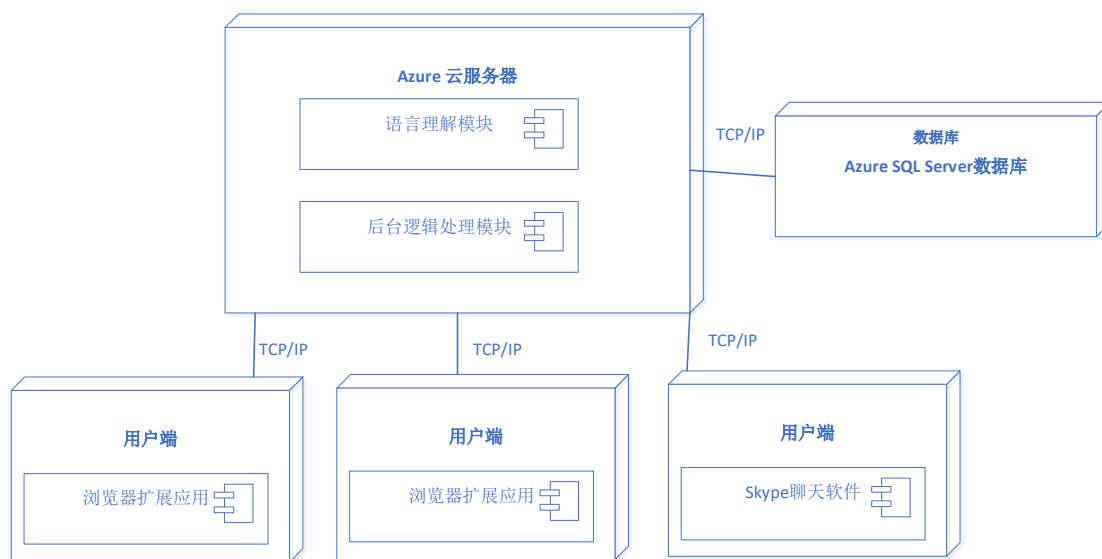


图 4.2 软件 UML 部署图

4.3 功能模块概要设计

系统功能上，按照需要满足的用户需求类型，可以分为导航模块、航班预订推荐模块、酒店预订推荐模块、旅行推荐模块、铁路服务预订模块。各个功能模块逻辑关系框架如图 4.3 所示。

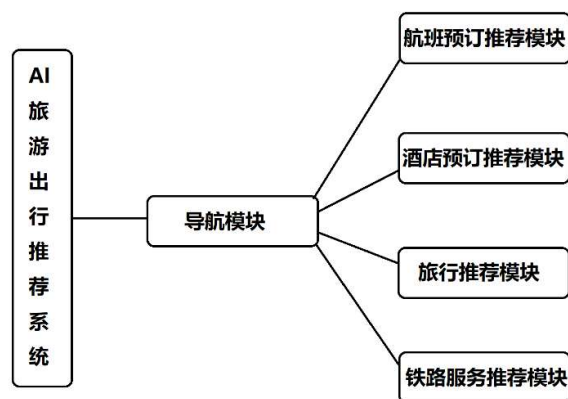


图 4.3 模型功能框架图

4.3.1 导航模块

导航模块在用户进入后便进入本模块。主要功能是向用户简要介绍本系统，并为用户提供一些功能导航按钮引导用户进入对应推荐模块。当用户点击按钮回复或输入文字回复后，将用户回复内容带入语言理解模块判断其意向以得到其所需意向服务类型，根据不同意向类型将对话带入不同的推荐模块进行进一步提问和推荐。

4.3.2 航班预订推荐模块

航班预订推荐模块在导航模块中判断出用户回复意向为航班预定后进入本模块。主要功能是进一步询问用户航班预定相关信息，并根据这些信息向用户推荐符合用户需求的航班预订服务。当进入对话，依次询问用户预期航班的起降地、出发时间等信息，解析用户回复并保存。汇总信息后，请求数据库层查询符合用户需要的航班预定链接，以回复方式提供给用户。

4.3.3 酒店预订推荐模块

在主对话中，当用户答复内容意向被判断为酒店预订，则进入酒店预订推荐模块。本模块主要功能是为用户推荐满足其要求的酒店预定服务。进入对话后，首先逐步向用户询问所需预定酒店所在城市，酒店位置所需满足大体区域、入住时间等信息，解析用户回复内容。之后根据用户需求请求数据库层，查询满足要

求的酒店预定链接，基于网站链接调取数据库或直接生成预定链接，将得到的链接以聊天方式提供给用户。

4.3.4 旅行推荐模块

当主对话判断用户意向为旅行推荐，则会跳入旅行推荐模块。旅行推荐模块主要功能是推荐用户需要的旅行信息情报、相关景点。进入对话后，向用户询问所需旅行咨询地区，得到用户需求后，基于咨询城市地区生成有关该地区的搜索链接并异步请求该链接。获取页面反馈后解析反馈获取到有关该地景点、博客等数据，通过聊天返回给用户。

4.3.5 铁路预订推荐模块

铁路预订推荐模块在主对话判断出用户意向为铁路预定后进入本模块。主要功能是向用户推荐符合用户需求的铁路预订服务。当进入对话，依次询问用户预期火车的出发地、目的地、出发时间等信息，将用户回复依次记录。汇总信息后，请求数据库层查询符合用户需要的铁路预定链接，以聊天回复方式将链接提供给用户。

4.4 数据库概要设计

数据库是用于存储系统后台数据的信息管理系统。数据库设计是在现有数据需求下设计数据表、通过数据表储存数据、建立数据库的过程。在构建数据库前需要根据需求分析，对需要使用的数据和未来运营过程中产生的数据进行全面的了解^[19]。根据系统数据存储与查询相关方面需求，设计实现符合要求的数据库，完善系统各方面的数据功能要求。

4.4.1 数据表设计

在数据库使用与管理中，核心原则是维护数据表之间的关系。根据数据需求，建立城市信息数据表、地标信息数据表、机场信息数据表、火车站信息数据表。各个数据表之间的 E-R 关系图如图 4.4，由于一个城市会包含多个地标，多个机

场及多个火车站，所以其他数据表在 E-R 关系内隶属于城市信息数据表。

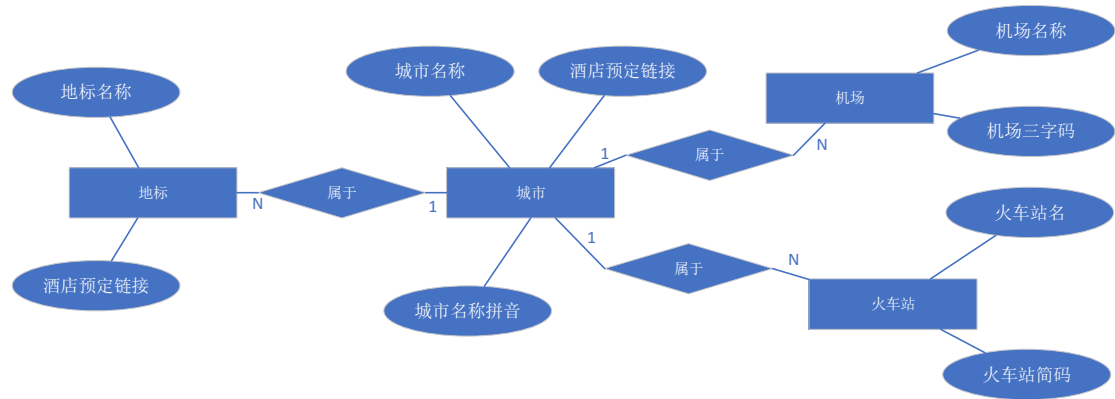


图 4.4 数据表 E-R 关系图

城市信息数据表，存储了各个城市相关数据信息，用于后期服务生成所需要的相关数据，包括城市名、部分网站的酒店预订链接、机场三字代码、城市名称拼音。如表 4.1 所示。

表 4.1 城市信息数据表字段简介

字段名	类型	长度	字段注释	是否可为空
City Name	Char	9	城市名称	不为空
Hotel Link	Char	40	城市范围内酒店预订链接	可为空
City Pinyin	Char	12	城市名称拼音	不为空

地标信息数据表存储了各个城市不同区域地标信息，用于后期服务生成所需要用的相关数据，包括地标名称、部分网站对应地标酒店预订链接、所属城市名称。如表 4.2 所示。

表 4.2 地标信息数据表字段简介

字段名	类型	长度	字段注释	是否可为空
Location Name	Char	15	地标名称	不为空
Hotel Link	Char	40	地标附近酒店预订链接	可为空
City Name	Char	9	地标所属城市名称	不为空

机场信息数据表存储了城市中不同机场信息,用于后期航班预定服务链接生成时,提供所需要的机场信息数据,包括机场名称、机场三字码、所属城市名称。如表 4.3 所示。

表 4.3 机场信息数据表字段简介

字段名	类型	长度	字段注释	是否可为空
Airport Name	Char	15	机场名称	不为空
Airport Code	Char	3	机场三字码	不为空
City Name	Char	9	机场所属城市名称	不为空

火车站信息数据表存储了城市中城市不同火车站信息,用于后期火车票预定服务链接生成时,所需要的火车站相关数据,包括火车站名称、火车站简码、所属城市名称。如表 4.4 所示。

表 4.4 火车站信息数据表字段简介

字段名	类型	长度	字段注释	是否可为空
Station Name	Char	15	火车站名称	不为空
Station Code	Char	5	火车站简码	不为空
City Name	Char	9	所属城市名称	不为空

4.4.2 数据库的创建与连接

Azure SQL 具有较完善的数据库设计管理功能,支持在多种开发工具中使用。在 Azure SQL 数据库中,按照设计的数据表创建数据表并按照数据表之间的关系建立数据库。

SqlConnectionStringBuilder 类是 C#中用于数据库操作管理的类。在业务逻辑层中,使用 SqlConnectionStringBuilder 设置对 Azure SQL 数据库访问。通过 SqlConnection 连接数据库,同时使用 SqlCommand 类编辑 SQL 数据库中的 SELECT 查询命令,实现在后台聊天逻辑模块在生成信息服务时,对数据库中数据进行查找操作。

4.5 接口概要设计

根据上文不同系统功能模块的概要设计，由于不同模块相互分离，模块之间内部逻辑相互并不影响，需要设计一定接口完成模块之间连接通信。本系统包含的交互接口主要是连接后台聊天逻辑模块与前端扩展应用模块的聊天逻辑 Bot Framework 接口，和连接语言理解模块与后台聊天逻辑模块的 LUIS 语言理解接口。

4.5.1 聊天逻辑 Bot Framework 接口

聊天逻辑 Bot Framework 接口是前端界面层捕获到用户行为后，将用户行为发送至后台聊天逻辑模块的通信接口，同时从接口请求反馈中可获取到聊天逻辑模块返回给用户的内容，方便前端界面层将处理后的信息展现给用户。本接口由 Bot Framework 框架提供。

请求详情如表 4.5，将用户对话内容，将用户唯一 ID，用户发送文本以及用户发送内容类型等信息，按照一定格式要求生成 json 数据，使用 POST 请求方式发送给聊天机器人。

表 4.5 聊天逻辑接口请求信息表

HTTP 请求方式	POST
URL	http://{域名}/api/messages
请求数据	<pre>{ "from": { "id": "KU6EIHje85", "role": "user" }, "text": "帮我预定去成都机票", "localTimestamp": "2020-09-13T21:20:59+08:00", "locale": "zh-CN", "type": "message" }</pre>

聊天逻辑接口请求数据中不同参数说明如表 4.6:

表 4.6 聊天逻辑接口请求数据参数说明表

参数名	描述定义
from	信息请求用户来源，包含用户表示唯一 id 及用户角色
Text	请求内容，即用户向 Bot 框架发送的信息
Type	请求类型

返回数据举例如表 4.7，其中包含了聊天机器人回复，用户 ID 等信息：

表 4.7 聊天逻辑接口请求返回数据样例

<pre>{ "from": { "id": "KUb6EIHje85", "role": "bot" }, "text": "请输入到达城市", "localTimestamp": "2020-09-13T21:21:02+08:00", "locale": "zh-CN", "type": "message" }</pre>

4.5.2 LUIS 语言理解接口

LUIS 语言理解接口是后台聊天逻辑模块在处理用户语言时，使用 LUIS 语言理解模块分析用户意向并获取用户实体内容的接口。当 LUIS 语言理解模块开发完成并发布至 Azure 云服务器以后，会自动生成 LUIS 接口供后台聊天逻辑模块决策使用，接口主要信息如下：

如表 4.8，用户以 GET 形式向 LUIS 模型发送语言解析请求，将数据以 URL 参数形式传递给 LUIS 语言理解模块。

表 4.8 LUIS 语言理解接口请求表

HTTP 请求方式	GET
URL	<p>http://{域名}/luis/v2.0/apps/{luisid}?</p> <p>timezoneOffset=480&q=预定下周三成都的宾馆</p>

在请求 URL 中，各个请求参数说明如表 4.9：

表 4.9 LUIS 语言理解接口参数说明表

参数名	描述定义
luisid	由 LUIS 生成的 LUIS 模型唯一 Id
timezoneOffset	与太平洋标准时间对比时差
q	需要 LUIS 分析的用户语言

返回数据举例如表 4.10，在数据中，包含每一种意向概率以及从语言中提取出的不同意向类型，解析返回数据即可得到用户语言中的意向与实体内容：

表 4.10 LUIS 语言理解返回数据样例

<pre>{ "intents": { "FlightBooking": "0.00710499566", "HotelBooking": "0.930015445", "TrainBooking": "0.00621636352", "TravelAdvance": "0.04392783", "None": "0.00487960968", }, "entities": [{ "text": "下周三", "timex": ["2020-09-23"], "type": "datetime", }, { "text": "成都", "type": "location", }], "text": "预定下周三成都的宾馆", "localTimestamp": "2020-09-13T21:27:31+08:00", "locale": "zh-CN", "type": "message" }</pre>
--

4.6 本章小结

在前一章需求分析的基础上，本章从体系结构设计、功能性结构设计、数据库设计、接口设计四个方面设计了聊天旅游出行推荐系统的架构。在体系结构方面，将模型分为浏览器界面层、聊天逻辑层和数据库层。在功能性方面，分为导航模块、航班预订推荐模块、酒店预订推荐模块、旅行推荐模块和铁路预订推荐模块。至此，得到了旅行推荐聊天机器人软件的系统的概要设计。

第五章 系统详细设计

上一章主要介绍了旅游出行推荐 AI 聊天机器人系统的整体结构、功能结构、接口结构、数据库结构的相关设计，本章将介绍系统架构中所有功能模块的详细设计及实现，主要是以下四个层次模块的设计与实现。

