



人工智能实验室

Bunkws 中文使用文档

文档版本 V1.7

发布日期 2025-08-28

目录

目录.....	1
版本历史.....	1
第 1 章 Bunkws 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 Bunkws 安装及环境配置.....	4
步骤二、安装显卡驱动.....	5
步骤三、解压 Bunkws 工程.....	6
步骤四、安装及迁移工作环境.....	6
第 3 章 Bunkws 工程说明.....	8
目录结构.....	8
工程示例.....	8
Bunkws 训练一个新的项目教程.....	9
步骤一、新建项目工程.....	9
步骤二、配置数据存储路径.....	9
步骤三、创建需要的唤醒词、指令词.....	10
步骤四、运行 Bunkws 训练目标模型，检查报告输出.....	10
输出文件说明.....	11
如何使用自有指令词数据，进行训练.....	12

版本历史

版本	修改日期	修改人员	修改描述
v1.0	2025-03-26	chenxie	初始发布版本
v1.1	2025-03-31	chenxie	添加中文 120KB 模型的例子；解决单个模型训练内存占用过大导致无法同时训练两个模型的问题
v1.2	2025-04-10	chenxie	将 <code>anaconda</code> 改成 <code>miniconda</code> ；修复 bug
v1.3	2025-04-28	chenxie	增加用户自有语料训练流程
v1.4	2025-05-19	chenxie	将 <code>anaconda</code> 改成 <code>miniforge3</code> ；增加 GX8005/GX8006 系列芯片支持
V1.5	2025-06-30	chenxie	新增 8301A 低功耗蓝牙芯片模型支持；减少负样本训练数据，加快训练速度；优化模型编译选项；新增英文预训练工程 v0.1.1 版本
V1.6	2025-08-05	chenxie	新增语料生成工具 <code>gxtts_v3</code>
V1.7	2025-08-28	chenxie	优化 <code>gxtts_v3</code> 速度；解决 bug

第 1 章 Bunkws 介绍

简介

Bunkws 是杭州国芯微自研的一款端到端唤醒词训练框架，其独创的从"数据生成->模型训练->测试报告输出"全链路自动化框架，让零算法基础的开发者也能完成从训练到芯片级模型落地，开启智能硬件开发的极简时代。

核心特性

- 端到端唤醒词训练框架
- 支持中文 && 英文两种语种的模型
- 超轻量级模型，NPU 编译器编译后模型大小约 150KB、300KB

支持芯片

- GX8002
- GX8005
- GX8006
- GX8301A

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. Bunkws 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 Bunkws 安装及环境配置

Bunkws 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:

<code>SenseVoice_py311.tar.gz</code>	<code>#python</code> 环境
<code>bunkws.tar.gz</code>	<code># Bunkws</code> 框架核心文件
<code>corpus.tar.gz</code>	<code># Bunkws</code> 语料库 (比较大)
<code>Miniforge3-Linux-x86_64.sh</code>	<code>#</code> 轻量级的 Python 环境管理工具安装脚本
<code>cosyvoice.gz</code>	<code># Python3</code> 环境
<code>py37_tf114.tar.gz</code>	<code># python3</code> 环境
<code>th_12.tar.gz</code>	<code># python3</code> 环境

- 下载命令

```
wget --ftp-user=test --ftp-password=test@1234 ftp://ftp.nationalchip.com/cosyvoice.gz .
wget --ftp-user=test --ftp-password=test@1234 ftp://ftp.nationalchip.com/SenseVoice_py311.tar.gz .
wget --ftp-user=test --ftp-password=test@1234 ftp://ftp.nationalchip.com/bunkws.tar.gz .
wget --ftp-user=test --ftp-password=test@1234 ftp://ftp.nationalchip.com/corpus.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/py37_tf114.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 Bunkws 工程

1. 解压主工程文件

```
tar xzvf bunkws.tar.gz -C ~
```

2. 解压 Bunkws 语料库

```
tar xzvf corpus.tar.gz -C ~
```

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 py37_tf114.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.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 `liushk` 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/liushk/miniforge3/bin/conda' 'shell.bash' 'hook' 2> /dev/null)"
if [ $? -eq 0 ]; then
    eval "$__conda_setup"
else
    if [ -f "/home/liushk/miniforge3/etc/profile.d/conda.sh" ]; then
        . "/home/liushk/miniforge3/etc/profile.d/conda.sh"
    else
        export PATH="/home/liushk/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 bunkws/envs/cache/* ~/.cache/  
#路径适配修改  
sed -i "s|/home/sundy|$HOME|g" ~/miniforge3/envs/py37_tf114/bin/pip  
sed -i "s|/home/sundy|$HOME|g" ~/miniforge3/envs/py37_tf114/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 py37_tf114  
cd bunkws/tools/logfbank  
python setup.py develop  
sudo apt install ffmpeg
```

第 3 章 Bunkws 工程说明

目录结构

主要工程目录介绍

```
liushk@SPD-R730xd:~/bunkws$ tree -L 1
├─ egs                #用户训练工程主目录
├─ envs
├─ local              #本地化脚本
├─ README.md
├─ tools              #核心工具链
└─ bunkws             #唤醒词引擎
```

工程示例

1. 中文项目：earphone

该项目下暂有两个版本，根据需要选取：

- zh_010 版本，用于低功耗 GX8002 芯片，编译后模型文件大小为 165KB 左右
- zh_011 版本，用于低功耗 GX8002 芯片，编译后模型文件大小为 120KB 左右；用于 8005/8006 芯片的模型，编译后模型文件大小为 200KB 左右
- zh_012 版本，用于低功耗 GX8301A 蓝牙芯片，编译后模型文件大小为 120KB 左右；用于低功耗 GX8002 芯片，编译后模型文件大小为 150KB 左右

