# 数据预处理

## 离线数据

## 在线数据

# 离线ASR训练工具

## 模型训练

### 基本信息【133】

|  |
| --- |
| # 代码目录  /wp/aster/zhongke/南网第二阶段文档、源码交付物/二阶段代码-南网202111-jiami/端到端语音识别训练工具/离线/  eteh-v2-release-JXJK2021\_orig\_v2\_release |

### 数据来源【226】

|  |
| --- |
| # Json数据  /home/blue/ssd/thinkit/data/95598\_longwav/dataEnv/preparejson/split\_train\_dev |
| #特征数据  /home/blue/ssd/thinkit/data/95598\_longwav/dataEnv/feats/mfcc-hires |

### 容器【147】

|  |
| --- |
| # 加载镜像文件 **147/data/thinkit\_train/system/env**  docker load -i espnet-gpu-cuda10.1-cudnn7-u18-local-nw.tar |
| # 镜像：espnet  docker images | grep espnet  espnet gpu-cuda10.1-cudnn7-u18-local 6d3a635a5ed 15 months ago 22.5GB |
| # 开启容器：e2e-finetune-env  NV\_GPU='0,1' nvidia-docker run  --ipc=host  --name e2e-finetune-env  -it -d  -v /data/thinkit\_train:/data/thinkit\_train  --entrypoint="/bin/bash"  6d3a635a5ed1 |
|  |
| # 进入容器  docker exec -it e2e-finetune-env bash  # 容器路径  /home/Ehe/WorkSpace/eteh-v2-release-JXJK2021\_orig\_v2\_release |

### 配置修改

#### 数据配置

|  |
| --- |
| #文件  example/hkust\_egs/conf/data.yaml |
| clean\_source:  1:  type:json  name:kefu\_cts  json:/data/Ehe/json3/data.1.json  2:  type:json  name:kefu\_cts  json:/data/Ehe/json3/data.2.json  valid\_source:  1:  type:json  name:dev  json:/data/Ehe/json3/data.3.json |
| 备注：将json数据放到/data/Ehe/josn3目录下，可以自定义，特征数据路面必须跟json里面路径一致 |
|  |

#### GPU配置

|  |
| --- |
| # 查看GPU信息  nvidia-smi |
| # 文件  example/hkust\_egs/path.sh |
| MAIN\_ROOT=/home/gaochangfeng/docker/env #PLAT\_ROOT=$MAIN\_ROOT/eteh\_all/eteh2\_20210129 PLAT\_ROOT=../../ KALDI\_ROOT=/opt/kaldi ESPNET\_ROOT=$MAIN\_ROOT/espnet\_utils:$MAIN\_ROOT FLAC\_ROOT=$MAIN\_ROOT/flac\_1.3.2/bin FAIRSEQ\_ROOT=$MAIN\_ROOT/fairseq/fairseq\_20200808 LOCAL=./bin UTIL\_ROOT=$MAIN\_ROOT/utils/ exportPYTHONPATH=$LOCAL:$FAIRSEQ\_ROOT:$ESPNET\_ROOT:$PLAT\_ROOT:$PYTHONPATH  ##[-f$KALDI\_ROOT/tools/env.sh]&&.$KALDI\_ROOT/tools/env.sh  #exportPATH=$PWD/utils/:$KALDI\_ROOT/tools/openfst/bin:$KALDI\_ROOT/tools/sctk/bin:$PWD:$PATH #[!-f$KALDI\_ROOT/tools/config/common\_path.sh]&&echo>&2"Thestandardfile$KALDI\_ROOT/tools/config/common\_path.shisnotpresent->Exit!"&&exit1 #.$KALDI\_ROOT/tools/config/common\_path.sh  exportPATH=$MAIN\_ROOT/utils/:$FLAC\_ROOT:$ESPNET\_ROOT:$PATH exportCUDA\_VISIBLE\_DEVICES=0,1 |
|  |
| #文件  example/hkust\_egs/run\_train.sh |
| ../path.sh  epochs=40 init\_method='tcp://127.0.0.1:8888'  exp\_dir=output/exp train\_config=conf/ce\_espnet\_baseline\_fintune.yaml data\_conf=conf/data.yaml checkpoint=baseModel/checkpoint.29  mkdir-p${exp\_dir}  python${PLAT\_ROOT}/bin/train.py-train\_config${train\_config}\  -data\_config${data\_conf}\  -train\_nameHKUST\  -task\_filebin.taskegs.pytorch\_backend.task\_ctc\_att\  -num\_gpu2\  -task\_nameCtcAttTask\  -exp\_dir${exp\_dir}\  -num\_epochs${epochs}\  -seed100\  -checkpoint$checkpoint\  --resume\_progress\  --split  :<<eof python${PLAT\_ROOT}/bin/train.py-train\_config${train\_config}\  -data\_config${data\_conf}\  -train\_nameHKUST\  -task\_filebin.taskegs.pytorch\_backend.task\_ctc\_att\  -num\_gpu1\  -task\_nameCtcAttTask\  -exp\_dir${exp\_dir}\  -num\_epochs${epochs}\  -seed100\  -checkpoint$checkpoint\  --resume\_optimizer\  --resume\_progress\  --split eof |

