

运行环境

win10 及以上、python3.7.5 及以上、TensorFlow2.1 及以上

注：其他运行依赖 python 包，请根据报错提示，执行 `pip install xxx` 进行安装。

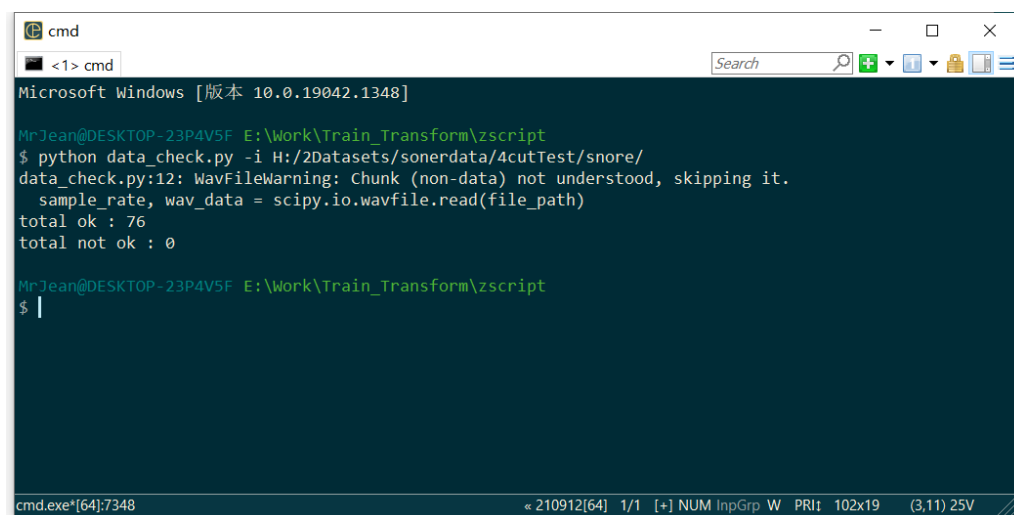
数据准备

1. 目前数据集包含 5 个类别

类别	说明
0	夜晚睡眠背景声，接近静音
1	重鼾声（音频最大值不小于 1600）
2	环境噪声
3	人的说话声
4	重鼾声（音频最大值小于 1600，大于 800）

2. 在向数据集中添加音频之前，使用脚本“data_check.py”先对要添加的音频数据检查其格式（单声道，16KHz，3.5s）是否正确。如下，检查某个目录下所有音频文件（切好的 3.5s 文件）是否满足格式要求：

示例：`python data_check.py -i H:/2Datasets/sonerdata/4cutTest/snore/`



```
cmd
<1> cmd
Microsoft Windows [版本 10.0.19042.1348]

MrJean@DESKTOP-23P4V5F E:\Work\Train_Transform\zscript
$ python data_check.py -i H:/2Datasets/sonerdata/4cutTest/snore/
data_check.py:12: WavFileWarning: Chunk (non-data) not understood, skipping it.
  sample_rate, wav_data = scipy.io.wavfile.read(file_path)
total ok : 76
total not ok : 0

MrJean@DESKTOP-23P4V5F E:\Work\Train_Transform\zscript
$ |
```

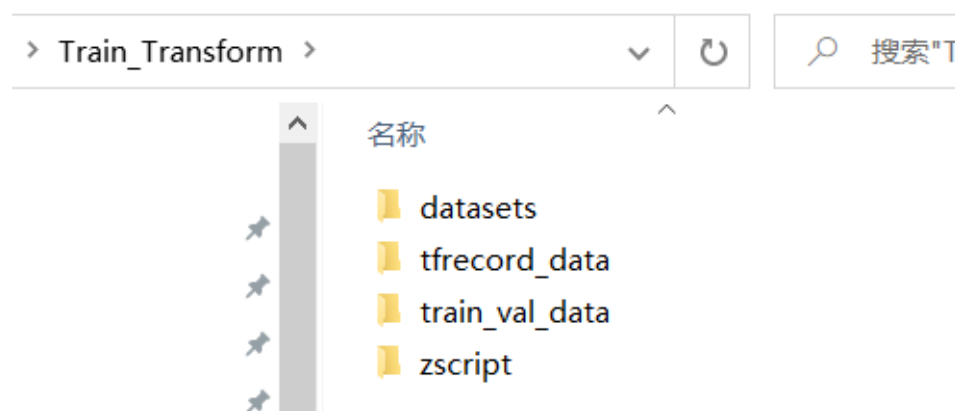
检查格式无误后，在按类别加入到数据集对应的文件夹内。

3. 生成模型训练用数据结构

调用脚本“data_process.py”生成数据标签文件，及训练用的数据结构。默认保存数据标签文件到目录“train_val_data”下，训练用的数据结构(tfreCORDs)保存在目录“tfrecord_data”下。

