This tutorial demonstrates how to run the Mask RCNN (https://arxiv.org/abs/1703.06870) model using Cloud TPU with the COCO (http://cocodataset.org) dataset.

Mask RCNN is a deep neural network designed to address object detection and image segmentation, one of the more difficult computer vision challenges.

The Mask RCNN model generates bounding boxes and segmentation masks for each instance of an object in the image. The model is based on the Feature Pyramid Network (FPN) (https://arxiv.org/abs/1612.03144) and a ResNet50 (/tpu/docs/tutorials/resnet) backbone.

This tutorial uses `tf.contrib.tpu.TPUEstimator` (https://www.tensorflow.org/api_docs/python/tf/estimator/Estimator) to train the model. The TPUEstimator API is a high-level TensorFlow API and is the recommended way to build and run a machine learning model on Cloud TPU. The API simplifies the model development process by hiding most of the low-level implementation, which makes it easier to switch between TPU and other platforms such as GPU or CPU.

Before starting this tutorial, check that your Google Cloud project is correctly set up.

If you plan to train on a TPU Pod slice, review Training on TPU Pods (/tpu/docs/training-on-tpu-pods) to understand parameter changes required for Pod slices.

Before starting this tutorial, check that your Google Cloud project is correctly set up.

**Warning:** Mask RCNN uses a third-party dataset. Google provides no representation, warranty, or other guarantees about the validity, or any other aspects of this dataset.

- Create a Cloud Storage bucket to hold your dataset and model output
- Prepare the COCO dataset
- Set up a Compute Engine VM and Cloud TPU node for training and evaluation
- Run training and evaluation on a single Cloud TPU or a Cloud TPU Pod

This tutorial uses billable components of Google Cloud, including:

- Compute Engine
- Cloud TPU
- Cloud Storage

Use the pricing calculator (/products/calculator/) to generate a cost estimate based on your projected usage. New Google Cloud users might be eligible for a free trial (/free/).

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.

1. Open a Cloud Shell window.

   Open Cloud Shell (https://console.cloud.google.com/?cloudshell=true)

2. Create a variable for your project's name.

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

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

★ **Note:** In the following command, replace ***your-bucket-name*** with the name you want to assign to your bucket.

This Cloud Storage bucket stores the data you use to train your model and the training results.

Your Compute Engine VM, your Cloud TPU node and your Cloud Storage bucket should all be located in the same region.

5. Launch a Compute Engine VM using the `ctpu up` command.

★ **Note:** If you have more than one project, you must specify the project name. If `--name` is not specified, it defaults to your username.

After running `ctpu up`, a message appears showing your proposed configuration and you are prompted to confirm the configuration:

6. Press `y` to create your Compute Engine VM.

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.

If you are not connected to the Compute Engine instance, you can do so by running the following command:

★ **Note:** The first time you run `ctpu up` on a project it takes several minutes to perform startup tasks such as SSH key propagation and API turnup. Occasionally, during the SSH key propagation, an error message is returned. If that happens, rerun the `ctpu up` command to see if that clears the problem.

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

1. Run the `download_and_preprocess_coco.sh` script to convert the COCO dataset into a set of TFRecords (`*.tfrecord`) that the training application expects.

   This installs the required libraries and then runs the preprocessing script. It outputs a number of `*.tfrecord` files in your local data directory. The COCO download and conversion script takes approximately 1 hour to complete.

2. Copy the data to your Cloud Storage bucket

   After you convert the data into TFRecords, copy them from local storage to your Cloud Storage bucket using the `gsutil` command. You must also copy the annotation files. These files help validate the model's performance.

The training runs for 22,500 steps and takes approximately 5 hours on a v2-8 TPU node and approximately 3 and 1/2 hours on a v3-8 TPU node.

1. Run the following command to create your Cloud TPU.

| Parameter | Description |
| --- | --- |
| `tpu-size` | Specifies the size of the Cloud TPU. This tutorial uses a v3-8 TPU size for the single device training and evaluation. |
| `zone` | The zone where you plan to create your Cloud TPU. This should be the same zone you used for the Compute Engine VM. For example, `europe-west4-a`. |

After running `ctpu up` to start the Cloud TPU, a message appears showing your proposed configuration, including the Compute Engine VM. You are prompted to confirm the configuration:

2. Press `y` to create your Cloud TPU. It will take several minutes to create your Cloud TPU.

The Mask RCNN training application requires several extra packages. Install them now:

This tutorial requires a long-lived connection to the Compute Engine instance. To ensure you aren't disconnected from the instance, run the following command:

Next, you need to define several parameter values. You use these parameters to train and evaluate your model.

