

Retrieval-based-Voice-Conversion-WebUI / infer-web.py

...

 **github-actions[bot]** chore(format): run black on main (#1851) 6061f63 · last year 

1619 lines (1539...

```
1  import os
2  import sys
3  from dotenv import load_dotenv
4
5  now_dir = os.getcwd()
6  sys.path.append(now_dir)
7  load_dotenv()
8  from infer.modules.vc.modules import VC
9  from infer.modules.uvr5.modules import uvr
10 from infer.lib.train.process_ckpt import (
11     change_info,
12     extract_small_model,
13     merge,
14     show_info,
15 )
16 from i18n.i18n import I18nAuto
17 from configs.config import Config
18 from sklearn.cluster import MiniBatchKMeans
19 import torch, platform
20 import numpy as np
21 import gradio as gr
22 import faiss
23 import fairseq
24 import pathlib
25 import json
26 from time import sleep
27 from subprocess import Popen
28 from random import shuffle
29 import warnings
30 import traceback
31 import threading
32 import shutil
33 import logging
34
35
36 logging.getLogger("numba").setLevel(logging.WARNING)
37 logging.getLogger("librosa").setLevel(logging.WARNING)
```

```
37 logging.getLogger("httpx").setLevel(logging.WARNING)
38
39 logger = logging.getLogger(__name__)
40
41 tmp = os.path.join(now_dir, "TEMP")
42 shutil.rmtree(tmp, ignore_errors=True)
43 shutil.rmtree("%s/runtime/Lib/site-packages/infer_pack" % (now_dir), ignore_errors=True)
44 shutil.rmtree("%s/runtime/Lib/site-packages/uvr5_pack" % (now_dir), ignore_errors=True)
45 os.makedirs(tmp, exist_ok=True)
46 os.makedirs(os.path.join(now_dir, "logs"), exist_ok=True)
47 os.makedirs(os.path.join(now_dir, "assets/weights"), exist_ok=True)
48 os.environ["TEMP"] = tmp
49 warnings.filterwarnings("ignore")
50 torch.manual_seed(114514)
51
52
53 config = Config()
54 vc = VC(config)
55
56
57 if config.dml == True:
58
59     def forward_dml(ctx, x, scale):
60         ctx.scale = scale
61         res = x.clone().detach()
62         return res
63
64     fairseq.modules.grad_multiply.GradMultiply.forward = forward_dml
65 i18n = I18nAuto()
66 logger.info(i18n)
67 # 判断是否有能用来训练和加速推理的N卡
68 ngpu = torch.cuda.device_count()
69 gpu_infos = []
70 mem = []
71 if_gpu_ok = False
72
73 if torch.cuda.is_available() or ngpu != 0:
74     for i in range(ngpu):
75         gpu_name = torch.cuda.get_device_name(i)
```



```

1257         choices=["pm", "harvest", "dio", "rmvpe", "rmvpe_gpu"],
1258         value="rmvpe_gpu",
1259         interactive=True,
1260     )
1261     gpus_rmvpe = gr.Textbox(
1262         label=i18n(
1263             "rmvpe卡号配置：以-分隔输入使用的不同进程卡号,例如0-0-1使用在卡0上跑2个进程和卡1上跑1个进程"
1264         ),
1265         value="%s-%s" % (gpus, gpus),
1266         interactive=True,
1267         visible=F0GPUVisible,
1268     )
1269     but2 = gr.Button(i18n("特征提取"), variant="primary")
1270     info2 = gr.Textbox(label=i18n("输出信息"), value="", max_lines=8)
1271     f0method8.change(
1272         fn=change_f0_method,
1273         inputs=[f0method8],
1274         outputs=[gpus_rmvpe],
1275     )
1276     but2.click(
1277         extract_f0_feature,
1278         [
1279             gpus6,
1280             np7,
1281             f0method8,
1282             if_f0_3,
1283             exp_dir1,
1284             version19,
1285             gpus_rmvpe,
1286         ],
1287         [info2],
1288         api_name="train_extract_f0_feature",
1289     )
1290     with gr.Group():
1291         gr.Markdown(value=i18n("step3: 填写训练设置, 开始训练模型和索引"))
1292         with gr.Row():
1293             save_epoch10 = gr.Slider(
1294                 minimum=1,
1295                 maximum=50,
1296                 step=1,
1297                 label=i18n("保存频率save_every_epoch"),
1298                 value=5,
1299                 interactive=True,
1300             )
1301             total_epoch11 = gr.Slider(
1302                 minimum=2,
1303                 maximum=1000,
1304                 step=1