示例：python data_process.py

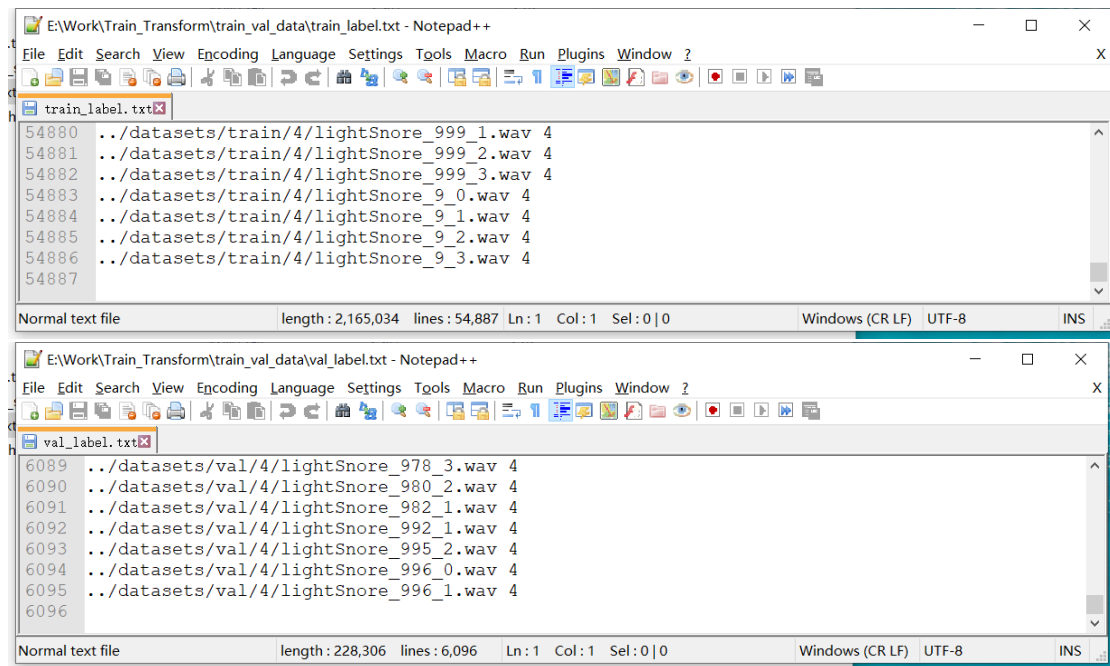
```
MrJean@DESKTOP-23P4V5F E:\Work\Train_Transform\zscript
$ python data_process.py
2021-11-13 00:28:30.150991: I tensorflow/stream_executor/platform/default/dso_loader.cc:48] Successfully opened dynamic library cudart64_101.dll
('this file index of: ', 1)
('this file index of: ', 2)
('this file index of: ', 3)
('this file index of: ', 4)
('this file index of: ', 5)
('this file index of: ', 6)
('this file index of: ', 7)
('this file index of: ', 8)
('this file index of: ', 9)
('this file index of: ', 10)
('this file index of: ', 11)
('this file index of: ', 12)
('this file index of: ', 13)
('this file index of: ', 14)
('this file index of: ', 15)
('this file index of: ', 16)
('this file index of: ', 17)
('this file index of: ', 18)
('this file index of: ', 19)
('this file index of: ', 20)
('this file index of: ', 21)
('this file index of: ', 22)
('this file index of: ', 23)
('this file index of: ', 24)
('this file index of: ', 25)
('this file index of: ', 26)
```