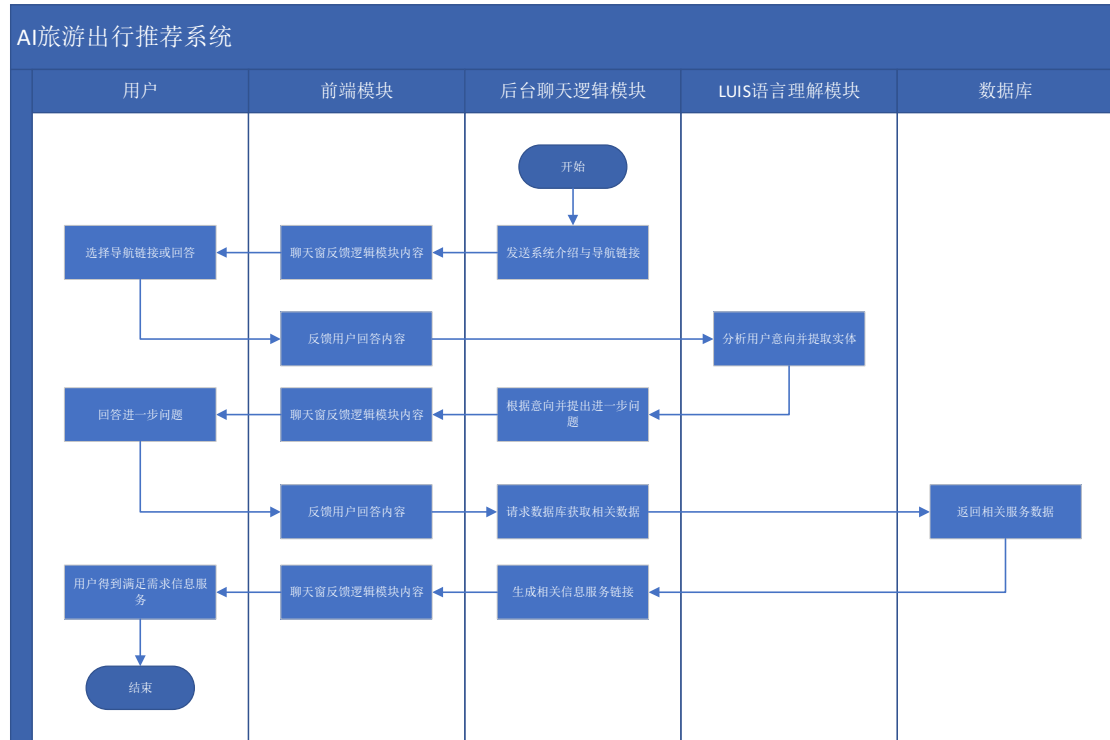


图 5.1 系统逻辑图

如图 5.1，系统包含前端模块、后台聊天逻辑模块、LUIS 语言理解模块以及数据库模块四个模块，各个模块功能作用如下：

– 数据库模块：根据已有的数据表，分析数据需求并根据需求通过爬虫技术获取相关数据，将数据按照数据表格式上传至数据库，方便后台聊天逻辑模块在旅行信息服务生成过程中对数据的调取与更新。

– LUIS 语言理解模块：设计与实现语言理解模型，从用户语言中分析语言意图并提取用户语言相关内容实体，后台逻辑模块根据语言理解模块对用户语言分析结果进行相关决策，将用户语言中相关信息转化为数据。

– 后台聊天逻辑模块：用户对话交流的逻辑层，决策与用户对话逻辑，根据用户回答由浅至深逐步提出问题，根据回答判断用户意向、获取用户需求，生成

用户所需信息与服务。

– 前端模块：在浏览器中向用户提供聊天交互界面，连接用户与后台聊天逻辑模块，实现用户与系统交互过程。

5.1 数据库模块的设计与实现

5.1.1 数据需求

本系统为生成用户所需的旅游相关的信息或服务，需要一定的数据储备。为得到相关数据，可通过爬虫的方法，使用广度优先算法对目标网站的所有网页批量化抓取网页信息，在获取未遍历链接的同时继续发送请求，获取新的未遍历网页相关网页信息，继续查找从网页信息中获得的新的可爬网页链接，重复该过程，获取到该网站内所有相关数据信息时结束程序。

在生成用户火车票预订、酒店预订等相关服务时，需要一个含有城市地区相关信息的数据集合，数据集合还需包含对应城市地区中，一定的地标区域数据，例如地标区域“虹桥”属于城市地区“上海”。还需要获取到该地区旅行攻略、酒店预订信息的对应查询访问的链接地址等信息，就要获取一些网站数据用来生成相关链接。为得到这些数据，选择当前某主流旅行门户网站作为目标网站，批量化获取目标网站相应数据信息，按照设计的数据表存储在数据库内。在生成用户机票预订信息时还需要国内所有机场及其三字码的信息数据，生成火车票预订信息需国内所有火车站及其代码信息，在爬虫解析页面时从页面一并获取，并按照设计的数据表存储在数据库当中。

5.1.2 离线数据爬取

分析目标门户网站的酒店分类，每一个城市的酒店预定网页，含以下几类信息方便后期使用（以上海市页面为例）：

- 城市不同位置区域（迪士尼度假区、浦东机场核心区、人民广场地区等）
- 周边城市（苏州、太仓、昆山、嘉善等）
- 酒店等级（经济、舒适、高档、豪华）

为获取全国完整城市列表及每个城市相关信息数据，可以使用广度优先遍历算法，依次获得目标网站的酒店分页中各个城市地区及其地标区域的相关数据。

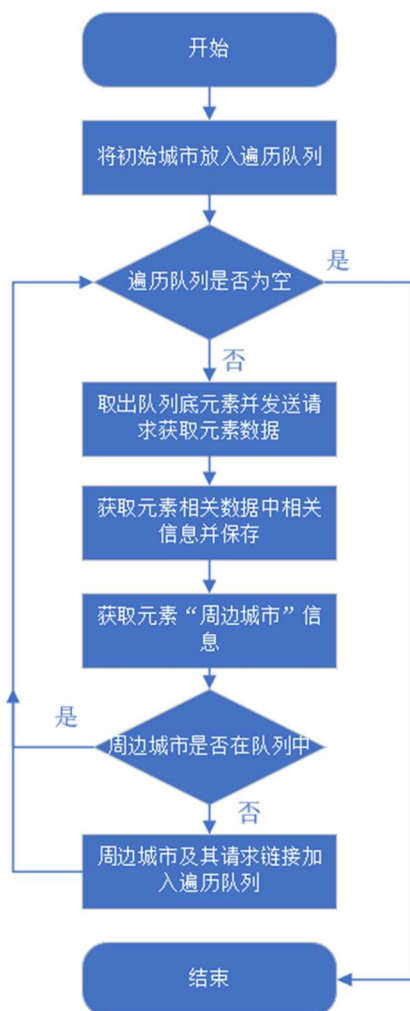


图 5.2 广度优先遍历获取城市数据爬虫流程图

如图 5.2，利用广度优先遍历获取城市数据的具体流程如下：

(1) 建立广度优先搜索的遍历队列，队列中每一个元素储存城市地区名称及其对应目标网站对应城市酒店列表链接。

(2) 选择上海作为起点，并将网站上海分站链接设置为上海访问链接，将上海压入遍历队列。

(3) 当遍历队列不为空时则取出队列头部元素城市地区，使用 Python 中 Urllib 库中函数向该城市对应 URL 链接发送请求，在请求反馈中即可获取该城

市网页 HTML 代码。

(4) 使用 BeautifulSoup 库可以将得到的旅行信息目标网页根据网页结构重新架构成 Struct Dom 树结构, 在 Struct Dom 树中通过目标信息在网页中的类名、标签名等标志查找“城市区域位置”数据在 Struct Dom 树中的位置节点, 从该位置节点中提取需要的数据进行处理, 将部分合并数据拆分成多组数据。例如将从目标页面中获取到的地标信息“静安寺/南京西路”格式的数据, 按照斜杠拆分成“静安寺”和“南京西路”两组数据。最后在数据库中保存相关数据和该地标所属城市, 方便聊天系统能够更准确识别用户表达的地标信息。

(5) 继续使用 BeautifulSoup 库中方法根据类名或者标签名来获取周边城市数据块在 Struct Dom 树中所在位置。从该位置提取所有周边城市及其链接信息, 依次判断周边城市是否在遍历队列中或已经被遍历访问过, 若未访问, 则将周边城市及其链接创建新的队列元素, 将新元素放入广度优先搜索遍历队列尾部等待遍历。

(6) 当广度优先搜索遍历队列为空时, 证明所有城市相关数据已全部获取, 算法结束。

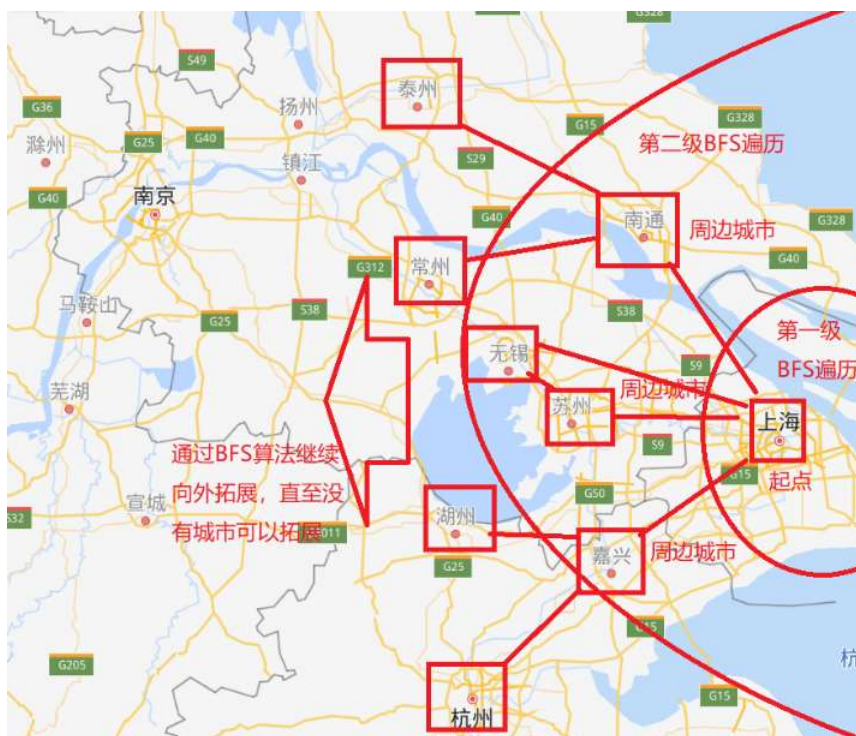


图 5.3 BFS 算法利用周边城市爬取城市效果示意图

如图 5.3，算法通过网页中的“周边城市”不断向下一级未标记的城市扩展。当使用广度优先算法实现的爬虫系统请求运行结束后，可得到一个基于目标旅行网站的全国所有城市地区及其相关地标区域信息的数据集合，并在数据库中根据数据表的数据格式存储这些数据。在其后的聊天逻辑模块实现中，通过解析对话获取数据索引，并利用索引请求数据库，以获取满足用户需求的旅游服务或信息。综上，利用广度优先搜索获取城市数据具体实现核心代码如下：

```
def BFS_Spider():  
    city_queue = queue.Queue()  
    # 选取上海作为起点，并且加入遍历队列尾部  
    city_queue.put(city('上海', 'https://{域名}/hotel/shanghai2'))  
    while city_queue.empty(): #当队列不为空，一直遍历队列  
        city_item = city_queue.get() #获取位于队列头部城市  
        try: # 成功请求页面  
            city_request = urllib.request.Request(url=city_item.url)  
            city_response = urllib.request.urlopen(city_request)  
        except: #页面请求失败处理  
            print('请求链接失败，失败链接: ' + city_item.url)  
            continue  
        try: #从页面反馈中获取页面爬取 HTML 代码  
            html = city_response.read()  
        except Exception as e:  
            html = e.partial  
        #解析网页，页面中未遍历城市地区加入队列，此函数实现过程此处省略  
        processPageElement(html, city_queue)
```

通过以上方法，得到 2091 个城市地区及不同城市地区中 4739 个地标位置以及相关机场信息，预定链接等辅助信息。将得到的数据按照设计好的数据

表，使用 `SqlConnectionStringBuilder` 类将数据批量上传至 Azure SQL 数据库当中。

5.2 LUIS 语言理解模块的设计与实现

5.2.1 LUIS 模块概述

在聊天机器人开发过程中，保证用户语言分析的准确无误是聊天机器人逻辑正常运行的充分条件，否则难以保证机器人逻辑可按照开发者设计所运行^[20]。为了开发出精确的语言分析理解模型，使用 LUIS 开发语言理解模块。LUIS 语言理解模块使用深度神经网络、自然语言处理等相关技术，可对语言进行进行多个维度的解析，分析语言并转换成数据。开发者可在 LUIS 平台上根据设计需求，自行创建语义理解模块。

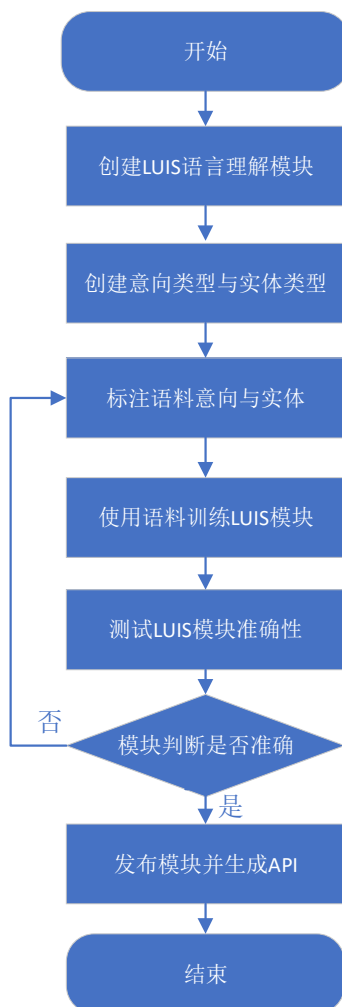


图 5.4 搭建 LUIS 模块流程图

如图 5.4, 在搭建 LUIS 模块的过程中, 首先在模块中自主建立所需的意向和实体, 批量化标注一定语料, 标注语料中的意向并标记语料中的实体内容作为训练数据, 最终通过模型训练得到能够判断用户语言表达意向和从用户语言中提取实体内容数据的自然语言模型。发布后, 在后台聊天逻辑模块中通过调用 LUIS 模块生成的 API 接口, 将用户语言等信息通过 URL 参数写入 API, 请求 API 得到响应后, 即可得到一个 XML 格式的反馈, 通过解析 XML 格式反馈得到基于 LUIS 模块的语言理解结果, 包含用户回复更偏向的服务意向与用户语言中所能提取到的实体数据信息。

5.2.2 LUIS 模块的意向与实体定义

为满足逻辑模块的决策需要, 根据后台聊天逻辑模块的四种不同二级对话类型以及需要从语言中提取的不同信息类型, 建立五个意向类型来判断用户聊天内容中所表达的意图, 并新建三种实体类型来提取用户话语中的重要信息数据。具体介绍如下:

- 航班预订意向 (Flight Booking): 当用户语言中表达出想预订飞机航班的意向, 则认定为此意向。例如“帮我看看下周去哈尔滨的机票”。
- 火车预订意向 (Train Booking): 当用户语言中表达出想预订火车票或高铁票的意向, 则归类为此意向。例如“查询一下后天去重庆的高铁”。
- 酒店预订意向 (Hotel Booking): 当用户语言中表达出想预订酒店或宾馆的意向, 则判断为此意向。例如“帮我订下周苏州金鸡湖附近的酒店”。
- 旅行推荐意向 (Travel Advice): 当用户的语言中表达出希望得到一定的旅行目的地介绍、旅行推荐、旅游攻略推荐等意向, 则认定为此意向。例如“去兰州该玩哪些地方”。
- 出发地实体 (Start Location Entity) 和目的地实体 (End Location Entity): 用户语言中所携带的所有有关地点地名的信息, 当地点在语句中表达的是出发概念, 则为出发地实体内容, 否则为目的地实体内容。在

酒店预订意向和旅行推荐意向中的地点则均为目的地实体。例如，“预订从成都到上海的航班”中，成都标注为出发地实体，上海标注为目的地实体。

- 时间实体 (Datetime Entity): LUIS 语义理解模型中，不需要人工语料训练，平台自带的一种实体类型，它可以从用户的语言中提取时间短语，并根据当前时区以及时间，计算并生成用户所说时间具体指代的对应公历日期，并随反馈数据一并返回。使用 LUIS 模型对用户所说的“查一下周三去西安的飞机票”进行解析。模型则可以从语言中提取到“下周三”这个内容符合一个时间实体，提取出实体后，可以根据当前时间，返回语言中下周三具体指代的公历日期并返回。

```

    "$instance": {
      "datetime": [
        {
          "endIndex": 3,
          "startIndex": 0,
          "text": "下周三",
          "type": "builtin.datetimeV2.date"
        }
      ]
    },
    "datetime": [
      {
        "timex": [
          "2020-10-14"
        ],
        "type": "date"
      }
    ]
  },

```

图 5.5 LUIS 模块返回 json 格式时间实体数据

如图 5.5，语言理解模块不仅可以从语言中提取出时间实体关键词“下周三”，还可以基于用户所处的时间与所在时区信息计算出其时间实体实际所代表的公历日期为“2019-10-09”，方便后台聊天逻辑模块在生成相关旅行信息时，填入符合用户需求的时间信息。

在使用默认设置语言理解模块情况下，一些特定时间分析结果与真实时间误差一天。经过测试，发现主要是 LUIS 语义理解模块使用的是太平洋标准时间，与我们的北京时间存在误差造成的。为解决这一问题，可以发送请求时在 API 在

请求设置了 8 个小时时差，即在语义理解请求 URL 中将参数 `timezoneoffset` 中设置 480，使得服务请求符合北京时间。

5.2.3 LUIS 模块语料标注与训练

在自定义 LUIS 模块中建立以上意向与实体以后，选择一部分能够涵盖所有意向判断与实体获取的语料，作为训练数据放入模型中，在语料中标注训练语料所属意向，并标记训练语料中不同类型的实体内容，标记过程如图 5.6。经过 LUIS 服务器使用训练数据对 LUIS 语言理解模块进行训练后，最终得到能够判断用户语言意向并提取语言实体内容，在后台聊天逻辑模块中用于逻辑决策的 LUIS 语言理解模块。

<input type="checkbox"/> Example utterance	Score ?
Enter an example of what a user might say and hit Enter.	
<input type="checkbox"/> 订一张 datetimeV2 去 location 的机票	: 0.98
订 datetimeV2 去 location 的机票	0.98
订一张去 location 的飞机票	0.99
订 datetimeV2 去 location 机票	0.97
订一张去 location 的机票	0.98

图 5.6 LUIS 模型训练数据标注页面

如图 5.6，对于训练样本“订一张明天去深圳的机票”，首先需要标注其意向为航班预定意向。由于时间实体是系统自带的实体，可以从语言中自动识别，“明天”就会被自动识别为时间实体。此时需要手动标注“深圳”为自定义的目的地实体。

5.2.4 LUIS 模块测试

经过以上步骤搭建语言理解模块，对训练集数据进行训练后，可以得到满足

后期需求的能够进行意向判断与实体提取的 LUIS 模块,接下来需要对 LUIS 语言理解模块进行测试。测试过程则继续在 LUIS 平台端进行。测试环境如图 5.7,选取针对不同意向判断和实体提取的测试集,输入语言理解模块进行测试,查看生成的语言理解模块对每个意向判断的状况和实体获取内容是否准确。

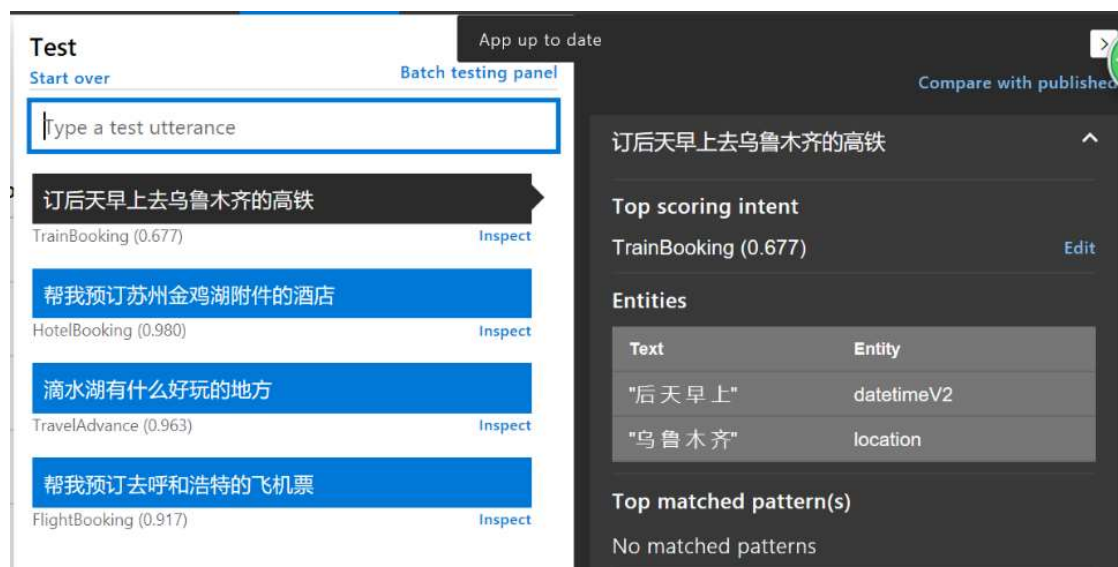


图 5.7 LUIS 模块测试页面

测试结果如表 5.1。根据如上测试结果,当前得到的 LUIS 语言理解模块满足后台聊天逻辑模块在意向判断与实体获取方面的相关设计需要,可以在发布之后供后台聊天逻辑模块在模型进行逻辑决策过程中使用。

表 5.1 LUIS 模块测试结果表

测试样例	意向判断	意向概率	实体提取
订后天去乌鲁木齐的高铁	火车预订意向	67.7%	目的地实体: 乌鲁木齐 时间实体: 后天
帮我预订苏州同里古镇附件的酒店	酒店预订意向	98.0%	目的地实体: 苏州同里古镇
滴水湖有什么好玩的地方	旅行推荐意向	96.3%	目的地实体: 滴水湖
帮我预订去呼和浩特的飞机票	航班预订意向	91.7%	目的地实体: 呼和浩特

5.2.5 LUIS 模块发布

将在 LUIS 平台中搭建的语言理解模块发布至 Azure 云服务器,即可生成远程调用模块所需要的密钥,通过该密钥,可实现在 Bot Framework 工程中调用 LUIS 语言理解模块。即在后台聊天逻辑模块的代码中通过调用密钥使用 LUIS 语言理解模块,使模块可以根据用户的语言内容提取需求信息并做出一定决策。发布后线上产品如图 5.8,请求 URL 即可得到 LUIS 分析结果。

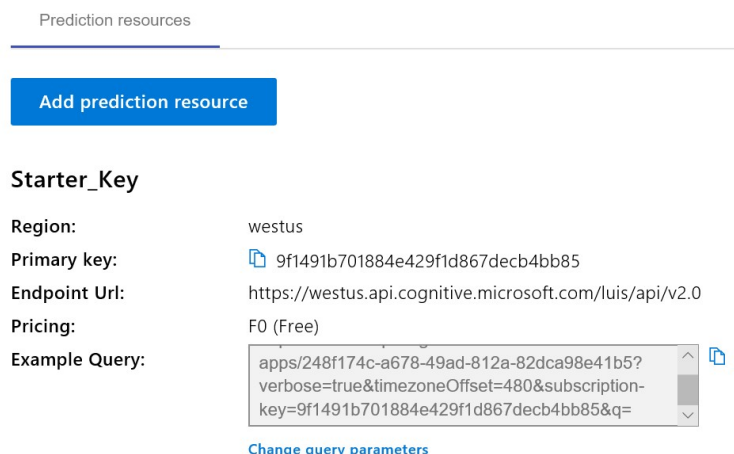


图 5.8 LUIS 发布情况详细信息图

5.3 后台聊天逻辑模块的设计与实现

后台聊天逻辑模块是系统中负责聊天逻辑的决策模块,是整个软件的核心。主要职责为设计尽可能符合用户对话习惯的聊天执行逻辑,同时基于聊天逻辑,处理与用户之间交互流程,通过与用户聊天获取服务需求意向,基于意向提出进一步问题,根据用户回答提取到的相关数据,结合数据库中信息生成用户所需的旅游信息或服务,最终以聊天回复的方式将生成的结果发送给用户。为实现以上功能,使用 Bot Framework 框架来开发实现后台聊天逻辑模块。

5.3.1 后台聊天模块开发概述

为设计通过与用户聊天获取需求并提供相应服务的聊天机器人逻辑,需要根据用户交互习惯设计并开发符合用户习惯的信息数据获取与信息服务提供的逻辑。

辑流程^[21]。首先用户进入聊天窗口后，向用户简要介绍本系统并提供给用户一些初始的交互选择以引导用户正确使用，例如让用户选择所需服务大类，首先建立一级对话主对话（Main Dialog），当用户加入聊天后，即在 Bot 框架中的 OnMembersAddedAsync 函数被触发时候，引导程序进入主对话模块。主对话模型提供一些系统介绍后，等待用户通过选择回复方式或者文本回复方式的回答，根据用户在主对话中回复的结果，带入 LUIS 模型得到语言分析结果。

```
App ID: 248f174c-a678-49ad-812a-82dca98e41b5
Recognizer Result  Raw Response
{
  "entities": [
    {
      "endIndex": 8,
      "entity": "成都",
      "score": 0.9950387,
      "startIndex": 7,
      "type": "location"
    },
    {
      "endIndex": 5,
      "entity": "后天",
      "resolution": {
        "values": [
          {
            "timex": "2020-03-31",
            "type": "date",
            "value": "2020-03-31"
          }
        ]
      }
    },
    {
      "startIndex": 4,
      "type": "builtin.datetimeV2.date"
    }
  ],
  "intents": [
    {
      "intent": "FlightBooking",
      "score": 0.9537165
    },
    {
      "intent": "HotelBooking",
      "score": 0.02952381
    },
    {
      "intent": "TrainBooking",
      "score": 0.01216451
    },
    {
      "intent": "TravelAdvance",
      "score": 0.00403513946
    },
    {
      "intent": "None",
      "score": 0.0004832104
    }
  ],
  "query": "帮我预订后天去成都的机票"
}
```

图 5.9 LUIS 模型反馈数据示意图

例如输入“预订后天去成都机票”，LUIS 返回结果如图 5.9，LUIS 模型可以

判断出用户机票预定意向为 95.37%，同时提取出目的地实体“成都”以及时间实体“后天”，将时间实体转化成对应公历时间：2020 年 3 月 31 日。

通过 LUIS 模型结果判断用户意向，根据模型反馈不同的意向结果，调入不同的二级对话进行进一步的交流。根据用户表现出来的不同的意向需求，可以将对话分为以下四种类型的二级对话：

(1) 航班对话 (Flight Dialog)：当用户在语言中表达的意向被判断为想要预订机票、航班。对话中需要进一步提问航班起降地、时间等信息。

(2) 火车对话 (Train Dialog)：当用户意向判断为购买火车票、高铁。对话中需要进一步询问火车起点终点及出发时间等信息。

(3) 酒店对话 (Hotel Dialog)：当用户意向判断为预订酒店、宾馆。对话中需要进一步询问酒店所在城市地区、宾馆附近地标区域、入住时间及酒店等级等信息。

(4) 旅行推荐对话 (Travel Advance Dialog)：当用户意向判断为景点推荐、攻略搜索。在对话中需要进一步向用户询问有关目标地点的信息。

如图 5.10，在不同的二级对话中，与用户进行进一步聊天对话，获取用户相关需求数据，在二级对话中根据用户需求，生成用户所需信息或服务，最终以聊天回复的方式推荐给用户。

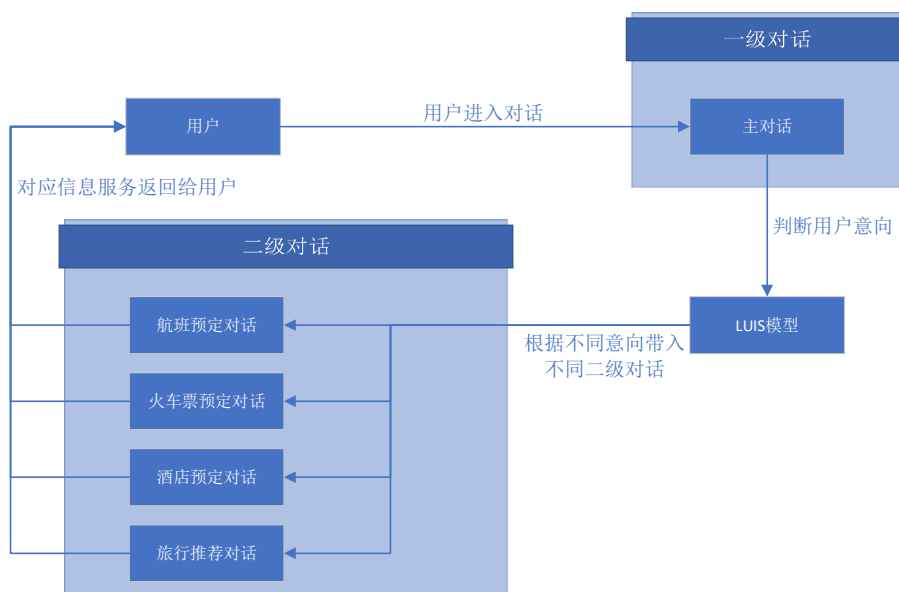


图 5.10 后台聊天逻辑模块对话逻辑图

5.3.2 创建新的 Bot 工程

Bot 框架的开发可通过 C#或者 Node.js 通过 BotBuilder 模板中使用 Echo Bot 工程实现^[22]。选择在 Visual Studio 2017 IDE 中设计并开发后台聊天逻辑模块,下载并安装 BotBuilder 模板,以支持在 IDE 中对 Bot 框架工程的开发。安装完成后,创建 Echo Bot 工程,得到一个空白的聊天框架模型。在 Bot 工程中编写相关代码并创建不同对话,以实现能够满足需求的逻辑。根据概述中所描述的五种对话类型,在 Bot Framework 工程中创建对应的对话流模块。

