人工智能实验室

Alchemy 中文使用文档

文档版本　　V3.2.0

发布日期　　2025-7-16

目录

目录 1
版本历史 1
第 1 章 Alchemy 介绍 2
简介 2
核心特性 2
支持芯片 2
目标用户 2
核心优势 2
系统要求 3
最低配置 3
推荐显卡及驱动配置 3
第 2 章 Alchemy 安装及环境配置 4
步骤二、安装显卡驱动 5
步骤三、解压 Alchemy 工程 6
步骤四、安装及迁移工作环境 6
第 3 章 Alchemy 工程说明 8
目录结构 8
工程示例 8
Alachemy 快速跑通示例工程教程 8
步骤一、新建项目工程 8
步骤三、配置指令词 9
步骤三、运行 Alchemy 快速示例工程，检查报告输出 10
Alachemy 训练一个新的项目--实战教程 10
步骤一、新建项目工程 10
步骤二、配置数据存储路径 11
步骤三、设置指令词，模型以及项目输出路径 11
步骤四、运行 Alchemy 训练目标模型，检查报告输出 12
输出文件说明 16

版本历史

版本	修改日期	修改人员	修改描述
v1.0	2023-04-17	wangzhx	初始发布版本
v2.0	2025-05-14	wangzhx	更新框架，支持一键 run
V3.1.0	2025-01-20	songyun	更新框架，支持自动数据合成，支持自动清洗，支持一键输出多版本
V3.2.0	2025-07-17	songyun	新增自动数据合成方法，更新自动清洗，新增 model_k

第 1 章 Alchemy 介绍

简介

Alchemy 是杭州国芯微基于 tensorflow 框架自研的一个相对成熟稳定的 KWS(关键词识别)模型微调和部署系统，可以帮助用户微调 KWS 模型以使用特定的应用场景。

核心特性

- 自动化数据处理： 支持 TTS 合成，支持音频数据的自动读取和预处理。
- 灵活的微调配置：提供多种模型结构和训练参数，通过简单配置即可启动微调。
- 从训练到模型导出：显示训练进度，损失函数变化，自动合成测试集，自动测试并支持导出部署文件。

支持芯片

- GX8002B（低功耗）
- GX8002D（非低功耗）
- GX8008C
- GX8003
- GX8005/GX8006
- GX8302B

目标用户

无算法背景的软件工程师

核心优势

一键式自动化流程:数据生成 → 数据准备 → 模型训练 → NPU 部署文件生成 → 测试报告输出

系统要求

最低配置

- 系统: ubuntu-18.04
- CPU: x86_64 32 核
- 内存: 128GB
- 存储: 4T-SSD
- GPU: 2080Ti
- CUDA 版本: > 11.0

注意：

1. Alchemy 当前已适配过 Tesla P40、 2080Ti、4090Ti 显卡，其他显卡我们暂未验证。
2. 要求存储硬盘是 SSD，如果是机械硬盘，那么训练速度会慢八倍左右。

推荐显卡及驱动配置

- 2080Ti/4090Ti
  - NVIDIA-SMI 驱动版本: 525.105.17
  - CUDA 版本: 12.0
- Tesla P40
  - NVIDIA-SMI 驱动版本: 550.90.07
  - CUDA 版本: 12.4

注意：驱动版本的确定，需要根据系统版本，显卡型号和 CUDA 版本 来确定，搜索方法请参考视频教程

第 2 章 Alchemy 安装及环境配置

Alchemy 相关文件均放置在一个 ftp 服务器上，并对外开源，用户只要登入该服务器，下载如下相关文件即可。

- ftp 链接：ftp.nationalchip.com
- 用户名：test
- 密码：test@1234
- 需下载文件：

```
alchemy_V3.2.0_250715_pub.tar.gz      # Alchemy 框架核心文件
corpus_data.tar.gz                    # 基础语料库
Miniforge3-Linux-x86_64.sh            # 轻量级的Python 环境管理工具安装脚本
py36_tf115.tar.gz                     # Python3 环境
SenseVoice_py311.tar.gz               # python3 环境
cosyvoice.tar.gz                      # python3 环境
th_12.tar.gz                          # python3 环境
```

- 下载命令

```
wget --ftp-user=test --ftp-password=test@1234
ftp://ftp.nationalchip.com/alchemy_prj_all_source_packages/alchemy_code/alchemy_V3.2.0_20250715_pub.tar.gz
 .
wget --ftp-user=test --ftp-password=test@1234
ftp://ftp.nationalchip.com/alchemy_prj_all_source_packages/env_source/corpus_data.tar.gz .
wget --ftp-user=test --ftp-password=test@1234 ftp://ftp.nationalchip.com/Miniforge3-Linux-x86_64.sh .
wget --ftp-user=test --ftp-password=test@1234
ftp://ftp.nationalchip.com/alchemy_prj_all_source_packages/env_source/py36_tf115.tar.gz .
wget --ftp-user=test --ftp-password=test@1234
ftp://ftp.nationalchip.com/alchemy_prj_all_source_packages/env_source/SenseVoice_py311.tar.gz .
wget --ftp-user=test --ftp-password=test@1234
ftp://ftp.nationalchip.com/alchemy_prj_all_source_packages/env_source/cosyvocie.tar.gz .
wget --ftp-user=test --ftp-password=test@1234 ftp://ftp.nationalchip.com/th_12.tar.gz .
```

