|  |  |  |  |
| --- | --- | --- | --- |
|  | | |  |
|  | | | |
|  | | | |
| 计算机编程手册 | | | |
|  | | | |
|  | | | |
| R/ | | | |
| V1.00.01/2021年12月 11日 | | | |
| 共 册 | 第 册 | 共 页 | |
|  | | | |
|  | | | |
| 技术研究所 | | | |
| 2021年12月 | | | |

版 本 说 明

|  |  |  |  |
| --- | --- | --- | --- |
| 版本号 | 简要说明 | 日期 | 更改人 |
| A1.00.00 | 首次建立 | 20211120 |  |
| A1.00.01 | QA审查后修改 | 20211126 |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

目 次

[1 范围 1](#_Toc108534400)

[1.1 标识 1](#_Toc108534401)

[1.2 系统概述 1](#_Toc108534402)

[1.3 文档概述 1](#_Toc108534403)

[2 引用文档 1](#_Toc108534404)

[3软件编程环境 1](#_Toc108534405)

[3.1系统配置 1](#_Toc108534406)

[3.2操作信息 1](#_Toc108534407)

[3.3编译、汇编和连接 2](#_Toc108534408)

[4编程信息 2](#_Toc108534409)

[4.1编程特征 2](#_Toc108534410)

[4.2程序指令 2](#_Toc108534411)

[4.3输入和输出控制 2](#_Toc108534412)

[4.4其他编程技术 2](#_Toc108534413)

[4.5编程示例 2](#_Toc108534414)

[4.6错误检测和诊断特征 2](#_Toc108534415)

[5注释 2](#_Toc108534416)

# 1 范围

# 1.1 标识

a）软件名称：

# 1.2 系统概述

# 1.3 文档概述

本文档描述了。

# 2 引用文档

本文档的引用文档参见表 1。

表 1引用文档

| 文档编号 | 标题 | 编写单位 |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

# 3软件编程环境

## 3.1系统配置

系统配置见表 2。

表 2系统配置清单

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 版本 | 说明 | 备注 |
| KylinOS | V10 | 操作系统 |  |
| PaddlePaddle | 2.0、2.1、2.2. | 深度学习框架 |  |
| 登临加速卡 | V1-L-256-32G | 加速卡 |  |
| 昆仑加速卡 | k200 | 加速卡 |  |
| Cmake | v3.23.1 | 编译软件 |  |
| Windows | 7 | 操作系统 |  |
| JDK | 1.8 | Java软件开发工具包 |  |
| Tomcat | 8 | Web应用服务器 |  |
| Mysql | 5.7 | 关系型数据库 |  |
| 达梦 | 7.6 | 关系型数据库 |  |
| IntelliJ IDEA | Community（社区版） | Java集成开发环境 |  |
| VS Code | 1.7 | 可视化前端开发工具 |  |

## 3.2操作信息

### 3.2.1多源异构军用数据处理通用算法与工具

软件操作方式：打开浏览器，输入软件访问地址，选择相应功能，输入参数：输入文本，请求关键词个数等，点击分析按钮。返回分析结果。

1. 错误处理方式：
2. 访问不存在的网址，返回页面不存在
3. 输入输出错误，页面提示错误信息

## 3.3编译、汇编和连接

|  |  |  |
| --- | --- | --- |
| 名称 | 版本号标识 | 描述 |
| 编译器gcc | 8.2 | 编译依赖库 |
| boost | 1.6.3 | 编译和打包后端代码 |
| opencv | 3.4.5.2 | 编译和打包后端代码 |
| nodejs | 12.22 | 编译可视化前端代码 |
| maven | 3.3.3 | 编译和打包java代码 |
| jenkins | 1.84 | 持续集成工具 |
| python | 3.7 | 算法集成代码python语言 |

# 4编程信息

## 4.1编程特征

### 4.1.1多源异构军用数据处理通用算法与工具

软件结构体系组成部分包括：

1、处理构件：负责对[数据](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE/5947370" \t "/Users/v_wenruisong/Documents\\x/_blank)进行加工。

