Training MnasNet on Cloud TPU

This tutorial shows you how to train the Tensorflow MnasNet model (https://github.com/tensorflow/tpu/tree/master/models/official/mnasnet) using a Cloud TPU device or Cloud TPU Pod slice (multiple TPU devices). You can apply the same pattern to other TPU-optimised image classification models that use TensorFlow and the ImageNet dataset.

Disclaimer

This tutorial uses a third-party dataset. Google provides no representation, warranty, or other guarantees about the validity, or any other aspects of this dataset.

Model description

The model in this tutorial is based on MnasNet: Platform-Aware Neural Architecture Search for Mobile (https://arxiv.org/pdf/1807.11626.pdf), which first introduces the AutoML mobile neural network (MnasNet) architecture. The tutorial uses the state-of-the-art variant, 'mnasnet-a1', and demonstrates training the model using TPUEstimator (https://www.tensorflow.org/api_docs/python/tf/contrib/tpu/TPUEstimator).

Special considerations when training on a Pod slice (v2-32/v3-32 and above)

If you plan to train on a TPU Pod slice, please make sure you go over this document (https://cloud.google.com/tpu/docs/training-on-tpu-pods) that explains the special considerations when training on a Pod slice.

Set up your project

Before you start the tutorial, check that your Google Cloud project is set up correctly or set up a new project.

1. Sign in (https://accounts.google.com/Login) to your Google Account.

   If you don't already have one, sign up for a new account (https://accounts.google.com/SignUp).

2. In the Cloud Console, on the project selector page, select or create a Cloud project.

★ **Note**: If you don't plan to keep the resources that you create in this procedure, create a project instead of selecting an existing project. After you finish these steps, you can delete the project, removing all resources associated with the project.

GO TO THE PROJECT SELECTOR PAGE (HTTPS://CONSOLE.CLOUD.GOOGLE.COM/PROJECTSELECT

3. Make sure that billing is enabled for your Google Cloud project. Learn how to confirm billing is enabled for your project (https://cloud.google.com/billing/docs/how-to/modify-project).
4. Verify (https://cloud.google.com/tpu/docs/quota#requesting_additional_quota) that you have sufficient quota to use either TPU devices or Pods.

Caution: This walkthrough uses billable components of Google Cloud. Check the Cloud TPU pricing page (https://cloud.google.com/tpu/docs/pricing) to estimate your costs. Be sure to clean up (#clean_up) resources you create when you've finished with them to avoid unnecessary charges.

Set up your resources

This section provides information on setting up Cloud Storage storage, VM, and Cloud TPU resources for tutorials.

Important: Set up all resources in the same region/zone to reduce network latency and network costs.

Create a Cloud Storage bucket

You need a Cloud Storage bucket to store the data you use to train your model and the training results. The `ctpu up` tool used in this tutorial sets up default permissions for the Cloud TPU

service account. If you want finer-grain permissions, review the access level permissions (https://cloud.google.com/tpu/docs/storage-buckets).

The bucket location must be in the same region as your virtual machine (VM) and your TPU node. VMs and TPU nodes are located in specific zones (https://cloud.google.com/tpu/docs/types-zones#types), which are subdivisions within a region.

1. Go to the Cloud Storage page on the Cloud Console.

   GO TO THE CLOUD STORAGE PAGE (HTTPS://CONSOLE.CLOUD.GOOGLE.COM/STORAGE/BROWSER

2. Create a new bucket, specifying the following options:
   - A unique name of your choosing.
   - Default storage class: `Standard`
   - Location: Specify a bucket location in the same region where you plan to create your TPU node. See TPU types and zones (https://cloud.google.com/tpu/docs/types-zones#types) to learn where various TPU types are available.

Use the `ctpu` tool

This section demonstrates using the Cloud TPU provisioning tool (https://github.com/tensorflow/tpu/tree/master/tools/ctpu) (`ctpu`) for creating and managing Cloud TPU project resources. The resources are comprised of a virtual machine (VM) and a Cloud TPU resource that have the same name. **These resources must reside in the same region/zone as the bucket you just created.**

You can also set up your VM and TPU resources using `gcloud` commands or through the Cloud Console (https://console.cloud.google.com/). For more information, see the creating and deleting TPUs (https://cloud.google.com/tpu/docs/creating-deleting-tpus) page for details.

Run `ctpu up` to create resources

1. Open a Cloud Shell window.

   OPEN CLOUD SHELL (HTTPS://CONSOLE.CLOUD.GOOGLE.COM/?CLOUDSHELL=TRUE)

2. Run `gcloud config set project <Your-Project>` to use the project where you want to create Cloud TPU.

3. Run `ctpu up` specifying the flags shown for either a Cloud TPU device or Pod slice. Refer to CTPU Reference (https://cloud.google.com/tpu/docs/ctpu-reference) for flag options and descriptions.
4. Set up either a Cloud TPU device or a Pod slice:

TPU DEVICE | TPU POD

Set up a Cloud TPU device:

```
$ ctpu up
```

Note: If you have more than one project, you must specify the project name.