步骤二、安装显卡驱动

查看是否已经安装 NVIDIA 驱动

```
nvidia-smi -L

# 如下打印表示没安装
Command 'nvidia-smi' not found, but can be installed with:

apt install nvidia-340             # version 340.108-0ubuntu5.20.04.2, or
apt install nvidia-utils-390        # version 390.157-0ubuntu0.20.04.1
apt install nvidia-utils-450-server  # version 450.248.02-0ubuntu0.20.04.1
apt install nvidia-utils-470        # version 470.256.02-0ubuntu0.20.04.1
apt install nvidia-utils-470-server  # version 470.256.02-0ubuntu0.20.04.1
apt install nvidia-utils-535        # version 535.183.01-0ubuntu0.20.04.1
apt install nvidia-utils-535-server  # version 535.230.02-0ubuntu0.20.04.3
apt install nvidia-utils-550-server  # version 550.144.03-0ubuntu0.20.04.1
apt install nvidia-utils-565-server  # version 565.57.01-0ubuntu0.20.04.2
apt install nvidia-utils-570-server  # version 570.86.15-0ubuntu0.20.04.5
apt install nvidia-utils-435        # version 435.21-0ubuntu7
apt install nvidia-utils-440        # version 440.82+really.440.64-0ubuntu6
apt install nvidia-utils-418-server  # version 418.226.00-0ubuntu0.20.04.2
```

在 NVIDIA 官网上寻找适配的驱动版本：NVIDIA 官网链接。
在目标系统上安装 NVIDIA 显卡驱动，如下以我电脑为例子，安装如下：

> Tesla P40:
> NVIDIA-SMI 驱动版本:推荐 550.90.07
> CUDA 版本:推荐 12.4

```
#禁用默认驱动
sudo vi /etc/modprobe.d/blacklist.conf，将 blacklist nouveau 这句加到该文件中
sudo update-initramfs -u
#重启目标系统
sudo reboot
#安装 NVIDIA 显卡驱动
sudo ./NVIDIA-Linux-x86_64-550.90.07.run  -no-x-check -no-nouveau-check -no-opengl-files
# 重启目标系统
sud reboot
```

步骤三、解压 Alchemy 工程

1. 解压主工程文件

```
tar -zxvf alchemy_V3.2.0_20250715_pub.tar.gz -C ~/
```

2. 解压 AlchemyAI 语料库

```
tar xzvf corpus_data.tar.gz -C ~  #解压得到 corpus 和 tts 两个文件夹
cd corpus     # 迁移语料库路径
bash corpus_data_setting.sh ###运行脚本
```

步骤四、安装及迁移工作环境

1. 安装 miniforge3

```
./Miniforge3-Linux-x86_64.sh
source ~/.bashrc
#解压环境
tar xzvf py36_tf115.tar.gz -C ~/miniforge3/envs
tar xzvf th_12.tar.gz -C ~/miniforge3/envs
tar xzvf SenseVoice_py311.tar.gz -C ~/miniforge3/envs
tar xzvf cosyvoice.tar.gz -C ~/miniforge3/envs

#设置 miniforge3 环境变量
#把如下内容，追加到 ~/.bashrc 文件中，并且把 `pub` 改成自己的目标系统用户名
# >>> conda initialize >>>
# !! Contents within this block are managed by 'conda init' !!
__conda_setup="$('/home/pub/miniforge3/bin/conda' 'shell.bash' 'hook' 2> /dev/null)"
if [ $? -eq 0 ]; then
    eval "$__conda_setup"
else
    if [ -f "/home/pub/miniforge3/etc/profile.d/conda.sh" ]; then
        . "/home/pub/miniforge3/etc/profile.d/conda.sh"
    else
        export PATH="/home/pub/miniforge3/bin:$PATH"
    fi
fi
unset __conda_setup
# <<< conda initialize <<<
```

```
#添加完成后生效
source ~/.bashrc
```

2. 配置 Torch 模型缓存及路径适配

```
mkdir -p ~/.cache/
cp -r corpus/cache/* ~/.cache/
#路径适配修改
sed -i "s|/home/sundy|$HOME|g" ~/miniforge3/envs/py36_tf115/bin/pip
sed -i "s|/home/sundy|$HOME|g" ~/miniforge3/envs/py36_tf115/bin/gxnpuc
sed -i "s|/home/sundy|$HOME|g" ~/miniforge3/bin/conda
```

3. 激活环境安装依赖库

```
source ~/.bashrc
conda activate py36_tf115
cd alchemy/tools/logfbank
python setup.py develop
sudo apt install ffmpeg
```

第 3 章 Alchemy 工程说明

目录结构

主要工程目录介绍

```
pub@SPD-R730xd:~/alchemy$ tree -L 1
├── example_local          #快速跑通流程示例目录
├── example_0              #用户实际项目训练工程目录
├── docs                   #使用手册
└── alchemy                #唤醒词引擎
```

工程示例

1. 快速默认示例: example_local
   - 快速跑通整个流程的示例,用于检查工作环境是否正常,体验模型从无到有的过程。
2. 实际项目示例: example_0
   - 可以达到实际应用效果的示例, 方便新项目开发。

Alachemy 快速跑通示例工程教程

步骤一、新建项目工程

拷贝/新建新工程方法

```
# 我的 home 路径如下: /home/pub
# 我的alchemy 解压存储路径: /home/pub/workspace

#进入示例模版工程
cd ~/workspace/alchemy
cp -r example_local user1_prj  # 拷贝快速示例为自定义示例, user1_prj 可自定义
```