2、数据构件：存储被加工的信息。

3、连接构件：把体系结构的不同部分组合连接起来。

4、算法构件：加工数据，提取数据中的关键信息。

### 4.1.2

软件结构体系组成部分包括：

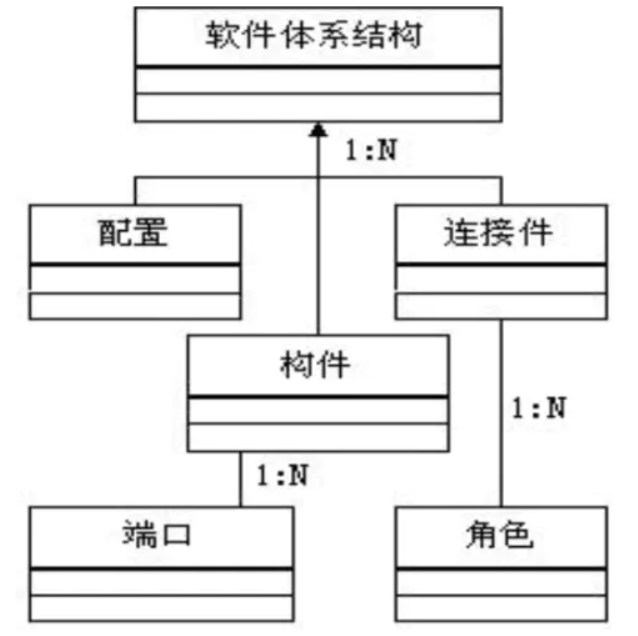
1、构件（Component）：一组代码，或程序块或一个独立的程序。（如SQL服务器）

2、连接件（Connector）：关系的抽象，用以表示构件之间的相互作用。（如过程调用、管道等）

3、配置（Configuration）：用于对构件与连接件的语义说明。

4、端口：表示构件与外部环境的交互点。

5、角色：连接件的接口。



## 4.2程序指令

### 4.2.1多源异构军用数据处理通用算法与工具

1. 程序部署和启动指令：

docker run -itd  -v /work/NLP/25001\_ner\_bert\_crf:/root/NLP/25001\_ner\_bert\_crf --name test1 --net host --restart=always bq:1 /bin/bash /root/NLP/25001\_ner\_bert\_crf/run.sh cpu  
docker run -itd -v /work/NLP/mrc25002:/root/NLP/mrc25002 --name test2 --net host --restart=always --privileged=True bq:2 /bin/bash /root/NLP/mrc25002/run.sh xpu  
docker run -itd -v /work/NLP/gen25003:/root/NLP/gen25003 --name test3 --net host --restart=always --privileged=True bq:3 /bin/bash /root/NLP/gen25003/run.sh xpu  
docker run -itd -v /work/NLP/seg25004\_bert:/root/NLP/seg25004\_bert --name test4 --net host --restart=always --privileged=True bq:4 /bin/bash /root/NLP/seg25004\_bert/run.sh xpu  
docker run -itd -v /work/NLP/key25005:/root/NLP/key25005 --name test5 --net host --restart=always --privileged=True bq:5 /bin/bash /root/NLP/key25005/run.sh xpu  
docker run -itd -v /work/NLP/sum25006:/root/NLP/sum25006 --name test6 --net host --restart=always --privileged=True bq:6 /bin/bash /root/NLP/sum25006/run.sh xpu  
docker run -itd -v /work/NLP/com25007:/root/NLP/com25007 --name test7 --net host --restart=always --privileged=True bq:7 /bin/bash /root/NLP/com25007/run.sh xpu   
docker run -itd -v /work/NLP/mrc25002:/root/NLP/mrc25002 --name test9 --net host --restart=always --privileged=True bq:9 /bin/bash /root/NLP/mrc25002/run\_odqa.sh xpu  
docker run -itd -v /work/NLP/25011vec:/root/NLP/25011vec --name test11 --net host --restart=always bq:11 /bin/bash /root/NLP/25011vec/run.sh   
docker run -itd -v /work/NLP/info\_gen25020:/root/NLP/info\_gen25020 --name test20 --net host --restart=always --privileged=True bq:20 /bin/bash /root/NLP/info\_gen25020/run.sh xpu

