Alpha

This product or feature is in a pre-release state and might change or have limited support. For more information, see the product launch stages (/products/#product-launch-stages).

Transformer is a neural network architecture that solves sequence to sequence problems using attention mechanisms. Unlike traditional neural seq2seq models, Transformer does not involve recurrent connections. The attention mechanism learns dependencies between tokens in two sequences. Since attention weights apply to all tokens in the sequences, the Transformer model is able to easily capture long-distance dependencies.

Transformer's overall structure follows the standard encoder-decoder pattern. The encoder uses self-attention to compute a representation of the input sequence. The decoder generates the output sequence one token at a time, taking the encoder output and previous decoder-outputted tokens as inputs.

The model also applies embeddings on the input and output tokens, and adds a constant positional encoding. The positional encoding adds information about the position of each token.

Warning: 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.

This tutorial uses billable components of Google Cloud, including:

- Compute Engine
- Cloud TPU

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/).

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

Before starting this tutorial, follow the steps below to check that your Google Cloud project is correctly set up.

Important: To reduce network latency and network costs, set up your Compute Engine VM and your Cloud TPU node in the same time zone.

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 ***bucket-name*** with the name you want to assign to your bucket.

5. Launch a Compute Engine VM using the `ctpu up` command. This example sets the zone to europe-west4-a, but you can set it to whatever zone you are planning to use for the Compute Engine VM and Cloud TPU.

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

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.

★ **Note:** If you are not connected to the Compute Engine VM instance, you can connect by running the following command, replacing ***vm-name*** with the name of your VM:

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

On your Compute Engine VM:

1. Create the following environment variables:

Parameter	Description
PARAM_SET	Parameter set to use when creating and training the model. Options are `base` and `big` (default).
DATA_DIR	Specifies the directory where training data are stored.
VOCAB_FILE	Path to subtoken vocabulary file. If data_download is used, the file will be in `data_dir`.

2. Change directory to the training directory:

3. Generate the training dataset:

`data_download.py` downloads and preprocesses the training and evaluation WMT datasets. After the data is downloaded and extracted, the training data is used to generate a vocabulary of subtokens. The evaluation and training strings are tokenized, and the resulting data is sharded, shuffled, and saved as TFRecords.

1.75GB of compressed data is downloaded. In total, the raw files (compressed, extracted, and combined files) take up 8.4GB of disk space. The resulting TFRecord and vocabulary files are 722MB. The script takes around 40 minutes to run, with the bulk of the time spent downloading and ~15 minutes spent on preprocessing.

To run the Transformer model on a TPU, you must set `--distribution_strategy=tpu`, `--tpu=$TPU_NAME`, and `--use_ctl=True` where `$TPU_NAME` is the name of your Compute Engine VM that you specified with `ctpu up`. The Cloud TPU should be the same as the VM name.

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

Run the following commands on your Compute Engine VM:

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

Parameter	Description
`tpu-size`	Specifies the type of Cloud TPU, for example v3-8.
`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`.

2. The configuration you specified appears. Enter y to approve or n to cancel.

   You will see a message: `Operation success; not ssh-ing to Compute Engine VM due to --tpu-only flag`. Since you previously completed SSH key propagation, you can ignore this message.

3. Set the Cloud TPU name variable. This will either be a name you specified with the `--name` parameter to `ctpu up` or the default, your username:

4. Run the training script:

By default, the model will evaluate after every 2000 steps. In order to train to convergence, change `train_steps` to 200000.

This training takes approximately 7 minutes on a v3-8 Cloud TPU. When the training and evaluation complete, a message similar to the following appears:

- Training with steps:
    - `--train_steps`: Sets the total number of training steps to run.
    - `--steps_between_evals`: Number of training steps to run between evaluations.

Use these flags to compute the BLEU when the model evaluates:

- `--bleu_source`: Path to file containing text to translate.
- `--bleu_ref`: Path to file containing the reference translation.

When running English to German translation use the flags: `--bleu_source=$DATA_DIR/newstest2014.en --bleu_ref=$DATA_DIR/newstest2014.de`

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 Transformer model can work with the following Pod slices:

- v2-32
- v3-32

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. 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 `tpu-only` flag and the `--zone` flag you used when you set up the Cloud TPU. This deletes only your Cloud TPU.

3. Reconnect to the Compute Engine instance by running the following command, replacing ***vm-name*** with the name of your VM:

4. 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.

5. Export storage bucket variables:

6. Change directory to the training directory:

7. Run the Pod training script:

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 ***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 more [Tensor2Tensor models for TPU][t2tcloudtpu].
- Experiment with more TPU samples (/tpu/docs/tutorials/).
- Explore the TPU tools in TensorBoard (/tpu/docs/cloud-tpu-tools).