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

1619 lines (1539…

```python
import os
import sys
from dotenv import load_dotenv

now_dir = os.getcwd()
sys.path.append(now_dir)
load_dotenv()
from infer.modules.vc.modules import VC
from infer.modules.uvr5.modules import uvr
from infer.lib.train.process_ckpt import (
    change_info,
    extract_small_model,
    merge,
    show_info,
)
from i18n.i18n import I18nAuto
from configs.config import Config
from sklearn.cluster import MiniBatchKMeans
import torch, platform
import numpy as np
import gradio as gr
import faiss
import fairseq
import pathlib
import json
from time import sleep
from subprocess import Popen
from random import shuffle
import warnings
import traceback
import threading
import shutil
import logging


logging.getLogger("numba").setLevel(logging.WARNING)
```

```python
logging.getLogger("httpx").setLevel(logging.WARNING)

logger = logging.getLogger(__name__)

tmp = os.path.join(now_dir, "TEMP")
shutil.rmtree(tmp, ignore_errors=True)
shutil.rmtree("%s/runtime/Lib/site-packages/infer_pack" % (now_dir), ignore_errors=True)
shutil.rmtree("%s/runtime/Lib/site-packages/uvr5_pack" % (now_dir), ignore_errors=True)
os.makedirs(tmp, exist_ok=True)
os.makedirs(os.path.join(now_dir, "logs"), exist_ok=True)
os.makedirs(os.path.join(now_dir, "assets/weights"), exist_ok=True)
os.environ["TEMP"] = tmp
warnings.filterwarnings("ignore")
torch.manual_seed(114514)


config = Config()
vc = VC(config)


if config.dml == True:

    def forward_dml(ctx, x, scale):
        ctx.scale = scale
        res = x.clone().detach()
        return res

    fairseq.modules.grad_multiply.GradMultiply.forward = forward_dml
i18n = I18nAuto()
logger.info(i18n)
# 判断是否有能用来训练和加速推理的N卡
ngpu = torch.cuda.device_count()
gpu_infos = []
mem = []
if_gpu_ok = False

if torch.cuda.is_available() or ngpu != 0:
    for i in range(ngpu):
        gpu_name = torch.cuda.get_device_name(i)
```

```
            f0_dir = "%s/2a_f0" % (exp_dir)
            f0nsf_dir = "%s/2b-f0nsf" % (exp_dir)
            names = (
                set([name.split(".")[0] for name in os.listdir(gt_wavs_dir)])
                & set([name.split(".")[0] for name in os.listdir(feature_dir)])
                & set([name.split(".")[0] for name in os.listdir(f0_dir)])
                & set([name.split(".")[0] for name in os.listdir(f0nsf_dir)])
            )
        else:
            names = set([name.split(".")[0] for name in os.listdir(gt_wavs_dir)]) & set(
                [name.split(".")[0] for name in os.listdir(feature_dir)]
            )
        opt = []
        for name in names:
            if if_f0_3:
                opt.append(
                    "%s/%s.wav|%s/%s.npy|%s/%s.wav.npy|%s/%s.wav.npy|%s"
                    % (
                        gt_wavs_dir.replace("\\", "\\\\"),
                        name,
                        feature_dir.replace("\\", "\\\\"),
                        name,
```

```
                    name,
                    f0_dir.replace("\\", "\\\\"),
                    name,
                    f0nsf_dir.replace("\\", "\\\\"),
                    name,
                    spk_id5,
                )
            )
        else:
            opt.append(
                "%s/%s.wav|%s/%s.npy|%s"
                % (
                    gt_wavs_dir.replace("\\", "\\\\"),
                    name,
                    feature_dir.replace("\\", "\\\\"),
                    name,
                    spk_id5,
                )
            )
    fea_dim = 256 if version19 == "v1" else 768
    if if_f0_3:
        for _ in range(2):
            opt.append(
                "%s/logs/mute/0_gt_wavs/mute%s.wav|%s/logs/mute/3_feature%s/mute.npy|%s/logs/mut
                % (now_dir, sr2, now_dir, fea_dim, now_dir, now_dir, spk_id5)
            )
    else:
        for _ in range(2):
            opt.append(
                "%s/logs/mute/0_gt_wavs/mute%s.wav|%s/logs/mute/3_feature%s/mute.npy|%s"
                % (now_dir, sr2, now_dir, fea_dim, spk_id5)
            )
    shuffle(opt)
    with open("%s/filelist.txt" % exp_dir, "w") as f:
        f.write("\n".join(opt))
    logger.debug("Write filelist done")
    # 生成config#无需生成config
    # cmd = python_cmd + " train_nsf_sim_cache_sid_load_pretrain.py -e mi-test -sr 40k -f0 1 -bs
    logger.info("Use gpus: %s", str(gpus16))
    if pretrained_G14 == "":
        logger.info("No pretrained Generator")
    if pretrained_D15 == "":
        logger.info("No pretrained Discriminator")
    if version19 == "v1" or sr2 == "40k":
        config_path = "v1/%s.json" % sr2
    else:
        config_path = "v2/%s.json" % sr2
    config_save_path = os.path.join(exp_dir, "config.json")
    if not pathlib.Path(config_save_path).exists():
        with open(config_save_path, "w", encoding="utf-8") as f:
            json.dump(
                config.json_config[config_path],
                f,
```

```
                    ensure_ascii=False,
                    indent=4,
                    sort_keys=True,
                )
                f.write("\n")
        if gpus16:
            cmd = (
                '"%s" infer/modules/train/train.py -e "%s" -sr %s -f0 %s -bs %s -g %s -te %s -se %s
                % (
                    config.python_cmd,
                    exp_dir1,
                    sr2,
                    1 if if_f0_3 else 0,
                    batch_size12,
                    gpus16,
                    total_epoch11,
                    save_epoch10,
                    "-pg %s" % pretrained_G14 if pretrained_G14 != "" else "",
                    "-pd %s" % pretrained_D15 if pretrained_D15 != "" else "",
                    1 if if_save_latest13 == i18n("是") else 0,
                    1 if if_cache_gpu17 == i18n("是") else 0,
                    1 if if_save_every_weights18 == i18n("是") else 0,
                    version19,
                )
            )
        else:
            cmd = (
```

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

↑ Top

Code Blame Raw

```
                    sr2,
                    1 if if_f0_3 else 0,
                    batch_size12,
                    total_epoch11,
                    save_epoch10,
                    "-pg %s" % pretrained_G14 if pretrained_G14 != "" else "",
                    "-pd %s" % pretrained_D15 if pretrained_D15 != "" else "",
                    1 if if_save_latest13 == i18n("是") else 0,
                    1 if if_cache_gpu17 == i18n("是") else 0,
                    1 if if_save_every_weights18 == i18n("是") else 0,
                    version19,
                )
            )
        logger.info("Execute: " + cmd)
        p = Popen(cmd, shell=True, cwd=now_dir)
        p.wait()
        return "训练结束, 您可查看控制台训练日志或实验文件夹下的train.log"


    # but4.click(train_index, [exp_dir1], info3)
    def train_index(exp_dir1, version19):
        # exp_dir = "%s/logs/%s" % (now_dir, exp_dir1)
```

```
        exp_dir = "logs/%s" % (exp_dir1)
        os.makedirs(exp_dir, exist_ok=True)
        feature_dir = (
            "%s/3_feature256" % (exp_dir)
            if version19 == "v1"
            else "%s/3_feature768" % (exp_dir)
        )
        if not os.path.exists(feature_dir):
            return "请先进行特征提取!"
        listdir_res = list(os.listdir(feature_dir))
        if len(listdir_res) == 0:
            return "请先进行特征提取！"
        infos = []
        npys = []
        for name in sorted(listdir_res):
            phone = np.load("%s/%s" % (feature_dir, name))
            npys.append(phone)
        big_npy = np.concatenate(npys, 0)
        big_npy_idx = np.arange(big_npy.shape[0])
        np.random.shuffle(big_npy_idx)
        big_npy = big_npy[big_npy_idx]
        if big_npy.shape[0] > 2e5:
            infos.append("Trying doing kmeans %s shape to 10k centers." % big_npy.shape[0])
            yield "\n".join(infos)
            try:
                big_npy = (
                    MiniBatchKMeans(
                        n_clusters=10000,
                        verbose=True,
                        batch_size=256 * config.n_cpu,
                        compute_labels=False,
                        init="random",
                    )
                    .fit(big_npy)
                    .cluster_centers_
                )
            except:
                info = traceback.format_exc()
                logger.info(info)
                infos.append(info)
                yield "\n".join(infos)

        np.save("%s/total_fea.npy" % exp_dir, big_npy)
        n_ivf = min(int(16 * np.sqrt(big_npy.shape[0])), big_npy.shape[0] // 39)
        infos.append("%s,%s" % (big_npy.shape, n_ivf))
        yield "\n".join(infos)
        index = faiss.index_factory(256 if version19 == "v1" else 768, "IVF%s,Flat" % n_ivf)
        # index = faiss.index_factory(256if version19=="v1"else 768, "IVF%s,PQ128x4fs,RFlat"%n_ivf)
        infos.append("training")
        yield "\n".join(infos)
        index_ivf = faiss.extract_index_ivf(index)  #
        index_ivf.nprobe = 1
        index.train(big_npy)
```

```
        faiss.write_index(
            index,
            "%s/trained_IVF%s_Flat_nprobe_%s_%s_%s.index"
            % (exp_dir, n_ivf, index_ivf.nprobe, exp_dir1, version19),
        )
        infos.append("adding")
        yield "\n".join(infos)
        batch_size_add = 8192
        for i in range(0, big_npy.shape[0], batch_size_add):
            index.add(big_npy[i : i + batch_size_add])
        faiss.write_index(
            index,
            "%s/added_IVF%s_Flat_nprobe_%s_%s_%s.index"
            % (exp_dir, n_ivf, index_ivf.nprobe, exp_dir1, version19),
        )