5.3.3 主对话开发

主对话的主要作用是提供功能介绍和功能导航,向用户介绍整个系统并通过提问引导用户通过文字输入或按钮点击方式,选择其所需要的信息服务类型。根据用户所点击选择按钮或输入的文字内容,带入 LUIS 语言理解模型判断用户意向。根据语言理解模型结果选择符合其意向的二级对话,将用户带入对应的二级对话。由于此步骤为对话起始,将主对话逻辑放在用户进入后执行,即当 Bot 框架中 OnMembersAddedAsync 函数被触发的时候,便开始执行主对话逻辑。

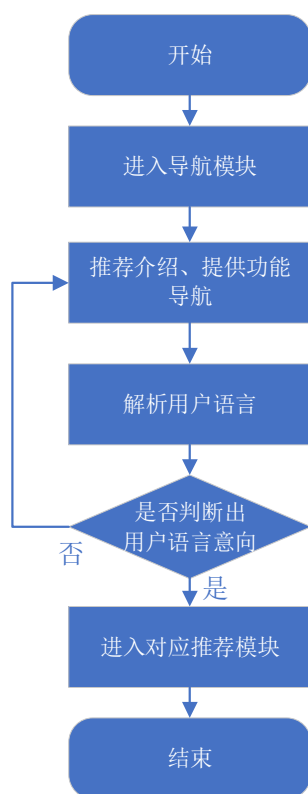


图 5.11 导航模块流程图

基于以上功能需求,选用瀑布型对话步骤。如图 5.11,为引导用户了解聊天机器人并与其聊天交流,使用 Bot Framework 中提供的 HeroCard 类来完成以上工作。在 HeroCard 中,通过标题和简介向用户简单介绍本系统,并向用户提供四个回复型选择按钮,代表系统的四种不同功能推荐。HeroCard 推荐卡片如图 5.12。当用户选择其中一种推荐点击按钮后,以文本方式回复按钮中对应的文字内容。同时支持用户直接在聊天窗口自行输入表达意向,越过引导直接提出需求。



图 5.12 主对话用户引导 HeroCard 界面

在得到用户自行输入的文字反馈或按钮点击的选择反馈后,我们将用户的回复内容以文字形式带入请求 LUIS 语言理解模块判断用户意向服务,语言理解模块则会返回 json 格式反馈,通过解析 json 格式的反馈,即可提取出模型给出的用户语言中最有可能的旅游相关意向,根据意向判断结果选择对应的二级对话,并跳转至该二级对话,结束主对话。

5.3.4 航班预定对话开发

当用户进入航班预定对话,即在主对话中,用户语言在自定义 LUIS 语言理解模型中被判断出意向服务类型为航班预订,为了能够提供相关航班预订链接,

我们需要在与用户交流的过程中得到航班计划时间、航班始发地、航班终到地等信息，对航班起始地分别带入数据库模块，检查城市是否存在，同时判断城市是否有机场信息，保证用户回复合理性，通过以上数据生成相关机票预订链接。将航班对话设置为五步瀑布模型步骤。分别是航班开始步骤、目的地处理步骤、出发地处理步骤、时间处理步骤以及总结步骤。

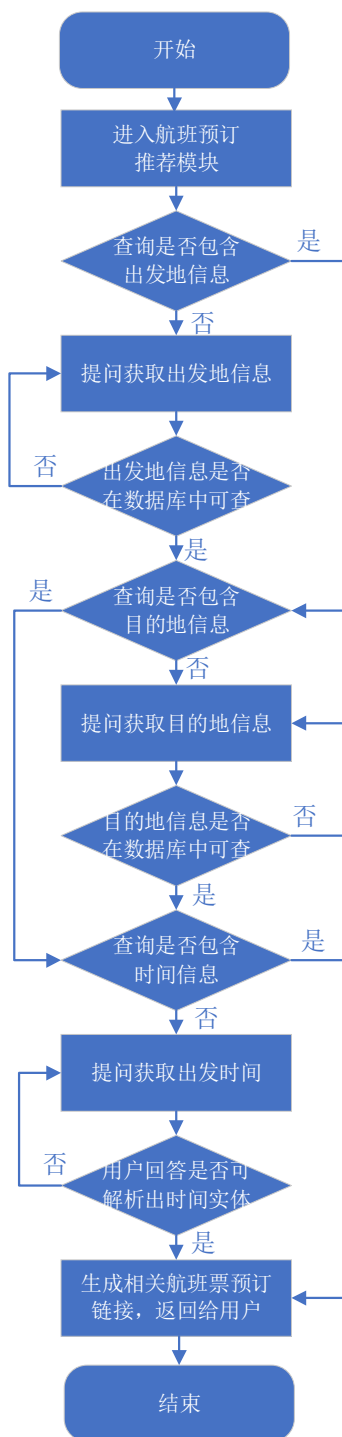


图 5.13 航班预定对话处理流程图

如图 5.13，航班对话逻辑流程如下：

- 航班开始步骤：目的为整合在主对话中航班相应的信息数据内容。将用户主对话中的回复，在 LUIS 语言理解模型提取结果进行进一步分析。从反馈中提取相关实体内容，检查反馈中是否包含始发地、终到地、出发日期等信息，即是否提取到出发地实体、目的地实体、时间实体内容。将提取到的实体对应数据在不同的实体类型分别保存以供后续步骤使用。
- 目的地处理步骤：目的为获取用户预订航班目的地相关信息。判断之前步骤是否已获取过目的地信息。查看已保存数据，是否包含目的地信息，如已经保存过目的地数据，直接跳至下一步对话当中，如未保存相关数据，通过聊天回复向用户询问需要预订航班目的地，将用户反馈带入数据库模块中检查，若找到符合城市并含有机场信息，将目的地保存后进入下一步。如未找到或无机场信息，继续重复询问本步骤。
- 出发地处理步骤：目的为获取用户预订的航班起飞地相关信息。判断之前步骤是否已获取过出发地信息。查看已经保存的用户数据，是否保存过出发地信息，如包含出发地信息，跳至下一步对话。若没有，通过聊天回复继续向用户提问需要预订飞机的出发地，将用户反馈带入数据库模块，若找到符合城市及机场数据，将目的地保存后进入下一步。如果未找到或无机场三字码，继续重复询问本步骤。
- 时间处理步骤：目的为获取用户预订航班的时间相关信息。检查之前的对话步骤是否已获取时间信息。查看已保存数据，是否包含时间信息。若包含，直接跳至下一步，否则通过聊天回复继续向用户询问出发时间，并将用户反馈再次带入 LUIS 语言理解模块，解析 LUIS 模型所反馈的 json 数据，从中提取时间实体，若存在时间实体，继续解析 json 数据从而获取用户时间实体所代表的公历日期，保存 LUIS 模型中所判断出来的公历时间。若 LUIS 模块的反馈中无法提取到时间实体，继续重复询问本步骤。
- 总结步骤：目的为确认航班预订信息并生成航班预订链接。将之前步骤获取信息汇总，通过聊天回复将获取到的信息发送给用户，提供选择按钮，

询问用户是否确认航班预订信息，若选择否，删除所有用户数据重新开始航班对话。若选择是，选取当前主流机票预订网站，分析每个网站机票预订网页 URL 及其参数构成，使用从用户已经获取到的需求信息，生成满足用户需求且可以直接进行机票预订的链接，创建 HeroCard 推荐所生成的链接，在 HeroCard 中向用户提供打开链接方式按钮，将生成的链接依次放入按钮中。用户点击对应的按钮，即可打开其所需要的机票订购页面，此时结束对话。用户可以通过聊天对话框打开的机票订购网页直接查找合适航班并提交订单购买机票。

航班对话模块开发完成后，对用户语言导航式航班意向对话逻辑进行测试，测试结果如图 5.14：



图 5.14 航班预订聊天测试结果图

5.3.5 酒店预订对话开发

当用户进入酒店预订对话，即在主对话中，LUIS 语言理解模块根据用户语言判断出用户意向为酒店预订服务。为向用户提供相关酒店宾馆的预订链接，需继续与用户聊天，获取酒店的入住时间、入住城市地区、入住附近地标区域、酒店目标舒适度等信息，生成符合用户需要的相关酒店预订链接。利用在数据库中查找到的城市地区数据信息，生成酒店预订链接。综上所述，酒店对话设计为以下四步的瀑布模型步骤。

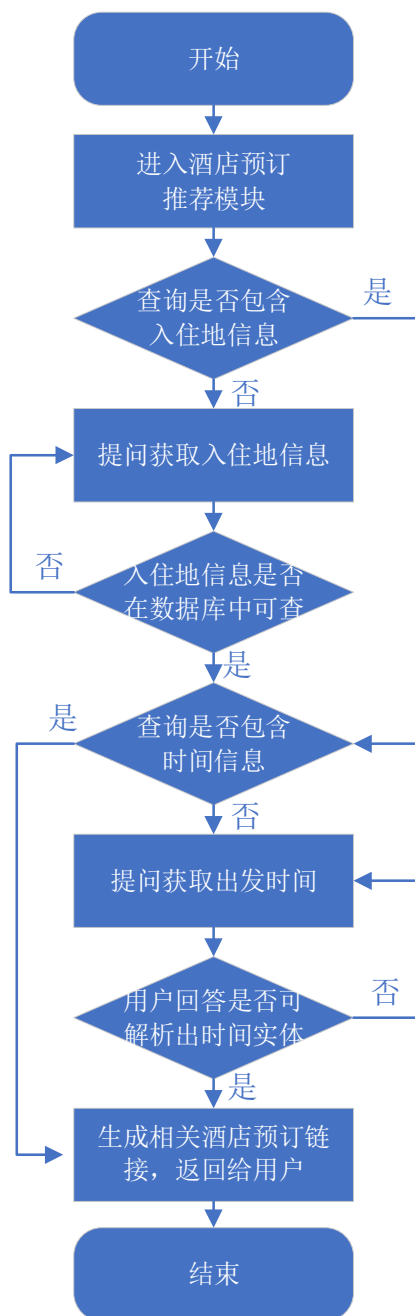


图 5.15 酒店预订对话流程图

如图 5.15，酒店对话聊天逻辑流程如下：

- 酒店对话开始步骤：目的为整合在主对话中宾馆相应的信息数据内容。对主对话中 LUIS 语言理解模型所返回的提取结果进行进一步分析。从反馈中提取出相关实体内容，检查反馈中是否包含入住城市、入住区域、入住日期等信息，即从反馈中是否可以提取到目的地实体、时间实体等内容。若提取到，继续提取实体在语言中对应的提取值，将提取值依照不同的实体类型分别保存，以供下游步骤使用。当获取到对应实体，则不再进行对应进一步对话。
- 地址处理步骤：目的为获取用户预订的宾馆地点相关信息。判断之前步骤是否已经获取目的地信息。查看已保存的数据，是否包含宾馆地点信息，若包含地址信息，跳至下一步操作，若未找到地点信息，通过聊天回复向用户询问需要预订酒店的目的地信息，并将用户反馈中的目的地实体带入数据库模块，查找是否存在匹配城市地区。若未找到，继续重复询问本步骤。若找到匹配城市，保存目标城市地区，继续在数据库中匹配所有从属关系的地标信息，查看是否可以继续匹配该城市某地标。若找到匹配地标，保存新获取到的区域地标信息。
- 时间处理步骤：目的是获取用户预订酒店的时间信息。判断在已经进行过的对话回复中，是否已经获取入住时间信息。即查找已经通过对话获取到的数据，是否包含时间实体。若包含相关信息，直接跳至下一步，否则通过聊天回复的方式继续向用户询问出发时间，并将用户反馈再次带入 LUIS 语言理解模块，解析 LUIS 模型所反馈的 json 数据，从中提取时间实体。若存在时间实体，继续解析 json 数据获取用户时间实体所代表的公历日期。将 json 中所提取到的公历时间保存。若 LUIS 模块的反馈中无法提取到时间实体，继续重复询问本步骤。
- 总结步骤：目的为确认酒店预订信息的同时，生成酒店预订链接。将之前步骤获取到的信息汇总，通过聊天的方式将获取到的信息展示给用户，提供选择按钮由用户进行确认，询问用户是否确认宾馆预订信息，若选择

否，则删除所有数据重新开始酒店预订对话。若选择是，选取多个当前主流酒店预订网站，分析其酒店预订网页的 URL 及其参数构成，使用从用户已获取的酒店时间、酒店预订城市、酒店预订附近地标区域等信息。基于已有的城市数据集来直接获取或生成满足用户需求的可直接预订酒店的链接。创建 HeroCard 推荐所生成的宾馆预订链接，在 HeroCard 中向用户提供打开链接形式的功能按钮，将生成的链接依次放入按钮中。用户点击对应的按钮，即可打开根据用户要求所生成的酒店预订页面，此时结束对话。用户可在打开的酒店预订网页直接提交订单预订酒店。

酒店对话模块开发完成后，对导航模式对话逻辑进行测试，具体测试如图

5.16:



图 5.16 酒店预订聊天测试结果图

5.3.6 旅行推荐对话开发

当用户进入旅行推荐对话即在主对话中，LUIS 语言理解模块判断出用户意向为旅行相关的信息推荐。为向用户提供旅行相关信息资讯，需进一步与用户聊天获取到旅行计划地信息。为向用户推荐相关旅行介绍，需要一个数据源提供相关旅行推荐数据。分析某旅行攻略网站的链接构成，例搜索上海，分析目标网站的搜索页面的地址参数信息：“https://{域名}/search/q.php?q=上海”。通过链接，从用户回答内容中获取到的目的地实体信息，写入 URL 链接中的 q 参数，即可合成查找目标地点搜索链接，通过请求该链接得到关于目的地搜索结果页面。解析页面得到用户需要的信息，通过对话方式返回给用户。

