PositNN: Training Deep Neural Networks with Mixed Low-Precision Posit

IEEE ICASSP 2021

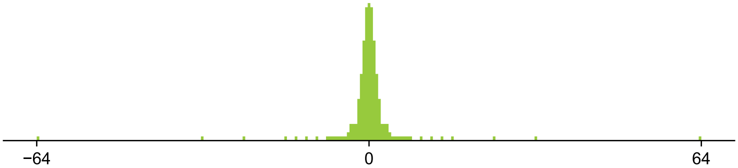
Gonçalo Raposo
goncalo.cascalho.raposo@tecnico.ulisboa.pt

Pedro Tomás
pedro.tomas@inesc-id.pt

Nuno Roma
nuno.roma@inesc-id.pt

INESC-ID, Instituto Superior Técnico, Universidade de Lisboa, Portugal

June, 2021



Introduction

Motivation

- Deep Learning (DL) requires lots of computing power and energy (e.g., training GPT-3 would cost $4.6M [1])
- Low-precision formats are an efficient way to reduce the memory footprint and power consumption
- The novel Posit format is designed as a direct drop-in replacement for the IEEE floating-point, providing higher accuracy in certain application domains for lower energy consumption [2]

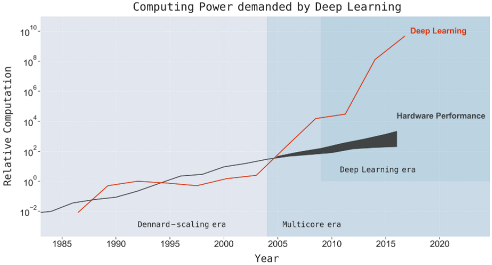


Figure: Growth of the computing power demanded by DL against the hardware performance [3].

Introduction

Related Work

Table: Related work regarding neural network training using Posits.

Montero et al. (2019) [4]	Langroudi et al. (2019) [5]	Lu et al. (2020) [6]	Murillo et al. (2020) [7]
▶ Trained a Fully Connected Neural Network (FCNN) using posits; ▶ Evaluated {32, 16, 14, 12, 10, 8}-bit posits; ▶ Irregular convergence for posit(10, 0) and posit(8, 0).	▶ Trained a FCNN using {32, 16}-bit posits; ▶ Evaluated the MNIST and Fashion MNIST datasets.	▶ Trained Convolutional Neural Networks (CNNs) using posits; ▶ 16-bit posits for the optimizer and last layer, and 8-bit posits everywhere else; ▶ Used floats for the 1st epoch and intermediate calculations.	▶ Trained CNNs using {32, 16}-bit posits and quires; ▶ Evaluated CIFAR-10 dataset; ▶ Posit(8, 0) did not converge.

X All works are unable to completely and properly train a DNN using posits smaller than 16 bits.

Posit Numbering System

- Proposed in 2017 by Dr. John L. Gustafson [8]
- Any precision/number of bits (nbits) and exponent size (es) – Posit(nbits, es)
- No overflow nor underflow and tapered precision – numbers near 1 are more accurate

$$p = (-1)^{\text{sign}} \times 2^{2^{es} \times k} \times 2^{\text{exponent}} \times (1 + \text{fraction})$$

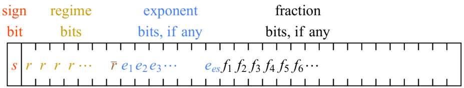


Figure: Generic posit format [8].

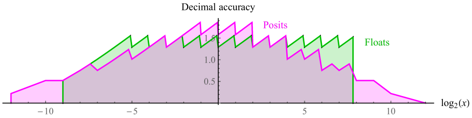


Figure: Comparison of posit(8, 1) and float decimal accuracies [8].