步骤三、配置指令词

对于一个项目，最主要的参数是**指令词**、**模型型号**和**项目输出路径**，只要设置好这三个参数就可以了。这三个参数默认设置如下：

```
# 我的工作路径如下: /home/pub/workspace/alchemy/user1_prj
    cd /home/pub/workspace/alchemy/user1_prj  # 进入工作目录
    $ vi prj_conf/prj_conf.yaml # 输入命令，打开参数设置文件
# 显示如下:
    1 project_conf:  # 项目参数
    2     work_space: './test_work_space' # 项目输出路径
    3     project_name: 'TEST_project_local'  # 项目名称
    4     gpu_id: 0
    5 model_conf:   # 模型参数
    6     model_name: 'model_f'     # 模型型号
    7     chip: 'gx8002'        # 芯片型号
    8 cmd_conf:    # 指令词参数
    9     main_cmd:   # 主唤醒词
    10         - 小华小华
    11         - 小奥小奥
    12         - 小迪小迪
    13     other_cmd:  # 二级指令词
    14         - 中速风
    15         - 低速风
    16         - 关闭净化
    17         - 音量最小
    18         - 风扇暂停
    # 备注：暂不支持英文指令词，以及不支持指令词中带各种标点符号
```

根据自己的喜好设置项目参数，模型参数以及指令词参数：

```
# 比如我将 prj_conf/prj_conf.yaml 修改为如下的样子:
    1 project_conf:  # 项目参数
    2     work_space: './test_project' # 项目输出路径
    3     project_name: 'self_project'  # 项目名称
    4     gpu_id: 0
    5 model_conf:   # 模型参数
    6     model_name: 'model_k'     # 模型型号
    7     chip: 'gx8002'        # 芯片型号
    8 cmd_conf:    # 指令词参数
    9     main_cmd:   # 主唤醒词，如有更多在后面跟上即可
    10         - 小树同学
    11         - 你好小奥
    12         - 小艺小艺
```

```
    other_cmd:  # 二级指令词
        - 打开风扇
        - 打开消毒
        - 打开净化
        - 音量最大
# 备注：暂不支持英文指令词，以及不支持指令词中带各种标点符号
```

以上即完成项目参数，模型参数以及指令词参数的这些项目必须的设置，可以开始跑代码了。

步骤三、运行 Alchemy 快速示例工程，检查报告输出

1. 运行训练脚本

一键训练 (一键完成，数据生成、数据准备、模型训练、模型测试和部署)

```
bash local_run.sh  # 一键训练
```

2. 运行结束

经过 10 分钟左右，Alchemy 即可跑完整个快速示例，会输出几份模型和测试报告，根据测试报告结果，挑出你最认可的模型即可

- **注意：报告输出路径，会在运行结束时候打印出来，报告有打印输出，也有详细的 xlsx 表格根据需要获取即可**
- **以上示例中的项目输出路径为：/home/pub/workspace/alchemy/user1_prj/test_project/project/self_project**
- **$项目输出路径/corpus 为 生成的指令词 wav,数据增强 tfr,以及测试集**
- **$项目输出路径/version 下有 v0.1.1 和 v0.1.2 两个文件夹，分别为两个版本的输出模型，其中测试报告路径为：v0.1.1/result/ori/default/*excel，可部署的模型文件路径为：v0.1.1/result/ori/default/*default/model*.h**

Alachemy 训练一个新的项目--实战教程

步骤一、新建项目工程

拷贝/新建新工程方法

```
# 我的 home 路径如下：/home/pub/
#进入示例模版工程
cd ~/workspace/alchemy/
cp -r example_0 dryer_prj  #  从项目实战示例拷贝，比如这里创建了晾衣架项目
#进入新创建的项目工作目录
cd dryer_prj
```