```

```

1304         step=1,
1305         label=i18n("总训练轮数total_epoch"),
1306         value=20,
1307         interactive=True,
1308     )
1309     batch_size12 = gr.Slider(
1310         minimum=1,
1311         maximum=40,
1312         step=1,
1313         label=i18n("每张显卡的batch_size"),
1314         value=default_batch_size,
1315         interactive=True,
1316     )
1317     if_save_latest13 = gr.Radio(
1318         label=i18n("是否仅保存最新的ckpt文件以节省硬盘空间"),
1319         choices=[i18n("是"), i18n("否")],
1320         value=i18n("否"),
1321         interactive=True,
1322     )
1323     if_cache_gpu17 = gr.Radio(
1324         label=i18n(
1325             "是否缓存所有训练集至显存。10min以下小数据可缓存以加速训练，大数据缓存会炸显存也
1326         ),
1327         choices=[i18n("是"), i18n("否")],
1328         value=i18n("否"),
1329         interactive=True,
1330     )
1331     if_save_every_weights18 = gr.Radio(
1332         label=i18n(
1333             "是否在每次保存时间点将最终小模型保存至weights文件夹"
1334         ),
1335         choices=[i18n("是"), i18n("否")],
1336         value=i18n("否"),
1337         interactive=True,
1338     )
1339     with gr.Row():
1340         pretrained_G14 = gr.Textbox(
1341             label=i18n("加载预训练底模G路径"),
1342             value="assets/pretrained_v2/f0G40k.pth",
1343             interactive=True,
1344         )
1345         pretrained_D15 = gr.Textbox(
1346             label=i18n("加载预训练底模D路径"),
1347             value="assets/pretrained_v2/f0D40k.pth",
1348             interactive=True,
1349         )
1350     sr2.change(
1351         change_sr2,
1352         [sr2, if_f0_3, version19],
1353         [pretrained_G14, pretrained_D15],
1354     )
1355     version19.change(
1356         change_version19,
1357         [
1358             sr2, if_f0_3, version19,
1359             pretrained_G14, pretrained_D15,
1360         ],
1361     )

```

Code

Blame

Raw



```
1362         [if_f0_3, sr2, version19],
1363         [f0method8, gpus_rmvpe, pretrained_G14, pretrained_D15],
1364     )
1365     gpus16 = gr.Textbox(
1366         label=i18n(
1367             "以-分隔输入使用的卡号，例如    0-1-2    使用卡0和卡1和卡2"
1368         ),
1369         value=gpus,
1370         interactive=True,
1371     )
1372     but3 = gr.Button(i18n("训练模型"), variant="primary")
1373     but4 = gr.Button(i18n("训练特征索引"), variant="primary")
1374     but5 = gr.Button(i18n("一键训练"), variant="primary")
1375     info3 = gr.Textbox(label=i18n("输出信息"), value="", max_lines=10)
1376     but3.click(
1377         click_train,
1378         [
1379             exp_dir1,
1380             sr2,
1381             if_f0_3,
1382             spk_id5,
1383             save_epoch10,
1384             total_epoch11,
1385             batch_size12,
1386             if_save_latest13,
1387             pretrained_G14,
1388             pretrained_D15,
1389             gpus16,
1390             if_cache_gpu17,
1391             if_save_every_weights18,
1392             version19,
1393         ],
1394         info3,
1395         api_name="train_start",
1396     )
1397     but4.click(train_index, [exp_dir1, version19], info3)
1398     but5.click(
1399         train1key,
1400         [
1401             exp_dir1,
1402             sr2,
1403             if_f0_3,
1404             trainset_dir4,
1405             spk_id5,
1406             np7,
1407             f0method8,
1408             save_epoch10,
1409             total_epoch11,
```

```
1410         batch_size12,  
1411         if_save_latest13,  
1412         pretrained_G14,  
1413         pretrained_D15,  
1414         gpus16,  
1415         if_cache_gpu17,  
1416         if_save_every_weights18,  
1417         version19,  
1418         gpus_rmvp,  
1419     ],  
1420     info3,  
1421     api_name="train_start_all",  
1422 )  
1423  
1424 with gr.TabItem(i18n("ckpt处理")):  
1425     with gr.Group():  
1426         gr.Markdown(value=i18n("模型融合, 可用于测试音色融合"))  
1427         with gr.Row():  
1428             ckpt_a = gr.Textbox(  
1429                 label=i18n("A模型路径"), value="", interactive=True  
1430             )  
1431             ckpt_b = gr.Textbox(  
1432                 label=i18n("B模型路径"), value="", interactive=True  
1433             )  
1434             alpha_a = gr.Slider(  
1435                 minimum=0,  
1436                 maximum=1,  
1437                 label=i18n("A模型权重"),  
1438                 value=0.5,  
1439                 interactive=True,  
1440             )  
1441         with gr.Row():  
1442             sr_ = gr.Radio(  
1443                 label=i18n("目标采样率"),  
1444                 choices=["40k", "48k"],  
1445                 value="40k",  
1446                 interactive=True,  
1447             )  
1448             if_f0_ = gr.Radio(  
1449                 label=i18n("模型是否带音高指导"),  
1450                 choices=[i18n("是"), i18n("否")],  
1451                 value=i18n("是"),  
1452                 interactive=True,  
1453             )  
1454             info__ = gr.Textbox(  
1455                 label=i18n("要置入的模型信息"),  
1456                 value="",  
1457                 max_lines=8,  
1458                 interactive=True,  
1459             )  
1460             name_to_save0 = gr.Textbox(  
1461                 label=i18n("保存的模型名不带后缀"),  
1462                 value="",
```

```

1463         max_lines=1,
1464         interactive=True,
1465     )
1466     version_2 = gr.Radio(
1467         label=i18n("模型版本型号"),
1468         choices=["v1", "v2"],
1469         value="v1",
1470         interactive=True,
1471     )
1472     with gr.Row():
1473         but6 = gr.Button(i18n("融合"), variant="primary")
1474         info4 = gr.Textbox(label=i18n("输出信息"), value="", max_lines=8)
1475     but6.click(
1476         merge,
1477         [
1478             ckpt_a,
1479             ckpt_b,
1480             alpha_a,
1481             sr_,
1482             if_f0_,
1483             info_,
1484             name_to_save0,
1485             version_2,
1486         ],
1487         info4,
1488         api_name="ckpt_merge",
1489     ) # def merge(path1,path2,alpha1,sr,f0,info):
1490     with gr.Group():
1491         gr.Markdown(
1492             value=i18n("修改模型信息(仅支持weights文件夹下提取的小模型文件)")
1493         )
1494     with gr.Row():
1495         ckpt_path0 = gr.Textbox(
1496             label=i18n("模型路径"), value="", interactive=True
1497         )
1498         info_ = gr.Textbox(
1499             label=i18n("要改的模型信息"),
1500             value="",
1501             max_lines=8,
1502             interactive=True,
1503         )
1504         name_to_save1 = gr.Textbox(
1505             label=i18n("保存的文件名，默认空为和源文件同名"),
1506             value="",
1507             max_lines=8,
1508             interactive=True,
1509         )

