# 软件需求规格说明



|  |  |
| --- | --- |
| **软件名称：** | **智能对联生成系统** |
| **学院：** | **计算机科学与技术学院** |
| **作者：** | **智能对联生成系统开发小组** |
| **指导教师：** | **张琛** |
| **日期：** | **2020.04.06** |
| **版本：** | **1.1** |

## 小组成员

|  |  |  |
| --- | --- | --- |
| 姓名 | 学号 | 分工 |
| 张笑天 | 17030140014 | 组长；后端；机器学习模型 |
| 张震宇 | 17030140013 | 前端 |
| 王重阳 | 17030140020 | 机器学习模型 |
| 王艺静 | 17030140069 | 后端 |

目录

[软件需求规格说明 1](#_Toc43319771)

[小组成员 2](#_Toc43319772)

[1. 范围 4](#_Toc43319773)

[a) 标识 4](#_Toc43319774)

[b) 系统概述 4](#_Toc43319775)

[c) 文档概述 4](#_Toc43319776)

[d) 基线 4](#_Toc43319777)

[2. 引用文件 4](#_Toc43319778)

[3. 需求 4](#_Toc43319779)

[a) 所需的状态和方式 4](#_Toc43319780)

[b) 概述需求 4](#_Toc43319781)

[c) 需求规格 5](#_Toc43319782)

[d) CSCI能力需求 8](#_Toc43319783)

[e) CSCI外部接口需求 9](#_Toc43319784)

[f) CSCI内部接口需求 11](#_Toc43319785)

[g) CSCI内部数据需求 11](#_Toc43319786)

[h) 适应性需求 11](#_Toc43319787)

[i) 保密性需求 11](#_Toc43319788)

[j) 保密性和私密性需求 11](#_Toc43319789)

[k) CSCI环境需求 11](#_Toc43319790)

[l) 计算机资源需求 11](#_Toc43319791)

[m) 软件质量因素 12](#_Toc43319792)

[n) 设计和实现的约束 12](#_Toc43319793)

[o) 数据 12](#_Toc43319794)

[p) 操作 13](#_Toc43319795)

[q) 故障处理 13](#_Toc43319796)

[r) 算法说明 13](#_Toc43319797)

[s) 有关人员需求 13](#_Toc43319798)

[t) 有关培训的需求 13](#_Toc43319799)

[u) 有关后勤需求 13](#_Toc43319800)

[v) 其他需求 13](#_Toc43319801)

[w) 包装需求 13](#_Toc43319802)

[x) 需求的优先次序和关键程度 13](#_Toc43319803)

[4. 合格性规定 13](#_Toc43319804)

[5. 需求可追踪性 14](#_Toc43319805)

[6. 尚未解决的问题 14](#_Toc43319806)

[7. 注解 14](#_Toc43319807)

**正文**

## 范围

### 标识

软件标题：智能对联生成系统

版本号：v1.0

### 系统概述

使用项目：智能对联生成系统

软件用途：通过网页端可以获取到根据已有上联只能生成的下联。

开发历史：本项目未曾有前置版本。但在服务器搭建，Tensorflow使用上已有经验。

投资方：开发小组自费

需方：西安电子科技大学计算机科学与技术学院软件工程课程

用户：网页使用者

开发方：开发小组，成员：张笑天，王重阳，王艺静，张震宇

支持机构：西安电子科技大学计算机科学与技术学院软件工程

当前运行现场：虚拟机VMWare中Ubuntu 19.10，Windows 10平台Anaconda

计划运行现场：阿里云 Ubuntu 18.04

### 文档概述

本文档为项目 智能对联生成系统 的软件需求规格说明，用于描述对计算机软件配置项CSCI的需求，及确保每个要求得以满足的所使用的方法。

保密性：该文档可以公开于网络，但应注意项目以及该文档本身均遵循GPLv3协议开源。

<http://www.gnu.org/licenses/quick-guide-gplv3.html>

### 基线

无

## 引用文件

GPLv3协议：

<http://www.gnu.org/licenses/quick-guide-gplv3.html>

## 需求

### 所需的状态和方式

默认状态为活动

### 概述需求

#### 需求获取：通过比较类似的娱乐人工智能产品，例如换脸、人工智能写诗等获取用户需求。

#### 目标

1. 产品开发可以使用户可以在网页上方便地输入自自定义的上联得到下联，起到丰富精神生活作用。只需用户懂得基础的网页操作便可使用。
2. 用户进行数据的输入，经后端处理，网页展现相应的数据输出。
3. 系统与其他相关产品无相关关系，是独立产品。

