BERT FineTuning with Cloud TPU: Sentence and Sentence-Pair Classification Tasks (TF 2.1)

This tutorial shows you how to train the Bidirectional Encoder Representations from Transformers (BERT) model on Cloud TPU.

Set up Cloud Storage and Compute Engine VM

1. Open a cloud shell window
2. Create a variable for the project's id:

```
export PROJECT_ID=your-project_id
```

3. Configure `gcloud` command-line tool to use the project where you want to create Cloud TPU.

```
gcloud config set project ${PROJECT_ID}
```

4. Create a Cloud Storage bucket using the following command:

```
gsutil mb -p ${PROJECT_ID} -c standard -l europe-west4 -b on gs://your-bucket-name
```

This Cloud Storage bucket stores the data you use to train your model and the training results. 5. Launch a Compute Engine VM and Cloud TPU using the ctpu up command.

```
ctpu up --tpu-size=v3-8 \
 --machine-type=n1-standard-8 \
 --zone=europe-west4-a \
 --tf-version=2.1 [optional flags: --project, --name]
```

6. The configuration you specified appears. Enter y to approve or n to cancel.
7. When the ctpu up command has finished executing, verify that your shell prompt has changed from username@project to username@tpuname. This change shows that you are now logged into your Compute Engine VM.

```
gcloud compute ssh vm-name --zone=europe-west4-a
(vm)$ export TPU_NAME=vm-name
```

As you continue these instructions, run each command that begins with `(vm)$` in your VM session window.

Prepare the Dataset

1. From your Compute Engine virtual machine (VM), install requirements.txt.

```
(vm)$ cd /usr/share/models
(vm)$ sudo pip3 install -r official/requirements.txt
```

2. Optional: download download_glue_data.py

This tutorial uses the General Language Understanding Evaluation (GLUE) benchmark to evaluate and analyze the performance of the model. The GLUE data is provided for this tutorial at gs://cloud-tpu-checkpoints/bert/classification.

Define parameter values

Next, define several parameter values that are required when you train and evaluate your model:

```
(vm)$ export PYTHONPATH="$PYTHONPATH:/usr/share/tpu/models"
(vm)$ export STORAGE_BUCKET=gs://your-bucket-name
(vm)$ export BERT_BASE_DIR=gs://cloud-tpu-checkpoints/bert/keras_bert/uncased_L-24_H-
1024_A-16
(vm)$ export MODEL_DIR=${STORAGE_BUCKET}/bert-output
(vm)$ export GLUE_DIR=gs://cloud-tpu-checkpoints/bert/classification
(vm)$ export TASK=mnli
```

Train the model

From your Compute Engine VM, run the following command.

```
(vm)$ python3 official/nlp/bert/run_classifier.py \
  --mode='train_and_eval' \
  --input_meta_data_path=${GLUE_DIR}/${TASK}_meta_data \
  --train_data_path=${GLUE_DIR}/${TASK}_train.tf_record \
  --eval_data_path=${GLUE_DIR}/${TASK}_eval.tf_record \
  --bert_config_file=$BERT_BASE_DIR/bert_config.json \
  --init_checkpoint=$BERT_BASE_DIR/bert_model.ckpt \
  --train_batch_size=32 \
  --eval_batch_size=32 \
  --learning_rate=2e-5 \
  --num_train_epochs=3 \
  --model_dir=${MODEL_DIR} \
  --distribution_strategy=tpu \
  --tpu=${TPU_NAME}
```

Verify your results

The training takes approximately 1 hour on a v3-8 TPU. When script completes, you should see results similar to the following:

```
Training Summary:
{'train_loss': 0.28142181038856506,
'last_train_metrics': 0.9467429518699646,
'eval_metrics': 0.8599063158035278,
'total_training_steps': 36813}
```

Clean up

To avoid incurring charges to your GCP account for the resources used in this topic:

1. Disconnect from the Compute Engine VM:

```
(vm)$ exit
```

2. In your Cloud Shell, run ctpu delete with the --zone flag you used when you set up the Cloud TPU to delete your Compute Engine VM and your Cloud TPU:

```
$ ctpu delete --zone=your-zone
```

3. Run ctpu status specifying your zone to make sure you have no instances allocated to avoid unnecessary charges for TPU usage. The deletion might take several minutes. A response like the one below indicates there are no more allocated instances:

```
$ ctpu status --zone=your-zone
```

4. Run gsutil as shown, replacing your-bucket with the name of the Cloud Storage bucket you created for this tutorial:

```
$ gsutil rm -r gs://your-bucket
```