Source code

- https://github.com/huggingface/transformers
- commit 079bfb3

Quantization

- Using NVIDIA's internal quantization tool on top of PyTorch
- All quantizers are symmetric scale-only per-tensor
    - x_q = round(clip(s * x, -127, 127))

Model changes

- Added fake quantizers to the output of each embedding layer to simulate quantized embeddings:
    - bert.embeddings.word_embeddings
    - bert.embeddings.position_embeddings
      bert.embeddings.token_type_embeddings
- For each bert.encoder.layer added fake quantizers to the inputs / weights of the following modules / operations:
    - attention.self.query
    - attention.self.key
    - attention.self.value
    - attention.self attention_scores matmul
    - attention.self context_layer matmul
    - attention.output.dense
    - attention.output residual add operation
    - intermediate.dense
    - output.dense
    - output.dense residual add operation
- added a fake quantizer to the output of the final bert.encoder.layer

Calibration

- Weights: max range of each tensor
- Activation:
    - Histograms generated from 128 samples from the training set
    - ranges are chosen from the 99.99th percentile of histograms

Fine-tuning

```
 python -m torch.distributed.launch --nproc_per_node 8 run_squad.py
--model_type bert \
--model_name_or_path ./calib-percentile-99.99-128/bert-large-uncased
\
 --max_seq_length 384 \
 --doc_stride 128 \
--train_file $SQUAD_DIR/train-v1.1.json \
--predict_file $SQUAD_DIR/dev-v1.1.json \
--per_gpu_train_batch_size 2 \
--gradient_accumulation_steps 1 \
--do_train \
--do_lower_case \
--learning_rate 3e-5 \
--num_train_epochs 2 \
--fp16 \
--seed 0
```