### 参数说明

|  |
| --- |
| #模型训练脚本位于  example/hkust\_egs/run\_train.sh |
| #参数说明：  exp\_dir=output/exp 模型输出路径  train\_config=conf/ce\_espnet\_baseline\_fintune.yaml 模型训练配置  data\_conf=conf/data.yaml 训练数据yaml文件  checkpoint=baseModel/checkpoint.29 基线模型 |
| #运行方法：  ./run\_train.sh |
|  |

## 模型预测

|  |
| --- |
| #模型预测脚本位于  example/hkust\_egs/run\_predictor.sh |
| #参数说明：  exp\_dir=output/exp 模型输出路径  train\_config=conf/ce\_espnet\_baseline\_fintune.yaml 模型训练配置  data\_conf=conf/data.yaml 训练数据yaml文件  char\_list=baseModel/vocab.kefu\_cts.txt 字典文件  checkpoint=baseModel/checkpoint.29 基线模型 |
| #运行方法：  ./run\_predictor.sh |
|  |

## 模型评估

|  |
| --- |
| #模型评估脚本位于  example/hkust\_egs/run\_evaluate.sh |
| #参数说明：  rec=yourresultfile 结果文件  ref=yourreffile 答案文件 |
| #运行方法：  ./run\_evaluate.sh |
| #执行完成后，会在结果所在路径生成.sys文件，为识别率测试结果，如下：  SPKR|#Snt#Chr|CorrSubDelInsErrS.Err  Sum/Avg|2136108|86.510.53.02.616.069.0  #由上述结果可知，字正确率为86.5%，字错误率为16%。 |

# 在线ASR训练工具

## 模型训练

### 基本信息【133】

|  |
| --- |
| # 代码目录  /wp/aster/zhongke/南网第二阶段文档、源码交付物/二阶段代码-南网202111-jiami/端到端语音识别训练工具/在线/eteh-v2-release-JXJK2021\_orig\_online\_v2\_release |

### 数据来源【226】

|  |
| --- |
| # json数据  /home/blue/ssd/thinkit/fred/data\_product/online\_AM\_dataproduct\_nanwang\_1000\_20211015/json |
| # 特征数据  /home/blue/ssd/thinkit/fred/data\_product/online\_AM\_dataproduct\_nanwang\_1000\_20211015/json |

### 容器【147】

|  |
| --- |
| 同离线 |
| # 进入容器  docker exec -it e2e-finetune-env bash  # 容器路径  /home/Ehe/WorkSpace/eteh-v2-release-JXJK2021\_orig\_online\_v2\_release |

### 配置修改

#### 数据配置

|  |
| --- |
| #文件  example/hkust\_egs/conf/data.yaml |
| clean\_source:  1:  type:json  name:kefu\_cts  json:/data/Ehe/online/json/data.1.json  2:  type:json  name:kefu\_cts  json:/data/Ehe/online/json/data.2.json  3:  type:json  name:kefu\_cts  json:/data/Ehe/online/json/data.3.json  valid\_source:  1:  type:json  name:dev  json:/data/Ehe/online/json/data.4.json |
| 备注：将json数据放到/data/Ehe/online/json目录下，可以自定义，**特征数据路面必须跟json里面路径一致** |
| # 特征数据容器位置  /home/blue/ssd/thinkit/fred/data\_product/online\_AM\_dataproduct\_nanwang\_1000\_20211015/json |