```

```
1546         label=i18n("目标采样率"),
1547         choices=["32k", "40k", "48k"],
1548         value="40k",
1549         interactive=True,
1550     )
1551     if_f0__ = gr.Radio(
1552         label=i18n("模型是否带音高指导,1是0否"),
1553         choices=["1", "0"],
1554         value="1",
1555         interactive=True,
1556     )
1557     version_1 = gr.Radio(
1558         label=i18n("模型版本型号"),
1559         choices=["v1", "v2"],
1560         value="v2",
1561         interactive=True,
1562     )
1563     info__ = gr.Textbox(
1564         label=i18n("要置入的模型信息"),
1565         value="",
1566         max_lines=8,
1567         interactive=True,
1568     )
```

```

1568         but9 = gr.Button(i18n("提取"), variant="primary")
1569         info7 = gr.Textbox(label=i18n("输出信息"), value="", max_lines=8)
1570         ckpt_path2.change(
1571             change_info_, [ckpt_path2], [sr__, if_f0__, version_1]
1572         )
1573     but9.click(
1574         extract_small_model,
1575         [ckpt_path2, save_name, sr__, if_f0__, info____, version_1],
1576         info7,
1577         api_name="ckpt_extract",
1578     )
1579
1580 with gr.TabItem(i18n("Onnx导出")):
1581     with gr.Row():
1582         ckpt_dir = gr.Textbox(
1583             label=i18n("RVC模型路径"), value="", interactive=True
1584         )
1585     with gr.Row():
1586         onnx_dir = gr.Textbox(
1587             label=i18n("Onnx输出路径"), value="", interactive=True
1588         )
1589     with gr.Row():
1590         infoOnnx = gr.Label(label="info")
1591     with gr.Row():
1592         butOnnx = gr.Button(i18n("导出Onnx模型"), variant="primary")
1593         butOnnx.click(
1594             export_onnx, [ckpt_dir, onnx_dir], infoOnnx, api_name="export_onnx"
1595         )
1596
1597 tab_faq = i18n("常见问题解答")
1598 with gr.TabItem(tab_faq):
1599     try:
1600         if tab_faq == "常见问题解答":
1601             with open("docs/cn/faq.md", "r", encoding="utf8") as f:
1602                 info = f.read()
1603         else:
1604             with open("docs/en/faq_en.md", "r", encoding="utf8") as f:
1605                 info = f.read()
1606         gr.Markdown(value=info)
1607     except:
1608         gr.Markdown(traceback.format_exc())
1609
1610 if config.iscolab:
1611     app.queue(concurrency_count=511, max_size=1022).launch(share=True)
1612 else:
1613     app.queue(concurrency_count=511, max_size=1022).launch(
1614         server_name="0.0.0.0",
1615         inbrowser=not config.noautoopen,
1616         server_port=config.listen_port,
1617         quiet=True,
1618     )
1619

```