The following configuration message appears:

```
ctpu will use the following configuration:

Name: [your TPU's name]
Zone: [your project's zone]
GCP Project: [your project's name]
TensorFlow Version: 1.14
VM:
  Machine Type: [your machine type]
  Disk Size: [your disk size]
  Preemptible: [true or false]
Cloud TPU:
  Size: [your TPU size]
  Preemptible: [true or false]

OK to create your Cloud TPU resources with the above configuration? [Yn]:
```

Press `y` to create your Cloud TPU resources.

The `ctpu up` command creates a virtual machine (VM) and Cloud TPU services.

From this point on, a prefix of `(vm)$` means you should run the command on the Compute Engine VM instance.

Verify your Compute Engine VM

When the `ctpu up` command has finished executing, verify that your shell prompt is `username@tpuname`, which shows you are logged into your Compute Engine VM.

Export the storage bucket

To specify a storage bucket to use for storing checkpoints during training and for writing training logs, set up the STORAGE_BUCKET environment variable, replacing `YOUR-BUCKET-NAME` with the name of your Cloud Storage bucket:

```
(vm)$ export STORAGE_BUCKET=gs://YOUR-BUCKET-NAME
```

The training application expects your training data to be accessible in Cloud Storage.

(Optional) Set up TensorBoard

TensorBoard offers a suite of tools (https://cloud.google.com/tpu/docs/cloud-tpu-tools) designed to present TensorFlow data visually. When used for monitoring, TensorBoard can help identify bottlenecks in processing and suggest ways to improve performance.

If you don't need to monitor the model's output at this time, you can skip the TensorBoard setup steps.

If you want to monitor the model's output and performance, follow the guide to setting up TensorBoard (https://cloud.google.com/tpu/docs/tensorboard-setup).

Run the MnasNet model with fake_imagenet

In the following steps, a prefix of `(vm)$` means you should run the command on your Compute Engine VM:

1. Add the top-level `/models` folder to the Python path with the command

   ```
   (vm)$ export PYTHONPATH="$PYTHONPATH:/usr/share/tpu/models"
   ```

2. Navigate to the directory:

```
(vm)$ cd /usr/share/tpu/models/official/mnasnet/
```

3. Run the training script for either a single Compute Engine device or Pod as follows:

TPU DEVICE TPU POD

```
(vm)$ python mnasnet_main.py \
  --tpu=${TPU_NAME} \
  --data_dir=gs://cloud-tpu-test-datasets/fake_imagenet \
  --model_dir=${STORAGE_BUCKET}/mnasnet \
  --model_name='mnasnet-a1' \
  --skip_host_call=true \
  --train_batch_size=1024
```

- `--tpu` specifies the name of the Cloud TPU. Note that `ctpu` passes this name to the Compute Engine VM as an environment variable (`TPU_NAME`).
- `--data_dir` specifies the Cloud Storage path for training input.
- `--model_dir` specifies the directory where checkpoints and summaries are stored during model training. If the folder is missing, the program creates one. When using a Cloud TPU, the `model_dir` must be a Cloud Storage path (`gs://...`). You can reuse an existing folder to load current checkpoint data and to store additional checkpoints as long as the previous checkpoints were created using TPU of the same size and Tensorflow version.

For a single Cloud TPU device, the procedure trains the MnasNet model ('mnasnet-a1' variant) for 350 epochs and evaluates every fixed number of steps. Using the specified flags, the model should train in about 23 hours. With real imagenet data, the settings will reproduce the state-of-the-art research result, while users should be able to tune up the training speed.

Using the full Imagenet dataset

This tutorial uses a demonstration version of the full ImageNet dataset, referred to as *fake_imagenet*. These demonstration versions allow you to test the tutorials, while reducing the storage and time requirements typically associated with running a model against the full ImageNet dataset. If you want to see how the model runs against the full ImageNet dataset, follow these instructions.

Verify space requirements

You need about 300GB of space available on your local machine or VM to use the full ImageNet dataset.

Note: If you use `ctpu up` to set up your VM, it will allocate 250GB by default.

You can increase the size of the VM disk using one of the following methods:

- Specify the `--disk-size-gb` flag on the `ctpu up` command line with the size, in GB, that you want allocated.
- Follow the Compute Engine guide to add a disk (https://cloud.google.com/compute/docs/disks/add-persistent-disk) to your VM.
    - Set **When deleting instance** to **Delete disk** to ensure that the disk is removed when you remove the VM.
    - Make a note of the path to your new disk. For example: `/mnt/disks/mnt-dir`.

