Language-agnostic BERT Sentence Embedding

The repository contains the implementation and experiment definition of `LaBSE`, Language-agnostic BERT Sentence Embedding. The implementation is provided by the paper author, Yinfei Yang. Note that, the cross-accelerator batch softmax is not implemented by the author, so the implementation does not fully reproduce the paper yet.

Due to the data policy, the authors are not able to release the pre-training and fine-tuning data for `LaBSE` training.

Requirements

The starter code requires Tensorflow. If you haven't installed it yet, follow the instructions on tensorflow.org. This code has been tested with Tensorflow 2.8.0. Going forward, we will continue to target the latest released version of Tensorflow.

Please verify that you have Python 3.7+ and Tensorflow 2.8.0 or higher installed by running the following commands:

```
python --version
python -c 'import tensorflow as tf; print(tf.__version__)'
```

Refer to the instructions here for using the model in this repo. Make sure to add the models folder to your Python path.

Data

The pre-training data should be multi-lingual and the format is the same as BERT pre-training.

The fine-tuning data follows the format as below:

```
{   # (tensorflow.Example)
  features: {
    feature: {
      key  : "src_raw"
      value: {
        bytes_list: {
          value: [ "Foo. " ]
        }
      }
    }
    feature: {
      key  : "tgt_raw"
      value: {
        bytes_list: {
          value: [ "Bar. " ]
```

```
        }
      }
    }
  }
}
```

Train using the config file.

After you generated your pretraining data, run the following command to start pretraining:

```
TPU=local
VOCAB=???
INIT_CHECKPOINT=???
PARAMS="task.train_data.input_data=/path/to/train/data"
PARAMS="${PARAMS},task.train_data.vocab_file=${VOCAB}"
PARAMS="${PARAMS},task.validation_data.input_path=/path/to/validation/data"
PARAMS="${PARAMS},task.validation_data.vocab_file=${VOCAB}"
PARAMS="${PARAMS},task.init_checkpoint=${INIT_CHECKPOINT}"
PARAMS="${PARAMS},runtime.distribution_strategy=tpu"

python3 train.py \
  --experiment=labse/train \
  --config_file=./experiments/labse_bert_base.yaml \
  --config_file=./experiments/labse_base.yaml \
  --params_override=${PARAMS} \
  --tpu=${TPU} \
  --model_dir=/folder/to/hold/logs/and/models/ \
  --mode=train_and_eval
```

Implementation

We implement the encoder and layers using `tf.keras` APIs in NLP modeling library:

- dual_encoder.py contains the dual-encoder task used for labse training.
- config_labse.py registers the labse training experiment.
- train.py is the program entry.

Pre-trained model through TF-HUB

If you are looking for pre-trained models, please check out: https://tfhub.dev/google/LaBSE/2. The hub `SavedModels` are exported through the `export_tfhub.py` in this repository.