#### GPU配置

|  |
| --- |
| # 查看GPU信息  nvidia-smi |
| # 文件  example/hkust\_egs/path.sh |
| MAIN\_ROOT=/data/xiaosujie/AM/e2e\_finetune/online-jianhang/src\_code/eteh-v2-release-JXJK2021\_orig\_online\_v2\_release  #PLAT\_ROOT=$MAIN\_ROOT/eteh\_all/eteh2\_20210129  PLAT\_ROOT=../../  KALDI\_ROOT=/opt/kaldi  ESPNET\_ROOT=$MAIN\_ROOT/espnet\_utils:$MAIN\_ROOT  FLAC\_ROOT=$MAIN\_ROOT/flac\_1.3.2/bin  FAIRSEQ\_ROOT=$MAIN\_ROOT/fairseq/fairseq\_20200808  LOCAL=./bin  UTIL\_ROOT=$MAIN\_ROOT/utils/  exportPYTHONPATH=$LOCAL:$FAIRSEQ\_ROOT:$ESPNET\_ROOT:$PLAT\_ROOT:$PYTHONPATH  ##[-f$KALDI\_ROOT/tools/env.sh]&&.$KALDI\_ROOT/tools/env.sh  #exportPATH=$PWD/utils/:$KALDI\_ROOT/tools/openfst/bin:$KALDI\_ROOT/tools/sctk/bin:$PWD:$PATH  #[!-f$KALDI\_ROOT/tools/config/common\_path.sh]&&echo>&2"Thestandardfile$KALDI\_ROOT/tools/config/common\_path.shisnotpresent->Exit!"&&exit1  #.$KALDI\_ROOT/tools/config/common\_path.sh  exportPATH=$MAIN\_ROOT/utils/:$FLAC\_ROOT:$ESPNET\_ROOT:$PATH  exportCUDA\_VISIBLE\_DEVICES=0,1 |
|  |
| #文件  example/hkust\_egs/run\_train.sh |
| ../path.sh  epochs=70  init\_method='tcp://127.0.0.1:8888'  exp\_dir=output/exp  train\_config=conf/ce\_espnet\_online\_transformer\_nospecaug\_md.yaml  data\_conf=conf/data.yaml  checkpoint=baseModel/checkpoint.59  mkdir-p${exp\_dir}  python ${PLAT\_ROOT}/bin/train.py -train\_config ${train\_config} \  -data\_config ${data\_conf}\  -train\_name HKUST\  -task\_file bin.taskegs.py torch\_backend.task\_ctc\_att\_online \  -num\_gpu 2\  -task\_name CtcAttOnlineTask \  -exp\_dir ${exp\_dir} \  -num\_epochs ${epochs} \  -seed 100 \  -checkpoint $checkpoint \  --resume\_progress \  --split  :<<eof  python ${PLAT\_ROOT}/bin/train.py -train\_config ${train\_config} \  -data\_config ${data\_conf} \  -train\_name HKUST \  -task\_file bin.taskegs.pytorch\_backend.task\_ctc\_att \  -num\_gpu 1 \  -task\_name CtcAttTask \  -exp\_dir ${exp\_dir} \  -num\_epochs ${epochs} \  -seed 100 \  -checkpoint $checkpoint \  --resume\_optimizer \  --resume\_progress \  --split  eof |

## 模型预测

## 模型评估

|  |
| --- |
| #输出位置修改  example/hkust\_egs/run\_evaluate.sh |
| ../path.sh  init\_method='tcp://127.0.0.1:8888'  rec=/home/Ehe/WorkSpace/eteh-v2-release-JXJK2021\_orig\_online\_v2\_release/example/hkust\_egs/output/exp/predictor/result\_kefu\_cts.txt  ref=/home/Ehe/WorkSpace/eteh-v2-release-JXJK2021\_orig\_online\_v2\_release/example/hkust\_egs/output/exp/predictor/ref\_kefu\_cts.txt  Python ${PLAT\_ROOT}/bin/evaluate.py -rec ${rec} -ref ${ref} |