#### 运行环境：

实际运行硬件使用阿里云的云服务器ECS。

实例：共享计算型 n4系列 III1核 2GB

I/O 优化实例：I/O 优化实例

系统盘：高效云盘/dev/xvda40GB

带宽：1Mbps按固定带宽

CPU：1核

可用区：华东 2 可用区 B

操作系统：Linux64位Ubuntu 18.04 64位

内存：2GB

地域：华东 2

网络类型：专有网络

生产环境软件包括

系统：Ubuntu 18.04

解释器：Python 3.7

环境：Anaconda

机器学习框架：Tensorflow 1.14

代理服务器：Nginx

Web服务器：uWsgi

Web应用框架：Django

#### 用户的特点：

经过一定的指导，懂得合法操作网页的用户，但用户仍有可能误非法操作。

#### 关键点：

数据输入的有效性检测。

使用深度学习框架进行数据的处理，然后经网页将响应的下联展示给用户。

机器学习模型的训练。

后台大量数据的管理。

#### 约束条件：

GPU性能不足。

受制于服务器性能，生成可能较慢。

没有上联的生成系统。

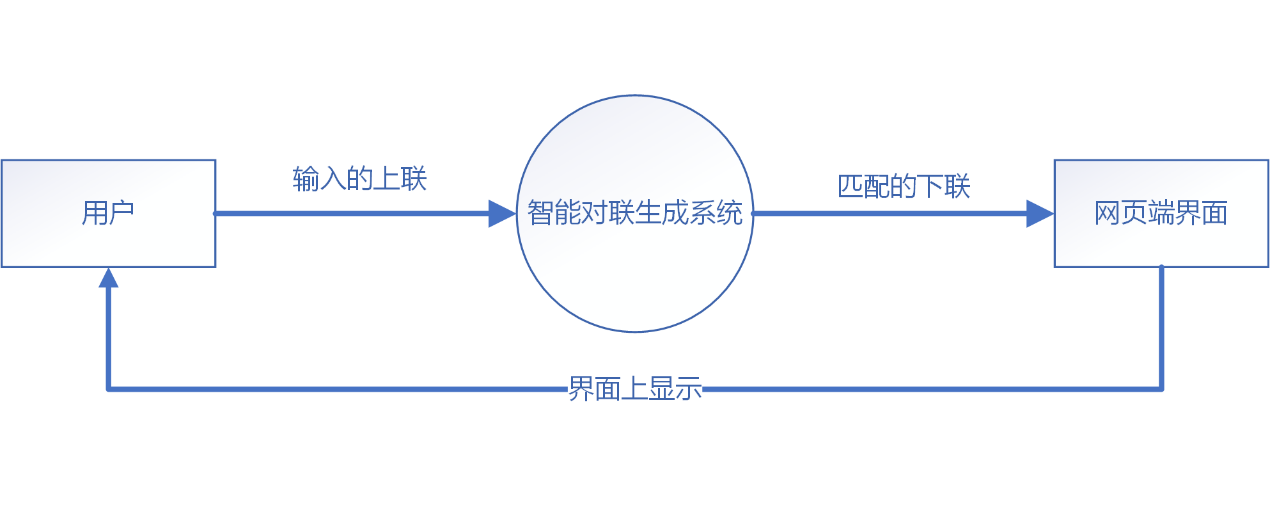
项目开发周期较短。

开发人员能力有限。

系统的功能较为单一，页面较为简单。

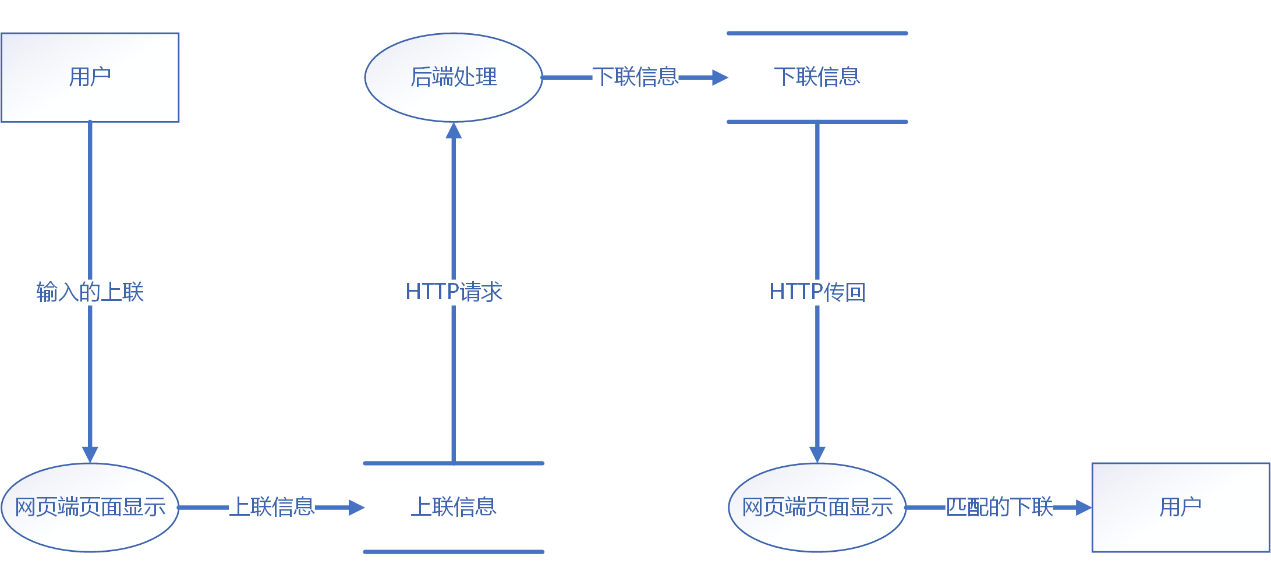
### 需求规格

#### 软件系统总体功能/对象结构：

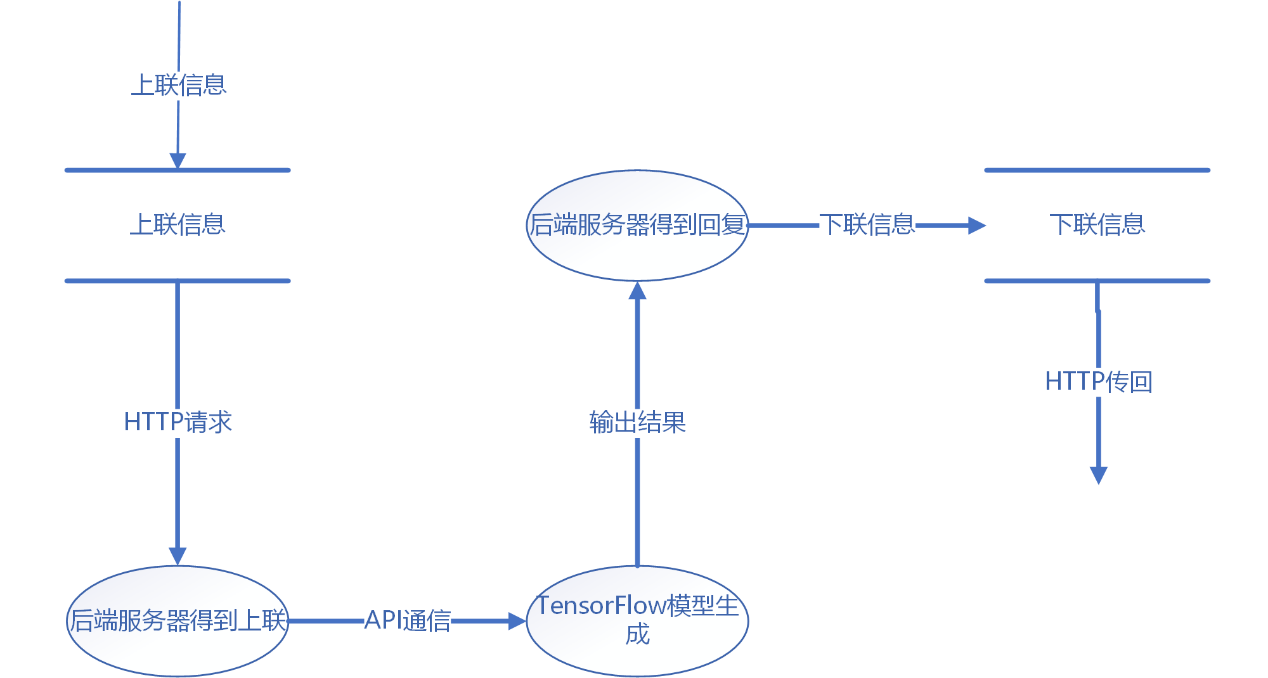


#### 软件子系统功能/对象结构：

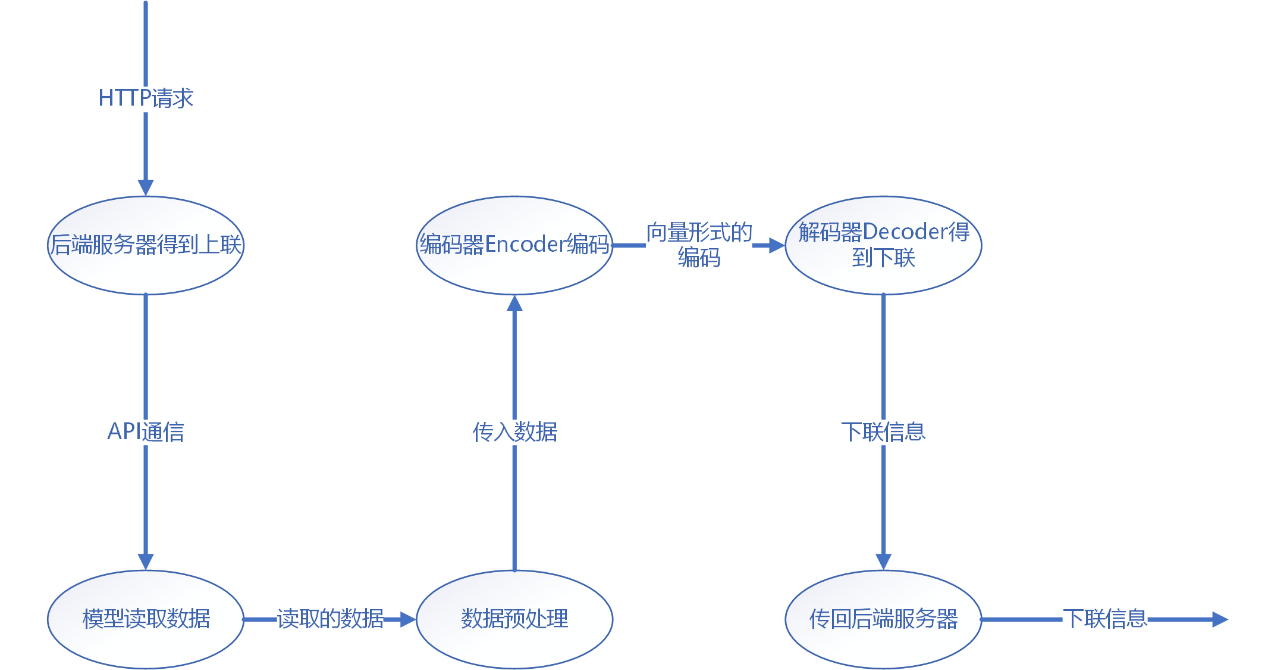
网络通信部分：



后端部分：



机器学习模型部分：



数据字典：

用户输入的上联

|  |  |
| --- | --- |
| 名字 | 用户输入的上联 |
| 分类 | 数据元素 |
| 描述 | 在前端由用户输入的上联 |
| 定义 | 用户输入的上联={utf-8汉字}7 |
| 位置 | 用户->前端 |

用户得到的下联

|  |  |
| --- | --- |
| 名字 | 用户得到的下联 |
| 分类 | 数据元素 |
| 描述 | 在前端由显示给用户的下联 |
| 定义 | 用户得到的下联={utf-8汉字}7 |
| 位置 | 前端->用户 |

前端传送到后端的上联

|  |  |
| --- | --- |
| 名字 | 前端传送到后端的上联 |
| 分类 | 数据流 |
| 描述 | 前端通过HTTP传送到后端的上联 |
| 定义 | 前端传送到后端的上联={HTTP规定} |
| 位置 | 前端->后端 |