运行完成后，检查文件夹“tfrecord_data”下是否存在“train.tfreCORDs”和“val.tfreCORDs”这两个文件，确保此步骤正确生成，后续训练需要使用。

模型训练

调用脚本“tftrain.py”进行模型的训练，需要指定参数训练用样本个数(tn)和验证用样本个数(vn)。训练用样本个数可打开目录“train_val_data”下的“train_label.txt 进行查看，验证用样本个数可打开目录“train_val_data”下的“val_label.txt 进行查看。如下：



训练用样本个数(tn)和验证用样本个数(vn)分别为：54886 和 6095。

示例：python tftrain.py -tn 54886 -vn 6095

```
cmd - python tftrain.py -tn 54886 -vn 6095

Microsoft Windows [版本 10.0.19042.1348]

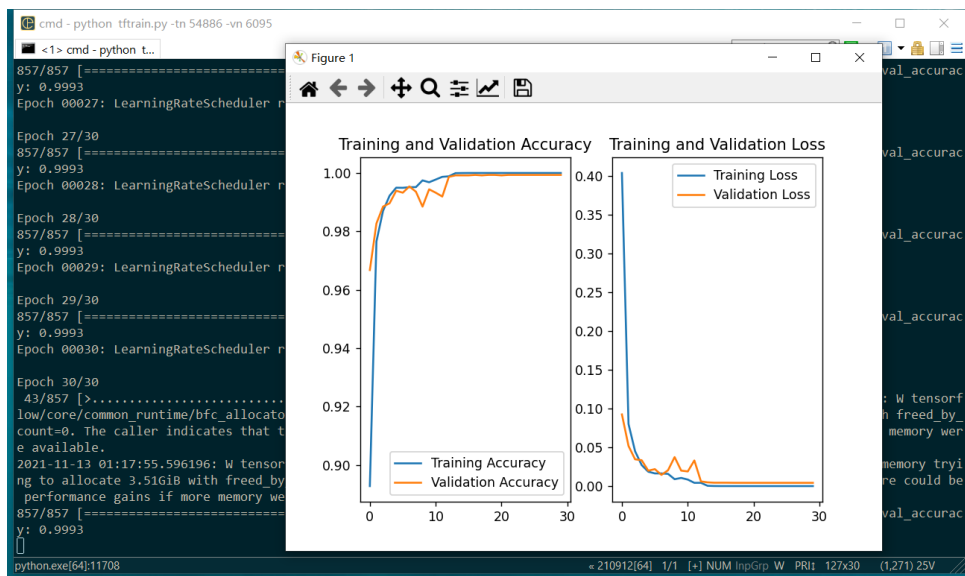
MrJean@DESKTOP-23P4V5F E:\Work\Train_Transform\zscript
$ python tftrain.py -tn 54886 -vn 6095
2021-11-13 00:40:14.761954: I tensorflow/stream_executor/platform/default/dso_loader.cc:48] Successfully op
ened dynamic library cudart64_101.dll
2021-11-13 00:40:16.783018: I tensorflow/stream_executor/platform/default/dso_loader.cc:48] Successfully op
ened dynamic library nvcuda.dll
2021-11-13 00:40:16.810405: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1716] Found device 0 with pr
operties:
pciBusID: 0000:01:00.0 name: GeForce GTX 1650 computeCapability: 7.5
coreClock: 1.56GHz coreCount: 16 deviceMemorySize: 4.00GiB deviceMemoryBandwidth: 119.24GiB/s
2021-11-13 00:40:16.811503: I tensorflow/stream_executor/platform/default/dso_loader.cc:48] Successfully op
ened dynamic library cudart64_101.dll

Dumped tool data for input_pipeline.pb to ..\logs\20211113-004016\train\plugins\profile\2021_11_12_16_40_26\DESK
TOP-23P4V5F.input_pipeline.pb
Dumped tool data for tensorflow_stats.pb to ..\logs\20211113-004016\train\plugins\profile\2021_11_12_16_40_26\DESK
TOP-23P4V5F.tensorflow_stats.pb
Dumped tool data for kernel_stats.pb to ..\logs\20211113-004016\train\plugins\profile\2021_11_12_16_40_26\DESK
TOP-23P4V5F.kernel_stats.pb

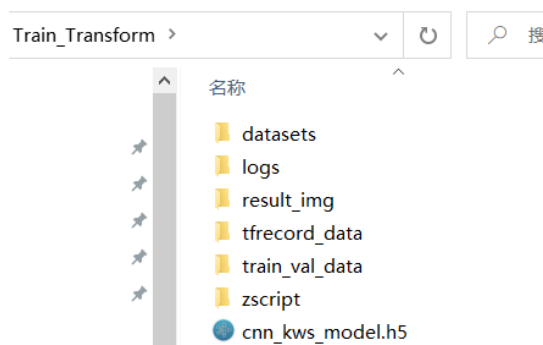
2/857 [.....] - ETA: 1:05 - loss: 9.6310 - accuracy: 0.2891WARNING:tensorflow:Callba
cks method `on_train_batch_end` is slow compared to the batch time (batch time: 0.0270s vs `on_train_batch_end`
time: 0.1253s). Check your callbacks.
534/857 [=====>.....] - ETA: 24s - loss: 0.5248 - accuracy: 0.8599

python.exe[64]:12028
```