1. 指令语法：

启动服务

docker start {容器名}

停止服务

docker stop{容器名}

重启服务

docker restart{容器名}

查看状态

docker ps

### 4.2.2

a)程序部署和启动指令：

docker run -itd  -v /work/NLP/25001\_ner\_bert\_crf:/root/NLP/25001\_ner\_bert\_crf --name test1 --net host --restart=always bq:1 /bin/bash /root/NLP/25001\_ner\_bert\_crf/run.sh cpu  
docker run -itd -v /work/NLP/mrc25002:/root/NLP/mrc25002 --name test2 --net host --restart=always --privileged=True bq:2 /bin/bash /root/NLP/mrc25002/run.sh xpu  
docker run -itd -v /work/NLP/gen25003:/root/NLP/gen25003 --name test3 --net host --restart=always --privileged=True bq:3 /bin/bash /root/NLP/gen25003/run.sh xpu  
docker run -itd -v /work/NLP/seg25004\_bert:/root/NLP/seg25004\_bert --name test4 --net host --restart=always --privileged=True bq:4 /bin/bash /root/NLP/seg25004\_bert/run.sh xpu  
docker run -itd -v /work/NLP/key25005:/root/NLP/key25005 --name test5 --net host --restart=always --privileged=True bq:5 /bin/bash /root/NLP/key25005/run.sh xpu  
docker run -itd -v /work/NLP/sum25006:/root/NLP/sum25006 --name test6 --net host --restart=always --privileged=True bq:6 /bin/bash /root/NLP/sum25006/run.sh xpu  
docker run -itd -v /work/NLP/com25007:/root/NLP/com25007 --name test7 --net host --restart=always --privileged=True bq:7 /bin/bash /root/NLP/com25007/run.sh xpu   
docker run -itd -v /work/NLP/mrc25002:/root/NLP/mrc25002 --name test9 --net host --restart=always --privileged=True bq:9 /bin/bash /root/NLP/mrc25002/run\_odqa.sh xpu  
docker run -itd -v /work/NLP/25011vec:/root/NLP/25011vec --name test11 --net host --restart=always bq:11 /bin/bash /root/NLP/25011vec/run.sh   
docker run -itd -v /work/NLP/info\_gen25020:/root/NLP/info\_gen25020 --name test20 --net host --restart=always --privileged=True bq:20 /bin/bash /root/NLP/info\_gen25020/run.sh xpu

查看日志：

docker logs -f {容器名}

## 4.3输入和输出控制

### 接口标识和接口图

表 3‑188 外部接口说明表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口名称 | 接口标识 | 功能描述 | 信息来源 | 信息目的地 |
| 中文分词短文本功能外部接口 | BDKJ\_SFGJ\_YSTD\_JSWBCLSF\_ZWFCDWBWBJK | 中文分词模型对文本的分词能力 | 智能算法分系统 | 交互式建模与可视化建模 |
| 军事领域命名实体识外部接口 | BDKJ\_SFGJ\_YSTD\_JSWBCLSF\_STSBWBJK | 军事领域命名实体识别模型对文本的实体识别能力 | 智能算法分系统 | 交互式建模与可视化建模 |
| 军事文本语义分析外部接口 | BDKJ\_SFGJ\_YSTD\_JSWBCLSF\_QGFXWBJK | 军事领域命语义分析模型对文本的语义分析能力 | 智能算法分系统 | 交互式建模与可视化建模 |
| 军事文本摘要外部接口 | BDKJ\_SFGJ\_YSTD\_JSWBCLSF\_WBJK | 军事文本摘要模型的摘要提取能力 | 智能算法分系统 | 交互式建模与可视化建模 |
| 军事文本主题词提取外部接口 | BDKJ\_SFGJ\_YSTD\_JSWBCLSF\_ZTCTQWBJK | 军事文本主题词提取模型提取主题词的能力 | 智能算法分系统 | 交互式建模与可视化建模 |
| 军事文本阅读理解外部接口 | BDKJ\_SFGJ\_YSTD\_JSWBCLSF\_WBYDLJWBJK | 军事文本阅读理解模型能力 | 智能算法分系统 | 交互式建模与可视化建模 |
| 文本情报生成外部接口 | BDKJ\_SFGJ\_YSTD\_JSWBCLSF\_QBSCWBJK | 军事文本情报生成模型能力 | 智能算法分系统 | 交互式建模与可视化建模 |
| 相似性搜索外部接口 | BDKJ\_SFGJ\_YSTD\_JSWBCLSF\_WBXSXSSWBJK | 军事领域文本相似性搜索处理的能力 |  |  |
| 语义推荐 | BDKJ\_SFGJ\_YSTD\_JSWBCLSF\_YYTJWBWBJK | 语义推荐处理文本的能力 |  |  |