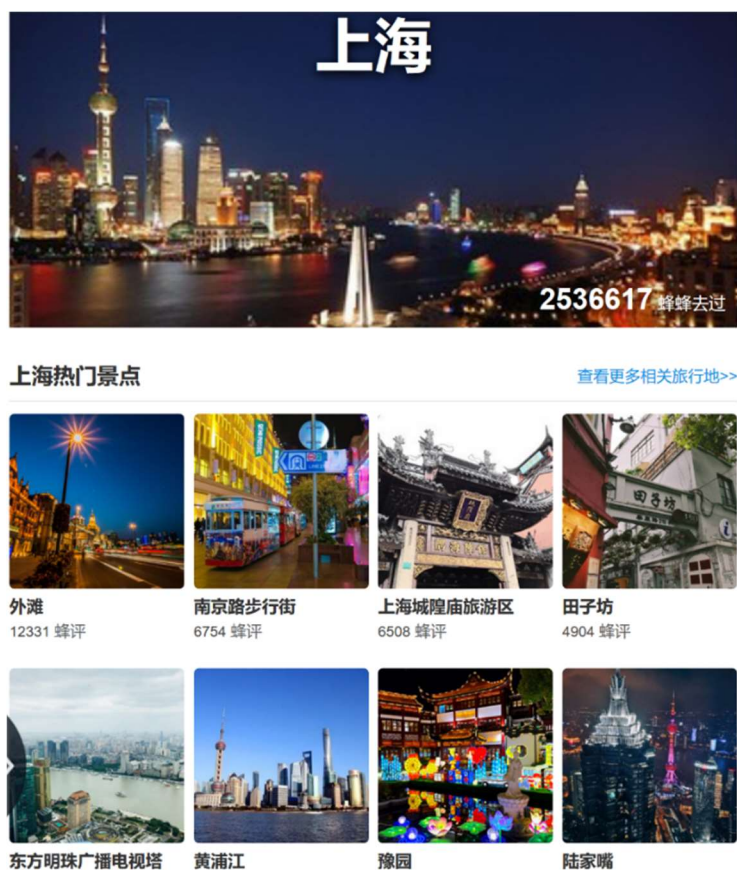


图 5.17 目标网站“上海”搜索结果页

如图 5.17 为需要爬取目标网站搜索结果页截图，从搜索结果页面中通过网页解析的方法，获取到城市图片、相关景点、相关博客、相关攻略等信息，即通过在线爬虫的手段可获取对应城市或地区的攻略信息，并利用 HeroCard 把获取

到的信息提供给用户，方便用户查看。

为进一步向用户获取数据，需针对旅行推荐继续向用户提问，将旅行推荐对话设置为两步的瀑布模型步骤。

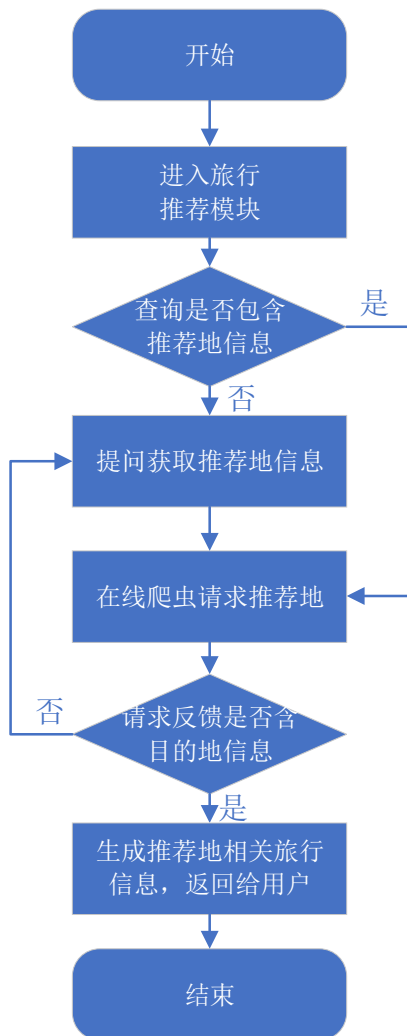


图 5.18 旅行推荐对话流程图

如图 5.18，旅行推荐对话逻辑流程如下：

- 旅行目的地处理步骤：目的为获取用户旅行攻略目的地信息并在线确认目的地信息有效性。利用主对话中请求 LUIS 语言理解模型所提取的结果进行进一步分析。从反馈中提取相关实体内容，检查反馈中是否包含旅行攻略目的地信息，即从反馈中是否可以提取到目的地实体内容。若可以提取到，获取目的地实体对应的提取值作为候选旅行目的地。若从反馈中无法提取到目的地实体，通过聊天回复方式向用户询问需要查询的旅行目

的地信息，此时使用候选目的地生成目标网站搜索结果链接，通过 C# 中 `HttpWebRequest` 类可以请求网页并获取网页 HTML 格式的页面内容^[23]。根据目标搜索结果链接构造，拼接生成目标网页链接，并使用 `HttpWebRequest` 类对页面进行在线抓取，获得 HTML 格式的目的地搜索结果页面，使用 `HTMLDocument` 类解析搜索结果页，基于网页中的类名和标签名定义指向页面中地点推荐块的 XPath，通过推荐块的 XPath 判断页面中是否存在地点推荐块。若页面中存在地点推荐块，保存目的地信息并进入下一步。若不存在目的地信息，表明搜索结果页面无法搜索到相关信息，即目的地不是一个地点，告知用户地址无效信息并继续重复询问本步骤。

- 总结步骤：目的为确认推荐信息并在线获取搜索结果网页相关信息，向用户提供旅行相关的景点攻略信息。向用户确认获取到的旅行目的地信息，通过聊天方式向用户展示提取到的信息，提供选择按钮，询问用户是否确认信息。若选择否，则删除所有数据重新开始旅行推荐对话。若选择是，继续解析在旅行目的地处理步骤中得到的地点推荐块。设计页面中的类名与属性标签定义指向地点推荐块中背景图片和地点列表块的 XPath，通过 XPath 从页面中提取相应信息数据，包含背景图片 URL 以及相关地点列表内所有景点名称及链接。获取数据后，使用 `HeroCard` 类，把从网页中解析出来的城市背景推荐图片 URL 设置为 `HeroCard` 展示背景图，将旅行目的地名设置为 `HeroCard` 标题。使用打开链接模式的按钮将网页中获取到的地点及其链接依次展示。当用户在聊天窗口中看到 `HeroCard` 的推送时，即可对旅行目的地有最基本了解。用户点击 `HeroCard` 中的按钮后，便可打开相应的目的地或景点介绍页面，可通过该页面更详细的了解的目的地及其景点，得到相关的博客攻略等推送信息，方便用户更深一步的获取目的地及景点的信息。

旅行推荐对话模块开发完成后，对其逻辑进行本地测试，具体测试如图

5. 19:

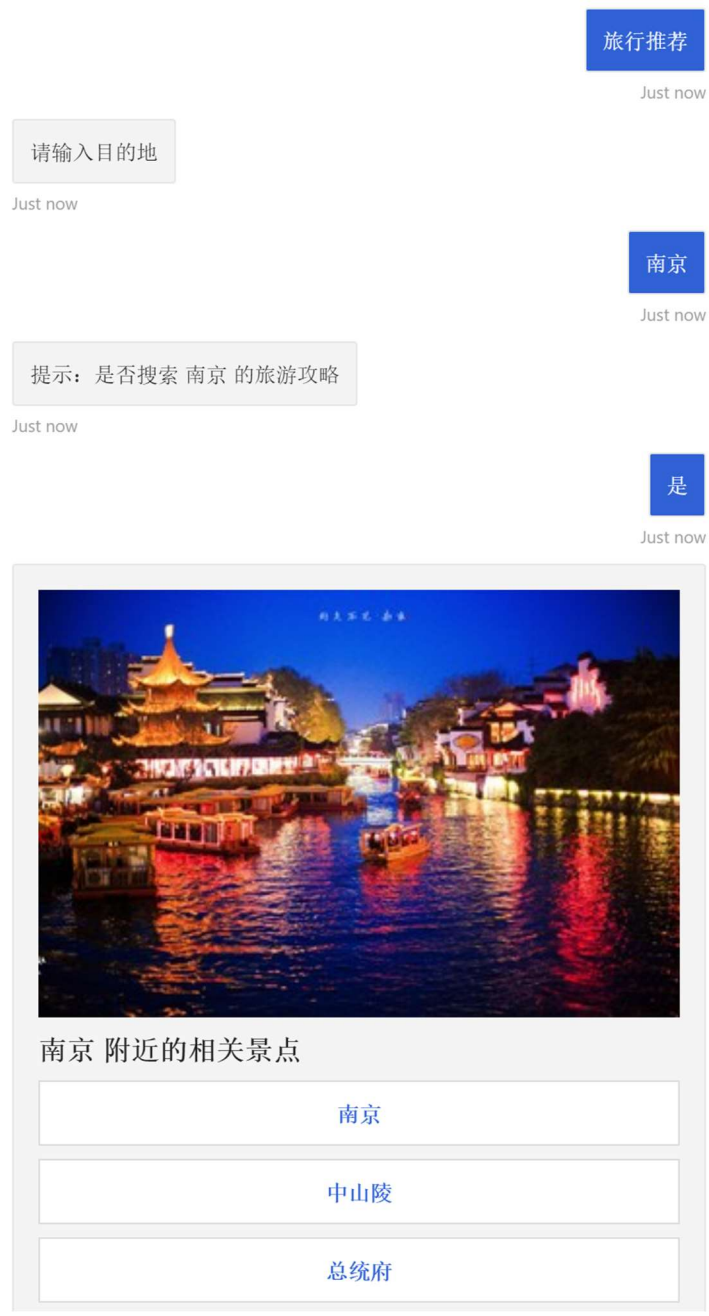


图 5.19 旅行推荐聊天测试结果图

5.3.7 火车票预定对话开发

当用户进入火车票预定对话，即在主对话 LUIS 语言理解模型中，用户意向被判断为火车高铁票预订意向。为了能够提供相关铁路预订服务链接，需要与用户进行进一步对话以得到出发时间、火车始发地、火车终到地等方面的信息，利用对话获取到的用户信息以及从数据库中查询的相关数据生成相关铁路预订链接。为实现以上对话，将铁路对话设置为五步瀑布模型步骤。具体流程如图 5.20：

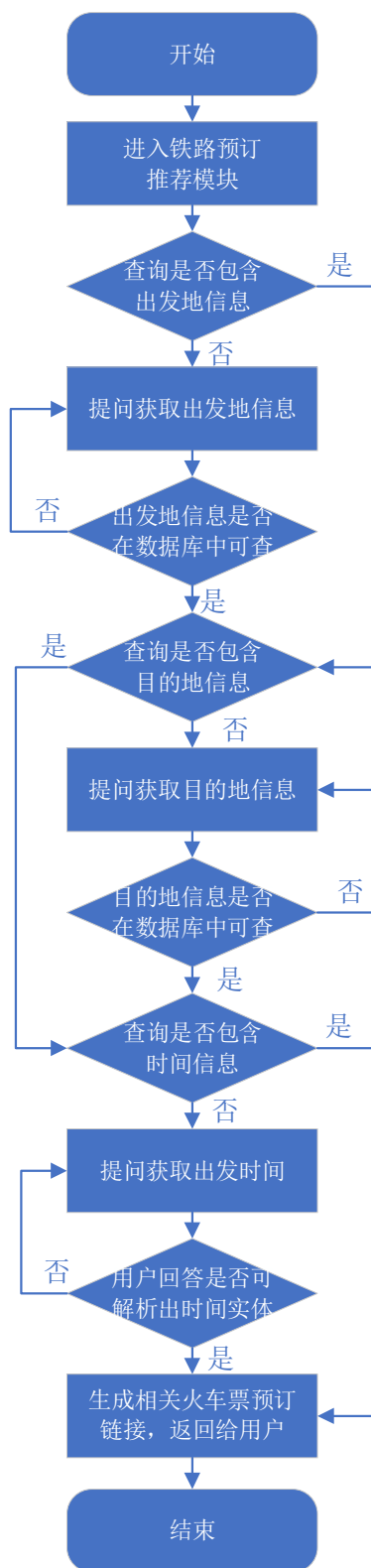


图 5.20 铁路预定对话流程图

- 开始步骤：目的为整合在主对话中铁路预定相应的信息数据内容。将用户
在主对话中请求 LUIS 语言理解模型所提取的结果进行进一步分析。检查

反馈中是否包含始发地、终到地、出发日期等信息，即是否提取到出发地实体、目的地实体、时间实体内容。若提取到，从反馈中获得实体对应的提取值，并将提取值依照不同的实体类型分别保存以便之后步骤使用。

- 目的地处理步骤：目的为获取用户预订的目的地相关信息。判断之前步骤是否已经获取目的地信息。查看已保存的数据，是否包含目的地信息。若包含该信息，跳至下一步，若不包含，通过聊天回复方式向用户询问需要预订铁路服务的到达站点信息，并将目的地实体带入数据库模块进行查找，若找到对应城市，将目的地保存后进入下一步，若不存在对应数据，继续重复询问本步骤。
- 出发地处理步骤：目的为获取用户预订的铁路出发地相关信息。判断已经结束的步骤中，是否已经提取过铁路出发地信息。查找之前对话中已保存信息，是否包含铁路出发地相关信息。若包含出发地数据，跳至下一步，其他情况则通过聊天回复的方式继续向用户询问需要预订火车票的出发地，并将用户的回答代入数据库模块，若找到符合城市或车站，将出发地保存后进入下一步对话。若未找到符合的城市或车站，重新询问本步骤。
- 时间处理步骤：目的为获取用户预订火车出行的时间相关信息。判断对话中运行结束的步骤中，是否已提取到出发时间相关信息。查看已保存数据，是否包含时间信息。若包含，直接跳至下一步，若不包含，通过聊天回复继续向用户询问出发时间，将用户反馈再次带入 LUIS 语言理解模块，解析 LUIS 模型所反馈的 json 数据，从中提取时间实体。若存在时间实体，继续解析 json 数据从而获取用户时间实体所代表的公历日期，保存在 LUIS 时间实体中，提取到对应公历时间。若 LUIS 模块的反馈中无法提取到时间实体，继续重复询问本步骤。
- 总结步骤：目的为确认火车预订信息同时生成火车预订服务链接。汇总之前步骤获取到的信息，利用聊天回复方式将生成的信息展示给用户，提供选择按钮，询问用户是否确认之前得到的预订信息。若选择否，则删除所有数据重新开始铁路预订对话。若选择是，选择当前主流铁路预订网站，

根据目标网站的火车票预订网页 URL 构成,使用从用户获取到的出发地、目的地及出发时间等信息生成订购目标网站火车票预订网页链接,以满足用户预订服务需要,创建 HeroCard,在 HeroCard 中向用户提供打开链接方式按钮,将生成的链接依次放入按钮中。当用户点击对应按钮,即可打开其所需要的火车票订购页面,此时结束整个对话。用户可以在打开的火车票预订网页直接提交订单购买火车票。

在 Bot Framework Emulator 中测试火车服务预订对话部分聊天,结果如图 5.21:



图 5.21 铁路服务预订聊天测试结果图

5.3.8 后台聊天逻辑模块整体测试

软件测试是软件工程的重要组成部分,通过软件测试可保证软件开发符合预期需求效果^[24]。当完成以上后台聊天逻辑模型的整体开发后,可使用 Bot

Framework Emulator 应用对实现模型后台聊天处理判断与决策逻辑进行整体调测。在 Visual Studio 中编译运行 Echo Bot 工程。工程编译运行成功后，生成本地聊天机器人接口。接口信息中包含本地聊天机器人服务的端口号，将生成的端口号填入 Bot Framework Emulator 中即可将其与聊天机器人模型链接并运行对话，与本地机器人聊天查看机器人对话内容是否按照所设计的逻辑执行，分别与机器人进行不同意向、不同内容涵盖的聊天，判断本地机器人是否符合设计需要。通过聊天窗口测试模块运行如图 5.22:

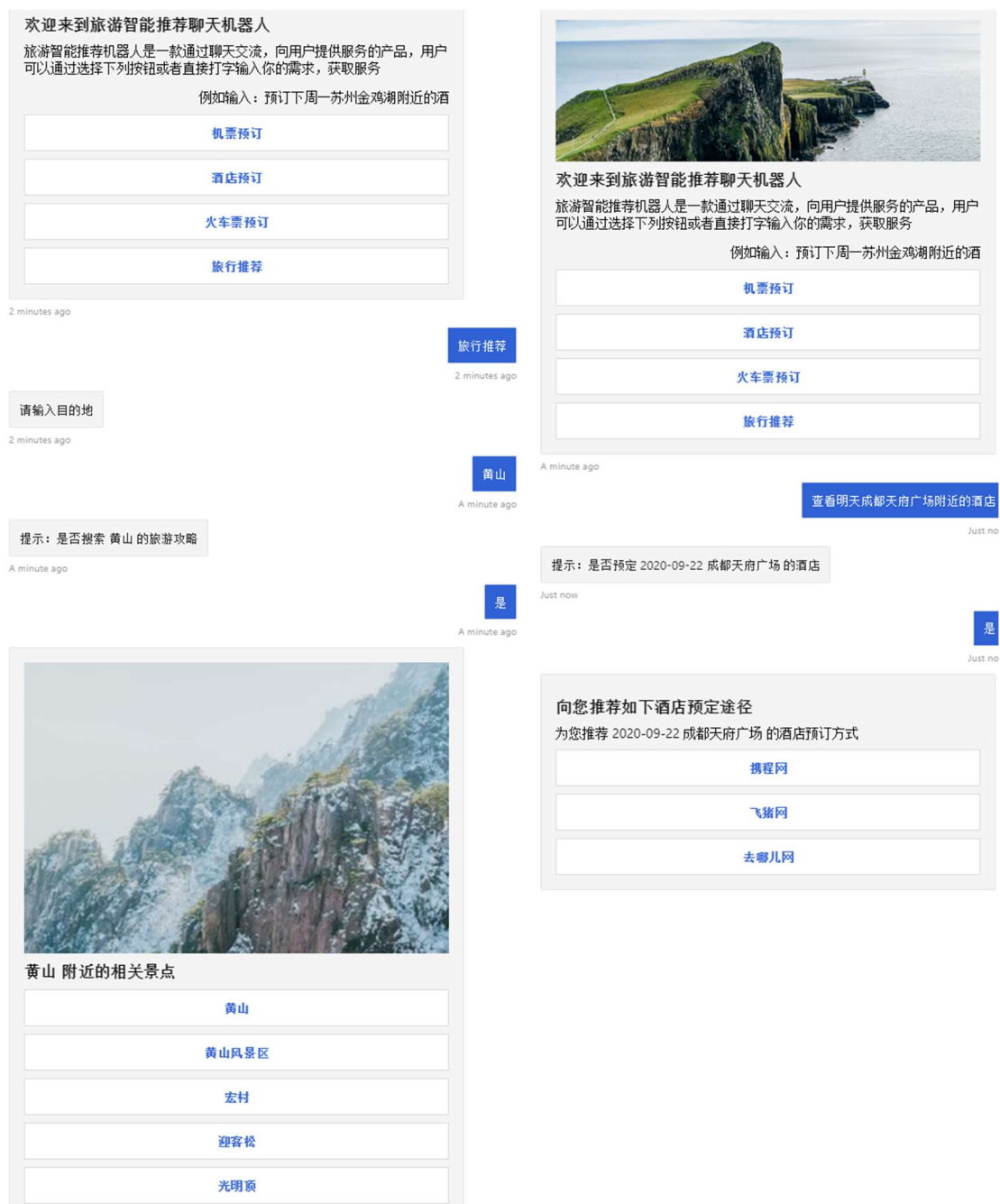


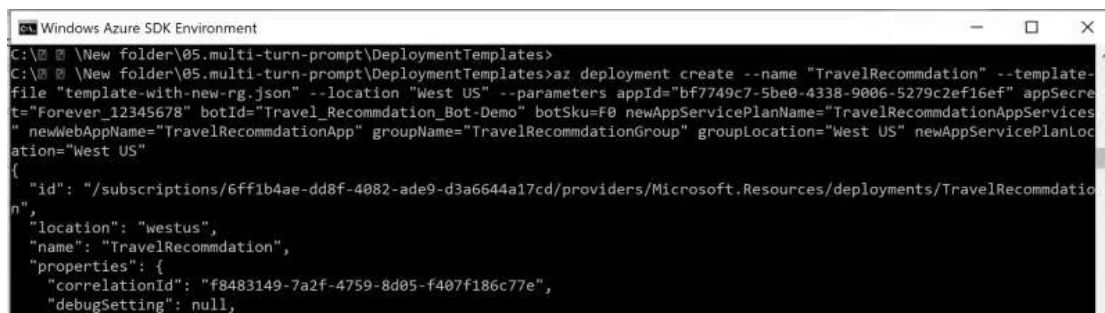
图 5.22 Bot Framework Emulator 测试后台聊天系统图

根据以上的逻辑测试结果, 可知后端聊天逻辑模块可以准确判断出用户旅行相关的意向, 根据判断的用户意向结果切换至对应的二级对话, 在二级对话中按照设计, 进一步向用户提问, 从用户回答内容中提取用户旅行相关的信息数据, 根据提取用户数据及数据库查询数据, 生成对应符合用户要求的旅行相关服务链接与信息, 通过聊天方式返回给用户。工程整体逻辑架构与执行符合设计要求, 可将后台聊天逻辑模块发布至 Azure 云服务器, 供前端模块中使用。

5.3.9 模型 Azure 发布

将聊天机器人发布在 Azure 供前端模块对后台聊天逻辑模块的调用, 注册 Azure 账户创建并管理云端工程。在电脑中安装 Azure CLI 命令行组件, 安装完成后在 Azure CLI 中使用命令行登陆 Azure 账户, 新建应用程序和 Bot 服务信道, 将本地聊天机器人上传至云端, 信道与应用程序创建命令结果如图

5.23:



```
Windows Azure SDK Environment
C:\New folder\05.multi-turn-prompt\DeploymentTemplates>az deployment create --name "TravelRecommndation" --template-file "template-with-new-rg.json" --location "West US" --parameters appId="bf7749c7-5be0-4338-9006-5279c2ef16ef" appSecret="Forever_12345678" botId="Travel_Recommndation_Bot-Demo" botSku=F0 newAppServicePlanName="TravelRecommndationAppServices" newWebAppName="TravelRecommndationApp" groupName="TravelRecommndationGroup" groupLocation="West US" newAppServicePlanLocation="West US"
{
  "id": "/subscriptions/6ff1b4ae-dd8f-4082-ade9-d3a6644a17cd/providers/Microsoft.Resources/deployments/TravelRecommndation",
  "location": "westus",
  "name": "TravelRecommndation",
  "properties": {
    "correlationId": "f8483149-7a2f-4759-8d05-f407f186c77e",
    "debugSetting": null,
  }
}
```

图 5.23 机器人使用 Azure SDK 命令发布截图

服务生成后, 命令行反馈信息包含新生成的 Azure 线上机器人应用程序的密钥 Id 和应用程序密码, 返回等待上传的 Echo Bot 工程中, 将服务所生成的应用程序密钥 Id 和密码填入 Bot Framework 工程 App Setting 中的密钥及密码字段当中, 线上注册该密钥作为机器人唯一的标示信息。如图 5.24, 最终在 Visual Studio 中, 将本地聊天机器人模块发布到新创建的 Azure Bot 服务信道。得到线上聊天机器人。

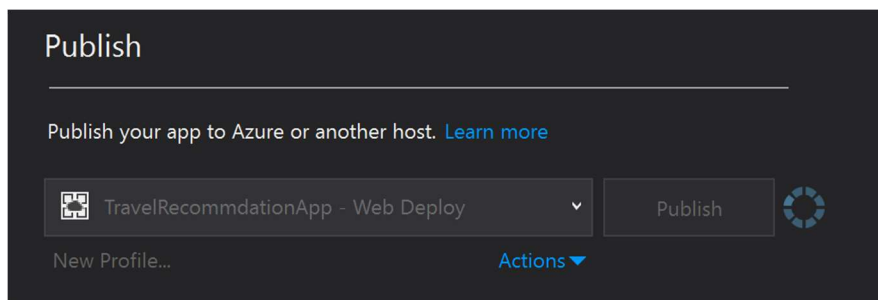


图 5.24 Visual Studio 中将聊天机器人发布到 Azure 执行图

5.3.10 Azure 线上聊天机器人服务测试

当后台聊天逻辑模块成功上传至 Azure 服务器后，即可在 Azure 控制台中找到 Bot 服务信道，在 Azure 平台的 Web Chat 环境中测试发布到线上的机器人，测试发布到线上机器人模型逻辑是否和本地机器人测试效果一致。

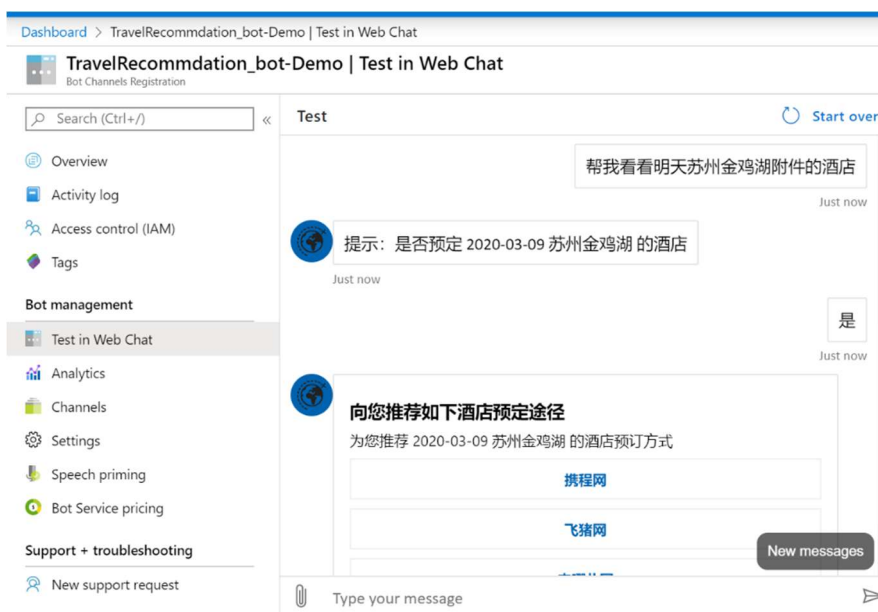


图 5.25 聊天机器人在 Azure 平台测试图

测试结果如图 5.25，根据以上测试结果，可知在 Azure 端的线上聊天机器人功能和本地开发的机器人逻辑与功能方面完全一致，符合预期设计要求。为线上机器人开通 Web Chat 信道和 Skype 信道，通过不同信道，Azure Bot 聊天机器人可以应用在不同类型终端使用。接下来需要设计前端模块，在前端界面中引入该线上机器人 Web Chat 信道，使用户通过信道入口实现与线上聊天机器

人的交互。

使用 Skype 聊天软件测试聊天机器人在 Skype 频道上的模型运行效果，在 Skype 聊天软件测试如图 5.26，经过测试可知，Skype 聊天软件端聊天机器人与本地开发机器人功能一致。

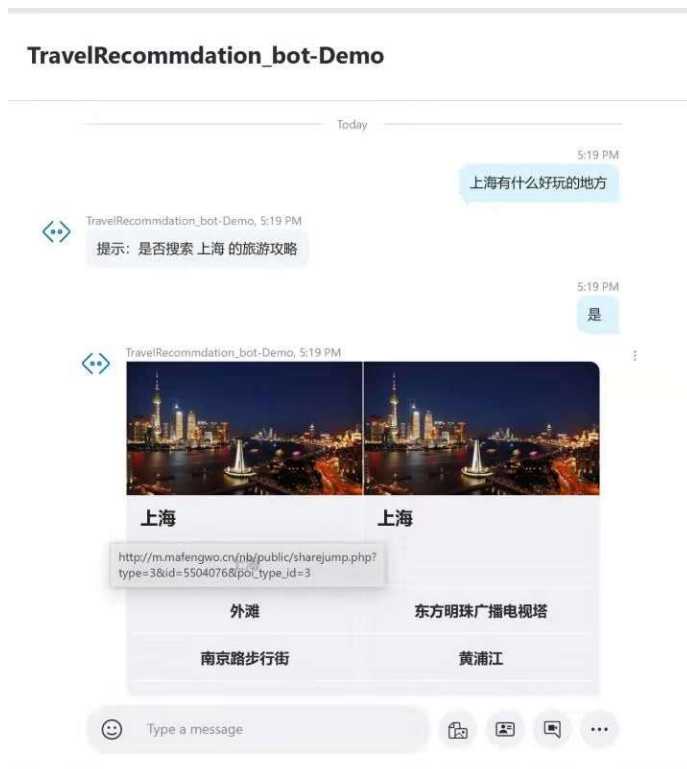


图 5.26 聊天机器人在 Skype 聊天软件测试图

5.4 前端模块的设计与实现

前端模块需要在 Chromium 内核的浏览器中开发扩展应用程序作为聊天机器人客户端，在前端页面中设计聊天窗口，使用户能够打开扩展应用后在聊天窗口中自由输入与发送信息。前端系统连接后台聊天逻辑模块，可随时将用户回答传送到后台聊天逻辑模块，供后台聊天逻辑模块决策，并将后台聊天逻辑模块最终反馈信息展示给用户。使用户能够通过扩展中聊天窗口和线上聊天机器人完成交互，成为连接用户与后台聊天逻辑模块的桥梁。

5.4.1 开发扩展框架

为了开发出完整的前端模块，可开发新的在 Chromium 内核浏览器中安装使

用的 Chrome 扩展应用。编写负责 Chrome 扩展应用整体框架设置的 manifest 文件，manifest 文件是 Chrome 扩展应用中一个 json 格式应用框架设置文件，这个文件指定了扩展应用整体框架及具体设置。为了使前端交互界面以及浏览器扩展应用符合设计要求。按照表 5.2 中的内容设置扩展应用。

