Wav2Vec2 Contrastive Loss PreTraining examples

The following example showcases how to pretrain a wav2vec2 model using the JAX/Flax backend. Pretraining Wav2Vec2 is rather complex, so it is highly recommended to read the official paper.

JAX/Flax allows you to trace pure functions and compile them into efficient, fused accelerator code on both GPU and TPU. Models written in JAX/Flax are **immutable** and updated in a purely functional way which enables simple and efficient model parallelism.

`run_wav2vec2_pretrain_flax.py` is a lightweight example of how to download and preprocess a dataset from the 🤗 Datasets library or use your own files (jsonlines or csv), then pretrain the wav2vec2 architectures above on it.

For custom datasets in `jsonlines` format please see: the Datasets documentation and you also will find examples of these below.

Let's start by creating a model repository to save the trained model and logs. Here we call the model `"wav2vec2-base-robust"`, but you can change the model name as you like.

You can do this either directly on huggingface.co (assuming that you are logged in) or via the command line:

```
huggingface-cli repo create wav2vec2-base-robust
```

Next we clone the model repository to add the tokenizer and model files.

```
git clone https://huggingface.co/<your-username>/wav2vec2-base-robust
```

To ensure that all tensorboard traces will be uploaded correctly, we need to track them. You can run the following command inside your model repo to do so.

```
cd wav2vec2-base-robust
git lfs track "*tfevents*"
```

Great, we have set up our model repository. During training, we will automatically push the training logs and model weights to the repo.

Next, let's add a symbolic link to the `run_wav2vec2_pretrain_flax`.

```
export MODEL_DIR="./wav2vec2-base-robust"
ln -s ~/transformers/examples/research_projects/jax-
projects/wav2vec2/run_wav2vec2_pretrain_flax.py ./
```

Create the model configuration

Let's first create the model configuration and store it in the model repository. Note that many training parameters can be set in the model configuration including the configuration about the masking distribution (`mask_time_length`, `mask_time_prob`), dropout (`attention_dropout`, ...), the trade-off between the contrastive loss and the diversity loss, etc... Mostly likely you will need to change these parameters depending on your use case. Again, we highly recommend to read the official paper to better understand which parameters can be set for pretraining.

For this example, we will be using a `"base"` -sized model of Wav2Vec2 with robust layer norm and keep most of the default settings.

```
model_dir="./wav2vec2-base-robust"

from transformers import Wav2Vec2Config
config = Wav2Vec2Config.from_pretrained(
    "facebook/wav2vec2-base",
    mask_time_length=10,
    mask_time_prob=0.05,
    diversity_loss_weight=0.1,
    num_negatives=100,
    do_stable_layer_norm=True,
    feat_extract_norm="layer",
)
config.save_pretrained(model_dir)
```

Create a feature extractor configuration

Before we can start the training, we need to define a feature extractor that takes care of normalization, etc...

Here we can also re-use the feature extractor of wav2vec2-base-960h while making sure that padding is allowed.

```
model_dir="./wav2vec2-base-robust"

from transformers import Wav2Vec2FeatureExtractor
config = Wav2Vec2FeatureExtractor.from_pretrained("facebook/wav2vec2-base",
return_attention_mask=True)
config.save_pretrained(model_dir)
```

Train the model

Finally, we can run the example script to train the model:

```
./run_wav2vec2_pretrain_flax.py \
    --output_dir=${MODEL_DIR} \
    --num_train_epochs="5" \
    --per_device_train_batch_size="32" \
    --per_device_eval_batch_size="32" \
    --learning_rate="5e-4" \
    --weight_decay="0.01" \
    --warmup_steps="2000" \
    --model_name_or_path=${MODEL_DIR} \
    --dataset_name="librispeech_asr" \
    --dataset_config_name="clean" \
    --train_split_name="train.100" \
    --preprocessing_num_workers="4" \
    --max_duration_in_seconds="10.0" \
    --adam_beta1="0.9" \
    --adam_beta2="0.98" \
```

```
    --pad_to_multiple_of="16384" \
    --push_to_hub
```

Note that this script is not fully tested yet, so we cannot ensure that the above script leads to satisfying results.