步骤二、配置数据存储路径

在前面的下载相关文件 步骤中，下载过通用语料 corpus.tar.gz，并且在 解压 Alchemy 工程 步骤中已经解压在 alchemy 工程所在路径下，那么我的配置路径方法如下：

```
# 我的 home 路径如下: /home/pub
# 我的 alchemy 工程路径如下: /home/pub/workspcae/alchemy
# 那么我的通用数据存储路径: /home/pub/workspace/corpus
# 创建模型训练数据存储路径
# 配置路径
vim prj_conf/prj_conf.yaml
#把该文件中 database_conf 如下内容修改成如下:
```

```
            cosyvoice_list_out_dir: '${project_corpus_path}/test/list/cosyvoice'
        valle_flag: true #false
        valle_conf:
            valle_max_wav_num: 150
            valle_wav_out_dir: '${project_corpus_path}/test/wav/valle'
            valle_list_out_dir: '${project_corpus_path}/test/list/valle'
        merge_flag: true
        merge_conf:
            merge_max_wav_num: 100
            merge_list_out_dir: '${project_corpus_path}/test/list/merge'
    database_conf:
        train_database_conf:
            hifi_asr_list_lib: '/home/pub/workspace/corpus/common_data/hifi_valle_database/hifi/train/list/asr'
            valle_asr_list_lib: '/home/pub/workspace/corpus/common_data/hifi_valle_database/valle/train/list/asr'
            cosyvoice_asr_list_lib: '/home/pub/workspace/corpus/common_data/hifi_valle_database/cosyvoice/train/list/asr'
            clone_source_male: '/home/pub/workspace/corpus/common_data/valle_deploy_source/source_list/train/adult_male.list'
            clone_source_female: '/home/pub/workspace/corpus/common_data/valle_deploy_source/source_list/train/adult_female.list'
            clone_source_child: '/home/pub/workspace/corpus/common_data/valle_deploy_source/source_list/train/child.list'
        test_database_conf:
            hifi_asr_list_lib: '/home/pub/workspace/corpus/common_data/hifi_valle_database/hifi/test/list/asr'
            valle_asr_list_lib: '/home/pub/workspace/corpus/common_data/hifi_valle_database/valle/test/list/asr'
            cosyvoice_asr_list_lib: '/home/pub/workspace/corpus/common_data/hifi_valle_database/cosyvoice/test/list/asr'
            clone_source_male: '/home/pub/workspace/corpus/common_data/valle_deploy_source/source_list/test/adult_male.list'
            clone_source_female: '/home/pub/workspace/corpus/common_data/valle_deploy_source/source_list/test/adult_female.list'
            clone_source_child: '/home/pub/workspace/corpus/common_data/valle_deploy_source/source_list/test/child.list'
dataset_conf:
    train_aug_conf:
        noise_dir: '/home/pub/workspace/corpus/common_data/noise/add_noise'
        ground_noise_dir: '/home/pub/workspace/corpus/common_data/noise/ground_noise'
        reverb_dir: '/home/pub/workspace/corpus/common_data/RIR'
        white_noise_dir: '/home/pub/workspace/corpus/common_data/noise/white_noise'
    domain_aug_conf:
        noise_dir: '/home/pub/workspace/corpus/common_data/noise/add_noise'
        ground_noise_dir: '/home/pub/workspace/corpus/common_data/noise/domain_noise'
        reverb_dir: '/home/pub/workspace/corpus/common_data/RIR'
        white_noise_dir: '/home/pub/workspace/corpus/common_data/noise/white_noise'
    common_tfr_conf:
        asr_tfr_dir: ['/home/pub/workspace/corpus/common_data/ASR']
deploy_conf:
    neg_path: '/home/pub/workspace/corpus/common_data/test_set/neg'
```

步骤三、设置指令词，模型以及项目输出路径

此步可参见快速示例中的指令词和模型项目输出路径参数设置。

比如我的设置如下所示：

```
# 打开项目参数设置文件
vim  /home/pub/workspace/alchemy/dry_prj/prj_conf/prj_conf.yaml
```

```yaml
project_conf:
    work_space: '/home/pub/workspace/kws'
    project_name: 'xiaowei_dryer'
    gpu_id: 0
model_conf:
    model_name: 'model_f'
    chip: 'gx8002'
cmd_conf:
    main_cmd:
        - 小维小维
        - 你好小维
    other_cmd:
        - 中速风
        - 低速风
        - 打开风扇
        - 关闭风扇
        - 打开净化
        - 关闭净化

```

步骤四、运行 Alchemy 训练目标模型，检查报告输出

3. 运行训练脚本

一键训练 (一键完成，数据生成、数据准备、模型训练、模型测试和部署)

```
bash run.sh  # 一键训练
```

4. 分阶段执行

可以自主控制运行步骤，主要为了在微调迭代使用，减少不必要的时间，但是请注意，步骤顺序有前后依赖，数据准备好，不可训练

```
# 在 run.sh 中，start_stage 与 end_stage 控制了运行的阶段，prject_conf 参数控制项目参数，
strategy_conf 控制了模型微调参数。
# 只进行数据生成，run.sh 设置如下：
```

```
project_conf=./prj_conf/prj_conf.yaml
conf_dir=./train_test_conf
conf=$conf_dir/v0.1.1.yaml
start_stage=1
end_stage=1 #2
python -u auto_finetune.py --project_config $project_conf \
          --strategy_config $conf \
          --start_stage $start_stage \
          --end_stage $end_stage

#conf=$conf_dir/v0.2.1.yaml
#start_stage=2
#end_stage=5
#python -u auto_finetune.py --project_config $project_conf \
#           --strategy_config $conf \
#           --start_stage $start_stage \
#           --end_stage $end_stage
#
#conf=$conf_dir/v0.1.1.yaml
#start_stage=3
#end_stage=5
#python -u auto_finetune.py --project_config $project_conf \
#           --strategy_config $conf \
#           --start_stage $start_stage \
#           --end_stage $end_stage
#
#conf=$conf_dir/v0.3.1.yaml
#start_stage=3
#end_stage=5
#python -u auto_finetune.py --project_config $project_conf \
#           --strategy_config $conf \
#           --start_stage $start_stage \
#           --end_stage $end_stage
#
#conf=$conf_dir/v0.4.1.yaml
#start_stage=3
#end_stage=5
```

