Date: 2023.12.07

The pytorch template may update in the future, but the code in this example is not affected by that.

IMDb

This is the record of how to tweak the pytorch template for this project, as well as what the training procedure looks like.

1. tweak code

remove the code unneeded, e.g., the code for cv, and the code about test dataset, as there's only train and valid data

tweak ./preprocess.py according to the dataset format and implement label transform

```
def preprocess nlp(
   train_text: list[str],
   lowercase: bool = True,
   rm punctuation: bool = True,
   rm stopword: bool = False,
   lemmatization: bool = True,
   min_freq: int = 1,
   max_tokens: int = 10000,
   """a function for preprocessing in NLP task
   return text transform, label transform, vocab, tokenizer
   tokenizer = Tokenizer(
       lowercase=lowercase,
       rm_punctuation=rm_punctuation,
        rm_stopword=rm_stopword,
       lemmatization=lemmatization,
   def yield_tokens(data: list[str]):
       for sentence in data:
           tokens = tokenizer(sentence)
           yield tokens
   token generator = yield tokens(data=train text)
   # special tokens
   specials = ['<unk>', ]
   vocab = build_vocab_from_iterator(
        iterator=token_generator,
```

tweak ./main.py, add code to load data, IMDb dataset will be loaded from Huggingface, and I only use a subset to save time

tweak ./dataset.py, according to the IMDb data format

```
class TextDataset(Dataset):
5
        def __init__(self, texts, labels, text_transform=None, label_transform=None):
            assert len(texts) == len(labels)
            self.texts = texts
            self.labels = labels
            self.text_transform = text_transform
            self.label_transform = label transform
        def len (self):
            return len(self.labels)
        def __getitem__(self, idx):
            text = self.texts[idx]
            label = self.labels[idx]
            if self.text_transform:
                text = self.text transform(text)
            if self.label_transform:
                label = self.label transform(label)
            return text, label
```

tweak **./main.py**, as the TextDataset changed, tweak the part about dataset

as ./model.py in template is for text classification task by default, so I just use it directly with no tweak

tweak ./config.yaml

tweak config for wandb

```
# config for wandb
wandb_cfg:
use_wandb: True
project: "IMDb"
notes: "training details on the process of global rank 0"
tags: ["IMDb dataset", "TransformerEncoder"]
```

the configs for preprocess, dataloader and model seem ok, so I just keep these

by default, the template use CrossEntropyLoss for criterion, AdamW for optimizer, CosineAnnealingWarmRestarts for lr scheduler, it seems ok, so I didn't change these, and just tweak the config for optimizer

```
# config for optimizer
config for optimizer
for optim
```

for the train_cfg, I set max_epoch to 20, do validation and test accuracy at every epoch, set save_dir and create the save_dir manually

```
train cfg:
 max epoch: 20
 do valid: True
 do test: True
 save log: True
 save best: True
 save checkpoint: True
 resume_checkpoint: False
 valid start: 1
 valid step: 1
 test_start: 1
 # required if `save_*` is True
save_dir: "./imdb_ckpt"
 # required if `save_best` is True
measure_best: "accuracy"
 measure_mode: "max"
# required if `save_checkpoint` is True
  checkpoint_latest: True
  checkpoint_list: [10, 15]
  # required if `resume checkpoint`
```

the Trainer is off-the-shelf, no need to change anything about it

as for the test method, I just want to test accuracy, which is already implemented by the template, so no change

lastly, tweak ./run.sh, as I will train on my laptop with single gpu

```
#!/bin/sh

#!/bin/sh

# torchrun automatically spawns the processes!

# single-node, multi-worker

# for example, 1 machine, which has 1 GPU

# run the command below

torchrun --standalone --nnodes=1 --nproc_per_node=1 ./main.py

# multi-node, multi-worker

# for example, 2 machines, where one has 4 GPUs and the other has

# run the fist command below on the first machine

# torchrun --nnodes=2 --node_rank=0 --nproc-per-node=4 --rdzv-id=$i

# torchrun --nnodes=2 --node_rank=0 --nproc-per-node=4 --rdzv-id=$i
```

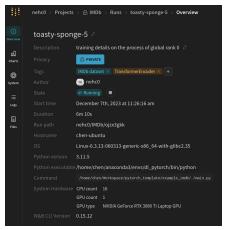
2. train procedure

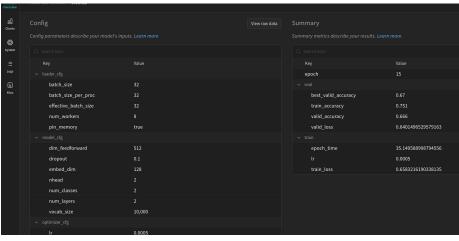
start training, terminal:

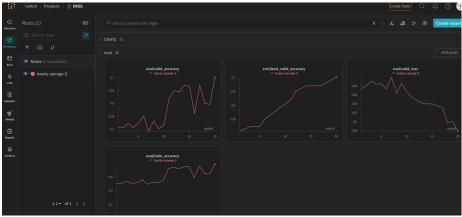
```
chen@chen-ubuntu:~/Workspace/pytorch_template/example_imdb
sh <u>./run.sh</u>
[2023-12-07 11:25:23,149] torch.distributed.run: [WARNING] master_addr is only used for static rdzv_backend and when rdzv
endpoint is not specified.
Found cached dataset imdb (/home/chen/Workspace/pytorch_template/example_imdb/.huggingface/imdb/plain_text/1.0.0/d613c88cf8fa3bab83b4ded3713f1f74830d1100e171db75bbddb80b3345c9c0)
                                                                                                                                                                           | 3/3 [00:00<00:00, 778.12it/s]
 Loading cached shuffled indices for dataset at /home/chen/Workspace/pytorch_template/example_imdb/.huggingface/imdb/plain_
text/1.0.0/d613c88cf8fa3bab83b4ded3713f1f74830d1100e171db75bbddb80b3345c9c0/cache-79aee49c9f40dc82.arrow
Loading cached shuffled indices for dataset at /home/chen/Workspace/pytorch_template/example_imdb/.huggingface/imdb/plain_
text/1.0.0/d613c88cf8fa3bab83b4ded3713f1f74830d1100e171db75bbddb80b3345c9c0/cache-5a09ddfc1bd0fbc8.arrow
Loading cached shuffled indices for dataset at /home/chen/Workspace/pytorch_template/example_imdb/.huggingface/imdb/plain_
text/1.0.0/d613c88cf8fa3bab83b4ded3713f1f74830d1100e171db75bbddb80b3345c9c0/cache-f131e6602007628b.arrow
text1.0.0/doi3cd8c18fa3babb3b4ded3/13f1f7483dd118de1/1doi/5bbddb8db343c9c0/cache-f131e000200/b28ba.arrow
/home/chen/anaconda3/envs/d_pytorch/rib/python3.11/site-package/tyrorch/rnn/modules/transformer.py:282: UserWarning: enable
_nested_tensor is True, but self.use_nested_tensor is False because encoder_layer.self_attn.batch_first was not True(use b
atch_first for better inference performance)
warnings.warn(f"enable_nested_tensor is True, but self.use_nested_tensor is False because {why_not_sparsity_fast_path}")
wandb: Currently logged in as: nehc0. Use `wandb login --relogin` to force relogin
wandb: wandb version 0.16.1 is available! To upgrade, please run:
wandb: $ pip install wandb --upgrade
wandb: $ Tacking run with wandb version 0.15.12
 wandb: Tracking run with wandb version 0.15.12
wandb: Run data is saved locally in /home/chen/
  wandb: Run `wandb offline` to turn off syncing.
 wandb: Syncing run toasty
wandb: ★ View project at
wandb: ※ View run at htt
                                                                                                                                                                         | 32/32 [00:05<00:00, 5.44it/s]
seed: 6
use wandb: True
notes: training details on the process of global rank 0
```