后端传送到前端的下联

|  |  |
| --- | --- |
| 名字 | 后端传送到前端的下联 |
| 分类 | 数据流 |
| 描述 | 后端通过HTTP返回到前端的下联 |
| 定义 | 后端传送到前端的下联={HTTP规定} |
| 位置 | 后端->前端 |

后端传送到机器学习模型的上联

|  |  |
| --- | --- |
| 名字 | 后端传送到机器学习模型的上联 |
| 分类 | 数据流 |
| 描述 | 后端通过API将上联给机器学习模型 |
| 定义 | 后端传送到机器学习模型的上联={API规定} |
| 位置 | 后端服务器->机器学习模型 |

机器学习模型传送到后端的下联

|  |  |
| --- | --- |
| 名字 | 机器学习模型传送到后端的下联 |
| 分类 | 数据流 |
| 描述 | 机器学习模型通过API传送给后端的下联 |
| 定义 | 机器学习模型传送到后端的下联={API规定} |
| 位置 | 机器学习模型->后端服务器 |

语义编码向量

|  |  |
| --- | --- |
| 名字 | 语义编码向量 |
| 分类 | 加工 |
| 描述 | 从Encoder生成到Decoder被解码 |
| 定义 | 语义编码向量={Encoder规定} |
| 位置 | Encoder->Decoder |