Deep Learning Posit Framework

PositNN Framework

- Popular DL frameworks: PyTorch and TensorFlow
- They do not natively support the novel posit format
- Supporting posits would require to reimplement most of their functions and operators

$\Downarrow$

- ✓ A new DL framework was developed – PositNN
- ✓ Supports posits and quires of any precision simulated via software with the Universal library [9]
- ✓ Implemented in C++ with multithreading support

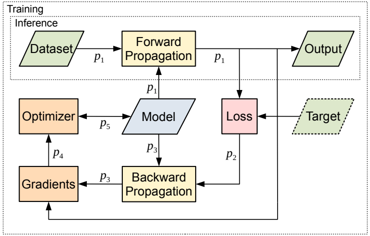


Figure: Block diagram of DNN training and inference.

Table: Supported functionalities of PositNN.

Posit Tensor	Layers	Activation Functions	Loss Functions	Optimizer
▶ Multidimensional arrays with posits ▶ Basic arithmetic operations ▶ Accumulate using quires ▶ Save and load to a binary file ▶ Convert from/to PyTorch tensor	▶ Linear: Equivalent to matrices operations ▶ Convolutional: Performs a convolution for a 3D input (e.g. image) ▶ Pooling operations ▶ Dropout	▶ ReLU ▶ Sigmoid ▶ TanH	▶ Mean Squared Error (MSE) ▶ Cross Entropy	▶ SGD: Momentum and Learning Rate (LR) scheduler

Training with Low-Precision Posits

Minimum Posit Precision

Training LeNet-5 on Fashion MNIST

- 16-bit posits achieved an accuracy equivalent to 32-bit floats
- 8-bit posits are unable to converge
- es=0 penalizes the achieved model accuracy (small dynamic range)

Table: Evaluation of how different posit precisions compare to 32-bit float for DNN training.

Format	Accuracy		
	$es = 0$	$es = 1$	$es = 2$
Float (FP32)		**90.28%**	
Posit16	88.23%	90.87%	90.55%
Posit12	66.66%	90.15%	90.26%
Posit10	19.86%	88.15%	88.52%
Posit9	11.65%	84.65%	82.50%
Posit8	10.00%	12.54%	12.55%

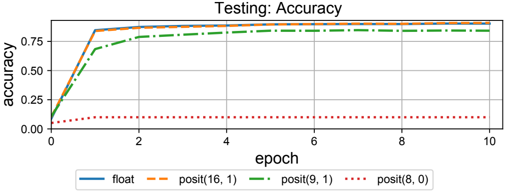


Figure: Training progress of posit and 32-bit float.

Training with Low-Precision Posits

Mixed Precision Configurations

Training LeNet-5 on Fashion MNIST

- The goal is to reduce the precision to 8 bits;
- The gradients decrease as the model converges – vanishing gradient problem;
- Insufficient dynamic range and resolution with narrow posit precisions;
- Increasing the optimizer and loss precisions to {16, 12}-bit posits is enough;

Table: Training with posit(8,2) for everything except optimizer and loss. Compared to 32-bit float.

Mixed Precision Configuration			
Optimizer (O)	Accuracy	Loss (L)	Accuracy
Float(FP32)	**90.28%**	**Float(FP32)**	**90.28%**
O16-L8_q	88.14%	O12-L16_q	90.03%
O12-L8_q	88.06%	O12-L12_q	90.07%
O10-L8_q	86.07%	O12-L10_q	90.13%
O9-L8_q	84.80%	O12-L9_q	89.35%
O8-L8_q	19.39%	O12-L8_q	88.0%

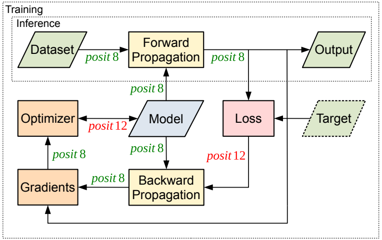


Figure: Mixed precision training configuration.

Experimental Evaluation

Experimental Setup

- Evaluation with various datasets and models
- DNN training using a mixed low-precision posit configuration (8-bit posit for everything except optimizer and loss)

Table: Considered datasets, models, and number of epochs used for training.

Dataset	Model	Epochs
MNIST	LeNet-5	10
Fashion MNIST	LeNet-5	10
CIFAR-10	CifarNet	20
CIFAR-100	CifarNet	20

Table: Configurations used for the training of the various CNNs. LR is for Learning Rate.