### (BDKJ\_SFGJ\_YSTD\_JSWBCLSF\_ZWFCDWBWBJK) 中文分词短文本功能外部接口

接口地址：http://host:25004/segmentation/predict

返回格式：JSON

请求方式：HTTP POST

请求头Key：application/json

请求参数说明：

表 3‑189请求参数说明

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | 类型 | 必填 | 说明 | 默认值 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
| text | string | 是 | 输入需要分词的句子 | 我爱北京天安门 |

返回参数说明：

表 3‑190返回参数说明

|  |  |  |
| --- | --- | --- |
| 名称 | 类型 | 说明 |
| data | dict | 返回结果，包含context和pdword  context：待分词的句子,类型为string；  pdword：分词结果的列表，类型为list。 |
| errCode | int | 错误码 |
| msg | string | 错误信息 |

错误码信息：

表 3‑191错误码信息说明

|  |  |
| --- | --- |
| 错误码 | 错误信息 |
| 200 | 返回成功 |
| 400 | 返回失败，请输入正确内容 |

### (BDKJ\_SFGJ\_YSTD\_JSWBCLSF\_STSBWBJK) 军事领域命名实体识别功能外部接口

接口地址：http://host:25001/ner/predict

返回格式：JSON

请求方式：HTTP POST

请求头Key：application/json

请求参数说明：

表 3‑192请求参数说明

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | 类型 | 必填 | 说明 | 默认值 |
| text | string | 是 | 输入需要进行实体识别抽取的句子 | ‘阿巴坎’98P6E非战略反导防御防空导弹系统是为击毁现代和未来的非战略性弹道导弹而研制的，以防御战略性军事工业设施、国家的政府和行政建筑。 |

返回参数说明：

表 3‑193返回参数说明

|  |  |  |
| --- | --- | --- |
| 名称 | 类型 | 说明 |
| data | dict | 返回结果，包含context和pdEntity  context：待实体识别的句子,类型为string；  pdEntity：实体识别结果的列表，类型为list。列表中的元素包含的字段有name（string类型，实体名称），index（实体在原句子中的索引，类型为int），tag（实体的实体类型，类型为string）。 |
| errCode | int | 错误码 |
| msg | string | 错误信息 |

错误码信息：

表 3‑194错误码信息说明

|  |  |
| --- | --- |
| 错误码 | 错误信息 |
| 200 | 返回成功 |
| 400 | 返回失败，请输入正确内容 |

### (BDKJ\_SFGJ\_YSTD\_JSWBCLSF\_QGFXWBJK) 军事文本语义分析外部接口

接口地址：http://host:25007/comprehension

返回格式：JSON

请求方式：HTTP POST

请求头Key：application/json

请求参数说明：

表 3‑195请求参数说明

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | 类型 | 必填 | 说明 | 默认值 |
| text | string | 是 | 输入需要进行分析的句子 | 我爱北京天安门 |

返回参数说明：

表 3‑193返回参数说明

|  |  |  |
| --- | --- | --- |
| 名称 | 类型 | 说明 |
| data | dict | 返回结果，包含emotional\_analysis，lexical\_analysis和syntactic\_analysis。emotional\_analysis：情感分析的结果,类型为string；  lexical\_analysis：词法分析的结果，类型为list。列表中的元素包含的字段有flag（string类型，词性结果），word（句子中的词语）；  syntactic\_analysis:句法分析的结果，类型为dict。  包含deprel，head和word。Deprel（list类型，元素为string类型，词语的句法关系），head（list类型，元素为string类型，该词对应的词的位置），word（句子中的词语）。 |
| errCode | int | 错误码 |
| context | string | 待分析的句子 |
| msg | string | 错误信息 |

