Text classification examples

GLUE tasks

Based on the script `run_flax_glue.py`.

Fine-tuning the library models for sequence classification on the GLUE benchmark: General Language Understanding Evaluation. This script can fine-tune any of the models on the hub and can also be used for a dataset hosted on our hub or your own data in a csv or a JSON file (the script might need some tweaks in that case, refer to the comments inside for help).

GLUE is made up of a total of 9 different tasks. Here is how to run the script on one of them:

```
export TASK_NAME=mrpc

python run_flax_glue.py \
  --model_name_or_path bert-base-cased \
  --task_name ${TASK_NAME} \
  --max_seq_length 128 \
  --learning_rate 2e-5 \
  --num_train_epochs 3 \
  --per_device_train_batch_size 4 \
  --eval_steps 100 \
  --output_dir ./$TASK_NAME/ \
  --push_to_hub
```

where task name can be one of cola, mnli, mnli_mismatched, mnli_matched, mrpc, qnli, qqp, rte, sst2, stsb, wnli.

Using the command above, the script will train for 3 epochs and run eval after each epoch. Metrics and hyperparameters are stored in Tensorflow event files in `--output_dir`. You can see the results by running `tensorboard` in that directory:

```
$ tensorboard --logdir .
```

or directly on the hub under *Training metrics*.

Accuracy Evaluation

We train five replicas and report mean accuracy and stdev on the dev set below. We use the settings as in the command above (with an exception for MRPC and WNLI which are tiny and where we used 5 epochs instead of 3), and we use a total train batch size of 32 (we train on 8 Cloud v3 TPUs, so a per-device batch size of 4),

On the task other than MRPC and WNLI we train for 3 these epochs because this is the standard, but looking at the training curves of some of them (e.g.,

SST-2, STS-b), it appears the models are undertrained and we could get better results when training longer.

In the Tensorboard results linked below, the random seed of each model is equal to the ID of the run. So in order to reproduce run 1, run the command above with `--seed=1`. The best run used random seed 3, which is the default in the script. The results of all runs are in this Google Sheet.

Task	Metric	Acc (best run)	Acc (avg/5runs)	Stdev	Metrics
CoLA	Matthews corr	60.57	59.04	1.06	tfhub.dev
SST-2	Accuracy	92.66	92.23	0.57	tfhub.dev
MRPC	F1/Accuracy	89.90/85.78	88.97/84.36	0.72/1.09	tfhub.dev
STS-B	Pearson/Spearman corr.	89.04/88.78	88.94/88.63	0.07/0.07	tfhub.dev
QQP	Accuracy/F1	90.81/87.58	90.76/87.51	0.05/0.06	tfhub.dev
MNLI	Matched acc.	84.10	83.80	0.16	tfhub.dev
QNLI	Accuracy	91.01	90.82	0.17	tfhub.dev
RTE	Accuracy	66.06	64.76	1.04	tfhub.dev
WNLI	Accuracy	46.48	37.01	6.83	tfhub.dev

Some of these results are significantly different from the ones reported on the test set of GLUE benchmark on the website. For QQP and WNLI, please refer to FAQ #12 on the website.

Runtime evaluation

We also ran each task once on a single V100 GPU, 8 V100 GPUs, and 8 Cloud v3 TPUs and report the overall training time below. For comparison we ran Pytorch's run_glue.py on a single GPU (last column).

Task	TPU v3-8	8 GPU	1 GPU	1 GPU (Pytorch)
CoLA	1m 42s	1m 26s	3m 9s	4m 6s
SST-2	5m 12s	6m 28s	22m 33s	34m 37s
MRPC	1m 29s	1m 14s	2m 20s	2m 56s
STS-B	1m 30s	1m 12s	2m 16s	2m 48s
QQP	22m 50s	31m 48s	1h 59m 41s	2h 54m
MNLI	25m 03s	33m 55s	2h 9m 37s	3h 7m 6s
QNLI	7m30s	9m 40s	34m 40s	49m 8s
RTE	1m 20s	55s	1m 10s	1m 16s

Task	TPU v3-8	8 GPU	1 GPU	1 GPU (Pytorch)
WNLI	1m 11s	48s	39s	36s
——-				
TOTAL	1h 03m	1h 28m	5h 16m	6h 37m

*All experiments are ran on Google Cloud Platform. GPU experiments are ran without further optimizations besides JAX transformations. GPU experiments are ran with full precision (fp32). "TPU v3-8" are 8 TPU cores on 4 chips (each chips has 2 cores), while "8 GPU" are 8 GPU chips.