比如在我这里，只进行数据生成这一步时的输出一部分如下所示：

```
$ bash run_local.sh
WARNING:tensorflow:Deprecation warnings have been disabled. Set TF_ENABLE_DEPRECATION_WARNINGS=1 to re-enable them.
### ================================================================ ###
### ==== stage 0: project path creating and checking!!! ==== ###
### ================================================================ ###
    ==> main cmd list is ['小华小华', '小奥小奥', '小迪小迪']
    ==> other cmd list is ['中速风', '低速风', '关闭净化', '音量最小', '风扇暂停']
    ==> project debug_modelk is created,located at /root/workspace/alchemy/example_local/test_work_space/project/debug_modelk/corpus
### ================================================================ ###
### ==== stage 1:  data prepare, train/test data clone and clean!!! ==== ###
### ================================================================ ###
    ==> train data prepare start!!!
      ==> total 8 cmds need to valle clone, which is : ['关闭净化', '小奥小奥', '音量最小', '小华小华', '风扇暂停', '小迪小迪', '中速风', '低速风']  !!!!! =====
        ==> batch_0 will clone 8 cmds:[['关闭净化', '小奥小奥', '音量最小', '小华小华', '风扇暂停', '小迪小迪', '中速风', '低速风']] ====
100%|████████████████████████████████████████████████████████████████████████| 20/20 [01:18<00:00,  3.90s/it]
      ==> fininshed valle clone, asr clean start !!
      ==> in file: 低速风.scp
rtf_avg: 0.038: 100%|████████████████████████████████████████████████████████| 20/20 [00:01<00:00, 14.92it/s]
      ==> in file: 风扇暂停.scp
rtf_avg: 0.003: 100%|████████████████████████████████████████████████████████| 20/20 [00:00<00:00, 102.54it/s]
      ==> in file: 中速风.scp
rtf_avg: 0.005: 100%|████████████████████████████████████████████████████████| 20/20 [00:00<00:00, 121.42it/s]
      ==> in file: 小华小华.scp
rtf_avg: 0.003: 100%|████████████████████████████████████████████████████████| 20/20 [00:00<00:00, 95.92it/s]
      ==> in file: 关闭净化.scp
rtf_avg: 0.003: 100%|████████████████████████████████████████████████████████| 20/20 [00:00<00:00, 87.32it/s]
      ==> in file: 音量最小.scp
rtf_avg: 0.003: 100%|████████████████████████████████████████████████████████| 20/20 [00:00<00:00, 93.36it/s]
      ==> in file: 小奥小奥.scp
rtf_avg: 0.004: 100%|████████████████████████████████████████████████████████| 20/20 [00:00<00:00, 126.99it/s]
      ==> in file: 小迪小迪.scp
rtf_avg: 0.003: 100%|████████████████████████████████████████████████████████| 20/20 [00:00<00:00, 93.37it/s]
Files merged and copied to /root/workspace/alchemy/example_local/test_work_space/project/debug_modelk/corpus/train/list/tmp/merge/hifi_valle_merge_1
Files merged and copied to /root/workspace/alchemy/example_local/test_work_space/project/debug_modelk/corpus/train/list/tmp/merge/hifi_valle_merge
    ==> test data prepare start!!!
      ==> total 8 cmds need to valle clone, which is : ['关闭净化', '小奥小奥', '音量最小', '小华小华', '风扇暂停', '小迪小迪', '中速风', '低速风']  !!!!! =====
        ==> batch_0 will clone 8 cmds:[['关闭净化', '小奥小奥', '音量最小', '小华小华', '风扇暂停', '小迪小迪', '中速风', '低速风']] ====
100%|████████████████████████████████████████████████████████████████████████| 10/10 [00:39<00:00,  3.93s/it]
      ==> fininshed valle clone, asr clean start !!
```

 # 只进行数据增强，run.sh 设置如下：

```
project_conf=./prj_conf/prj_conf.yaml
conf_dir=./train_test_conf
conf=$conf_dir/v0.1.1.yaml
start_stage=1
end_stage=2
python -u auto_finetune.py --project_config $project_conf \
          --strategy_config $conf \
          --start_stage $start_stage \
          --end_stage $end_stage

conf=$conf_dir/v0.2.1.yaml
start_stage=2
end_stage=2 #5
python -u auto_finetune.py --project_config $project_conf \
          --strategy_config $conf \
          --start_stage $start_stage \
          --end_stage $end_stage

#conf=$conf_dir/v0.1.1.yaml
#start_stage=3
#end_stage=5
#python -u auto_finetune.py --project_config $project_conf \
#          --strategy_config $conf \
#          --start_stage $start_stage \
#          --end_stage $end_stage
#
#conf=$conf_dir/v0.3.1.yaml
#start_stage=3
#end_stage=5
#python -u auto_finetune.py --project_config $project_conf \
#          --strategy_config $conf \
#          --start_stage $start_stage \
#          --end_stage $end_stage
#
#conf=$conf_dir/v0.4.1.yaml
#start_stage=3
#end_stage=5
#python -u auto_finetune.py --project_config $project_conf \
#          --strategy_config $conf \
#          --start_stage $start_stage \
#          --end_stage $end_stage
```