The variables you need to set are described in the following table:

| Parameter | Description |
| --- | --- |
| `STORAGE_BUCKET` | This is the name of the Cloud Storage bucket that you created in the **Before you begin** section. |
| `TPU_NAME` | This is the name of the Compute Engine VM and the the Cloud TPU. The Compute Engine VM and the Cloud TPU name must be the same. Since the Compute Engine VM was set to the default value, your username, set the Cloud TPU to the same value. |
| `ACCELERATOR_TYPE` | This is the accelerator version (/tpu/docs/supported-versions#supported-tpu) and the number of cores you want to use, for example, v2-128 (128 cores). In this tutorial, you will use a v3-8 TPU type when training on a single TPU device. |
| `GCS_MODEL_DIR` | This is the directory that contains the model files. This tutorial uses a folder within the Cloud Storage bucket. You do not have to create this folder beforehand. The script creates the folder if it does not already exist. |

| | |
|---|---|
| `CHECKPOINT` | This variable specifies a pre-trained checkpoint. The Mask RCNN model requires a pre-trained image classification model, such as ResNet, to be used as a backbone network. This tutorial uses a pre-trained checkpoint created with the ResNet demonstration model (/tpu/docs/tutorials/resnet). You can also train your own ResNet model and specify a checkpoint from your ResNet model directory. |
| `PATH_GCS_MASKRCNN` | This is the address of your Cloud Storage bucket where you want to store your model training artifacts. As with the `GCS_MODEL_DIR` variable, this tutorial uses a folder in the Cloud Storage bucket. |

Use the `export` command to set these variables.

**Note:** In the following code example, replace ***your-bucket-name*** with the name of your bucket and set the TPU_NAME variable to your username.

You are now ready to run the model on the preprocessed COCO data. To run the model, you use the `mask_rcnn_main.py` script.

1. First, add the top-level `/models` folder to the Python path with the command:

2. Run the following command to run both the training and evaluation.

From here, you can either conclude this tutorial and clean up (#cleanup) your GCP resources, or you can further explore running the model on a Cloud TPU Pod.

You can get results faster by scaling your model with Cloud TPU Pods. The fully supported Mask RCNN model can work with the following Pod slices:

- v2-32
- v3-32

When working with Cloud TPU Pods, you first train the model using a Pod, then use a single Cloud TPU device to evaluate the model.

The training runs for 11,250 steps and takes approximately 2 hours on a v3-32 TPU node.

**Note:** If you have already deleted your Compute Engine instance, create a new one following the steps in Set up your resources (#set_up_your_resources).

1. Delete the Cloud TPU resource you created for training the model on a single Cloud TPU device.

2. Run the `ctpu up` command, using the `tpu-size` parameter to specify the Pod slice you want to use. For example, the following command uses a v2-32 Pod slice.

3. Install the extra packages needed by Mask RCNN.

4. Update the keepalive values of your VM connection.

   This tutorial requires a long-lived connection to the Compute Engine instance. To ensure you aren't disconnected from the instance, run the following command:

5. Define the variables you need for training on a Pod. Use the `export` command to create multiple bash variables and use them in a configuration string.

★ **Note:** In the following code example, replace ***your-bucket-name*** with the name of your bucket and set the TPU_NAME variable to your username.

6. Add the top-level `/models` folder to the Python path.

7. Start the training script.

In this step, you use a single Cloud TPU node to evaluate the above trained model against the COCO dataset. The evaluation takes about 10 minutes.

1. Delete the Cloud TPU resource you created to train the model on a Pod.

2. Launch a new TPU device to run evaluation.

3. Install the packages necessary to run the evaluation.

4. Create your environmental variables.

★ **Note:** In the following code example, replace ***your-bucket-name*** with the name of your bucket and set the TPU_NAME variable to your username.

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

2. Start the evaluation.

To avoid incurring charges to your Google Cloud Platform account for the resources used in this tutorial:

1. Disconnect from the Compute Engine instance, if you have not already done so:

   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:

★ **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 the following command to verify the Compute Engine VM and Cloud TPU have been shut down:

The deletion might take several minutes. A response like the one below indicates there are no more allocated instances:

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

- 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 (/tpu/docs/cloud-tpu-tools).