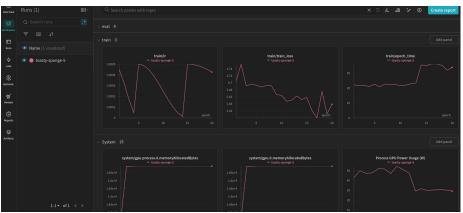
```
chen@chen-ubuntu:~/Workspace/pytorch_template/example_imdb
 save_best: True
save_checkpoint: True
resume_checkpoint: False
valid_start: 1
valid_step: 1
test_start: 1
test_step: 1
save_dir: ./imdb_ckpt
measure_best: accuracy
measure_mode: max
checkpoint_latest: True
checkpoint_list: [10, 15]
resume_path: None
}
    save_best: True
      vorld_size: 1
   2023-12-07 11:26:27 - INFO -
                                                                                                                                       Start of training, Good day
                                                                                                                                                                                                                                                                                                           32/32 [00:05<00:00, 5.75it/s]
32/32 [00:05<00:00, 5.83it/s]
32/32 [00:05<00:00, 5.76it/s]
32/32 [00:05<00:00, 5.73it/s]
loss: 0.6725325584411621 | Train
   100%|
100%|
100%|
     100%
   2023-12-07 11:26:50 - INFO - [GPU0] | Epoch 1/20 | Train loss: 0.7045852541923523 | Valid scores: accuracy: 0.507 | Valid scores: accuracy: 0.507 | Time/epoch: 22.2024 seconds 2023-12-07 11:26:50 - INFO - New best model: valid accuracy update from -inf to 0.507 2023-12-07 11:26:50 - INFO - Saving best model: ./imdb_ckpt/rung23120_11:26:27/best_mode
                                                                                                                                                                                                                                                                                                        el epoch1.pth ..
                                                                                                                                                                                                                                                                                                              2690011.ptn ...
32/32 [00:05<00:00, 6.16it/s]
32/32 [00:04<00:00, 6.41it/s]
32/32 [00:05<00:00, 5.93it/s]
32/32 [00:06<00:00, 5.30it/s]
   100%|
100%|
100%|
   | 2023-12-07 11:27:11 - INFO - [GPU0] | Epoch 2/20 | Train loss: 0.7534643411636353 | Valid
scores: accuracy: 0.507 | Valid scores: accuracy: 0.507 | Time/epoch: 21.62994 seconds
                                                                                                                                                                                                                                                                                                               loss: 0.6802284717559814 | Train
  32/32 [00:05<00:00, 6.10it/s]
32/32 [00:05<00:00, 5.49it/s]
   2023-12-07 11:31:15 - INFO - [GPU0] | Epoch 13/20 | Train loss: 0.6626368165016174 | Vali
n scores: accuracy: 0.719 | Valid scores: accuracy: 0.644 | Time/epoch: 23.4461 seconds
100%|
                                                                                                                                                                                                                                                                                                             32/32 [00:08<00:00, 3.78it/s]
32/32 [00:10<00:00, 3.07it/s]
32/32 [00:08<00:00, 3.98it/s]
32/32 [00:08<00:00, 3.73it/s]
loss: 0.639674961566925 | Trai
   100%
100%
100%
   | 2023-12-07 11:31:51 - INFO - Saving best model: ./imdb_ckpt/run@231207_11:26:27/best_model_epoch14.pth ... | 32/32 [00:10<00:00, 2.92it/s] | 100% | | 32/32 [00:10<00:00, 2.92it/s] | 32/32 [00:00<00:00, 2.92it/s] | 32/32 [00:00<00:00, 3.88it/s] | 100% | | 32/32 [00:00<00:00, 3.88it/s] | 32/32 [00:00<00:00, 3.88it/s] | 32/32 [00:00<00:00, 3.88it/s] | 32/32 [00:00<00:00, 3.90it/s] | 32/32 [00:00<00:00, 3.85it/s] | 32/32 [00:00<00:00, 3.85it/s] | 32/32 [00:00<00:00, 3.85it/s] | 32/32 [00:00<00:00, 3.85it/s] | 32/32 [00:00<00:00, 3.64it/s] | 32/32 [00:00<00:00, 3.64it/s]
   n score
100%|
100%|
                                                                                                                                                                                                                                                                                                            32/32 [00:09<00:00, 3.30it/s]
32/32 [00:09<00:00, 3.38it/s]
32/32 [00:07<00:00, 4.12it/s]
32/32 [00:09<00:00, 3.52it/s]
     100%
   100%
  2023-12-07 11:34:14 - INFO - [GPU0] | Epoch 18/20 | Train loss: 0.6756136417388916 | Valid loss: 0.5975479483604431 | Train scores: accuracy: 0.65 | Valid scores: accuracy: 0.601 | Time/epoch: 35.29723 seconds | 32/32 [00:09<00:00, 3.541t/s] | 32/32 [00:07<00:00, 4.24it/s]
                                                                                                                                                                                                                                                                                                             32/32 [00:09<00:00, 3.54it/s]
32/32 [00:07<00:00, 4.24it/s]
32/32 [00:07<00:00, 4.18it/s]
32/32 [00:07<00:00, 4.16it/s]
d loss: 0.6009341478347778 | Trai
  100%|
100%|
100%|
100%|
2023-12-07 11:34:46 - INFO - [GPU0] | Epoch 19/20 | Train loss: 0.6119271516799927 | Va
n scores: accuracy: 0.648 | Valid scores: accuracy: 0.596 | Time/epoch: 31.95647 second:
100%|
100%|
100%|
100%|
wandb: epoch wandb: epoch wandb: eval/best_valid_accuracy wandb: eval/train_accuracy wandb: eval/valid_accuracy wandb: eval/valid_accuracy wandb: eval/valid_loss wandb: train/epoch_time wandb: train/train_loss wandb: train/train_loss
                                                                                                                                                            +-
     wandb: Run summary:
       andb: epoch 20
andb: eval/best_valid_accuracy 0.702
andb: eval/train_accuracy 0.785
```

wandb:









saved logs and checkpoints:

```
pytorch_template

example_imdb

pycache__

huggingface

imdb_ckpt

run@231207_11:26:27

best_model_epoch20.pth

checkpoint_epoch10.pth

checkpoint_epoch15.pth

latest_checkpoint_epoch20.pth

train.log
```

```
train.log X
III Workspace > pytorch_template > example_imdb > imdb_ckpt > run@231207_11:26:27 > III train.log
  1 2023-12-07 11:26:27 - INFO - ----- config -----
      wandb_cfg: {
      use wandb: True
      project: IMDb
      notes: training details on the process of global rank 0
      tags: ['IMDb dataset', 'TransformerEncoder']
      preprocess_nlp_cfg: {
     lowercase: True
     rm punctuation: True
      rm_stopword: False
      lemmatization: True
      min_freq: 3
      max tokens: 10000
      loader_cfg: {
      batch_size: 32
```