Q8BERT: Quantized 8Bit BERT

Ofir Zafrir, Guy Boudoukh, Peter Izsak, Moshe Wasserblat
Intel AI Lab

Motivation

- BERT[1] shown great improvement in many Natural Language Processing (NLP) tasks
- BERT computational characteristics pose a challenge to deployment in real-time production environments
- 8bit fixed point supporting hardware and software exists and is already used in production environments

Method

- We propose to perform Quantization-Aware training[4] (QAT) while fine-tuning BERT
- 8bit Linear quantization has can accelerate inference by up to 4× using 25% of the memory footprint[3]

Quantization-Aware Training

- Quantization-aware training is a method of training a model to be quantized at the inference stage
- Fake quantization operation simulates the rounding effect of quantization

Quantization Scheme

We use symmetric linear quantization quantizing both weights and activations to 8bit Integers:

$$\boldsymbol{Quant(x|S^x, M) := Clamp(\lfloor x \times S^x \rceil, -M, M)}$$

$$\boldsymbol{Clamp(x, a, b) = \min(max(x, a), b)}$$

$$\boldsymbol{M = 2^{b-1} - 1}$$

Weights' scaling factor:

$$\boldsymbol{S^W = \frac{M}{\max(|W|)}}$$

Activations' scaling factor:

$$\boldsymbol{S^x = \frac{M}{EMA(max(|x|))}}$$

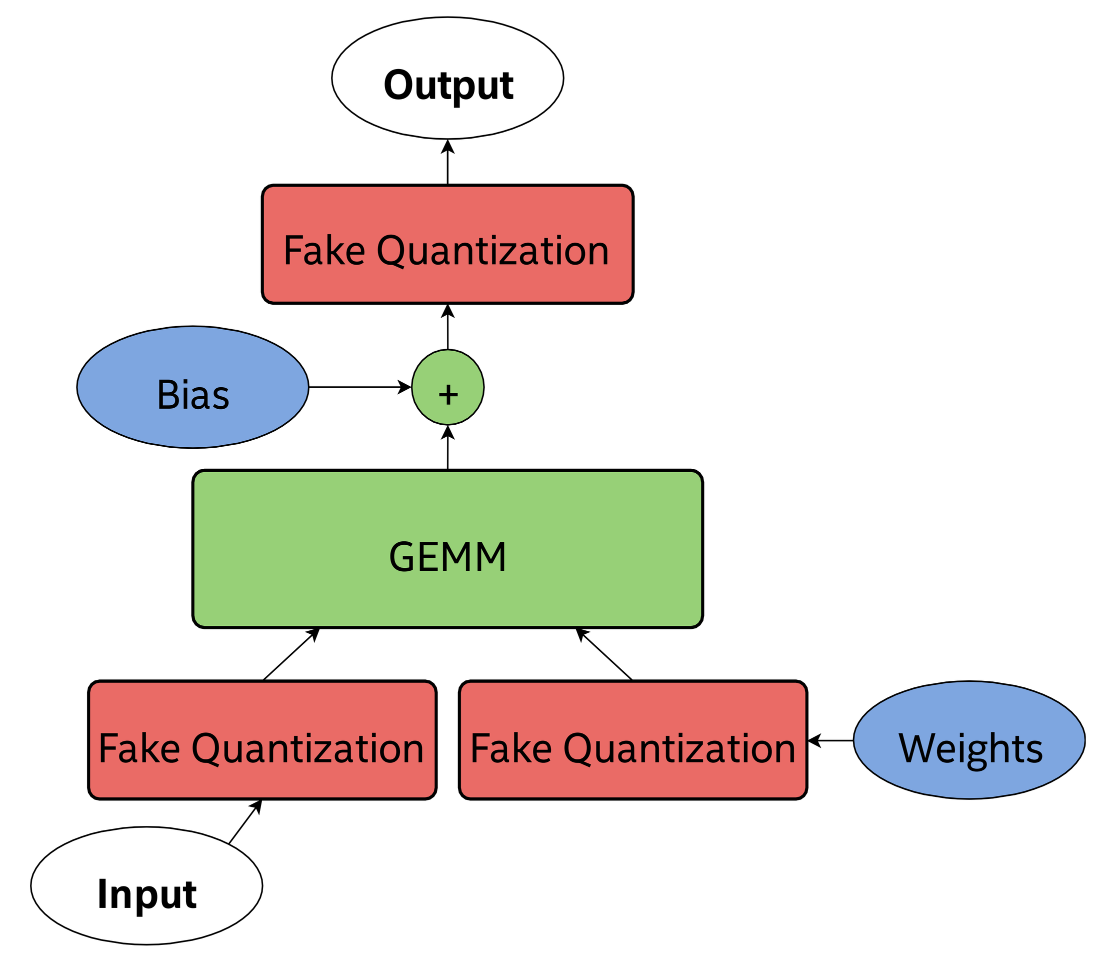
$\boldsymbol{EMA}$ – Exponential Moving Average