#### 描述约定：

数据约定为简体汉字，编码为utf-8。

### CSCI能力需求

#### CSCI能力：

1. 说明

目标：使得用户可以在网页上方便地输入自自定义的上联得到下联，起到丰富精神生活作用。

所采用技术和方法：

前端：HTML5+CSS+JavaScript

后端：Nginx+uWsgi+Django

深度学习框架：

Python3+Ananconda+Tensorflow+Tensor2Tensor，使用序列到序列模型

使用较为流行的Tensorflow作为学习框架，使用T2T深度学习模型库

功能意图：在学习和生活中，此系统可帮助学生进行文化学习，也可丰富精神生活。

1. 输入

简体汉字，编码为utf-8。

输入处理参见前端文档。

1. 处理

输入数据是否为简体汉字，编码是否为utf-8。

输入有效即可进行下联的匹配。

若输入错误，清空文本框重新输入。

汉字的数量不可超过规定值。

由后台进行处理上联，下联展现在网页端。

在后端检查输出数据是否符合语言和数据长度的规定。

1. 输出

输出在网页上，可以展现匹配对联给用户，若无法匹配，则显示匹配失败。提示用户重新输入。

参考资料详见后端设计文档。

### CSCI外部接口需求

#### **用户接口：用户输入输出汉字接口**

#### 优先级别

高，必须分配，确保随时可用。

#### 要实现的接口的类型的需求

数据的输入验证

#### CSCI必须提供、存储、发送、访间、接收的单个数据元素的特性

1. 名称/标识符

项目唯一标识符：interfaceAlphaUserInput

非技术名称：用户输入汉字接口

标准数据元素名称：字符串

技术名称：userInput （未来可能更改）

缩写名：用户输入

1. 数据类型

字符串，uft-8编码

1. 大小和格式

长度不超过7个字即7个code point

1. 计量单位

code point

1. 范围或可能的值

0-正无穷

1. 精度

1

1. 其他约束条件

输入时可以修改，提交后不可修改

1. 保密性和私密性的约束

只有该用户、前端和后端可见，其他用户不可见

1. 来源和接收者

来源为用户，接收者为前端程序

#### CSCI必须提供、存储、发送、访问、接收的数据元素集合体的特性

因为用户的输入数据设计上为单个输入，因此集合体与单个数据等价，因此同上。

#### CSCI必须为接口使用通信方法的特性

#### 项目唯一标识符：interfaceAlphaUserInput

其它通信特性由浏览器和前端解释器控制。

#### CSCI必须为接口使用协议的特性

通信协议由浏览器和前端解释器控制

#### 其他所需的特性

显示设备：PC显示屏或者移动端显示设备

#### **用户接口：输出汉字接口**

#### 优先级别

高，必须分配，确保随时可用。

#### 要实现的接口的类型的需求

数据的输出显示

#### CSCI必须提供、存储、发送、访间、接收的单个数据元素的特性

1. 名称/标识符

项目唯一标识符：interfaceBetaUserOutput

非技术名称：输出汉字接口

标准数据元素名称：字符串

技术名称：userOutput （未来可能更改）

缩写名：输出汉字

1. 数据类型

字符串，uft-8编码

1. 大小和格式

长度不超过7个字即7个code point

1. 计量单位

code point

1. 范围或可能的值

0-正无穷

1. 精度

1

1. 其他约束条件

不可修改

1. 保密性和私密性的约束

只有该用户、前端和后端可见，其他用户不可见

1. 来源和接收者

来源为前端程序，接收者为用户显示设备

#### CSCI必须提供、存储、发送、访问、接收的数据元素集合体的特性

因为用户的输入数据设计上为单个输出，因此集合体与单个数据等价，因此同上。

#### CSCI必须为接口使用通信方法的特性

#### 项目唯一标识符：interfaceBetaOutputInput

其它通信特性由浏览器和前端解释器控制。

#### CSCI必须为接口使用协议的特性

通信协议由浏览器和前端解释器控制

#### 其他所需的特性

显示设备：PC显示屏或者移动端显示设备

### CSCI内部接口需求

内部接口待设计时决定

内部接口包括：

前端与后端的通信接口

后端服务器与机器学习模型间的接口

### CSCI内部数据需求

用户的输入数据

用于机器学习的对联数据，该数据的预处理方法待定

机器学习模型本身的序列化数据

### 适应性需求

无

### 保密性需求

将用户所查询的上联进行保密管理，不可其信息随意提供给其他用户查看。

### 保密性和私密性需求

将后台数据进行管理，不同角色有不同的操作权限，只有后台管理员才有查看用户所有数据的权限。后台数据进行密码登录，不同用户以不同身份登录，不存在普通用户查看其它用户的记录。

