OpenNMT_Pytorch_Basic_Tutorial

March 23, 2025

1 I created the OpenNMT Pytorch tutorial using Colab.

First Go to Runtime and change the runtime type to GPU.

2 Git Clone

First Git clone the OpenNMT source

[ ]:
```
!git clone https://github.com/OpenNMT/OpenNMT-py
```

3 Please install requirements.txt use by pip

Error : You must restart the runtime in order to use newly installed versions. Solution : Click Restart Runtime => Redo

[ ]:
```
!pip install -r OpenNMT-py/requirements.txt
```

4 Theory explanation

Machine translation is a field of natural language processing, meaning that computers translate one language into another.

Rule based, and statistical based, and recently we are using Deep Learning-based machine translation.

Learn how to build a real machine translation system and how the system pipeline is structured. Most of these courses can be applied to basic natural language processing problems as well as machine translation.

Step

1. Data Collection

Parallel corpus is collected from various sources. It is possible to collect news texts, drama / movie subtitles, Wikipedia, etc., as well as data sets for evaluation of translation systems disclosed by WMT, a machine translation competition, and use them in translation systems.

2. Cleaning

The collected data must be refined. The refinement process includes sorting sentences by corpus in both languages, and eliminating noise such as special characters.

3. Subword Tokenization

Refine spacing using the POS tagger or segmenter for each language. English may have refinement issues in upper / lower case. After the spacing is refined, use Byte Pair Encoding (BPE) using public tools such as Subword or WordPiece. This allows you to perform additional segments and construct a vocabulary list. At this time, the segmented models learned for the BPE segment should be kept for future use.

4. Train

Train the seq2seq model using prepared datasets. Depending on the amount, you can train with a single GPU, or use multiple GPUs in parallel to reduce training time.

5. Translate

Now that the model has been created, you can start translating.

6. Detokenization

Even after the translation process is finished, it is still in a segment, so it is different from the actual sentence structure used by real people. Thus, when you perform a detoxification process, it is returned in the form of the actual sentence.

7. Evaluating

Quantitative evaluation is performed on the sentence thus obtained. BLEU is a quantitative evaluation method for machine translation. You can see which model is superior by comparing it to the BLEU score you are comparing.

5 Tutotorial Start

Assume that you have data collection and refinement and start the tutorial.

Use the data provided by OpenNMT-py.

Locate in **OpenNMT-py/data**

6 Subword Tokenization

We use Byte Pair Encoding for Subword Tokenization

https://www.aclweb.org/anthology/P16-1162

i => input o ==> Output(*.code) s ==> Symbol

learn_bpe ==> make code apply_bpe ==> apply subwordTokenization

src-train, src-val,test ==> Need to apply src.code tgt-train,tgt-val ==> Need to apply tgt.code

[ ]:
```
!python OpenNMT-py/tools/learn_bpe.py -i OpenNMT-py/data/src-train.txt -o␣
↪OpenNMT-py/data/src.code -s 10000
```

```
[ ]: !python OpenNMT-py/tools/learn_bpe.py -i OpenNMT-py/data/tgt-train.txt -o␣
     ↪OpenNMT-py/data/tgt.code -s 10000
```

```
[ ]: !python OpenNMT-py/tools/apply_bpe.py -c OpenNMT-py/data/src.code -i OpenNMT-py/
     ↪data/src-train.txt -o OpenNMT-py/data/src-train-bpe.txt
```

```
[ ]: !python OpenNMT-py/tools/apply_bpe.py -c OpenNMT-py/data/src.code -i OpenNMT-py/
     ↪data/src-val.txt -o OpenNMT-py/data/src-val-bpe.txt
```

```
[ ]: !python OpenNMT-py/tools/apply_bpe.py -c OpenNMT-py/data/src.code -i OpenNMT-py/
     ↪data/src-test.txt -o OpenNMT-py/data/src-test-bpe.txt
```

```
[ ]: !python OpenNMT-py/tools/apply_bpe.py -c OpenNMT-py/data/tgt.code -i OpenNMT-py/
     ↪data/tgt-train.txt -o OpenNMT-py/data/tgt-train-bpe.txt
```

```
[ ]: !python OpenNMT-py/tools/apply_bpe.py -c OpenNMT-py/data/tgt.code -i OpenNMT-py/
     ↪data/tgt-val.txt -o OpenNMT-py/data/tgt-val-bpe.txt
```

7 Preprocess the data

We will be working with some example data in data/ folder.