训练完成后会显示出训练过程中准确率和损失的变化情况，一般来说准确率会逐渐提高，而损失会逐渐降低。如下所示：



点击关闭这两个图，先不要之间关闭命令行窗口。检查“Train_Transform”文件夹下是否生成了模型文件“cnn_kws_model.h5”，若已生成，则训练完成。

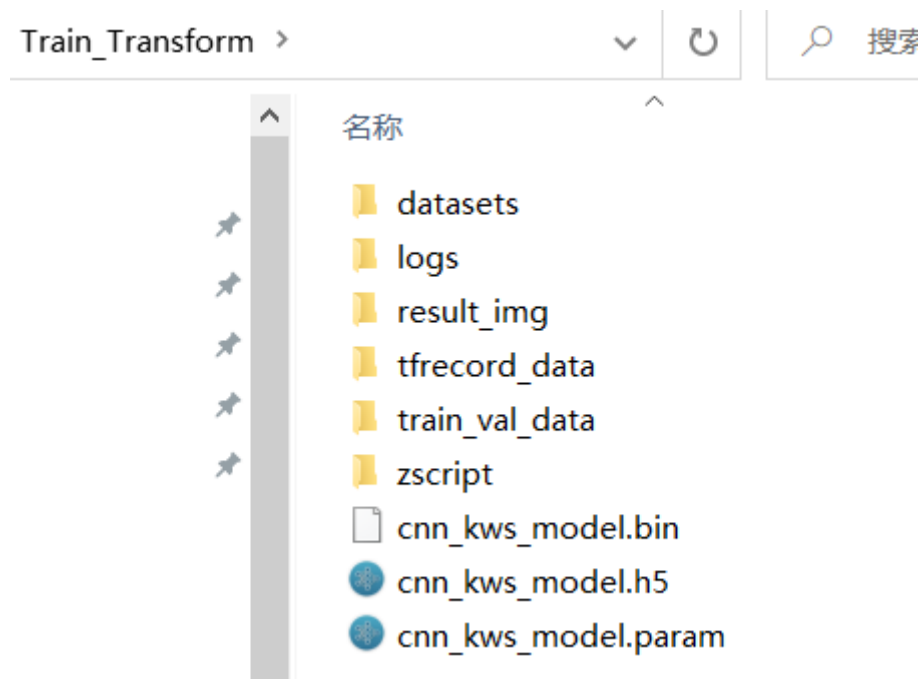


模型转换

调用脚本“tf2ncnn.py”，需要给定两个参数：-i 原模型路径，-o 转换后模型保存的目录。

示例：python tf2ncnn.py -i ../cnn_kws_model.h5 -o ../

```
MrJean@DESKTOP-23P4V5F E:\Work\Train_Transform\zscript
$ python tf2ncnn.py -i ../cnn_kws_model.h5 -o ../
Reading and parsing keras h5df file...
Start graph optimizing pass...
    Removing unused nodes...
    Removing squeeze reshape after pooling...
    Refreshing graph...
Converting keras graph to ncnn graph...
Start emitting to ncnn files.
    Emitting param...
    Emitting binary...
Done!
```



模型测试

按照之前给的测试代码进行测试即可。