错误码信息：

表 3‑194错误码信息说明

|  |  |
| --- | --- |
| 错误码 | 错误信息 |
| 200 | 返回成功 |
| 400 | 返回失败，请输入正确内容 |

### (BDKJ\_SFGJ\_YSTD\_JSWBCLSF\_WBJK) 军事文本抽取式摘要外部接口

接口地址：<http://host:25006/summ>ary

返回格式：JSON

请求方式：HTTP POST

请求头Key：application/json

请求参数说明：

表 3‑195请求参数说明

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | 类型 | 必填 | 说明 | 默认值 |
| number | int | 是 | 需要返回的抽取句子个数 | 1 |
| text | string | 是 | 输入需要进行分析的句子 | 这11月21日，伊拉克总理阿巴迪声称，在收复拉沃镇后，伊拉克已在军事上终结极端组织“伊斯兰国”。而“伊斯兰国”残余分子只能在西部沙漠地带躲避，待伊拉克政府军在西部沙漠地带的清剿行动结束，就是“伊斯兰国”最终失败的历史性时刻。 |

返回参数说明：

表 3‑193返回参数说明

|  |  |  |
| --- | --- | --- |
| 名称 | 类型 | 说明 |
| data | dict | 返回结果，包含text\_abstract。  text\_abstract：摘要的抽取结果,类型为string。 |
| errCode | int | 错误码 |
| msg | string | 错误信息 |

错误码信息：

表 3‑194错误码信息说明

|  |  |
| --- | --- |
| 错误码 | 错误信息 |
| 200 | 返回成功 |
| 400 | 返回失败，请输入正确内容 |

### (BDKJ\_SFGJ\_YSTD\_JSWBCLSF\_WBJK) 军事文本生成式摘要外部接口

接口地址：[http://host:25003/summ](http://host:25006/summ)ary/generate

返回格式：JSON

请求方式：HTTP POST

请求头Key：application/json

请求参数说明：

表 3‑195请求参数说明

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | 类型 | 必填 | 说明 | 默认值 |
| text | string | 是 | 输入需要进行分析的句子 | 这11月21日，伊拉克总理阿巴迪声称，在收复拉沃镇后，伊拉克已在军事上终结极端组织“伊斯兰国”。而“伊斯兰国”残余分子只能在西部沙漠地带躲避，待伊拉克政府军在西部沙漠地带的清剿行动结束，就是“伊斯兰国”最终失败的历史性时刻。 |

返回参数说明：

表 3‑193返回参数说明

|  |  |  |
| --- | --- | --- |
| 名称 | 类型 | 说明 |
| data | dict | 返回结果，包含npredict和source。  npredictt：摘要的生成结果,类型为string；  source：输入的原句子。 |
| errCode | int | 错误码 |
| msg | string | 错误信息 |

错误码信息：

表 3‑194错误码信息说明

|  |  |
| --- | --- |
| 错误码 | 错误信息 |
| 200 | 返回成功 |
| 400 | 返回失败，请输入正确内容 |

### (BDKJ\_SFGJ\_YSTD\_JSWBCLSF\_ZTCTQWBJK) 军事文本主题词提取外部接口

接口地址：[http://host:25005/](http://host:25006/summ)theme2

返回格式：JSON

请求方式：HTTP POST

请求头Key：application/json

请求参数说明：

表 3‑195请求参数说明

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | 类型 | 必填 | 说明 | 默认值 |
| number | int | 是 | 需要提取的主题词个数 | 3 |
| text | string | 是 | 输入需要进行分析的句子 | 这11月21日，伊拉克总理阿巴迪声称，在收复拉沃镇后，伊拉克已在军事上终结极端组织“伊斯兰国”。而“伊斯兰国”残余分子只能在西部沙漠地带躲避，待伊拉克政府军在西部沙漠地带的清剿行动结束，就是“伊斯兰国”最终失败的历史性时刻。 |