# 离线引擎容器

## 容器【147】

|  |
| --- |
| # 加载镜像文件  docker load -i offline\_base.tar |
| docker images | grep offline\_base  offline\_base v1.1 8bd3a0500d9f 8 months ago 3.1GB |
| docker run  -p 20100:20100  -itd  -v /data/thinkit/offline\_e2e/tempvoice:/root/tempvoice  -v /data/thinkit/offline\_e2e/tempvoice\_xml:/root/tempvoice\_Xml  -v /data/thinkit/offline\_e2e/thinkit\_offline\_systeme/:/root/thinkit\_offline\_system  --privileged=true  --name offline\_e2e  offline\_base:v1.1  /usr/sbin/init |
|  |
|  |

# 在线引擎容器

## 容器【147】

|  |
| --- |
| # 加载镜像文件  docker load -i online\_docker.tar |
| docker images | grep online  online v1.1 266f66bf8d3d 10 months ago 554MB |
| docker run  -p 20200:20200  -itd  -v /data/thinkit/online\_e2e/:/root  --name online\_e2e  online:v1.1  /bin/bash |
|  |

# 在线代码编译及服务集成

|  |
| --- |
| **源码位置：**133:/wp/aster/zhongke/三阶段代码和文档补充平台层0323/应用平台封装/在线/在线识别 |

## Mrcp

|  |
| --- |
| **源码位置：**100/home/ASR/应用平台封装/在线/在线识别/在线识别系统源码  **安装包位置：**100/home/ASR/Release\_MRCP-v2.7.1\_20210820\_20210916\_nolic\_nanwang.tar.gz |

**编译：**

|  |
| --- |
| cp -r /home/ASR/应用平台封装/在线/在线识别/在线识别系统源码/在线识别系统mrcp接口**/home/ASR/mrcp/**  mv 在线识别系统mrcp接口 unimrcpserver |
| #根据文档《在线代码编译及服务集成》进行编译 |
|  |
| 编译完成后，可以打包部署到其它服务器上  部署包路径：100:/root/local/unimrcp |
|  |

## 引擎封装tbnr

### 解码器内核

|  |
| --- |
| **源码位置：**100：/home/ASR/引擎封装tbnr/解码器内核  **输出目录：**100：/home/ASR/引擎封装tbnr/解码器内核/bin  将编译好的静态库文件librecengine.a、libsrilm.a、libtools.a、libtshare.a和libwfstdecoder.a链接至引擎封装工程TBNR中统一进行测试  **应用：**在线\引擎封装tbnr\在线语音识别引擎-总模块\TBNR\_API\release\_lib\_wfst\_dnn\  cd /home/ASR/引擎封装tbnr/引擎封装tbnr/在线语音识别引擎-总模块/TBNR\_API/release\_lib\_wfst\_dnn  cp /home/ASR/引擎封装tbnr/解码器内核/bin/lib\* . |

### ITN模块

|  |
| --- |
| **源码位置：**100：/home/ASR/引擎封装tbnr/引擎封装tbnr/在线语音识别引擎子模块-ITN模块/  **输出目录：**  /home/ASR/引擎封装tbnr/引擎封装tbnr/在线语音识别引擎子模块-ITN模块/ITN/lib/libthraxrewrite.so  /home/ASR/引擎封装tbnr/引擎封装tbnr/在线语音识别引擎子模块-ITN模块/ITN/lib/thraxrewrite-tester-rec  /home/ASR/引擎封装tbnr/引擎封装tbnr/在线语音识别引擎子模块-ITN模块/ITN/compile/compileFar/example.far  **应用：**  **libthraxrewrite.so**放到 /home/ASR/引擎封装tbnr/引擎封装tbnr/在线语音识别引擎-总模块/TBNR\_API/ITN/  cd /home/ASR/引擎封装tbnr/引擎封装tbnr/在线语音识别引擎子模块-ITN模块/ITN/lib/  cp libthraxrewrite.so /home/ASR/引擎封装tbnr/引擎封装tbnr/在线语音识别引擎-总模块/TBNR\_API/ITN/  **example.far**这个文件替换到Decoder/bin目录下  cd /home/ASR/Decoder/bin  cp /home/ASR/引擎封装tbnr/引擎封装tbnr/在线语音识别引擎子模块-ITN模块/ITN/lib/example.far . |

### 标点模块

|  |
| --- |
| **源码位置：**100：/home/ASR/引擎封装tbnr/引擎封装tbnr/在线语音识别引擎子模块-标点模块  **输出目录：**/home/ASR/引擎封装tbnr/引擎封装tbnr/在线语音识别引擎子模块-标点模块/Punc/LIB\_LINUX  **应用：**标点文件目录Decoder/model/punctuation  cd /home/ASR/Decoder/model/punctuation  cp /home/ASR/引擎封装tbnr/引擎封装tbnr/在线语音识别引擎子模块-标点模块/Punc/LIB\_LINUX/\* . |

