Mixed Precision Training

Bash, Daniel, Hali, Shaheen

From zeros to heros

IEEE 754

1985: 32bit single-precision format
2008: 64but double-precision format

????: various uses of 16bit half-precision formats.
- signal processing
- graphics

Unreal

nVIDIA
GeFORCE™
FX

CNN:

- non-trivial accuracy loss

- post-training quantization. impractical for larger models.

- better accuracy when quantizing weight and activation parameters, by increasing their width by 2-3x

RNN:

- quantize weights and activation outputs
- Leave gradients as single-precision (minimize loss during backpropagation)

Current proposal

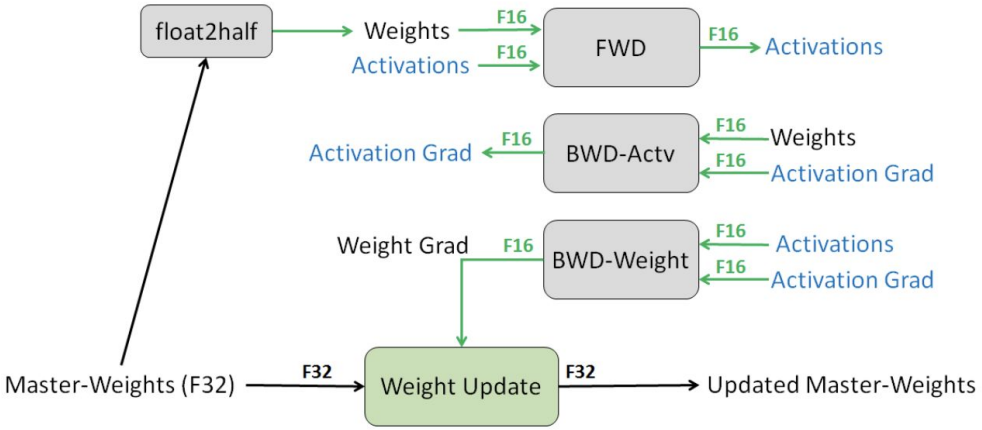
- All tensors and arithmetic are trained with quantization in mind
- no hyper-parameters (such as layer width) are adjusted
- no accuracy loss compared to single-precision baselines
- works on large datasets in variety of applications.

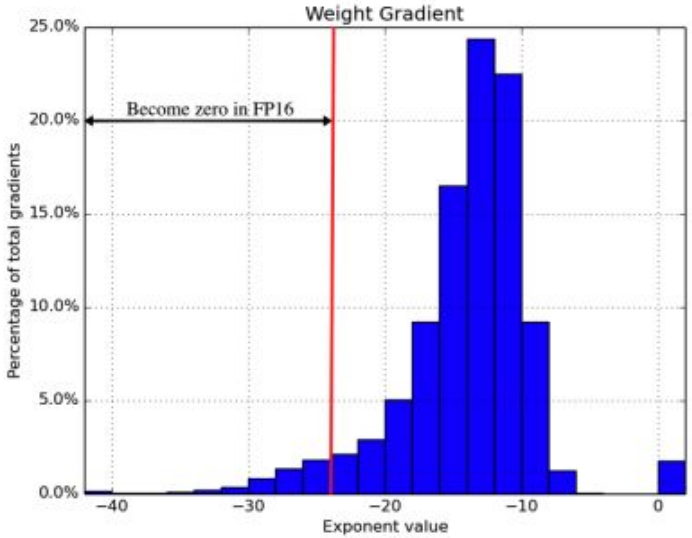
Implementation

- Goal: Train with FP16, match model accuracy in FP32
- Techniques used:
  - FP32 Master Weights
  - Loss Scaling
  - Arithmetic Precision

FP32 Master Weights

- “Full-fat” copy of weights stored
- Weight gradients stored in FP16
- Avoids two problems:
  - Gradients can become zero when multiplied with learning rate(~5%). Updating in FP32 avoids this
  - Very large weight, very small gradient (at end of training). Gradients can go to zero if ratio > 2048 (see addition via right-shifting bits)





Loss Scaling

- Much of upper end of range in FP16 unused by gradients
  - Shift them up
- Specifically:
  - Scale up FP16 loss value after forward pass
  - Backprop
  - Undo scaling before weight update
- Scaling factor = 8-32k
  - Just don't cause overflow during backprop

