IWSLT'14 German to English (Transformer)

Website: https://github.com/pytorch/fairseq/tree/main/examples/translation

One-time installation

Install Moses tokenizer **[Installed: click-8.0.3 joblib-1.1.0 sacremoses-0.0.47 six-1.16.0]**:

```
pip install sacremoses
```

Variable initialization

Set the model name, model description, source and target language

`model_folder_name=iwslt14.tokenized_bpe.hi-en` (If BPE is applied)

`model_folder_name=iwslt14.tokenized_bbpe_2e_2d.hi-en` (If BBPE is applied)

`model_desc=adam_weight_deacy_1e-4`

`source_lang=hi`

`target_lang=en`

`data_source=iitb-corpus`

Directory setup

Create separate folders for the model '**`model_folder_name`**' and mention whether to use pre-processed data/not, get data files or not.

```
bash create_setup_for_model.sh -n $model_folder_name -d $data_source -p [0/1]
```

-p 0 - for no pre-processing and 1 - for pre-processing

Plot Histogram for the data

```
python plot_histogram.py $data_folder_name $source_lang $target_lang $default_data_folder_path
```

Change Directory

```
cd examples/translation/
```

[One-time setup] Download tokenization and BPE code

```
bash setup_tokenization_and_bpe.sh
```

Keep train.$source_lang and train.$target_lang in $model_folder_name/tmp

Text Normalization

```
python preprocess_custom.py $model_folder_name $source_lang $target_lang
```

Break data into Train, Valid, Test and learn and apply either BPE or BBPE

`bash prepare-iwslt14_custom.sh -n $model_folder_name -s $source_lang -t $target_lang` (If BPE is applied)

`bash prepare-iwslt14_custom_bbpe.sh -n $model_folder_name -s $source_lang -t $target_lang` (If BBPE is applied)

Copy files from tmp to its parent directory

`cp $model_folder_name/tmp/*.* $model_folder_name`

Change Directory

`cd ../..`

Preprocess/binarize the data

`bash binarize_data.sh -n $model_folder_name -s $source_lang -t $target_lang`

Train a Transformer translation model over data segmented by either BPE or BBPE

```
CUDA_VISIBLE_DEVICES=0 fairseq-train data-bin/$model_folder_name --arch
transformer_iwslt_de_en --share-decoder-input-output-embed --optimizer adam
--adam-betas '(0.9, 0.98)' --clip-norm 0.0 --lr 5e-4 --lr-scheduler
inverse_sqrt --warmup-updates 4000 --dropout 0.3 --weight-decay 0.001
--criterion label_smoothed_cross_entropy --label-smoothing 0.1 --max-tokens
4096 --keep-last-epochs 1 --max-epoch 350 --fp16 >
log_custom_fldr/$model_folder_name/train_log.txt
```
(If BPE is applied)

```
CUDA_VISIBLE_DEVICES=0 fairseq-train data-bin/$model_folder_name --task
translation --user-dir examples/byte_level_bpe --arch gru_transformer
--encoder-layers 6 --decoder-layers 6 --dropout 0.3 --share-all-embeddings
--optimizer adam --adam-betas '(0.9, 0.98)' --lr 5e-4 --lr-scheduler
inverse_sqrt --warmup-updates 4000 --dropout 0.3 --weight-decay 0.001
--criterion label_smoothed_cross_entropy --label-smoothing 0.1 --save-dir
"checkpoints/${model_folder_name}" --batch-size 32 --max-epoch 350
--update-freq 2 --keep-last-epochs 1 --skip-invalid-size-inputs-valid-test
--fp16 > log_custom_fldr/$model_folder_name/train_log.txt
```
(If BBPE is applied)

Extract info (training and validation loss, etc.,) from our training log file

`bash get_model_stats.sh -n $model_folder_name`

Cleanup after training (move files in proper locations, so that there is no clash with training a new model)

`bash cleanup_after_training.sh -n $model_folder_name -d $model_desc`

Evaluate our trained model that uses either BPE or BBPE

`bash generate_outputs_for_model.sh -n $model_folder_name -d $model_desc -s $source_lang -t $target_lang` (If BPE is applied)

`BPE="--bpe byte_bpe --sentencepiece-model-path examples/translation/$model_folder_name/spm_bbpe4096.model"` (If BBPE is applied)

```
fairseq-generate data-bin/$model_folder_name --task translation --user-dir
examples/byte_level_bpe --source-lang $source_lang --gen-subset test
--sacrebleu --path "checkpoints/$model_folder_name/checkpoint_best.pt"
--tokenizer moses --target-lang $target_lang ${BPE}
--skip-invalid-size-inputs-valid-test
```
(If BBPE is applied)