### 总模块

|  |
| --- |
| **源码位置：**100：/home/ASR/引擎封装tbnr/引擎封装tbnr/在线语音识别引擎-总模块/  进入TBNR\_API目录，执行make -f Makefile\_TBNR\_new\_epd,待编译完成后，得到libTBNR\_API.so引擎库  **输出目录：**/home/ASR/引擎封装tbnr/引擎封装tbnr/在线语音识别引擎-总模块/TBNR\_API/libTBNR\_API.so  **应用：**libTBNR\_API.so在Decoder/model/lib下  cd /home/ASR/引擎封装tbnr/引擎封装tbnr/在线语音识别引擎-总模块/TBNR\_API  cp libTBNR\_API.so /home/ASR/Decoder/model/lib/ |

### 测试用例

|  |
| --- |
| **源码位置：**100：/home/ASR/引擎封装tbnr/引擎封装tbnr/在线语音识别引擎-测试用例/  进入TBNR\_API\_TBNRTestDemo/TBNR\_Test\_demo/TBNR\_Test\_Stream 目录，执行buildTest.sh 编译脚本得到测试用例可执行文件TBNR\_Test\_Stream  cd /home/ASR/引擎封装tbnr/引擎封装tbnr/在线语音识别引擎-测试用例/TBNR\_API\_TBNRTestDemo/TBNR\_Test\_demo/TBNR\_Test\_Stream |

### 运行环境

|  |
| --- |
| **源码位置：**100：/home/ASR/引擎封装tbnr/引擎封装tbnr/ |

## 工程源码编译及服务集成

|  |
| --- |
| **源码位置**：100/home/ASR/应用平台封装/在线/在线识别/在线识别系统源码/在线识别系统decoder接口  **安装包位置**：100/home/ASR/Rel\_OnlineASR-e2e\_v3.10.9\_20210708\_20210916\_nolic\_nanwang.tar.gz  **语音识别引擎sdk包**：133/wp/aster/zhongke/三阶段代码和文档202203/三阶段代码和文档/语音识别引擎(端到端）/在线/引擎封装tbnr/在线语音识别引擎-运行环境 |

**编译：**

|  |
| --- |
| cp -r /home/ASR/应用平台封装/在线/在线识别/在线识别系统源码/在线识别系统decoder接口**/home/ASR/decoder/** |
|  |
| #根据文档《在线代码编译及服务集成》进行编译 |
|  |
| 编译完成后，可以打包部署到其它服务器上  部署包路径：100/home/ASR/Decoder |

## http源码编译

|  |
| --- |
| /home/ASR/应用平台封装/在线/在线识别/在线识别系统源码/在线识别系统http能力接口  Pom.xml 添加几行代码：       <resource>       <directory>src/main/resources</directory>       <includes>        <include>\*\*/\*.\*</include>       </includes>      </resource>  只需要将编译后的war包替换到apache-tomcat-9.0.35/webapp目录下即可 |

## WebSocket源码编译

|  |
| --- |
| /home/ASR/应用平台封装/在线/在线识别/在线识别系统源码/在线识别系统websocket能力接口  配置文件：  /root/local/WebSocket/config/application.yml  **服务启停：**  1、startup.sh  2、先kill掉websocketMonitor.sh 然后再执行stop脚本  验证路径：  /home/asr/online/rel\_onlineasr/tools/SDK/JAVA/WebSocketDemo/websocketASR\_demo  **配置文件：**  /home/asr/online/rel\_onlineasr/tools/SDK/JAVA/WebSocketDemo/websocketASR\_demo/config/application.properties  服务启停：start.sh、stop.sh |

## 系统Docker镜像

**100服务器：**

|  |
| --- |
| # 100构建基础镜像  mv dockerfile jdk-8u171-linux-x64.tar.gz redis-3.2.0.tar.gz apache-tomcat-9.0.35.20210702.tar.gz /home/ASR/Docker/  # 构建镜像  docker build -t online\_base:v1.0 .  cd dockersave /  ~~docker save -o online\_asr.tar online\_base:v1.0~~  ~~rm online\_asr.tar~~  # 添加了基本命令  docker commit -a "Ehe" -m "online\_base\_test" online\_base\_ehe online\_base\_ehe:v1  # 镜像文件：100/home/dockersave  Docker save -o online\_base\_ehe.tar online\_base\_ehe:v1  Docker exec -it 7701484e03ce /bin/bash |
|  |
|  |