表 5.2 扩展应用 manifest 文件设置介绍表

Manifest 变量名	变量介绍及作用	设置值
Name	扩展应用名称	旅行推荐聊天机器人
Description	扩展应用简介	可以与用户通过聊天方式 向用户推荐用户所需的旅行信息服务的聊天机器人
Permissions	扩展应用所需权限	activeTab 活动标签 storage 储存
Version	扩展应用版本	1.0
Browser action default_popup	应用默认弹出窗口 当扩展应用弹出窗口打开 时，在弹出窗口中展示此处 设置本地页面	Travel Recommendation HomePage.html
Browser action default_icon	应用标签按钮图片 在浏览器应用扩展块中扩展 应用的图标	/images/travel.png
Browser action default_title	在应用图标悬停时，展现的 标题	旅游辅助推荐系统
Converted from user_script	是否使用用户内联脚本 使扩展应用支持 Web Chat 控 件所需要使用内联脚本	True

通过以上设置，新建设置中所需引用的本地文件，会得到一套完整的包含

前端网页的 Chrome 扩展应用框架。安装该扩展应用至 Chromium 内核的浏览器后，当再次打开该浏览器时，浏览器右上方扩展应用列表中可以看到扩展应用图标，点击扩展应用图标，弹出浮动在浏览器之上的浮动网页窗口，浮动网页内容即为应用中设置的本地网页。搭建完 Chrome 扩展应用框架后，下一步需要实现在打开的浮动页面展现聊天窗中连接后台聊天逻辑模块。

5.4.2 引入 Web Chat 嵌套网页

Iframe 标签在 HTML 中规定了将外部框架嵌入在网页中的内联框架，在网页中构建一个沙盒模型使之不受标签以外的前端框架行为控制^[25]。当搭建完成扩展框架后，在扩展应用中指定的 HTML 文件中，调用 Bot Framework 中提供的 Web Chat 相关嵌套网页，使用 iframe 标签把 Web Chat 嵌套内容内联至本地的前端页面当中，然后在 Azure 平台聊天机器人应用设置中创建后端逻辑模块的 Web Chat 频道，在频道中生成 Token Id 作为聊天机器人在 Web Chat 接口中使用的唯一标识，将经过测试的 Azure 线上聊天机器人 Web Chat 频道的 Token Id 设置在本地 Web Chat 嵌套引用链接中，使前端 Web Chat 控件能够通过 Token Id 识别并调用线上聊天机器人，完成后台系统与前端系统的连接。当获取到用户输入后，将用户回复内容反馈给后台聊天逻辑模块。将后台聊天逻辑模块做出的回复同样通过聊天窗口展示给用户。当用户在浏览器中打开扩展应用后，即可得到一个聊天窗口，通过与其聊天便可实现用户与线上聊天机器人聊天交互的过程。

超链接标签是用户在网页中重定向的标签。在页面下方添加超链接标签，在 Azure 平台聊天机器人应用设置中创建 Skype 频道，生成接通线上聊天机器人的 Skype 频道链接，将 Azure 中 Skype 频道的指向链接作为一个点击标签放在聊天框下方，当用户点击后即可跳转至 Skype 聊天软件，将线上聊天机器人添加至 Skype 聊天软件，方便用户通过 Skype 聊天软件，而不是单一的网页聊天方式与线上机器人进行交互。在浏览器中打开包含 Web Chat 的扩展应用如图 5.27。

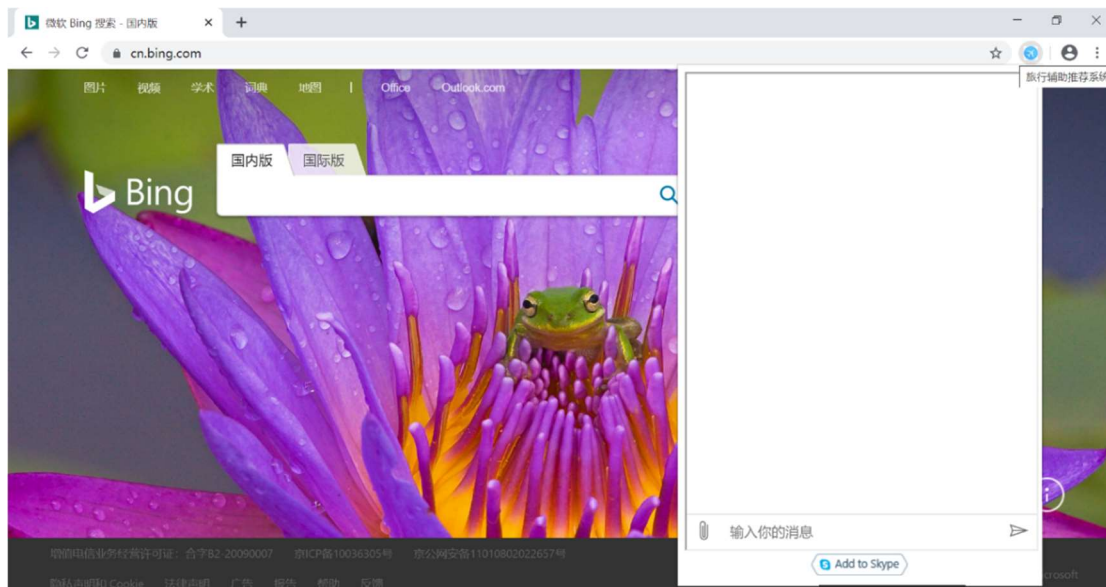


图 5.27 浏览器扩展应用中打开聊天窗口效果图

5.4.3 安装扩展至 Chromium 内核浏览器

选择 Chrome 浏览器作为旅行推荐聊天机器人的扩展应用安装测试工具，在 Chrome 浏览器中将旅行推荐聊天机器人扩展应用程序目录打包，可得到一个 CRX 安装文件，在 Chromium 内核浏览器的扩展程序设置中，安装旅行推荐聊天机器人扩展应用安装包。完成安装后可在扩展应用列表看到如图 5.28 的应用信息。

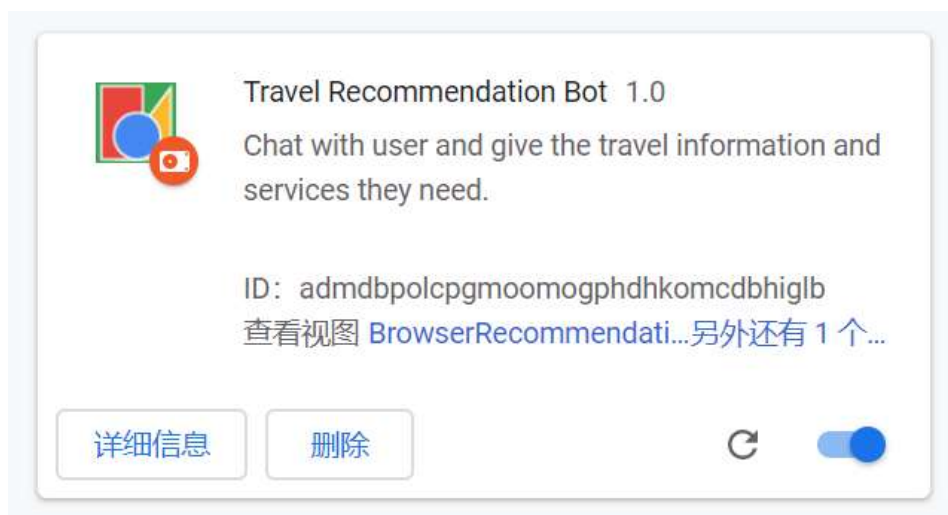


图 5.28 扩展应用在 Chrome 浏览器上安装成功图

程序安装后，重启浏览器，如图 5.29，浏览器的扩展应用列表中出现扩展

应用对应启动按钮。点击扩展应用标签，即可打开扩展应用，弹出聊天框方便用户使用。



图 5.29 安装扩展应用的浏览器效果图

5.5 本章小结

通过本章介绍的模型详细设计开发流程，经过批量化网页请求及数据获取、数据库构建、语言理解模型设计与开发、后台聊天逻辑模型的设计与实现，提供用户交互前端界面的扩展应用设计与实现，得到嵌入在浏览器扩展应用和 Skype 聊天软件的可通过与用户聊天获取用户需求并提供旅行相关信息服务的聊天对话机器人。

第六章 系统测试

6.1 测试目的

经过以上过程的开发，我们已经完成所有的系统模块开发，得到了旅行推荐聊天机器人扩展应用。为了确认模型设计完全符合设计预期，需要对模型进行整体测试。包含对所有正常功能模块逻辑、错误信息处理的逻辑测试以及对系统整体的压力测试，以验证最终模型逻辑上是否符合设计需求，性能上是否符合承载要求。

6.2 测试步骤

黑盒测试即对软件逻辑不了解的情况下，通过设计输入检查软件输出是否符合预期效果。软件测试需要根据黑盒原则，测试软件所有不同逻辑输入得到的结果是否符合预期^[26]。主要是对联调后最终得到的 Chrome 扩展应用不同功能模块分别进行整体测试，整体测试的具体步骤如下：

- 打开已经安装了旅行推荐机器人扩展应用的浏览器。
- 在浏览器扩展应用菜单列表中找到旅行推荐聊天机器人扩展应用图标，点击图标打开扩展应用。
- 浏览器弹出聊天窗口，通过弹出的聊天窗口页面咨询旅行服务，按照聊天指示与聊天窗口对话或主动输入完整需求，分别测试基于机票预订、酒店预订、旅行推荐、火车票预订四种不同目的的聊天，分别测试用户直接回复和经过选择引导回复不同方式测试模型。
- 得到最终信息服务反馈后，检查窗口最终提供服务是否满足输入的内容要求，检查意向判断是否准确、信息提取是否匹配、推荐的内容是否满足用户需求。

6.3 测试结果分析

6.3.1 航班预订对话测试

首先测试机票预订服务聊天流程，分别测试对话含有实体文字回复和不含

实体按钮点击回复两种场景下机器人回复是否符合预期设计。



图 6.1 聊天机器人航班对话测试图

如图 6.1，对含有实体的对话分析，输入“帮我预订后天去北京的机票”，模型准确分析出航班意向并进入航班对话，提取出目的地实体“北京”和时间实体“后天”，并跳过目的地询问和出发时间询问，最终返回上海至北京 4 月 12 日机票预订链接。点击不同链接，均可在浏览器内部跳转至不同网站 4 月 12 日上海至北京机票预定链接。对不含实体的对话分析，用户点击机票预定按钮后，按照航班对话顺序，依次询问出发地、目的地、时间并确认信息，最终返

回 4 月 12 日上海至成都机票预订链接。不同链接均可打开不同网站 4 月 12 日上海至成都机票预定链接。经过测试，航班预订模块对话内容与返回链接符合要求。

6.3.2 酒店预订对话测试

测试酒店预订服务对话，分别测试有实体文本回复和导航点击回复对话。



图 6.2 聊天机器人酒店预订对话测试图

如图 6.2，对含有实体的对话分析，输入“帮我预订明天苏州观前街附件的酒店”，模型准确分析出酒店意向并进入酒店对话，提取出目的地实体“苏州”、目的地区域“观前街”和时间实体“明天”，并跳过入住城市询问和入住时间询问，最终返回 4 月 11 日苏州观前街附件的酒店预订链接。点击后可返回

不同网站 4 月 11 日苏州观前街附近的酒店预订链接。对不含实体的对话分析，用户点击“酒店预订”按钮后，按照酒店对话顺序，依次询问出发地，目的地，时间并确认信息，最终返回 4 月 13 日重庆的酒店预订链接。点击后可以打开重庆 4 月 13 日不同网站的酒店预订链接。经测试，酒店预订模块对话内容与返回链接符合要求。

6.3.3 旅行推荐对话测试

测试旅行推荐对话，分别测试对聊天机器人进行实体式对话和向导式对话。

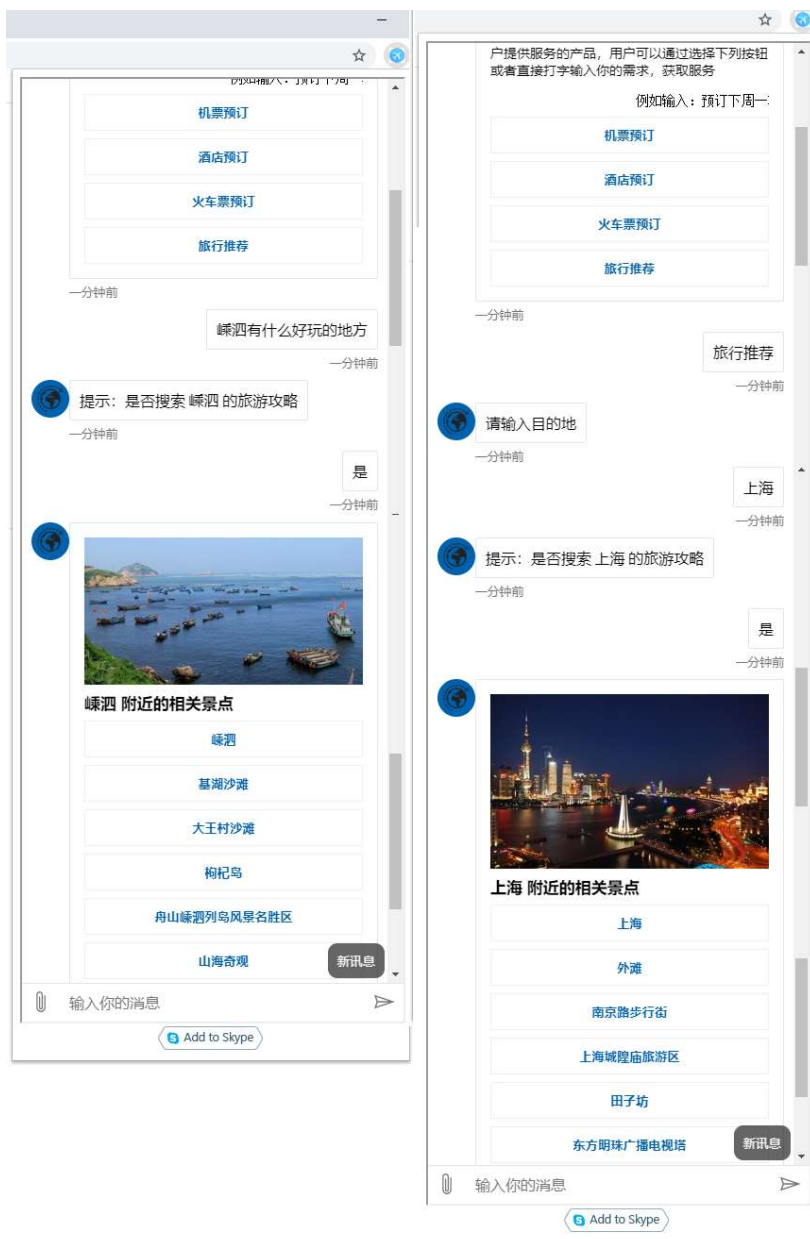


图 6.3 聊天机器人旅行推荐对话测试图

如图 6.3，对含有实体的对话分析，输入“嵊泗有什么好玩的地方”，模型准确分析出旅行推荐意向并进入旅行推荐对话，同时提取出目的地实体“嵊泗”，最终返回嵊泗旅行的景点推荐链接，点击不同链接可跳转至嵊泗对应不同景点介绍及攻略页面。对不含实体的对话分析，用户点击旅行推荐按钮后，按照旅行推荐对话顺序，询问出游目的地后确认信息，最终返回上海旅行的景点推荐链接。点击对应链接可打开上海对应不同景点介绍及旅行攻略页面。经测试，旅行推荐模块对话内容与返回链接符合要求。