start_stage=2
end_stage=2

此部分注释掉

数据增强步骤示例输出如下：

```
  ==> stage1: data prepare cost 520.03 s
### ============================================================ ###
### ==== stage 2:  wav aug !!! ==== ###
### ============================================================ ###
      ==> here start to default wav aug !!!
      Loading all the .wav files in the path:/root/workspace/alchemy/example_local/test_work_space/data/common_data/noise/add_noise, and create the materials wa
v_db
100%|| 8/8 [00:00<00:00, 122.88it/s]
      Finish loading all the .wav files in the path above!
      Loading all the .wav files in the path:/root/workspace/alchemy/example_local/test_work_space/data/common_data/noise/ground_noise, and create the materials
 wav_db
100%|| 3/3 [00:00<00:00, 27.60it/s]
      Finish loading all the .wav files in the path above!
      Loading all the .wav files in the path:/root/workspace/alchemy/example_local/test_work_space/data/common_data/RIR, and create the materials wav_db
100%|| 2/2 [00:00<00:00, 1357.38it/s]
      Finish loading all the .wav files in the path above!
      Loading all the .wav files in the path:/root/workspace/alchemy/example_local/test_work_space/data/common_data/noise/white_noise, and create the materials
wav_db
100%|| 1/1 [00:00<00:00,  7.25it/s]
      Finish loading all the .wav files in the path above!
      ==> Loading matericals costs time:0.32
      Do wav augumentation for list: 小迪小迪.list
      wav_list: 小迪小迪.list, list_len: 11, repeat_num:18, result_wav_num: 198
      Warning:the wave is longer than 7.6 seconds.Skip it. /root/workspace/alchemy/example_local/test_work_space/project/debug_modelk/corpus/train/wav/valle/小
迪小迪/ID0463W0386.wav
      ====== Aug list 小迪小迪.list done! ======
      ====== Write the tfr of the last batch =====
[Batch-end] Processing the 1/1 tfr of the cmd:小迪小迪
Write /root/workspace/alchemy/example_local/test_work_space/project/debug_modelk/corpus/train/tfr/merge/default/main/小迪小迪/小迪小迪_HamLogFbank_end_0.tfr.
===== All the tfr of 小迪小迪 has been written! =====
      write tfr done in dir:/root/workspace/alchemy/example_local/test_work_space/project/debug_modelk/corpus/train/tfr/merge/default/main/小迪小迪 .
      Do wav augumentation for list: 风扇暂停.list
      wav_list: 风扇暂停.list, list_len: 15, repeat_num:13, result_wav_num: 195
      Warning:the wave is longer than 7.6 seconds.Skip it. /root/workspace/alchemy/example_local/test_work_space/project/debug_modelk/corpus/train/wav/valle/风
扇暂停/ID0463W0386.wav
      ====== Aug list 风扇暂停.list done! ======
      ====== Write the tfr of the last batch =====
[Batch-end] Processing the 1/1 tfr of the cmd:风扇暂停
Write /root/workspace/alchemy/example_local/test_work_space/project/debug_modelk/corpus/train/tfr/merge/default/others/风扇暂停/风扇暂停_HamLogFbank_end_0.tfr.
===== All the tfr of 风扇暂停 has been written! =====
      write tfr done in dir:/root/workspace/alchemy/example_local/test_work_space/project/debug_modelk/corpus/train/tfr/merge/default/others/风扇暂停 .
      Do wav augumentation for list: 小奥小奥.list
      wav_list: 小奥小奥.list, list_len: 9, repeat_num:22, result_wav_num: 198
      Augmentaion-Processed:小奥小奥.list  [198/198]
      ====== Aug list 小奥小奥.list done! ======
      ====== Write the tfr of the last batch =====
[Batch-end] Processing the 1/1 tfr of the cmd:小奥小奥
```

 # 只进行模型训练，run.sh 设置如下：

```
project_conf=./prj_conf/prj_conf.yaml
conf_dir=./train_test_conf
#conf=$conf_dir/v0.1.1.yaml
#start_stage=1
#end_stage=2
#python -u auto_finetune.py --project_config $project_conf \
#           --strategy_config $conf \
#           --start_stage $start_stage \
#           --end_stage $end_stage

conf=$conf_dir/v0.2.1.yaml
start_stage=3 #2
end_stage=3 #2 #5
python -u auto_finetune.py --project_config $project_conf \
           --strategy_config $conf \
           --start_stage $start_stage \
           --end_stage $end_stage

conf=$conf_dir/v0.1.1.yaml
start_stage=3
end_stage=3 #5
python -u auto_finetune.py --project_config $project_conf \
           --strategy_config $conf \
           --start_stage $start_stage \
           --end_stage $end_stage

conf=$conf_dir/v0.3.1.yaml
start_stage=3
end_stage=3 #5
python -u auto_finetune.py --project_config $project_conf \
           --strategy_config $conf \
           --start_stage $start_stage \
           --end_stage $end_stage

conf=$conf_dir/v0.4.1.yaml
start_stage=3
end_stage=3 #5
python -u auto_finetune.py --project_config $project_conf \
           --strategy_config $conf \
           --start_stage $start_stage \
           --end_stage $end_stage
```