# 在线语音识别系统安装

**144服务器：加载容器**

|  |
| --- |
| # 将online\_base\_ehe.tar拷贝到144：/data/Ehe/dockersave  Docker load -i online\_base\_ehe.tar |
| # 运行容器  ~~docker run -p 20051:20051 -i -t -d -v /home/data/ASR:/home/data/ASR --name online\_base\_eheonline\_base:v1.0 /bin/bash~~ |
| docker run -p 20051:20051 -i -t -d -v /data/Ehe/ASR:/data/Ehe/ASR --name online\_base\_ts online\_base\_ehe:v1 /bin/bash |
|  |
| docker commit -a "" -m "" online\_base\_ts online\_base\_ehe:v1.2  docker save -o online\_base\_ts\_v1.2.tar online\_base\_ehe:v1.2 |

**容器内安装**

|  |
| --- |
| **进入容器online\_base\_ts：**docker exec -it 7701484e03ce /bin/bash  **容器目录：**/home/asr/online/rel\_onlineasr  **安装文档：**在线语音识别系统安装手册  进入/home/asr/online/rel\_onlineasr/tools/tools\_install目录，执行脚本install\_All.sh，解压tomcat、redis、jdk、nginx到~/local目录下 |

## jdk安装

|  |
| --- |
|  |

## nginx安装

|  |
| --- |
| 配置文件：/root/local/nginx/nginx/server\_conf\_asr.txt  配置参数：server=127.0.0.1:20200 weight=2  启动命令：cd /root/local/nginx/nginx  ./sbin |

## redis单机安装

|  |
| --- |
| 配置文件：/root/local/redis/redis-3.2.0/redis.conf  配置参数：bind 127.0.0.1  Port 30010  启动命令：cd /root/local/redis/redis-3.2.0  nohup ./redis-server redis.conf > /dev/null 2>&1 &  验证命令：ps -ef | grep redis  ./redis-server redis.conf &  ./redis-cli -h 127.0.0.1 -p 30010  monitor |

## Tomcat安装

|  |
| --- |
| **配置文件1**：/root/local/tomcat/apache-tomcat-9.0.35/webapps/asrability/WEB-INF/classes/redis.properties  **配置参数**：ASR\_REDISSERVER=127.0.0.1  ASR\_REDISPORT=30010  ASR\_REDIS\_SINGLE = 127.0.0.1:30010 # 对应redis  ASR\_REDISSETLIST=ASR\_SERVICEREQ:test01 # Decoder用到  ~~ASR\_REDISADD= \*\*\*\*\*~~  # 对应redis集群  ASR\_INCLUDEPASSWORD=false # 是否有密码，集群有密码，单机模型没有密码  ASR\_REDISPASSWORD=foobared  ASR\_CLUSTER=false#是否启用集群  Ps:集群模式不一样  **配置文件2：**/root/local/tomcat/apache-tomcat-9.0.35/conf/server.xml  **配置参数**：<Connector port="20051" protocol="org.apache.coyote.http11.Http11NioProtocol" # 对外映射端口  **启动命令**：cd /root/local/tomcat/apache-tomcat-9.0.35/bin  sh shutdown.sh  sh startup.sh  **验证命令**：ps -ef | grep tomcat |

