Fine-tuning

Obviously, when you opened this page, you were not satisfied with the performance of the few-shot pre-trained model. You want to fine-tune a model to improve its performance on your dataset.

In current version, you only need to finetune the 'LLAMA' part.

Fine-tuning LLAMA

1. Prepare the dataset

```
.
├── SPK1
│   ├── 21.15-26.44.lab
│   ├── 21.15-26.44.mp3
│   ├── 27.51-29.98.lab
│   ├── 27.51-29.98.mp3
│   ├── 30.1-32.71.lab
│   └── 30.1-32.71.mp3
└── SPK2
    ├── 38.79-40.85.lab
    └── 38.79-40.85.mp3
```

You need to convert your dataset into the above format and place it under `data`. The audio file can have the extensions `.mp3`, `.wav`, or `.flac`, and the annotation file should have the extensions `.lab`.

> **Dataset Format**
>
> The `.lab` annotation file only needs to contain the transcription of the audio, with no special formatting required. For example, if `hi.mp3` says "Hello, goodbye," then the `hi.lab` file would contain a single line of text: "Hello, goodbye."

Warning

It's recommended to apply loudness normalization to the dataset. You can use fish-audio-preprocess to do this.

```
fap loudness-norm data-raw data --clean
```

2. Batch extraction of semantic tokens

Make sure you have downloaded the VQGAN weights. If not, run the following command:

```
huggingface-cli download fishaudio/fish-speech-1.5 --local-dir
checkpoints/fish-speech-1.5
```

You can then run the following command to extract semantic tokens:

```
python tools/vqgan/extract_vq.py data \
    --num-workers 1 --batch-size 16 \
    --config-name "firefly_gan_vq" \
    --checkpoint-path "checkpoints/fish-speech-1.5/firefly-gan-vq-fsq-8x1024-
21hz-generator.pth"
```

Note

You can adjust `--num-workers` and `--batch-size` to increase extraction speed, but please make sure not to exceed your GPU memory limit.
For the VITS format, you can specify a file list using `--filelist xxx.list`.

This command will create `.npy` files in the `data` directory, as shown below:

```
.
├── SPK1
│   ├── 21.15-26.44.lab
│   ├── 21.15-26.44.mp3
│   ├── 21.15-26.44.npy
│   ├── 27.51-29.98.lab
│   ├── 27.51-29.98.mp3
│   ├── 27.51-29.98.npy
│   ├── 30.1-32.71.lab
│   ├── 30.1-32.71.mp3
│   └── 30.1-32.71.npy
└── SPK2
    ├── 38.79-40.85.lab
    ├── 38.79-40.85.mp3
    └── 38.79-40.85.npy
```

3. Pack the dataset into protobuf

```
python tools/llama/build_dataset.py \
    --input "data" \
    --output "data/protos" \
    --text-extension .lab \
    --num-workers 16
```

After the command finishes executing, you should see the `quantized-dataset-ft.protos` file in the `data` directory.

4. Finally, fine-tuning with LoRA

Similarly, make sure you have downloaded the `LLAMA` weights. If not, run the following command:

```
huggingface-cli download fishaudio/fish-speech-1.5 --local-dir
checkpoints/fish-speech-1.5
```

Finally, you can start the fine-tuning by running the following command:

```
python fish_speech/train.py --config-name text2semantic_finetune \
    project=$project \
    +lora@model.model.lora_config=r_8_alpha_16
```

> **Note**
>
> You can modify the training parameters such as `batch_size`, `gradient_accumulation_steps`, etc. to fit your GPU memory by modifying `fish_speech/configs/text2semantic_finetune.yaml`.

> **Note**
>
> For Windows users, you can use `trainer.strategy.process_group_backend=gloo` to avoid `nccl` issues.

After training is complete, you can refer to the inference section to generate speech.

Info

By default, the model will only learn the speaker's speech patterns and not the timbre. You still need to use prompts to ensure timbre stability. If you want to learn the timbre, you can increase the number of training steps, but this may lead to overfitting.

After training, you need to convert the LoRA weights to regular weights before performing inference.

```
python tools/llama/merge_lora.py \
    --lora-config r_8_alpha_16 \
    --base-weight checkpoints/fish-speech-1.5 \
    --lora-weight results/$project/checkpoints/step_000000010.ckpt \
    --output checkpoints/fish-speech-1.5-yth-lora/
```

Note

You may also try other checkpoints. We suggest using the earliest checkpoint that meets your requirements, as they often perform better on out-of-distribution (OOD) data.