Loss	Optimizer	Initial LR	LR Scheduler	Momentum	Batch Size
Cross Entropy	SGD	$1/16$	Divide by 2 after every 4 epochs	0.5	64

Experimental Evaluation

DNN Training Evaluation

Table: Accuracy evaluation of using posits for training with mixed precision and various datasets and models. The obtained results were compared against the same models trained with 32-bit floats with PyTorch.

Format	MNIST (LeNet-5)	Fashion MNIST (LeNet-5)	CIFAR-10 (CifarNet)		CIFAR-100 (CifarNet)	
	Accuracy	Accuracy	Top-1	Top-3	Top-1	Top-5
Float (FP32)	**99.21%**	**90.28%**	**70.79%**	**92.64%**	**36.35%**	**66.92%**
Posit8 and O16-L16$_q$	99.19%	90.46%	71.30%	92.65%	35.41%	67.00%
Posit8 and O16-L12$_q$	99.17%	90.14%	71.09%	92.83%	35.27%	66.57%
Posit8 and O12-L12$_q$	99.20%	90.07%	68.28%	91.22%	25.85%	57.77%
Posit8 and O12-L10$_q$	99.17%	90.13%	68.41%	91.41%	25.37%	56.21%

- Mixed precision posit configuration allows to replace 32-bit floats for DNN training
- $85 - 95\%$ of the computations are performed with only 8-bit posits, $\sim 4\times$ less memory
- Langroudi et al. (2019) [5] observed an accuracy loss of $\sim 7\%$ for 16-bit floats

Conclusion

Main Contributions

- Proposed a new DNN framework – PositNN[1] – for training and inference using posits and quires
- Evaluated multiple CNNs and datasets using posits of various precisions
- 8-bit posits can replace 32-bit floats in a mixed precision configuration for DNN training (accuracy degradation $< 1\%$)

Future Work

- Evaluate these results in a hardware implementation of a posit unit, its critical path (time) and energy consumption (ongoing)
- Compare posits to other numerical formats, such as block floating-point
- Explore adapting the posit precision during run-time

[1] Available at: `https://github.com/hpc-ulisboa/posit-neuralnet`

References

[1] C. Li, "OpenAI's GPT-3 Language Model: A Technical Overview," June 2020. Accessed on 2020-10-13.

[2] R. Chaurasiya, J. Gustafson, R. Shrestha, J. Neudorfer, S. Nambiar, K. Niyogi, F. Merchant, and R. Leupers, "Parameterized Posit Arithmetic Hardware Generator," in *2018 IEEE 36th International Conference on Computer Design (ICCD)*, pp. 334–341, IEEE, Oct. 2018.

[3] N. C. Thompson, K. Greenewald, K. Lee, and G. F. Manso, "The Computational Limits of Deep Learning," *arXiv: 2007.05558*, July 2020.

[4] R. M. Montero, A. A. D. Barrio, and G. Botella, "Template-Based Posit Multiplication for Training and Inferring in Neural Networks," *arXiv: 1907.04091*, July 2019.

[5] H. F. Langroudi, Z. Carmichael, D. Pastuch, and D. Kudithipudi, "Cheetah: Mixed Low-Precision Hardware & Software Co-Design Framework for DNNs on the Edge," *arXiv: 1908.02386*, pp. 1–13, Aug. 2019.

[6] J. Lu, C. Fang, M. Xu, J. Lin, and Z. Wang, "Evaluations on Deep Neural Networks Training Using Posit Number System," *IEEE Transactions on Computers*, vol. 14, no. 8, pp. 1–1, 2020.

[7] R. Murillo, A. A. D. Barrio, and G. Botella, "Deep PeNSieve: A deep learning framework based on the posit number system," *Digital Signal Processing*, vol. 102, p. 102762, jul 2020.

[8] J. L. Gustafson and I. Yonemoto, "Beating Floating Point at its Own Game: Posit Arithmetic," *Supercomputing Frontiers and Innovations*, vol. 4, pp. 71–86, June 2017.

[9] Stillwater Supercomputing, Inc., "stillwater-sc/universal: Universal Number Arithmetic - GitHub," 2020. Accessed on 2020-11-02.