### CSCI环境需求

实际运行硬件使用阿里云的云服务器ECS。

实例：共享计算型 n4系列 III1核 2GB

I/O 优化实例：I/O 优化实例

系统盘：高效云盘/dev/xvda40GB

带宽：1Mbps按固定带宽

CPU：1核

可用区：华东 2 可用区 B

操作系统：Linux64位Ubuntu 18.04 64位

内存：2GB

地域：华东 2

网络类型：专有网络

生产环境软件包括

系统：Ubuntu 18.04

解释器：Python 3.7

环境：Anaconda

机器学习框架：Tensorflow 1.14

代理服务器：Nginx

Web服务器：uWsgi

Web应用框架：Django

### 计算机资源需求

#### 计算机硬件需求

实际运行硬件使用阿里云的云服务器ECS。

实例：共享计算型 n4系列 III1核 2GB

I/O 优化实例：I/O 优化实例

系统盘：高效云盘/dev/xvda40GB

带宽：1Mbps按固定带宽

CPU：1核

可用区：华东 2 可用区 B

操作系统：Linux64位Ubuntu 18.04 64位

内存：2GB

地域：华东 2

网络类型：专有网络

#### 计算机硬件资源利用需求

带宽：1Mbps按固定带宽

CPU：1核

可用区：华东 2 可用区 B

内存：2GB

#### 计算机软件需求

系统：Ubuntu 18.04

解释器：Python 3.7

环境：Anaconda

机器学习框架：Tensorflow 1.14

代理服务器：Nginx

Web服务器：uWsgi

Web应用框架：Django

#### 计算机通信需求

传输技术：HTTPS

传输速率：1Mb/s

数据的类型与容量：HTTPS设置

数据诊断功能：对于数据的使用和服务器流量有监控与日志功能

### 软件质量因素

可靠性：软件能够出错时尽量做到异常处理，包括在前端的输入检查，在后端防止服务器报错停止。；

可维护性：文档齐全，易于维护；

可重用性：可以被多个应用，多个设备使用；

可测试性：易于充分测试；

易用性：易于学习和使用；

可拓展性：

功能扩展，例如古诗生成，图片的情景联想；

纵向拓展，例如移动端APP，桌面端GUI应用，微信小程序等；

### 设计和实现的约束

易于扩展，方便日后添加功能；

### 数据

系统输入：用户自身的上联

系统输出：系统通过NLP模型得到的下联

对于数据有一定缓冲功能，防止大量数据将系统崩溃；

处理速度尽量快；

### 操作

常规操作：正常执行

特殊操作：系统可以甄别用户输入是否合法

初始化，管理员可以快速重新部署软件系统

### 故障处理

软件系统问题：可重用系统，之前移动互联网导论课程中搭建的Django后端平台

发生错误时，例如输入不合法时可以恰当修改，尽可能保护用户界面不会因为异常关闭

发生错误时可能采取的补救措施：重新输入合法数据进行下一次匹配

发送后端系统崩溃故障时，管理员可以快速重新部署软件系统

### 算法说明

Sequence to Sequence模型

https://papers.nips.cc/paper/5346-sequence-to-sequence-learning-with-neural-networks.pdf

### 有关人员需求

人员数量：小组成员4人；

技能等级：计科大数据方向在读本科生；

责任期：直至本次软件工程大作业考核结束；

培训需求：4人平时查阅网上资料互相交流；

对用户数量无需求，用户只需熟悉简单的网页操作即可；

### 有关培训的需求

无

### 有关后勤需求

ECS的续费；

后续功能的添加；

对于后端状态的检测；

机器学习模型的改进；

### 其他需求

无

### 包装需求

初定Git以及GitHub交付

### 需求的优先次序和关键程度

从上到下排序：

机器学习模型；后端；简单的前端交互；

进阶的前端交互与UI；后端数据库；

## 合格性规定

演示：使用合法输入输出得到期望结果；

测试：使用非法输入输出；对于代码进行单元测试；

分析：根据合法输入输出结果进行改进；

审查：利用代码审查工具审查代码错误与规范；通过小组成员互相审查确保代码逻辑正确；

## 需求可追踪性

所有需求均可跟踪；

接口需求在设计时会被考虑与跟踪；

## 尚未解决的问题

该系统未来的扩展方向；

内部接口的设计方式；

## 注解

NLP：自然语言处理

云服务器（ECS）: 一种简单高效、安全可靠、处理能力可弹性伸缩的计算服务

CPU：中央处理器，计算机系统的运算和控制核心，是信息处理、程序运行的最终执行单元

GPU：图形处理器，一种专门在个人电脑、工作站、游戏机和一些移动设备上做图像和图形相关运算工作的微处理器。

X86：一系列基于Intel 8086且向后兼容的中央处理器指令集架构

Linux：一套免费使用和自由传播的类UNIX操作系统。

Ubuntu：一个以桌面应用为主的Linux操作系统。

Windows：美国微软公司研发的一套操作系统

Python：一种跨平台的计算机程序设计语言

VMware Workstation：一款功能强大的桌面虚拟计算机软件，提供用户可在单一的桌面上同时运行不同的操作系统

Anaconda：开源的Python包管理器

Tensorflow：TensorFlow™是一个基于数据流编程的符号数学系统

PyTorch：是一个开源的Python机器学习库

GPL：GNU General Public License，GNU通用公共许可证

IDE：集成开发环境，Integrated Development Environment是用于提供程序开发环境的应用程序，一般包括代码编辑器、编译器、调试器和图形用户界面等工具

PyCharm：一种Python集成开发工具

IDLE：Integrated Development and Learning Environment，集成开发和学习环境，是Python的集成开发环境

Visual Studio：美国微软公司的开发工具包系列产品

Visual Studio Code：跨平台源代码编辑器

Vim：功能强大、高度可定制的文本编辑器

Google Chrome：是一款由Google公司开发的网页浏览器

Firefox：是一个自由及开放源代码的网页浏览器

Microsoft Edge：是一款由Microsoft公司开发的网页浏览器

Git：一个开源的分布式版本控制系统

GitHub：一个面向开源及私有软件项目的托管平台

Nginx：一个高性能的HTTP和反向代理web服务器

uWsgi：一个Web服务器与web应用通信的规范实现

Django：一个开放源代码的Web应用框架

Flask：一个使用 Python 编写的轻量级 Web 应用框架

HTML5：超文本标记语言5，构建Web内容的一种语言描述方式

CSS：层叠样式表，Cascading Style Sheets是一种用来表现HTML或XML等文件样式的计算机语言

JavaScript：是一种具有函数优先的轻量级，解释型或即时编译型的编程语言

Tensor2Tensor：一套基于TensorFlow的深度学习系统