返回参数说明：

表 3‑193返回参数说明

|  |  |  |
| --- | --- | --- |
| 名称 | 类型 | 说明 |
| data | dict | 返回结果，list类型。元素为dict类型。 |
| errCode | int | 错误码 |
| msg | string | 错误信息 |

错误码信息：

表 3‑194错误码信息说明

|  |  |
| --- | --- |
| 错误码 | 错误信息 |
| 200 | 返回成功 |
| 400 | 返回失败，请输入正确内容 |

### (BDKJ\_SFGJ\_YSTD\_JSWBCLSF\_WBYDLJWBJK) 军事文本阅读理解外部接口

接口地址：[http://host:25002/](http://host:25006/summ)model/predict

返回格式：JSON

请求方式：HTTP POST

请求头Key：application/json

请求参数说明：

表 3‑195请求参数说明

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | 类型 | 必填 | 说明 | 默认值 |
| context | string | 是 | 输入的军事文本段落 | 近日据外媒报道，印度尼西亚国防部长在公开场合称，将向美国购买5架全新的C-130J运输机，用以替代老旧的C-130H运输机。众所周知，印度早在上个世纪就开始都准备C-130H中型运输机。这款运输机的原型早在上世纪50年代就开始使用，其产量高达2500架以上。 |
| questions | list | 是 | 输入需要进行提问的问题 | [{"question":"印度尼西亚向美国购买的5架C-130J运输机是用来替代哪个型号的运输机","id":"1"},{"question":"印尼计划向欧洲空中客车公司引进5架A-400M运输机价格多少","id":"2"}] |

返回参数说明：

表 3‑193返回参数说明

|  |  |  |
| --- | --- | --- |
| 名称 | 类型 | 说明 |
| data | list | 返回结果，list类型。元素包含answers，id和question。  answers:string类型，返回的答案；  id：问题的id；  question：提出的问题。 |
| errCode | int | 错误码 |
| msg | string | 错误信息 |

错误码信息：

表 3‑194错误码信息说明

|  |  |
| --- | --- |
| 错误码 | 错误信息 |
| 200 | 返回成功 |
| 400 | 返回失败，请输入正确内容 |

### (BDKJ\_SFGJ\_YSTD\_JSWBCLSF\_QBSCWBJK) 文本情报生成外部接口

接口地址：[http://host:25020/](http://host:25006/summ)intelligence\_generation/predict

返回格式：JSON

请求方式：HTTP POST

请求头Key：application/json

请求参数说明：

表 3‑195请求参数说明

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | 类型 | 必填 | 说明 | 默认值 |
| text | string | 是 | 状态信息 | 侦察 |
| Konwlege\_info | string | 是 | 知识信息 | -15E“攻击鹰”（Strike Eagle）战斗轰炸机是美国麦克唐纳·道格拉斯公司在F-15“鹰”式战斗机的基础上改型设计、以对地攻击为主要任务的双座超音速战斗轰炸机，兼具对地攻击和空中优势能力。长度19.43米，高度5.63米，空重14300千克，高度5.63米，最大速度2655千米/小时，翼展13.05米，最大航程3900千米。F-15E战斗轰炸机能够使用美国空军大多数的武器，包括AIM-7“麻雀”导弹、AIM-9“响尾蛇”导弹、AIM-120先进中程空对空导弹等以进行空战，并且仍装备一门20毫米M61A1机炮。 |
| image\_info | string | 是 | 图片信息 | F-15E |

返回参数说明：

表 3‑193返回参数说明

|  |  |  |
| --- | --- | --- |
| 名称 | 类型 | 说明 |
| data | dict | 返回结果，包含npredict和source。  npredictt：摘要的生成结果,类型为string；  source：输入的原句子。 |
| errCode | int | 错误码 |
| msg | string | 错误信息 |