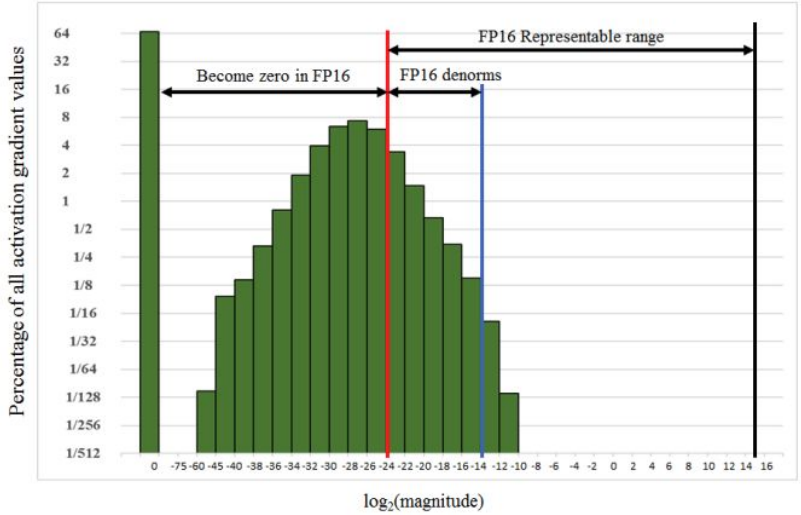


Figure 3: Histogram of activation gradient values during the training of Multibox SSD network.

Arithmetic Precision

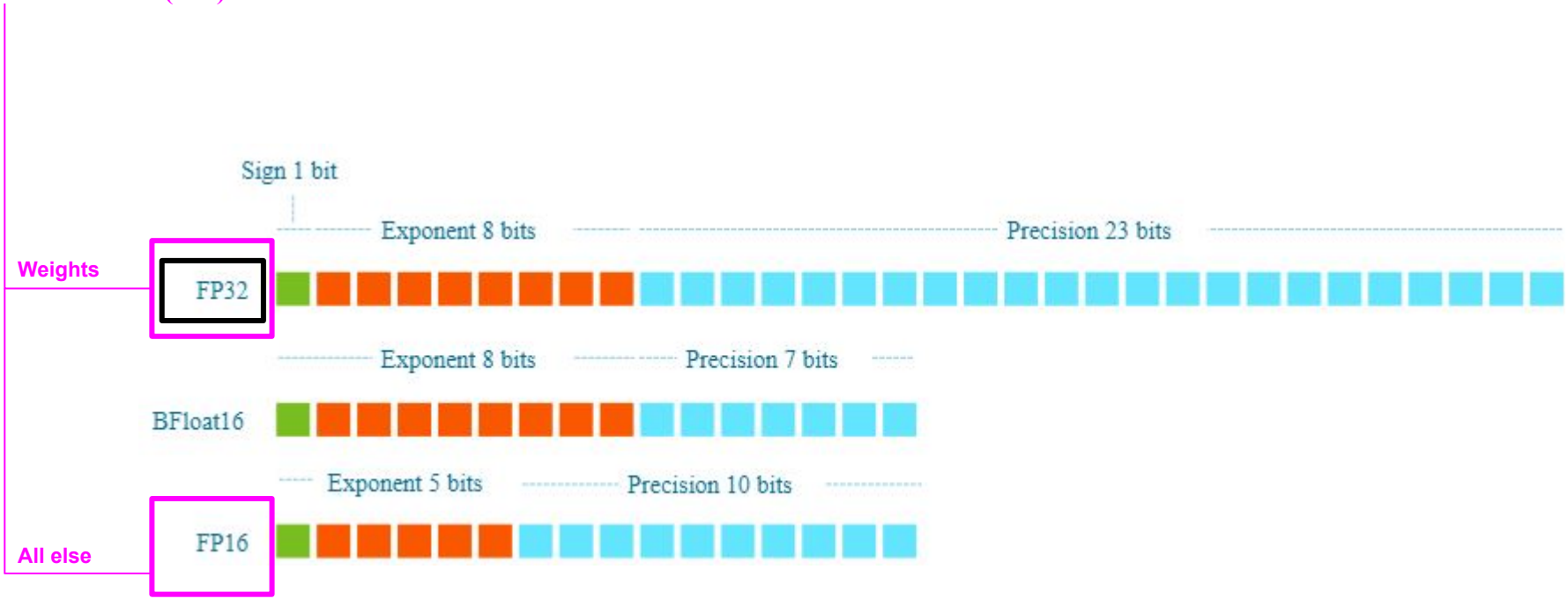
Different math operations require different precisions