2. 英文项目：hi_ella

该项目下有两个版本

- en_010 版本，用于低功耗 GX8002 芯片，编译后模型文件大小为 150KB ，
- en_011 版本，和 en_010 的唯一区别是**训练速度快**，大小差不多

Bunkws 训练一个新的项目教程

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/liushk
# 那么我的通用数据存储路径: /home/liushk/corpus

#进入示例模版工程
cd ~/bunkws/egs
cp -r earphone xxxx # 中文项目
# 或
cp -r hi_ella xxx # 英文项目
```

例如:

- 我本次训练的是一个中文项目, 名称我取名为 aiots, 根据工程示例 工程的解释, 我可以选择 zh_010 版本, 或者 zh_011 版本。
- 我本次项目空间大, 可以选择大一点模型, 那么我选择 zh_010 版本, 如下所有介绍均以该项目为例, 操作如下:

```
#进入示例模版工程
cd ~/bunkws/egs
cp -r earphone aiots # 中文项目
```

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 Bunkws 工程步骤中已经解压在 home 目录下, 那么我的配置路径方法如下:

```
# 我的 home 路径如下: /home/liushk
# 那么我的通用数据存储路径: /home/liushk/corpus
# 我选择该项目训练过程中产生的数据, 存储路径为 /home/liushk/corpus/kws/aiots
# 创建模型训练数据存储路径
mkdir -p /home/liushk/corpus/kws/aiots
cd ~/bunkws/egs/aiots/zh_010

# 配置路径
vim run.sh
#把该文件中如下内容修改成如下
corpus_dir=/home/liushk/corpus #通用语料路径
project_dir=/home/liushk/corpus/kws/aiots
```

步骤三、创建需要的唤醒词、指令词

在上述步骤中，我们创建了一个/home/liushk/corpus/kws/aiot 目录用于存储本项目的生成数据集过程中的语料，Bunkws 框架定义，在 project_dir 路径下，会自动读取 cmd.txt 内容作为唤醒词、指令词。**注意：英文指令词，多个单词之间需要以下划线连接，并且字母单词用小写**

创建方法如下：

```
# 我的 home 路径如下: /home/liushk
# 那么我的通用数据存储路径: /home/liushk/corpus
# 我选择该新项目训练过程中产生的数据，存储路径为 /home/liushk/corpus/kws/aiot
vim /home/liushk/corpus/kws/aiot/cmd.txt
#中文写入格式如下，一个指令、唤醒词一行，以换行分割，除唤醒词、指令词以外不要有其他内容
你好小树
打开灯光
关闭灯光
#英文写入格式如下，一个指令、唤醒词一行，以换行分割，除唤醒词、指令词以外不要有其他内容
hello_world
ok_google
play_next_song
```

步骤四、运行 Bunkws 训练目标模型，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)

```
./run.sh # 一键训练
```

2. 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

```
# 数据生成
./run.sh --stage -1 --stop_stage -1
# 数据准备
./run.sh --stage 0 --stop_stage 1
# 模型训练
./run.sh --stage 2 --stop_stage 2
# 模型测试和部署
./run.sh --stage 3 --stop_stage 3
```

3. 运行结束

Bunkws 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 zh_010 版本训练
 - 该版本支持 GX8002 芯片，模型较大，模型输出：model_grus.h
- 情况二、选择 zh_011 版本训练
 - 该版支持 GX8002 芯片，模型输出文件：model_grus.h
 - 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)
- 情况二、选择 zh_012 版本训练
 - 支持 GX8002 芯片，模型输出文件：model_grus.h
 - 支持 GX8301A 芯片，模型输出文件：model_apus.h(模型)，mean_std.txt (归一化参数)

测试报告:

report.xlsx

关键词列表:

keyword.txt

如何使用自有指令词数据，进行训练

- 自有数据质量要求

- 每条指令的 wav 要求:
 - ✓ 至少 300 人，每人 10 句，总数至少 3000 句
 - ✓ 男女比例 1: 1
 - ✓ 地域分布要均衡
 - ✓ 年龄段分布也要均衡，如特殊项目要倾斜，比如儿童玩具项目，那么儿童比例增加
 - ✓ 采样率为 16000
 - ✓ 单通道
 - ✓ int 16bit 或 float 32bit
- 数据尽量采用专业高保真麦克风录制，或手机录制，录制的软件不要过任何信号处理算法录制的数据，尽量不要有噪声(底噪也不要有)，信噪比大于 20dB

- 指令词数据的目录形式

假设当前指令词的父目录为 aito，指令词有两个: a 和 b，每个指令词有 2 条 wav，则目录形式为:

```
aito
├─ a
│   ├── 1.wav
│   └── 2.wav
└─ b
    ├── 1.wav
    └── 2.wav

2 directories, 4 files
```

- 如何在脚本里加入自有指令词数据:

假设在 earphone/zh_010 工程下, 添加自有数据, aito 的绝对路径: /home/liushk/aito

打开 egs/zh_010/run.sh 编辑如下字段

```
vi run.sh
corpus_type=hifi #说明语料类型(高保真)
hifi_corpus=/home/liushk/aito #添加语料路径
```

运行 run.sh

```
./run.sh # 运行训练模型, 和上述正常“一键训练”模型流程一致
```