Straight-Through Estimator[2] is used to estimate the gradient of fake quantization:

$$\boldsymbol{\frac{\partial x^q}{\partial x} = \vec{1}}$$

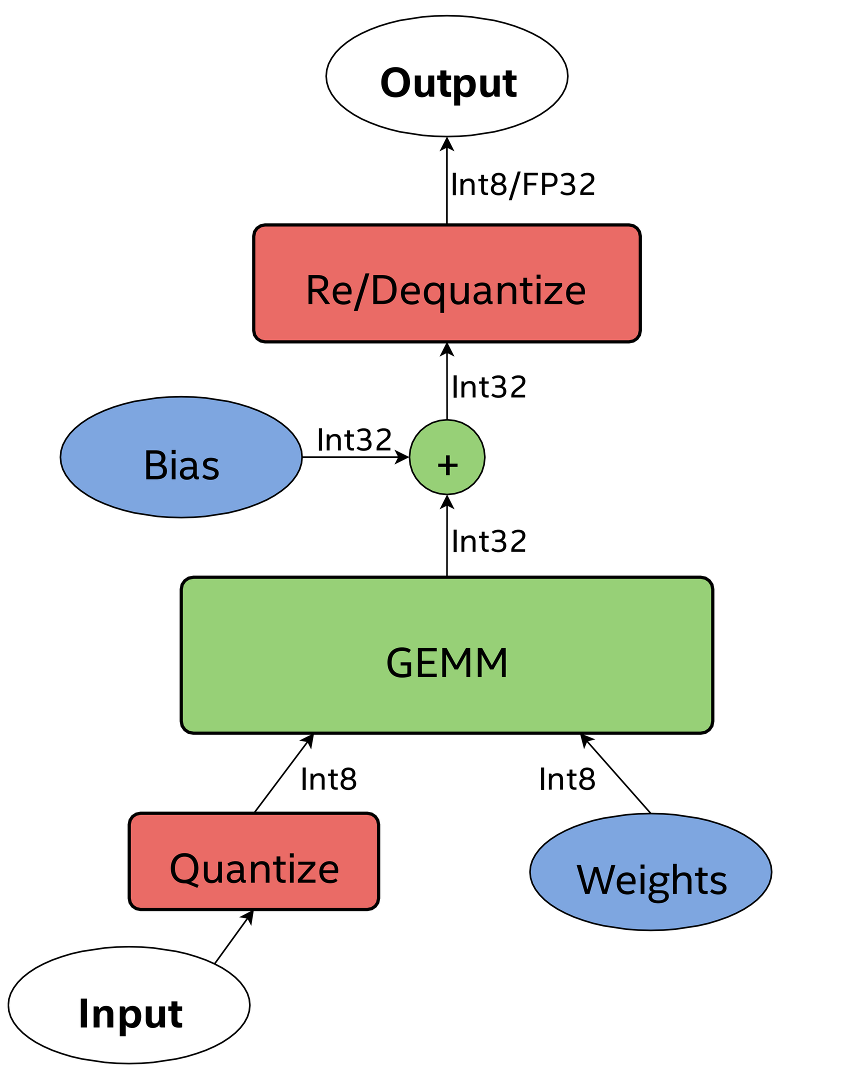
Operations that require higher precision are kept in FP32 during training and inference

Training Graph



The quantization at the output is removed where full precision is required

Inference Graph



This inference graph is the result of training with fake quantization

GLUE Benchmark and SQuAD Results

For each task we present the score of a baseline (FP32) model, of a QAT model quantized to 8bit. *L* means those models were trained with BERT-Large architecture.