- Vector dot products:
  - FP16 vectors, partial products in FP32
  - Otherwise accuracy degrades
  - Scale down to FP16 before writing to memory
- Large reductions (sums):
  - Use FP32, risk of high precision loss
- Point-wise operations (element-wise matrix products):
  - Lower risk of precision loss
  - Can use FP16 or FP32

For each application

Baseline (FP32)

Mixed Precision (MP):



Hardware

Baseline (FP32): **NVIDIA's Maxwell or Pascal GPU**

Mixed Precision (MP): **Volta V100 that accumulates FP16 products into FP32**

Mixed precision speech recognition experiments (Section 4.3) were conducted using Maxwell GPUs using FP16 storage only.

CNNs for ILSVRC: Classification Task

(ImageNet Large Scale Visual Recognition Challenge) for **image classification**

Loss-scaling was not required

Table 1: ILSVRC12 classification top-1 accuracy.

Model	Baseline	Mixed Precision	Reference
AlexNet	56.77%	56.93%	(Krizhevsky et al., 2012)
VGG-D	65.40%	65.43%	(Simonyan and Zisserman, 2014)
GoogLeNet (Inception v1)	68.33%	68.43%	(Szegedy et al., 2015)
Inception v2	70.03%	70.02%	(Ioffe and Szegedy, 2015)
Inception v3	73.85%	74.13%	(Szegedy et al., 2016)
Resnet50	75.92%	76.04%	(He et al., 2016b)

Detection CNNs - Regression + Classification Task

By losing small gradient values to zeros, as described in Section 3.2, poor weights are learned and training diverge

Table 2: Detection network average mean precision.

Model	Baseline	MP without loss-scale	MP with loss-scale
Faster R-CNN	69.1%	68.6%	69.7%
Multibox SSD	76.9%	diverges	77.1%

Speech Recognition - RNN - Classification

English mode: 115 million parameters

Mandarin model: 215 million parameters

Character Error Rate (CER)

