Speech Recognition Pre-Training

Wav2Vec2 Speech Pre-Training

The script `run_speech_wav2vec2_pretraining_no_trainer.py` can be used to pre-train a Wav2Vec2 model from scratch.

In the script `run_speech_wav2vec2_pretraining_no_trainer`, a Wav2Vec2 model is pre-trained on audio data alone using Wav2Vec2's contrastive loss objective.

The following examples show how to fine-tune a `"base"`-sized Wav2Vec2 model as well as a `"large"`-sized Wav2Vec2 model using `accelerate`.

NOTE 1

Wav2Vec2's pre-training is known to be quite unstable. It is advised to do a couple of test runs with a smaller dataset, *i.e.* `--dataset_config_names clean clean`, `--dataset_split_names validation test` to find good hyper-parameters for `learning_rate`, `batch_size`, `num_warmup_steps`, and the optimizer. A good metric to observe during training is the gradient norm which should ideally be between 0.5 and 2.

NOTE 2

When training a model on large datasets it is recommended to run the data preprocessing in a first run in a **non-distributed** mode via `--preprocessing_only` so that when running the model in **distributed** mode in a second step the preprocessed data can easily be loaded on each distributed device.

Demo

In this demo run we pre-train a `"base-sized"` Wav2Vec2 model simply only on the validation and test data of librispeech_asr.

The demo is run on two Titan RTX (24 GB RAM each). In case you have less RAM available per device, consider reducing `--batch_size` and/or the `--max_duration_in_seconds`.

```
accelerate launch run_wav2vec2_pretraining_no_trainer.py \
    --dataset_name="librispeech_asr" \
    --dataset_config_names clean clean \
    --dataset_split_names validation test \
    --model_name_or_path="patrickvonplaten/wav2vec2-base-v2" \
    --output_dir="./wav2vec2-pretrained-demo" \
    --max_train_steps="20000" \
    --num_warmup_steps="32000" \
    --gradient_accumulation_steps="8" \
    --learning_rate="0.005" \
    --weight_decay="0.01" \
    --max_duration_in_seconds="20.0" \
    --min_duration_in_seconds="2.0" \
    --logging_steps="1" \
    --saving_steps="10000" \
    --per_device_train_batch_size="8" \
    --per_device_eval_batch_size="8" \
    --adam_beta1="0.9" \
    --adam_beta2="0.98" \
    --adam_epsilon="1e-06" \
    --gradient_checkpointing \
```

The results of this run can be seen here.

Base

To pre-train `"base-sized"` Wav2Vec2 model, *e.g.* facebook/wav2vec2-base on librispeech_asr, the following command can be run:

```
accelerate launch run_wav2vec2_pretraining_no_trainer.py \
    --dataset_name=librispeech_asr \
    --dataset_config_names clean clean other \
    --dataset_split_names train.100 train.360 train.500 \
    --model_name_or_path="patrickvonplaten/wav2vec2-base-v2" \
    --output_dir="./wav2vec2-pretrained-demo" \
    --max_train_steps="200000" \
    --num_warmup_steps="32000" \
    --gradient_accumulation_steps="4" \
    --learning_rate="0.001" \
    --weight_decay="0.01" \
    --max_duration_in_seconds="20.0" \
    --min_duration_in_seconds="2.0" \
    --logging_steps="1" \
    --saving_steps="10000" \
    --per_device_train_batch_size="8" \
    --per_device_eval_batch_size="8" \
    --adam_beta1="0.9" \
    --adam_beta2="0.98" \
    --adam_epsilon="1e-06" \
    --gradient_checkpointing \
```

The experiment was run on 8 GPU V100 (16 GB RAM each) for 4 days. In case you have more than 8 GPUs available for a higher effective `batch_size`, it is recommended to increase the `learning_rate` to `0.005` for faster convergence.

The results of this run can be seen here and the checkpoint pretrained for 85,000 steps can be accessed here

Large

To pre-train `"large-sized"` Wav2Vec2 model, *e.g.* facebook/wav2vec2-large-lv60, on librispeech_asr, the following command can be run:

```
accelerate launch run_wav2vec2_pretraining_no_trainer.py \
    --dataset_name=librispeech_asr \
    --dataset_config_names clean clean other \
    --dataset_split_names train.100 train.360 train.500 \
    --output_dir=./test \
    --max_train_steps=200000 \
    --num_warmup_steps=32000 \
    --gradient_accumulation_steps=8 \
    --learning_rate=0.001 \
    --weight_decay=0.01 \
    --max_duration_in_seconds=20.0 \
    --min_duration_in_seconds=2.0 \
    --model_name_or_path=./
    --logging_steps=1 \
    --saving_steps=10000 \
    --per_device_train_batch_size=2 \
    --per_device_eval_batch_size=4 \
    --adam_beta1=0.9 \
    --adam_beta2=0.98 \
    --adam_epsilon=1e-06 \
    --gradient_checkpointing \
```

The experiment was run on 8 GPU V100 (16 GB RAM each) for 7 days. In case you have more than 8 GPUs available for a higher effective `batch_size`, it is recommended to increase the `learning_rate` to `0.005` for faster convergence.

The results of this run can be seen here and the checkpoint pretrained for 120,000 steps can be accessed here