The data consists of parallel source (src) and target (tgt) data containing one sentence per line with tokens separated by a space:

1. src-train.txt
2. tgt-train.txt
3. src-val.txt
4. tgt-val.txt

Train data and validataion data are required for machine translation training.

Validation files are required and used to evaluate the convergence of the training. It usually contains no more than 5000 sentences.

> If you think about it briefly, you can specify the path of train data and validation data, and specify the path and name to save in -save_data.
>
> If you want to set vocab size add below command -src_vocab_size 32000 -tgt_vocab_size 32000

The vocab size is usually 32000.

```
[ ]: !python OpenNMT-py/preprocess.py -train_src OpenNMT-py/data/src-train-bpe.txt␣
     ↪-train_tgt OpenNMT-py/data/tgt-train-bpe.txt -valid_src OpenNMT-py/data/
     ↪src-val-bpe.txt -valid_tgt OpenNMT-py/data/tgt-val-bpe.txt -save_data␣
     ↪OpenNMT-py/data/demo
```

8 Train the data(Basic)

This is simple Train command use 2-layer LSTM with 500 hidden units on both the encoder/decoder.

If you want to use GPU , try add below command (example use 2 GPU) >-world_size 2 -gpu_ranks 0 1

Let's Check Available GPU

[ ]:
```
!nvidia-smi
```

[ ]:
```
!python OpenNMT-py/train.py -data OpenNMT-py/data/demo -save_model OpenNMT-py/
↪demo-model
```

9 Train the data(Transformer)

https://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf

If you get GPU-related errors, try halving batch_size

Below is the full command, and if you want to know more about it, search about Transformer.

!python OpenNMT-py/train.py -data OpenNMT-py/data/demo -save_model OpenNMT-py/data/model/model -layers 6 -rnn_size 512 -word_vec_size 512 -transformer_ff 2048 -heads 8 -encoder_type transformer -decoder_type transformer -position_encoding -train_steps 200000 -max_generator_batches 2 -dropout 0.1 -batch_size 4096 -batch_type tokens -normalization tokens -accum_count 2 -optim adam -adam_beta2 0.998 -decay_method noam -warmup_steps 8000 -learning_rate 2 -max_grad_norm 0 -param_init 0 -param_init_glorot -label_smoothing 0.1 -valid_steps 1000 -save_checkpoint_steps 1000 -world_size 1 -gpu_rank 0

[ ]:
```
!python OpenNMT-py/train.py -data OpenNMT-py/data/demo -save_model OpenNMT-py/
↪data/model/model -layers 6 -rnn_size 512 -word_vec_size 512 -transformer_ff␣
↪2048 -heads 8 -encoder_type transformer -decoder_type transformer␣
↪-position_encoding -train_steps 200000 -max_generator_batches 2 -dropout 0.1␣
↪-batch_size 4096 -batch_type tokens -normalization tokens -accum_count 2␣
↪-optim adam -adam_beta2 0.998 -decay_method noam -warmup_steps 8000␣
↪-learning_rate 2 -max_grad_norm 0 -param_init 0 -param_init_glorot␣
↪-label_smoothing 0.1 -valid_steps 1000 -save_checkpoint_steps 1000␣
↪-world_size 1 -gpu_rank 0
```

10 Translate

Now that you have your model, you can start translating.

-model ==> Setting your model

Output predictions into pred.txt

```
[ ]: !python OpenNMT-py/translate.py -model OpenNMT-py/data/model/
      ↪demo-model_YOUR_MODEL.pt -src OpenNMT-py/data/src-test.txt -output␣
      ↪OpenNMT-py/data/pred.txt -replace_unk -verbose
```

11 Detokenization

Even after the translation process is finished, it is still in a segment, so it is different from the actual sentence structure used by real people. Thus, when you perform a detoxification process, it is returned in the form of the actual sentence.

We Use "sed" for BPE Detokenization

```
[ ]: sed -i "s/@@ //g"  OpenNMT-py/data/pred.txt
```

12 Evaluation Using BLEU

Quantitative evaluation is performed on the sentence thus obtained. BLEU is a quantitative evaluation method for machine translation. You can see which model is superior by comparing it to the BLEU score you are comparing.

https://www.aclweb.org/anthology/P02-1040

```
[ ]: perl  OpenNMT-py/tools/multi-bleu.perl OpenNMT-py/data/ref.txt < OpenNMT-py/
      ↪data/pred.txt
```

If you have Any Question Please Email to "bcj1210@naver.com"