错误码信息：

表 3‑194错误码信息说明

|  |  |
| --- | --- |
| 错误码 | 错误信息 |
| 200 | 返回成功 |
| 400 | 返回失败，请输入正确内容 |

## 4.4其他编程技术

### 4.4.1多源异构军用数据处理通用算法与工具

1. Python
2. Java
3. c++
4. go
5. BRPC框架

### 4.4.2

1. Python
2. 大数据存储和计算框架
3. 分布式对象存储技术
4. 批计算框架、流计算框架、图处理框架
5. 消息队列框架

## 4.5编程示例

### 4.5.1多源异构军用数据处理通用算法与工具

# -\*- coding: utf-8 -\*-  
  
import requests  
import json  
  
def post\_data(url,datas):  
 headers = {'Content-Type': 'application/json'}  
 datas = json.dumps(datas)  
 r = requests.post(url, data=datas, headers=headers)  
 return r.text  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 # 实体识别  
 url\_ner = 'http://0.0.0.0:25001/ner/predict'  
 datas\_ner = {"text":"xxxx"}  
 return\_data\_ner = post\_data(url\_ner, datas\_ner)  
# 阅读理解  
 url\_mrc = 'http://0.0.0.0:25002/model/predict'  
 datas\_mrc = {"context":"xxxxxxxxxxxxxx","questions":[{"question":"xxxxx","id":"xxx"}]}  
 return\_mrc = post\_data(url\_mrc, datas\_mrc)  
# 生成式摘要  
 url\_generate = 'http://0.0.0.0:25003/summary/generate'  
 datas\_generate = {"text":"xxxxxxxxxxx"}  
 return\_generate = post\_data(url\_generate, datas\_generate)  
  
 # 分词  
 url\_segmentation = 'http://0.0.0.0:25004/segmentation/predic'  
 datas\_segmentation = {"text": "xxxxxxxxxxx"}  
 return\_segmentation = post\_data(url\_segmentation, datas\_segmentation)  
 # 关键词  
 url\_theme2 = 'http://0.0.0.0:25005/theme2'  
 datas\_theme2 = {"text":"xxxxxxxxxxxxxxxxxxxxxxx","number":5}  
 return\_theme2 = post\_data(url\_theme2, datas\_theme2)  
  
 # 抽取式摘要  
 url\_summary = 'http://0.0.0.0:25006/summary'  
 datas\_summary = {"text": "xxxxxxxxxxxxxxxxxxxxxxx", "number": 5}  
 return\_summary = post\_data(url\_summary, datas\_summary)  
 # 情报生成  
 url\_intelligence\_generation = 'http://0.0.0.0:25020/intelligence\_generation/predict'  
 datas\_intelligence\_generation ={"text":"xxxx","image\_info": "xxxx", "konwlege\_info": "xxx"}  
 return\_intelligence\_generation = post\_data(url\_intelligence\_generation, datas\_intelligence\_generation)  
 # 语义分析  
 url\_comprehension = 'http://0.0.0.0:25007/comprehension'  
 datas\_comprehension ={"text":"xxxx"}  
 return\_comprehension = post\_data(url\_comprehension, datas\_comprehension)

## 4.6错误检测和诊断特征

### 4.6.1异常处理规范

1. 捕获异常，要么处理，要么抛出给调用者；最外层的业务使用者，必须处理异常，将其转化为用户可以理解的内容。
2. 在异常处理块中，如果包含事务，一定要处理好提交和回滚逻辑。如有对IO资源操作，需要逐一关闭资源对象。

### 4.6.2异常处理规范

1. 不对大段代码进行try-catch，catch时请分清稳定代码和非稳定代码，稳定代码指的是无论如何不会出错的代码。对于非稳定代码的catch尽可能进行区分异常类型，再做对应的异常处理。
2. 捕获异常，要么处理，要么抛出给调用者；最外层的业务使用者，必须处理异常，将其转化为用户可以理解的内容。
3. 在异常处理块中，如果包含事务，一定要处理好提交和回滚逻辑。如有对IO资源操作，需要逐一关闭资源对象。

# 5注释

## 5.1注释说明

1. 文档注释：/\*\*…………\*/
2. 多行注释：/\*…………\*/
3. 单行注释：//

## 5.2注释的原则

1. 尽可能进行代码注释
2. 力求注释简洁、清楚、易懂
3. 先写注释、再写代码
4. 除了注释代码功能外，更重要是注释出代码逻辑，清晰描述出每段代码的目的。