Model/Dataset	Baseline	Mixed Precision
English	2.20	1.99
Mandarin	15.82	15.01

Machine Translation - Encoder/Decoder with LSTM cells

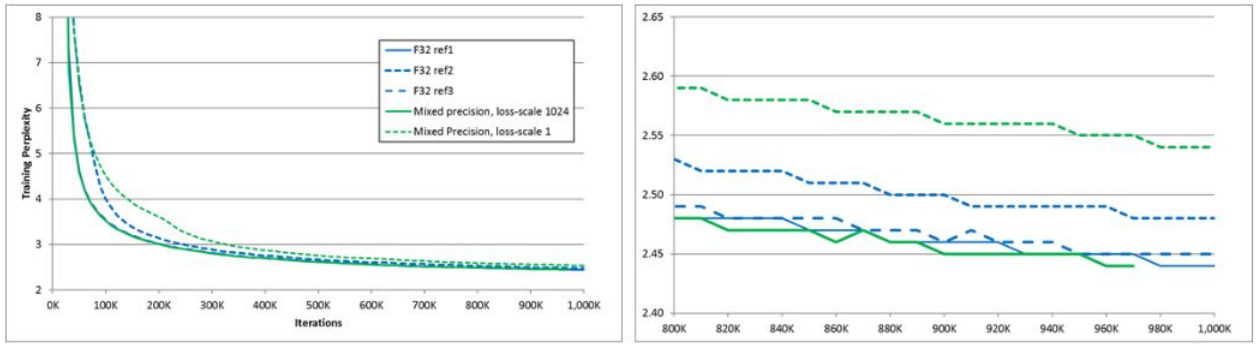
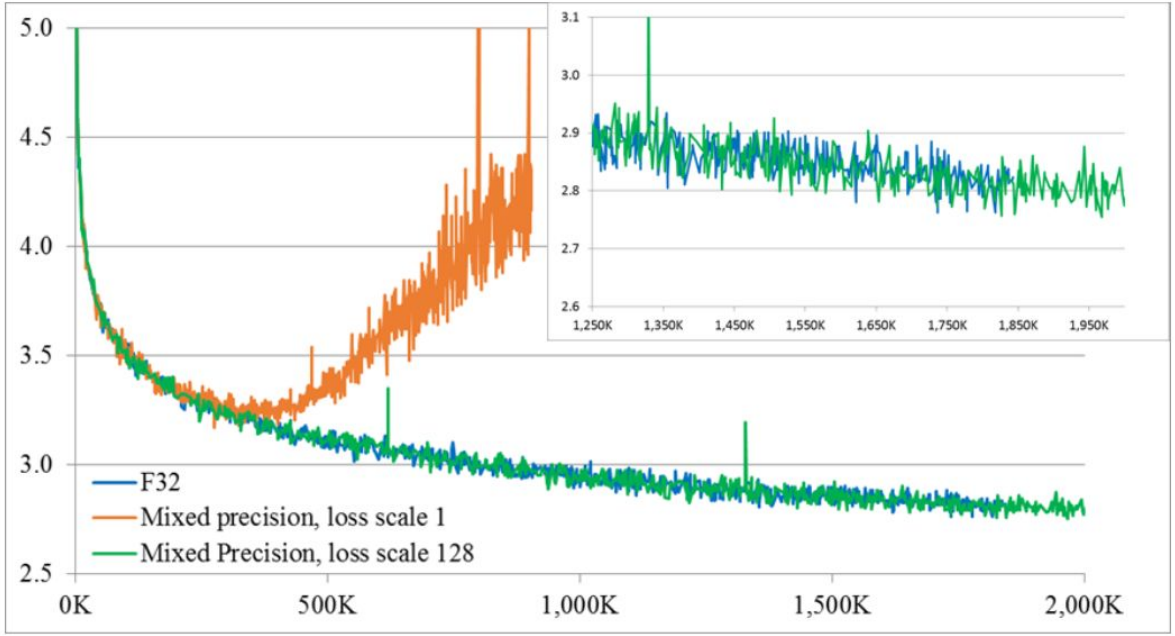


Figure 4: English to French translation network training perplexity, 3x1024 LSTM model with attention. Ref1, ref2 and ref3 represent three different FP32 training runs.

Language Modeling - bigLSTM

1 Billion word dataset



DCGAN - Generative Task

Generative Adversarial Networks (GANs): combine regression and discrimination tasks

Qualitatively the outputs of FP32 and mixed-precision training appear comparable. This network did not require loss-scaling to match FP32 results.

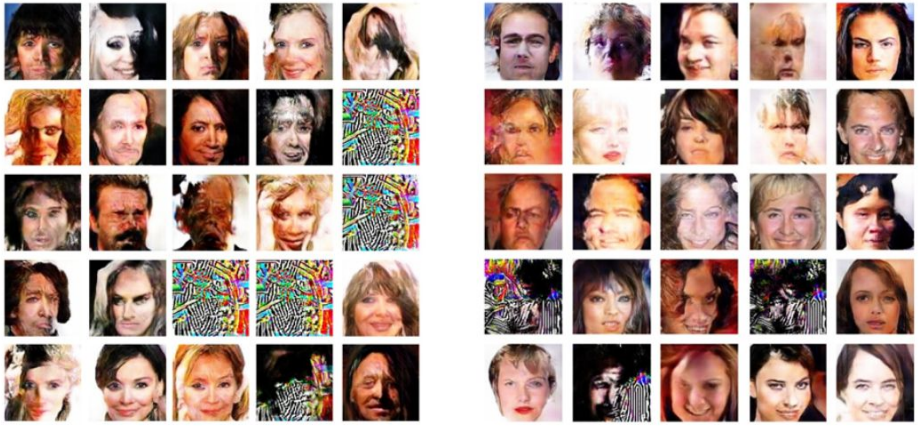


Figure 6: An uncurated set of face images generated by DCGAN. FP32 training (left) and mixed-precision training (right).

Conclusions and Future Work

- Constraints:
  - Memory bandwidth
  - Arithmetic bandwidth
  - Latency
- Speedups of 2-6x when limited by memory or arithmetic bandwidth
- Speedups are lower when operations are latency-limited
- Future work:
  - Full network training and inference speedups depend on library and framework optimizations
  - Include generative models
  - Automating loss-scaling factor selection
  -