Train Faster:
A Guide to TensorFlow Performance Optimization

Shriya Balaji Palsamudram, 04-12-2021
INTRODUCTION
Is optimization worth your time?

➤ Utilize resources optimally
➤ Get results faster
➤ Apply simple strategies to most models
➤ Add very few lines of code
Automatic Mixed Precision

Do we really need FP32 everywhere?

- Uses lower precision (e.g. FP16) where applicable (e.g. convolutions, matrix multiplies)
- Keeps certain operations in FP32
- Achieves the same accuracy as FP32 training using all the same hyper-parameters
- Reduces memory requirements
- Accelerates memory-intensive operations
Code Snippets

**TF 1.x**

```python
opt = tf.train.experimental.enable_mixed_precision_graph_rewrite(opt)
```

**TF 2.x**

```python
policy = tf.keras.mixed_precision.Policy('mixed_float16')
tf.keras.mixed_precision.set_policy(policy)
```

use mixed_bfloat16 when training with TPUs
NHWC

Which is the faster data format?

- Are you using AMP?
- Do you have convolutions in your model?
- Do you want to improve your tensor core utilization?
- Do you want to eliminate unnecessary data layout transposes?

```python
tf.keras.layers.Conv2D(..., data_format="channels_last")
```
Auto-clustering

Why run only one node at a time?

Use XLA to fuse kernels!

➤ Levels: -1 (disable), 1, 2
➤ Higher level = XLA is more aggressive

Note: XLA does not perform well for models with dynamic shapes (example: transformer) as XLA would have to compile a new kernel for every shape

More information: XLA Best practices
Bias

Why add bias when you do not need it?

Batch norm basically shifts the values by their mean and this shift removes the need to have a constant bias term.
Batch Size

Why not utilize your resources well?

<table>
<thead>
<tr>
<th>Training Efficient-Det D0 on a Tesla V100 32GB machine</th>
<th>Case #1</th>
<th>Case #2</th>
<th>Case #3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Batch size</td>
<td>1</td>
<td>16</td>
<td>64</td>
</tr>
<tr>
<td>Training speed (frames per second)</td>
<td>2 FPS</td>
<td>22 FPS</td>
<td>70 FPS</td>
</tr>
<tr>
<td>Training time for 1 coco epoch</td>
<td>16 hours</td>
<td>1.5 hours</td>
<td>28 mins</td>
</tr>
</tbody>
</table>

Note: Increasing batch size will change the model’s accuracy so the model needs to be scaled by tuning hyperparameters like learning rate, etc to meet the target accuracy.

Use nvidia-smi query to investigate memory utilization

```
nvidia-smi --query-gpu=utilization.memory,memory.total,memory.free,memory.used --format=csv
```
Threading

Why not use threads to improve performance?

**Intra-op:** Multiple threads → One operation

**Inter-op:** Independent operations → Run concurrently

Note: These threads depend on the number of CPU threads available

**GPU Thread mode:** Reduces kernel launch time during training

**GPU Thread count:** # Threads per GPU
Code Snippets

Parallel Threads

```python
import multiprocessing
config.intra_op_parallelism_threads = 1
config.inter_op_parallelism_threads = max(2, (multiprocessing.cpu_count() // hvd.size()) - 2)
```

GPU Threads

```bash
TF_GPU_THREAD_MODE=gpu_private
TF_GPU_THREAD_COUNT=1
```
Data Pipeline Optimization

Why not demystify the biggest bottleneck?

Use tf.data API !!!!!

Note: Benchmark only the data loader without the training and backpropagation steps to quantify the effect of the optimizations independently.
Data Pipeline Optimization
Why not demystify the biggest bottleneck?

```
tf.data.Dataset.interleave
```

Read TFRecords in parallel
Data Pipeline Optimization

Why not demystify the biggest bottleneck?

```
num_parallel_calls=tf.data.experimental.AUTOTUNE
```

Run operations in parallel
Data Pipeline Optimization

Why not demystify the biggest bottleneck?

dataset.shard()

Shard very early on
(While listing TFrecords)
Data Pipeline Optimization

Why not demystify the biggest bottleneck?

```python
dataset.prefetch(batch_size)
```

Prefetch right before returning the dataset
Data Pipeline Optimization

Why not demystify the biggest bottleneck?

```python
dataset.cache()
```

Only if the entire dataset can fit in memory
Data Pipeline Optimization
Why not demystify the biggest bottleneck?

tf.data.Options()

```python
options = tf.data.Options()
options.experimental_optimization.map_vectorization.enabled = True
options.experimental_optimization.map_parallelization = True
options.experimental_optimization.parallel_batch = True
```
Data Pipeline Optimization

Why not demystify the biggest bottleneck?

from_generator()

Do not use this!

Use TF operations instead to get better performance
Data Pipeline Optimization

Why not demystify the biggest bottleneck?

Do not use this!

py_func()

It cannot be serialized
Horovod

Why not make distributed training simple?

➤ Scale models easily
➤ Very few code changes
➤ Uses MPI model
➤ Its fast!
Understanding Horovod

How does horovod do it?

Horovod Mailbox

Tensor A

Tensor A

Tensor A

Tensor C

Tensor C

Tensor C

Tensor C

Tensor D

Tensor D

Tensor D

Tensor E

Horovod Clears Mailbox

Horovod Clears Mailbox

Horovod Clears Mailbox

Horovod Clears Mailbox

Horovod Clears Mailbox

Horovod Clears Mailbox

Horovod Clears Mailbox

Horovod Clears Mailbox

Horovod Clears Mailbox

Horovod Clears Mailbox

T = 0  T = 1  T = 2  T = 3  T = 4  T = 5  T = 6  T = 7  T = 8  T = 9  T = 10
Horovod Flags

How to get boost in performance?

HOROVOD_CYCLE_TIME

Set based on complexity of the model and training step duration
Horovod Flags
How to get boost in performance?

HOROVOD_TENSOR_THRESHOLD_FUSION

Buffer size to reduce multiple tensors
Horovod Flags
How to get boost in performance?

HOROVOD_AUTOTUNE

Let horovod find the best values based on your model

Note: Do not enable autotuning in production code as it affects the performance
Horovod Tensors

How to avoid time spent on transformation?

Convert sparse tensors to dense before allreduce

```python
optimizer = hvd.DistributedOptimizer(optimizer, sparse_as_dense=True)
```
Summary

Or is it the introduction again?

➤ Get results faster
➤ Utilize resources optimally
➤ Apply simple strategies to all models
➤ Add very few lines of code
Further Information

What next?

https://www.tensorflow.org
Further Information

What next?

- PyTorch Performance Tuning Guide [S31831]
- Algorithmic and Software Techniques to Optimize BERT Training and Inference [S31140]
- Profiling and Optimizing Deep Neural Networks with DLProf and PyProf [S31341]
- Deep Learning Performance Optimization with Profiling Tools [S31228]
- Fast, Accurate, Scalable: Building a Benchmark to Test Neural Time Series Models [S31950]
- Segmenting 3D Medical Images with nnU-Net and DGX A100, Using Tensor Cores [S31315]
Thank You