        infos.append(
            "成功构建索引 added_IVF%s_Flat_nprobe_%s_%s_%s.index"
            % (n_ivf, index_ivf.nprobe, exp_dir1, version19)
        )
        try:
            link = os.link if platform.system() == "Windows" else os.symlink
            link(
                "%s/added_IVF%s_Flat_nprobe_%s_%s_%s.index"
                % (exp_dir, n_ivf, index_ivf.nprobe, exp_dir1, version19),
                "%s/%s_IVF%s_Flat_nprobe_%s_%s_%s.index"
                % (
                    outside_index_root,
                    exp_dir1,
                    n_ivf,
                    index_ivf.nprobe,
                    exp_dir1,
                    version19,
                ),
            )
            infos.append("链接索引到外部-%s" % (outside_index_root))
        except:
            infos.append("链接索引到外部-%s失败" % (outside_index_root))

        # faiss.write_index(index, '%s/added_IVF%s_Flat_FastScan_%s.index'%(exp_dir,n_ivf,version19)
        # infos.append("成功构建索引，added_IVF%s_Flat_FastScan_%s.index"%(n_ivf,version19))
        yield "\n".join(infos)


# but5.click(train1key, [exp_dir1, sr2, if_f0_3, trainset_dir4, spk_id5, gpus6, np7, f0method8,
def train1key(
    exp_dir1,
    sr2,
    if_f0_3,
    trainset_dir4,
    spk_id5,
    np7,
    f0method8,
    save_epoch10,
```

```
        total_epoch11,
        batch_size12,
        if_save_latest13,
        pretrained_G14,
        pretrained_D15,
        gpus16,
        if_cache_gpu17,
        if_save_every_weights18,
        version19,
        gpus_rmvpe,
    ):
        infos = []

        def get_info_str(strr):
            infos.append(strr)
            return "\n".join(infos)

        # step1:处理数据
        yield get_info_str(i18n("step1:正在处理数据"))
        [get_info_str(_) for _ in preprocess_dataset(trainset_dir4, exp_dir1, sr2, np7)]
```

```
                    label=i18n("目标采样率"),
                    choices=["32k", "40k", "48k"],
                    value="40k",
                    interactive=True,
                )
                if_f0__ = gr.Radio(
                    label=i18n("模型是否带音高指导,1是0否"),
                    choices=["1", "0"],
                    value="1",
                    interactive=True,
                )
                version_1 = gr.Radio(
                    label=i18n("模型版本型号"),
                    choices=["v1", "v2"],
                    value="v2",
                    interactive=True,
                )
                info___ = gr.Textbox(
                    label=i18n("要置入的模型信息"),
                    value="",
                    max_lines=8,
                    interactive=True,
                )
```

```
                    )
                    but9 = gr.Button(i18n("提取"), variant="primary")
                    info7 = gr.Textbox(label=i18n("输出信息"), value="", max_lines=8)
                    ckpt_path2.change(
                        change_info_, [ckpt_path2], [sr__, if_f0__, version_1]
                    )
                but9.click(
                    extract_small_model,
                    [ckpt_path2, save_name, sr__, if_f0__, info___, version_1],
                    info7,
                    api_name="ckpt_extract",
                )

            with gr.TabItem(i18n("Onnx导出")):
                with gr.Row():
                    ckpt_dir = gr.Textbox(
                        label=i18n("RVC模型路径"), value="", interactive=True
                    )
                with gr.Row():
                    onnx_dir = gr.Textbox(
                        label=i18n("Onnx输出路径"), value="", interactive=True
                    )
                with gr.Row():
                    infoOnnx = gr.Label(label="info")
                with gr.Row():
                    butOnnx = gr.Button(i18n("导出Onnx模型"), variant="primary")
                butOnnx.click(
                    export_onnx, [ckpt_dir, onnx_dir], infoOnnx, api_name="export_onnx"
                )

            tab_faq = i18n("常见问题解答")
            with gr.TabItem(tab_faq):
                try:
                    if tab_faq == "常见问题解答":
                        with open("docs/cn/faq.md", "r", encoding="utf8") as f:
                            info = f.read()
                    else:
                        with open("docs/en/faq_en.md", "r", encoding="utf8") as f:
                            info = f.read()
                    gr.Markdown(value=info)
                except:
                    gr.Markdown(traceback.format_exc())

        if config.iscolab:
            app.queue(concurrency_count=511, max_size=1022).launch(share=True)
        else:
            app.queue(concurrency_count=511, max_size=1022).launch(
                server_name="0.0.0.0",
                inbrowser=not config.noautoopen,
                server_port=config.listen_port,
                quiet=True,
            )
```