# 软件设计说明



|  |  |
| --- | --- |
| **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：** | **1.0** |

## 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

[软件设计说明 1](#_Toc43215301)

[小组成员 2](#_Toc43215302)

[1. 引言 3](#_Toc43215303)

[a) 标识 3](#_Toc43215304)

[b) 系统概述 3](#_Toc43215305)

[c) 文档概述 3](#_Toc43215306)

[d) 基线 3](#_Toc43215307)

[2. 引用文件 3](#_Toc43215308)

[3. CSCI级设计决策 3](#_Toc43215309)

[a) 关于CSCI应接受的输入和产生的输出的设计决策 3](#_Toc43215310)

[b) 有关响应每个输入或条件的CSCI行为的设计决策 3](#_Toc43215311)

[c) 为满足安全性、保密性、私密性需求而选择的方法 3](#_Toc43215312)

[d) 对应需求所做的其他CSCI级设计决策 3](#_Toc43215313)

[4. CSCI体系结构设计 4](#_Toc43215314)

[a) 体系结构 4](#_Toc43215315)

[b) 全局数据结构说明 4](#_Toc43215316)

[c) CSCI部件 5](#_Toc43215317)

[d) 执行概念 5](#_Toc43215318)

[e) 接口设计 6](#_Toc43215319)

[5. CSCI详细设计 8](#_Toc43215320)

[6. 需求的可追踪性 9](#_Toc43215321)

[7. 注解 9](#_Toc43215322)

**正文**

## 引言

### 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 的软件设计说明，用于描述对计算机软件配置项CSCI的设计，它描述了CSCI级设计决策、CSCI体系结构设计(概要设计)和实现该软件所需的详细设计。

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>

## CSCI级设计决策

### 关于CSCI应接受的输入和产生的输出的设计决策

CSCI应该接受上联信息，返回下联信息或者必要的提示。

### 有关响应每个输入或条件的CSCI行为的设计决策

执行动作：对于输入上联显示对应下联

响应时间：10秒以内

所选择的算法：Transform算法

对不允许的输入或条件的说明：返回错误提示并提醒用户正确输入的格式

### 为满足安全性、保密性、私密性需求而选择的方法

本文档根据GPLv3开源，保密性由开源协议保护。

安全性与私密性由Nginx反向代理和Django以及服务器提供商即阿里云的安全措施保护。

### 对应需求所做的其他CSCI级设计决策

灵活性由前后端分离和Django框架设计分离性保证。

可用性、可维护性由Django框架设计保证。

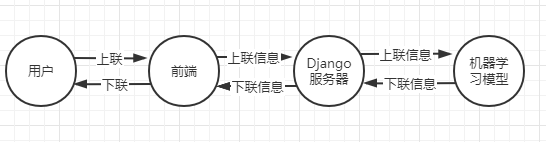
## CSCI体系结构设计

### 体系结构

#### 程序划分

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 标识名 | 功能 | 源可执行文件 |
| Django服务器 | Django服务器 | 提供网页前后端服务 | /wx\_food/manage.py |
| 机器学习模型 | LittleServer | 为Django服务器提供对联推理的API | /MyNLP/LittleServer.py |

#### 程序层次结构关系



### 全局数据结构说明

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 |

### CSCI部件

#### 机器学习模型训练配置

标识：inference.sh

静态关系：机器学习模型调用该配置

用途：

TRAIN\_DIR：训练得到的模型

DATA\_DIR：推理所用的数据

USR\_DIR：模型问题所在的目录

DECODE\_FILE：推理结果所在目录

PROBLEM：问题名称

MODEL：模型名称

HPARAMS：选用的参数集

BEAM\_SIZE、ALPHA：推理参数

开发状态：

新开发的配置项

计划使用的硬件资源：

服务器CPU的一个核心

典型用法：LittleServer.py利用Popen调用bash命令运行该文件得到下联。

该配置项放置在机器学习模型中。

#### Django的配置项

本配置项实际为Django框架的配置项，目的是为了配置前后端。

请查阅：<https://docs.djangoproject.com/zh-hans/3.0/>

### 执行概念

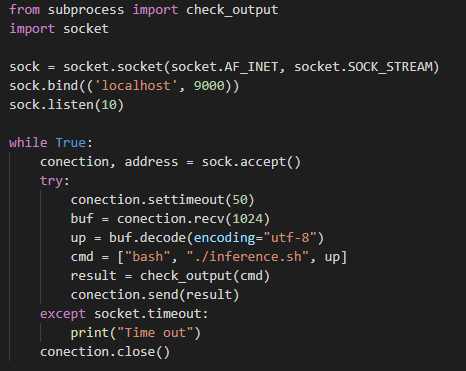
前后端交互：使用HTTP协议的GET。

Django服务器与机器学习模型交互：使用socket模型。

具体实现：

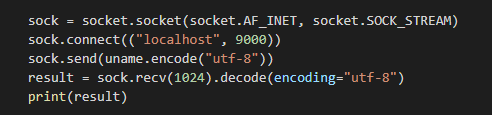
机器学习模型端：

使用socket监听并接收上联，随后向来源返回下联。



Django服务器端：

向机器学习模型端请求socket连接并传送上联，得到下联。



### 接口设计

#### **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

**内部接口：机器学习模型通信接口**

#### 优先级别

高

#### 接口类型

实时数据传输

#### 数据元素

1. 名称/标识符

标识符：LittleServerSocket

名称：机器学习模型接口

数据元素名称：byte[]

1. 数据类型

utf-8汉字编码成的byte[]

1. 大小与格式

汉字，utf-8编码，长度小于等于7个

1. 计量单位

个

1. 范围

汉字在utf-8编码中位置

1. 保密性及私密性约束

本数据仅Django服务器和机器学习模型可见

1. 来源：Django服务器 接收：机器学习模型

#### 通信方法：TCP通信

#### 通信协议：socket

## CSCI详细设计

#### Django服务器

1. 语言：Python3，该Web框架使用Python3
2. 接收输出数据请参见接口设计。
3. Django逻辑请参考<https://docs.djangoproject.com/zh-hans/3.0/>

本软件逻辑主要在/couplet/views中，通过判断前端数据，提供返回对联操作。

1. 异常与错误追踪由Django框架负责。

## 需求的可追踪性

可以根据《软件需求说明书》对应在本文档中找到需求。

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。

该目标已经实现。

运行环境已经在部署在阿里云后实现。

需求规格的结构已经实现，可以参考CSCI的结构。

能力需求、接口需求实现可以参考接口文档。

其他需求将在后续维护中保持。

## 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