此部分注释掉

start_stage=3

end_stage=3

仅模型训练

模型训练部分示例输出如下：

```
### ============================================================ ###
### ==== stage 3:  model training !!! ==== ###
### ============================================================ ###
Tensor("Model/Reshape:0", shape=(128, 760, 40, 1), dtype=float32)
Tensor("Model_1/Reshape:0", shape=(128, 760, 40, 1), dtype=float32)
## Load special ckpt from ../alchemy/common/pre_train/model_k/common_asr-model
Learning rate assgin to 0.000200
## Write summary to  /root/workspace/alchemy/example_local/test_work_space/project/debug_modelk/version/v0.2.1/summary
Initial Step=1
============================begin training============================
== Epoch:0/4, step=100,time=50.2s,loss=58.077,train_ler=0.263,eval_ler=0.226
== step=100,time=8.258,loss=65.080, ctc_loss=18.466, l2_loss=1.475
           main_chuan_softmax_loss = 4.415, main_ctc_loss = 2.668, main_loss = 7.083
           other1_chuan_softmax_loss = 6.696, other1_ctc_loss = 0.054, other1_loss = 6.750
           other2_chuan_softmax_loss = 11.305, other2_ctc_loss = 3.969, other2_loss = 15.274
           asr_chuan_softmax_loss = 11.076, asr_ctc_loss = 19.007, asr_loss = 30.083
== Epoch:0/4, step=200,time=33.8s,loss=42.880,train_ler=0.206,eval_ler=0.311
== step=200,time=0.237,loss=60.387, ctc_loss=11.885, l2_loss=1.478
           main_chuan_softmax_loss = 4.437, main_ctc_loss = 6.680, main_loss = 11.117
           other1_chuan_softmax_loss = 10.918, other1_ctc_loss = 2.822, other1_loss = 13.740
           other2_chuan_softmax_loss = 6.552, other2_ctc_loss = 0.009, other2_loss = 6.561
           asr_chuan_softmax_loss = 10.879, asr_ctc_loss = 12.175, asr_loss = 23.054
== Epoch:0/4, step=300,time=26.1s,loss=39.143,train_ler=0.186,eval_ler=0.303
== step=300,time=0.214,loss=39.138, ctc_loss=5.593, l2_loss=1.484
           main_chuan_softmax_loss = 3.809, main_ctc_loss = 0.837, main_loss = 4.646
           other1_chuan_softmax_loss = 6.530, other1_ctc_loss = 0.002, other1_loss = 6.532
           other2_chuan_softmax_loss = 6.525, other2_ctc_loss = 0.001, other2_loss = 6.525
           asr_chuan_softmax_loss = 10.377, asr_ctc_loss = 5.764, asr_loss = 16.141
== Epoch:0/4, step=400,time=25.8s,loss=33.222,train_ler=0.137,eval_ler=0.303
== step=400,time=0.241,loss=33.950, ctc_loss=3.433, l2_loss=1.490
           main_chuan_softmax_loss = 3.062, main_ctc_loss = 0.187, main_loss = 3.249
           other1_chuan_softmax_loss = 6.524, other1_ctc_loss = 0.001, other1_loss = 6.525
           other2_chuan_softmax_loss = 6.522, other2_ctc_loss = 0.000, other2_loss = 6.522
           asr_chuan_softmax_loss = 9.560, asr_ctc_loss = 3.541, asr_loss = 13.102
== Epoch:0/4, step=500,time=26.3s,loss=30.799,train_ler=0.132,eval_ler=0.352
== step=500,time=0.217,loss=56.609, ctc_loss=1.978, l2_loss=1.499
           main_chuan_softmax_loss = 6.020, main_ctc_loss = 12.910, main_loss = 18.930
           other1_chuan_softmax_loss = 10.754, other1_ctc_loss = 2.529, other1_loss = 13.283
           other2_chuan_softmax_loss = 6.522, other2_ctc_loss = 0.000, other2_loss = 6.522
           asr_chuan_softmax_loss = 8.475, asr_ctc_loss = 1.880, asr_loss = 10.355
!!! Model saved in file: /root/workspace/alchemy/example_local/test_work_space/project/debug_modelk/version/v0.2.1/result/ori/ckpt/model_k

== Epoch:0/4, step=600,time=31.2s,loss=29.085,train_ler=0.168,eval_ler=0.479
== step=600,time=0.210,loss=29.086, ctc_loss=0.984, l2_loss=1.509
           main_chuan_softmax_loss = 2.766, main_ctc_loss = 0.000, main_loss = 2.766
           other1_chuan_softmax_loss = 6.523, other1_ctc_loss = 0.000, other1_loss = 6.523
```

 # 只进行模型推理测试，run.sh 设置如下：

```
16
17 ### mkdir prj and get wav,tfr etc.
18 project_conf=./prj_conf/prj_conf.yaml
19 conf_dir=./train_test_conf
20 #conf=$conf_dir/v0.1.1.yaml
21 #start_stage=1
22 #end_stage=2
23 #python -u auto_finetune.py --project_config $project_conf \
24 #           --strategy_config $conf \
25 #           --start_stage $start_stage \
26 #           --end_stage $end_stage
27
28 conf=$conf_dir/v0.2.1.yaml
29 start_stage=4 #3 #2
30 end_stage=5
31 python -u auto_finetune.py --project_config $project_conf \
32            --strategy_config $conf \
33            --start_stage $start_stage \
34            --end_stage $end_stage
35
36 conf=$conf_dir/v0.1.1.yaml
37 start_stage=4
38 end_stage=5
39 python -u auto_finetune.py --project_config $project_conf \
40            --strategy_config $conf \
41            --start_stage $start_stage \
42            --end_stage $end_stage
43
44 conf=$conf_dir/v0.3.1.yaml
45 start_stage=4
46 end_stage=5
47 python -u auto_finetune.py --project_config $project_conf \
48            --strategy_config $conf \
49            --start_stage $start_stage \
50            --end_stage $end_stage
51
52 conf=$conf_dir/v0.4.1.yaml
53 start_stage=4
54 end_stage=5
55 python -u auto_finetune.py --project_config $project_conf \
56            --strategy_config $conf \
```