Task	Metric	Baseline Score (STD)	*Our* 8bit BERT Score (STD)	Relative Error
CoLA	MCC	**58.48 (1.54)**	**58.48 (1.32)**	0.00%
MRPC	F1	**90.00 (0.23)**	89.56 (0.18)	-0.49%
MRPC-*L*	F1	90.86 (0.55)	**90.90 (0.29)**	0.04%
QNLI	Acc.	90.30 (0.44)	**90.62 (0.29)**	0.35%
QNLI-*L*	Acc.	91.66 (0.15)	**91.74 (0.36)**	0.09%
QQP	F1	87.84 (0.19)	**87.96 (0.35)**	0.14%
RTE	Acc.	**69.70 (1.50)**	68.78 (3.52)	-1.32%
SST-2	Acc.	**92.36 (0.59)**	92.24 (0.27)	-0.13%
STS-B	PCC	**89.62 (0.31)**	89.04 (0.17)	-0.65%
STS-B-*L*	PCC	**90.34 (0.21)**	90.12 (0.13)	-0.24%
SQuADv1.1	F1	**88.46 (0.15)**	87.74 (0.15)	-0.81%

Effect of Quantization-Aware Training

- Compare to post training quantization
- Dynamic Quantization (DQ)

$$\boldsymbol{S^x = \frac{M}{\max(|x|)}}$$

Results:

Task	Metric	Baseline Score (STD)	DQ 8bit BERT Score (STD)	Relative Error
CoLA	MCC	**58.48 (1.54)**	56.74 (0.61)	-2.98%
MRPC	F1	**90.00 (0.23)**	87.88 (2.03)	-2.36%
MRPC-*L*	F1	**90.86 (0.55)**	88.18 (2.19)	-2.95%
QNLI	Acc.	**90.30 (0.44)**	89.34 (0.61)	-1.06%
QNLI-*L*	Acc.	**91.66 (0.15)**	88.38 (2.22)	-3.58%
QQP	F1	**87.84 (0.19)**	84.98 (0.97)	-3.26%
RTE	Acc.	**69.70 (1.50)**	63.32 (4.58)	-9.15%
SST-2	Acc.	**92.36 (0.59)**	91.04 (0.43)	-1.43%
STS-B	PCC	**89.62 (0.31)**	87.66 (0.41)	-2.19%
STS-B-*L*	PCC	**90.34 (0.21)**	83.04 (5.71)	-8.08%
SQuADv1.1	F1	**88.46 (0.15)**	80.02 (2.38)	-9.54%

Conclusion

- We have presented a method for quantizing BERT GEMM operations to 8bit for a variety of NLP tasks with minimum loss in accuracy
- We compared our QAT method to Post-Training Quantization method and shown our method produces significantly better results.
- We made our work available for the community in the open-source library NLP Architect
- We encourage the community to use our quantization method to implement efficient BERT inference

1. J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. ArXiv, abs/1810.04805, 2018.
2. Y. Bengio, N. Léonard, and A. Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. arXiv preprint arXiv:1308.3432, 2013.
3. V. Vanhoucke, A. Senior, and M. Z. Mao. Improving the speed of neural networks on cpus. In Deep Learning and Unsupervised Feature Learning Workshop, NIPS 2011, 2011.
4. B. Jacob, S. Kligys, B. Chen, M. Zhu, M. Tang, A. Howard, H. Adam, and D. Kalenichenko. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 2704–2713, 2018.

NLP ARCHITECT

Paper

The 5th Workshop on Energy Efficient Machine Learning and Cognitive Computing @ NeurIPS 2019