Download and convert the ImageNet data

Note: For the following commands, a prefix of (vm) means you should run the command on the Compute Engine VM instance. If a command does not have the (vm) prefix, run it on your local workstation.

1. Sign up for an ImageNet account (http://image-net.org/signup). Remember the username and password you used to create the account.
2. Set up a `DATA_DIR` environment variable pointing to a path on your Cloud Storage bucket:

```
(vm)$ export DATA_DIR=gs://storage-bucket
```

3. Download the `imagenet_to_gcs.py` script from GitHub:

```
$ wget https://raw.githubusercontent.com/tensorflow/tpu/master/tools/datasets/i
```

4. Set a `SCRATCH_DIR` variable to contain the script's working files. The variable must specify a location on your local machine or on your Compute Engine VM. For example, on your local machine:

```
$ SCRATCH_DIR=./imagenet_tmp_files
```

Or if you're processing the data on the VM:

```
(vm)$ SCRATCH_DIR=/mnt/disks/mnt-dir/imagenet_tmp_files
```

5. Run the `imagenet_to_gcs.py` script to download, format, and upload the ImageNet data to the bucket. Replace `[USERNAME]` and `[PASSWORD]` with the username and password you used to create your ImageNet account.

```
$ pip install google-cloud-storage
$ python imagenet_to_gcs.py \
  --project=$PROJECT \
  --gcs_output_path=$DATA_DIR \
  --local_scratch_dir=$SCRATCH_DIR \
  --imagenet_username=[USERNAME] \
  --imagenet_access_key=[PASSWORD]
```

Optionally if the raw data, in JPEG format, has already been downloaded, you can provide a direct `raw_data_directory` path. If a raw data directory for training or validation data is provided, it should be in the format:

- Training images (https://github.com/awslabs/deeplearning-benchmark/blob/master/tensorflow/inception/inception/data/build_imagenet_data.py) : train/n03062245/n03062245_4620.JPEG
- Validation images (https://github.com/tensorflow/models/blob/master/research/inception/inception/data/preprocess_imagenet_validation_data.py) : validation/ILSVRC2012_val_00000001.JPEG
- Validation labels (http://data.dmlc.ml/mxnet/models/imagenet/synset.txt): synset_labels.txt

The training subdirectory names (for example, n03062245) are "WordNet IDs" (wnid). The ImageNet API (http://www.image-net.org/download-API) shows the mapping of WordNet IDs to their associated validation labels in the `synset_labels.txt` file. A synset in this context is a visually-similar group of images.

Note: Downloading and preprocessing the data can take 10 or more hours, depending on your network and computer speed. Do not interrupt the script.

When the script finishes processing, a message like the following appears:

```
2018-02-17 14:30:17.287989: Finished writing all 1281167 images in data set.
```

The script produces a series of directories (for both training and validation) of the form:

```
${DATA_DIR}/train-00000-of-01024
${DATA_DIR}/train-00001-of-01024
 ...
${DATA_DIR}/train-01023-of-01024
```

and

```
${DATA_DIR}/validation-00000-of-00128
S{DATA_DIR}/validation-00001-of-00128
 ...
${DATA_DIR}/validation-00127-of-00128
```

After the data has been uploaded to your Cloud bucket, run your model and set `--data_dir=${DATA_DIR}`.

If you used the full ImageNet dataset to train the model, proceed to clean up (#clean-up).

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
   ```

   Your prompt should now be `user@projectname`, showing you are in the Cloud Shell.

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 [optional: --zone]
   ```

★ **Important:** If you set the TPU resources name when you ran `ctpu up`, you must specify that name with the `--name` flag when you run `ctpu delete` in order to shut down your TPU resources.

3. Run `ctpu status` 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:

```
2018/04/28 16:16:23 WARNING: Setting zone to "us-central1-b"
No instances currently exist.
        Compute Engine VM:     --
        Cloud TPU:             --
```

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

```
$ gsutil rm -r gs://YOUR-BUCKET-NAME
```

Note: For free storage limits and other pricing information, see the Cloud Storage pricing guide (https://cloud.google.com/storage/pricing).

What's next

- Learn more about `ctpu` (https://github.com/tensorflow/tpu/tree/master/tools/ctpu), including how to install it on a local machine.
- Explore the TPU tools in TensorBoard (https://cloud.google.com/tpu/docs/cloud-tpu-tools).

Except as otherwise noted, the content of this page is licensed under the Creative Commons Attribution 4.0 License (https://creativecommons.org/licenses/by/4.0/), and code samples are licensed under the Apache 2.0 License (https://www.apache.org/licenses/LICENSE-2.0). For details, see our Site Policies (https://developers.google.com/terms/site-policies). Java is a registered trademark of Oracle and/or its affiliates.

Last updated December 4, 2019.