## **Decoder**安装

|  |
| --- |
| 将过程源码编译的Decoder文件夹拷贝到/home/asr/online/rel\_onlineasr/  cp -R /data/Ehe/ASR/Decoder /home/asr/online/rel\_onlineasr/ |
| 进入目录Decoder/bin修改启动线程数可编辑脚本decoder.sh  ./decoder -s 10 -p 20051 |
| **配置文件：**/home/asr/online/rel\_onlineasr/Decoder/conf/decoder.conf  ，修改是否保存语音文件和识别结果  #是否保存语音，范围0－3，0代表不保存语音；1代表只保存接收到的pcm语音，2代表只保存转换后的pcm16语音；3代表同时保存pcm语音和pcm16语音。  SaveVoice=3  VoiceDir=../voice  #设置redis集群是否设置密码，1：设置，0：不设置  RedisPasswdOn=0  RedisCluster=127.0.0.1:30010  Ps:集群模式不一样，单机默认没有密码，集群要密码  #是否保存语音识别结果，0不保存，非0保存  SaveRegResult=1  RegResultPath=../txtFile  RedisKey=ASR\_SERVICEREQ:test01 |
| 添加环境变量  vi /root/.bashrc  exportLD\_LIBRARY\_PATH=$LD\_LIBRARY\_PATH:/home/asr/online/rel\_onlineasr/Decoder/model/lib  exportLD\_LIBRARY\_PATH=$LD\_LIBRARY\_PATH:/home/asr/online/rel\_onlineasr/Decoder/lib  source/root/.bashrc  授权执行:chmod -R +x Decoder  启动命令：cd /home/asr/online/rel\_onlineasr/Decoder/bin  ./shutdown.sh  ./start.sh  服务验证：ps -ef | grep decoder |
|  |
| 其它配置：  /home/asr/online/rel\_onlineasr/Decoder/model/scripts/WFSTDecoder\_onlyrec.cfg  # 标点  PrintPunctuation=true  # 数字  isDoITN=false |

## 重启服务

|  |
| --- |
| **重启nginx：**  cd /root/local/nginx/nginx  ./sbin  **重启redis：**  cd /root/local/redis/redis-3.2.0  nohup ./redis-server redis.conf > /dev/null 2>&1 &  **重启Tomcat：**  cd /root/local/tomcat/apache-tomcat-9.0.35/bin  sh startup.sh  **重启Decoder：**  cd /home/asr/online/rel\_onlineasr/Decoder/bin  sh start.sh |

# 集群部署

## 准备工作

|  |
| --- |
| # 容器内部映射端口 20200是http服务 8066是websocket服务 docker run -itd --name online\_e2e --restart=unless-stopped -p 27528:20200 -p 27530:8066 -v /wp/aster/ws/online\_asr\_e2e/rel\_onlineasr:/root/rel\_onlineaser offline\_base:v1.1  # 文件  133：/wp/dockersave/asr\_model/asr\_model/model/shenzhen\_asr/online  scp Rel\_OnlineASR-e2e\_v3.10.9\_20210708\_20220526\_withlic\_nanwang.tar.gz root@172.16.128.147:/data/Ehe/cluder\_online\_asr/ |

## 部署计划

* 133 服务器（主机）：
* online\_e2e\_redis
* online\_e2e\_bc：apache-tomcat-9.0.62、rel\_onlineaser
* 147 服务器
* online\_e2e\_bc：apache-tomcat-9.0.62、rel\_onlineaser
* 191 服务器
* online\_e2e\_bc：apache-tomcat-9.0.62、rel\_onlineaser

## 容器

193：

docker run -itd --name online\_e2e\_bc --restart=unless-stopped -p 27528:20200 -v /wp/Ehe/cluder\_online\_asr/rel\_onlineasr:/root/rel\_onlineaser offline\_base:v1.1

## Redis

|  |
| --- |
| 注意：  1、redis.conf端口要改成0.0.0.0 |
| 命令：  1、单节点测试：./redis-server cluster/7000.conf |

## Tomcat

## Decoder

# ASR准确率验证

|  |
| --- |
| 147服务器：  外包编译：/data/Ehe/ASR/asr\_api\_test/asr\_online\_ts  中科编译：/data/Ehe/ASR/asr\_api\_test/asr\_online\_ts\_zk  **asr\_online\_ts为例**：  short\_ivr7.mlf：标注结果  short\_ivr\_1\_2.mlf：转写结果  1\_2：原始语音文件  result\_1\_2：转写结果数据  get\_asr\_result.py：asr接口调用脚本，将1\_2中的录音转写结果写入到result\_1\_2  txt2mlf.py：将转写结果result\_1\_2中的文件全部写入到short\_ivr\_1\_2.mlf  HResults：计算准确率  ./HResults -t -I /data/Ehe/ASR/asr\_api\_test/asr\_online\_ts/short\_ivr7.mlf /dev/null /data/Ehe/ASR/asr\_api\_test/asr\_online\_ts/short\_ivr\_1\_2.mlf |