6.3.4 铁路预订对话测试

测试铁路服务预订对话，依次对含有实体和不含实体分别进行不同对话测试。



图 6.4 聊天机器人铁路预定对话测试图

如图 6.4，对含有实体的对话分析，输入“帮我预订下周去南京的火车”，模型准确分析出铁路意向并进入铁路对话，提取出目的地实体“南京”和时间实体“下周”，跳过对应询问，最终返回上海至南京 4 月 16 日铁路预订链接。点击不同链接可打开不同网站上海至南京 4 月 16 日火车票预定页面。对不含实体的对话分析，用户点击“火车票预定”按钮后，并依次询问出发地、目的地、时间并确认信息，最终返回 4 月 12 日上海至马鞍山铁路预订链接。所有链接返回无误。经测试，铁路预订模块对话内容与返回链接符合要求。

6.3.5 问题对话测试

测试逻辑中部分退出逻辑，用户输入错误是否能够按照符合设计要求。



图 6.5 聊天机器人实体无法识别及确认状态退出聊天测试图

如图 6.5，测试确认信息退出，当进入信息确认时，选择“否”，程序开始重新询问所有问题，运行符合设计预期。测试信息无法识别，在询问目的地时候，若回答“城市”，这个词不是一个城市名，不会出现在后台数据库记录当中，通过索引无法找到，机器人对相同问题再次询问，信息输入错误程序运行符合预期。

6.4 压力测试

压力测试是测试系统访问上限服务能力，测试在高并发的访问情况下是否会响应错误，不响应或运行错误等情况。

使用 Visual Studio 中 Web 性能和负载测试项目对线上 Bot 服务进行压力测试，在 Visual Studio 中安装 Web 性能与负载测试工具以后创建对应项目。并添加请求，请求 Azure Bot 服务 API 链接，设置 500 个虚拟用户同时不间断向 Azure Bot API 发送请求 5 分钟，进行压力测试。最终运行压力测试项目。

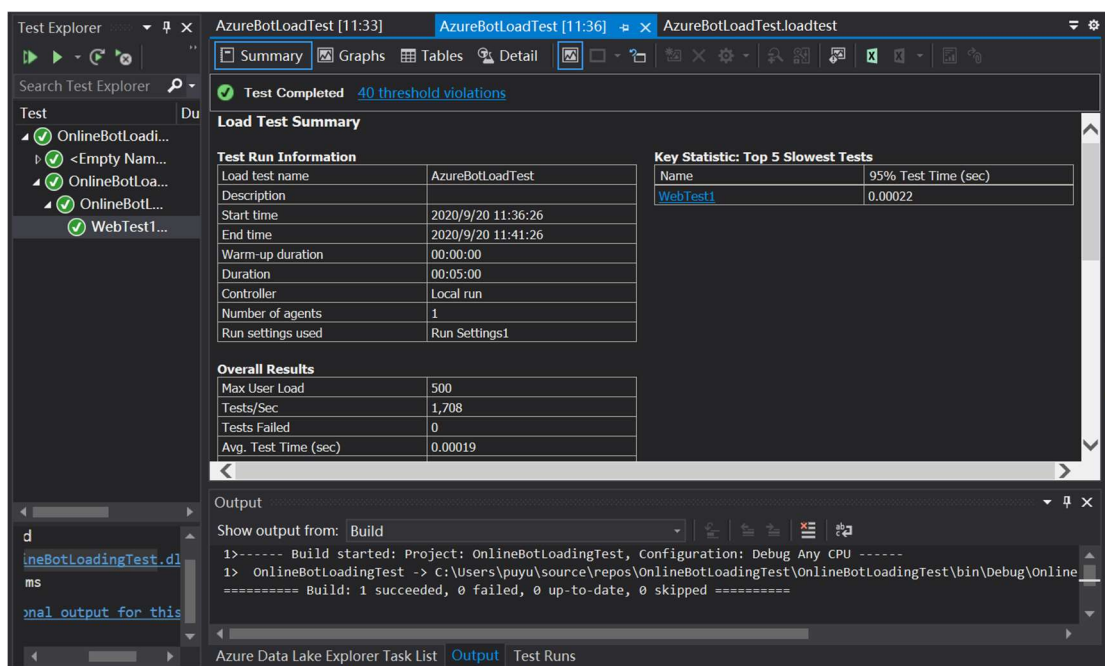


图 6.6 Visual Studio 负载测试结果图

如图 6.6，根据测试可知，500 个用户平均每秒发送 1708 个请求的情况下持续 5 分钟，请求成功率依然保持 100%，即模型支持每秒 1700 次以上的请求，符合需求设计中模型压力要求。

6.5 本章小结

经过以上测试，模型效果意向判断准确，实体获取无误，生成服务满足用户需求，浏览器扩展应用中聊天窗口工作正常、对话流畅，模型整体效果符合预期设计需求。

第七章 总结与展望

7.1 本文工作总结

本文首先分析了用户在互联网旅游方面的信息获取与检索相关服务中存在的问题。当前，主要的互联网旅游服务的预订服务步骤比较繁琐，用户需要经过大量的信息输入后进行搜索，造成过程复杂、交互体验较差。并且缺少旅行信息汇总平台以方便用户查询所需旅行服务，更缺少利用聊天机器人来辅助用户搜索获取全网旅行信息服务的软件。

基于互联网旅游服务检索的问题，本文设计并开发了一种能向用户推荐旅游相关服务与信息的智能聊天机器人软件。它通过与用户之间的“聊天对话”，与用户“提出问题”、“回答问题”，从基于意向判断的一级对话开始，针对用户不同意向进入不同的二级对话，在二级对话中继续进行基于信息获取的深层次对话，在对话中通过分析用户语言提取用户需求。利用获取到的用户需求以及数据库中存储的信息，生成符合用户需求的出行旅游服务或旅游信息，并通过聊天回复的方式反馈给用户。设计浏览器扩展应用，将聊天机器人嵌入到扩展应用中，简化用户交互流程。

本文主要完成了四个方面的模块设计开发工作，即数据库模块、LUIS 语言理解模块、后台聊天逻辑模块和前端扩展应用模块的设计与开发。

在数据库模块实现阶段中，使用爬虫技术，对选取目标网站利用广度优先搜索算法批量化请求网站中网页，通过解析网页，获取其中有用信息，根据设计的数据表创建数据库，将获取到的信息存储在数据库中供后续步骤中使用。

在 LUIS 语言理解模块开发阶段中，主要是开发一款能够理解用户语言的模型。在 LUIS 语言理解平台中，根据需要定义语言意向和实体内容，标注一定训练数据，通过训练得到一个可以判断用户意向、从语言中提取实体信息的语言理解模型，模型发布后可在之后的模型中通过 API 接口进行调用。

后台聊天逻辑模块是整个软件核心模块，开发主要难点在于设计的对话模型要尽可能符合用户习惯，并根据用户答复引导对话。在此利用 Bot Framework 逻辑框架进行开发，实现聊天对话逻辑。首先定义主对话，在用户

加入对话后，向用户提供功能导航信息。用户回复后使用语言理解模块判断用户旅行相关的服务意向要求，根据不同意向进入不同二级对话当中。例如航班预订对话、旅行推荐对话等二级对话。在二级对话中，针对不同服务继续提出相关问题，使用语言理解模型再次对用户回复进行解析。当得到完整的用户需求时，再根据需求从数据库中查找相关所有数据，根据用户数据和数据库数据生成符合用户需求的旅行相关服务与信息。通过聊天方式将结果反馈给用户。最后将聊天机器人发布至 Azure 云服务器上，生成线上聊天机器人。

在前端扩展应用模块开发阶段中，开发在 Chromium 为内核的浏览器中使用的扩展应用。首先设置扩展应用架构，设置为菜单栏式扩展应用。当用户在浏览器中点击扩展应用菜单中的旅游辅助推荐按钮时，可打开一个悬浮在浏览器之上的网页。在网页中，引用 Web Chat 嵌套网页，通过线上聊天机器人信道密钥连接聊天机器人。在 Web Chat 实现的对话框中可以直接与聊天机器人进行对话交流。添加 Azure 聊天机器人的 Skype 信道，方便用户在 Skype 中使用。

聊天对话机器人模式相较于传统旅游互联网网站用户填写输入需求模式的交互方式，简化了用户检索旅行信息与服务的过程，使用户尤其是老年用户更加便捷地搜索到其需要的旅行信息或服务，从而直面化、简化、优化了用户在获取互联网旅游有关信息服务的流程与体验。同时，人工智能的包容性更强，对比人工客服，聊天机器人在聊天过程中不会情绪化，会不停地根据指令帮助用户解决问题，也减少了客服方面人力和财力成本。

7.2 未来工作展望

本文设计实现了通过聊天向用户推荐旅行信息服务的聊天机器人软件。系统在设计实现上有以下三个方面需要在未来进行进一步提升。首先，当前聊天机器人只支持与用户文字交互，不支持语音交互。在未来工作中，可以接入语言模块提升用户体验。其次，当前软件只支持 Web Chat 和 Skype 聊天软件两种接入方式，接入手段较少。未来可考虑开发更多用户接入方式。最后，由爬虫得到的数据，可靠性无法完全保证，需要在未来接入更加可靠的商业数据供用户查询。

参考文献

- [1] 人民网. 文旅部: 2019 年国内旅游旅游收入超 6 万亿元[EB/OL]. 2020-03-11. <http://js.people.com.cn/BIG5/n2/2020/0311/c360311-33866145.html>.
- [2] 成都商报. 这届老年人很潮, 携程报告称 2/3 “老顽童” 每年出游超 3 次[EB/OL]. 2019-10-07. <https://news.sina.cn/2019-10-07/detail-iicezuev0567741.d.html>
- [3] 宋瑞. 旅游绿皮书: 2018-2019 年中国旅游发展分析与预测[M]. 北京: 社会科学文献出版社, 2020
- [4] 王浩畅, 李斌. 聊天机器人系统研究与进展[J]. 计算机应用与软件, 2018, 35(12): 1-6+89.
- [5] 快科技. 阿里巴巴发布首款人工智能新品: 天猫精灵[EB/OL]. 2017-07-05. <http://www.techweb.com.cn/it/2017-07-05/2552255.shtml>
- [6] Mierzwa Stanley, Souidi Samir, Conroy Terry, Abusyed Mohammad, Watarai Hiroki, Allen Taammy. On the Potential, Feasibility, and Effectiveness of Chat Bots in Public Health Research Going Forward[J]. Online journal of public health informatics, 2019, 11(2)
- [7] 王冬旭. 基于 Python 的旅游网站数据爬虫研究[D]. 沈阳理工大学, 2020.
- [8] 黎曦. 基于网络爬虫的论坛数据分析系统的设计与实现[D]. 华中科技大学, 2019
- [9] 刘汝佳. 算法竞赛入门经典 (第二版) [M]. 北京: 清华大学出版社, 2014
- [10] Christian Nagel. C#高级编程 (第 11 版) C# 7 & .Net Core 2.0[M]. 北京: 清华大学出版社, 2019
- [11] Kishore Gaddam. Building Bots with Microsoft Bot Framework[M]. Birmingham: Packt Publishing Limited, 2017
- [12] Charles Waghmare. Introducing Azure Bot Service[M]. Berkeley: Apress, 2017
- [13] 闵昭浩. LUIS 在 Microsoft Bot Framework 中的应用与研究[J]. 电脑迷.

2018(11):12

- [14] Manisha Biswas. Beginning AI Bot Frameworks[M]. Berkeley: Apress, 2018
- [15] 世纪互联蓝云公司. Microsoft Azure 管理与开发[M]. 北京: 电子工业出版社, 2018
- [16] 李喆. Chrome 扩展及应用开发[M]. 北京: 人民邮电出版社, 2014
- [17] Leszek Maciaszek. 需求分析与系统设计[M]. 北京: 机械工业出版社, 2009
- [18] 邓博. 基于特定领域客服机器人的研究与实现[D]. 西安电子科技大学, 2019
- [19] 钱进, 常玉慧, 叶飞跃. 数据库的设计与开发[M]. 北京: 科学出版社, 2020
- [20] 朱鹏臻. 人工智能产品经理: 人机对话系统设计逻辑探究[M]. 电子工业出版社, 2018
- [21] Amir Shevat. 聊天机器人: 对话式体验与产品设计[M]. 北京: 电子工业出版社, 2019
- [22] Szymon Rozga. 实用 Bot 开发指南: 基于 Node.js 与 Bot 框架设计并构建聊天机器人[M]. 北京: 机械工业出版社, 2019
- [23] 刘秋香, 王云, 姜桂洪, 刘树淑. Visual C# .Net 程序设计 (第二版) [M]. 北京: 清华大学出版社, 2017
- [24] Glenford J. Myers, Tom Badgett, Corey Sandler. 软件测试的艺术 (第三版) [M]. 北京: 机械工业出版社, 2012
- [25] 李银城. 高效前端: Web 高效编程与优化实践[M]. 北京: 机械工业出版社, 2018
- [26] 胡静. 浅析黑盒测试与白盒测试[J]. 衡水学院学报, 2008(01):30-32

致谢

三年的研究生阶段的学习将要画上句号，从入学时候对软件工程这门学科的一知半解，到现在能够在彭超导师的认真指导下完成毕业论文，我感觉到了我自己在基础知识的储备和专业实践技能的拓展等方面上有了很大的提升。由于自己基础的限制，我在学习的过程中也遇到了很多困难。但是现如今，面对学习中的困难，我也能够拥有了独立思考并自主解决问题的能力。在此我要首先感谢我的母校，来这里学习是我多年一来的梦想。如今这也将成为我生命中的宝贵财富。在我三年研究生的学习过程中，我的专业学识与实践能力的飞跃，能够将课本上软件工程相关知识应用到实际开发工作项目当中，这些能力的提升是在学院老师们的共同指导关心下才能够完成的成绩。在此我对学院的辛勤奉献的老师们先说一句诚挚的感谢。尤其感谢我的导师彭超在三年中对我的指导。

彭超导师学识渊博，治学严谨，与人和善，他的耐心教导在学习与生活中无时无刻激励着我好好学习，专注思考。在此我对我的导师说一声真诚的感谢，感谢您三年来对我的付出。

在此，我对我的导师，对学院的老师，对我的母校由衷的表示感谢，感谢他们三年来对我的关心，对我的爱，对我的教导，对我的启发，让我的三年研究生生活过的如此充实，自信。最终，在此对你们表达我最真诚的感谢。