此部分注释掉

仅模型推理测试

start_stage=4

end_stage=5

模型推理部分示例输出如下：

```
Compile OK.
### ========================================================== ###
### ==== stage 4:  model deploy !!! ==== ###
### ========================================================== ###
out avg ckpt is  /root/workspace/alchemy/example_local/test_work_space/project/debug_modelk/version/v0.2.1/result/ori/ckpt/average.ckpt
    ==> check apus model_apus.h at /root/workspace/alchemy/example_local/test_work_space/project/debug_modelk/version/v0.2.1/result/ori/x_deploy/default/
neg:2 wavs, pos:96 wavs
================================================
    ==> Test the ckpt dir: /root/workspace/alchemy/example_local/test_work_space/project/debug_modelk/version/v0.2.1/result/ori/ckpt
Memory allocation info:
Mem0(ops): 0
Mem1(data): 69120
Mem2(instruction): 9060
Mem3(in): 5264
Mem4(out): 4988
Mem5(tmp content): 30718
Mem6(weights): 487382
Total NPU Size (Mem0+Mem1+Mem2+Mem5+Mem6): 596280
Total Memory Size: 606532
----------------------
Compile OK.
    check grus model.h at /root/workspace/alchemy/example_local/test_work_space/project/debug_modelk/version/v0.2.1/result/ori/x_deploy/default
model_checkpoint_path: "/root/workspace/alchemy/example_local/test_work_space/project/debug_modelk/version/v0.2.1/result/ori/ckpt/average.ckpt"
    == Calculate neg phone prob !
100%|| 9996/9996 [00:29<00:00, 338.00it/s]
      == Neg: Calculate the CTC-Score
    neg inference done !! cost time:0.02 min
    == Calculate pos phone prob !
100%|| 2677/2677 [00:05<00:00, 476.07it/s]
      == Pos: Calculate the CTC-Score for the words contained within the wav file itself.
    pos inference done !! cost time:0.02 min
  ==> stage 4: model deploying cost 157.44 s
### ========================================================== ###
### ==== stage 5:  model decode and gen reports !!! ==== ###
### ========================================================== ###
neg:2 wavs, pos:96 wavs
    ==> in search thr part, score-len is 150, stride = 6, skip_win= 25
================================================================
================== Searching the THR of the CMDs.= ==================
================================================================
    ==> saved in path: /root/workspace/alchemy/example_local/test_work_space/project/debug_modelk/version/v0.2.1/result/ori/x_excel/default/Debug_modelk_
default_2025_07_17_model_k_阈值表.xlsx
neg:2 wavs, pos:96 wavs
write the probs into a temp file to share the memeory
    ==> avg_list:['average', 61.663, 0.979, 0.0, '', '', 0.937, 0.0, '', '', 0.964, 0.0, '', '', 0.971, 0.0, '', '', 0.717, 0.0, '', '', 0.958, 0.022, ''
.78, 0.0, '', '', 0.864, 0.007, '', '', 0.912, 0.036, '', '', 0.978, 0.0, '', '', 0.959, 0.007, '', '', 0.413, 0.0, '', '', 0.869, 0.006, '', 6.25]
================================================================
========================== FNR: done. ==========================
```

模型测试报告生成部分示例输出如下：

```
### ========================================================== ###
### ==== stage 5:  model decode and gen reports !!! ==== ###
### ========================================================== ###
neg:2 wavs, pos:96 wavs
    ==> in search thr part, score-len is 150, stride = 6, skip_win= 25
================================================================
================== Searching the THR of the CMDs.= ==================
================================================================
    ==> saved in path: /root/workspace/alchemy/example_local/test_work_space/project/debug_modelk/version/v0.2.1/result/ori/x_excel/default/Debug_modelk_v0.2.1_
default_2025_07_17_model_k_阈值表.xlsx
neg:2 wavs, pos:96 wavs
write the probs into a temp file to share the memeory
    ==> avg_list:['average', 61.663, 0.979, 0.0, '', '', 0.937, 0.0, '', '', 0.964, 0.0, '', '', 0.971, 0.0, '', '', 0.717, 0.0, '', '', 0.958, 0.022, '', '', 0
.78, 0.0, '', '', 0.864, 0.007, '', '', 0.912, 0.036, '', '', 0.978, 0.0, '', '', 0.959, 0.007, '', '', 0.413, 0.0, '', '', 0.869, 0.006, '', 6.25]
================================================================
```

5. 运行结束

Alchemy 运行完成之后，会输出几份模型和测试报告，根据测试报告结果，挑出你最认可的模型即可

- **注意：报告输出路径，会在运行结束时候打印出来，报告有打印输出，也有详细的 xlsx 表格根据需要获取即可**

输出文件说明

模型文件:

- 情况一、选择 model_b 版本训练
  - 该版本仅支持 GX8002b 芯片，模型输出：model.h
- 情况二、选择 model_c 版本训练
  - 该版本仅支持 GX8008C 芯片，模型输出：model.pb
- 情况三、选择 model_g 版本训练
  - 该版本仅支持 GX8002D 芯片，模型输出：model.h
- 情况四、选择 model_f 版本训练
  - 该版本支持 GX8003 芯片，模型输出文件：model_h

- 该版本支持 GX8005/GX8006 芯片，模型输出文件： model_fornax.h(模型)，mean_std.txt (归一化参数)

- 情况五、选择 model_k 版本训练
  - 该版本仅支持 GX8302B 芯片，模型输出文件：model_grus.h, model_apus.h(模型)，mean_std.txt (归一化参数)

用于部署的文件都在项目输出路径/version/版本号/result/ori/x_deploy/default 路径下。

测试报告:

实战示例中会输出 4 份测试报告，分别对应 4 个版本的模型

```
/home/pub/workspace/kws/dryer_prj/version/v0.1.1/result/ori/x_excel/default/*.xlsx
/home/pub/workspace/kws/dryer_prj/version/v0.2.1/result/ori/x_excel/default/*.xlsx
/home/pub/workspace/kws/dryer_prj/version/v0.3.1/result/ori/x_excel/default/*.xlsx
/home/pub/workspace/kws/dryer_prj/version/v0.4.1/result/ori/x_excel/default/*.xlsx
```