POSITNN: TRAINING DEEP NEURAL NETWORKS WITH MIXED LOW-PRECISION POSIT

Gonçalo Raposo, Pedro Tomás, Nuno Roma

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

ABSTRACT

Low-precision formats have proven to be an efficient way to reduce not only the memory footprint but also the hardware resources and power consumption of deep learning computations. Under this premise, the posit numerical format appears to be a highly viable substitute for the IEEE floating-point, but its application to neural networks training still requires further research. Some preliminary results have shown that 8-bit (and even smaller) posits may be used for inference and 16-bit for training, while maintaining the model accuracy. The presented research aims to evaluate the feasibility to train deep convolutional neural networks using posits. For such purpose, a software framework was developed to use simulated posits and quires in end-to-end training and inference. This implementation allows using any bit size, configuration, and even mixed precision, suitable for different precision requirements in various stages.

The obtained results suggest that 8-bit posits can substitute 32-bit floats during training with no negative impact on the resulting loss and accuracy.

Index Terms – Posit numerical format, low-precision arithmetic, deep neural networks, training, inference

POSIT NUMBERING SYSTEM

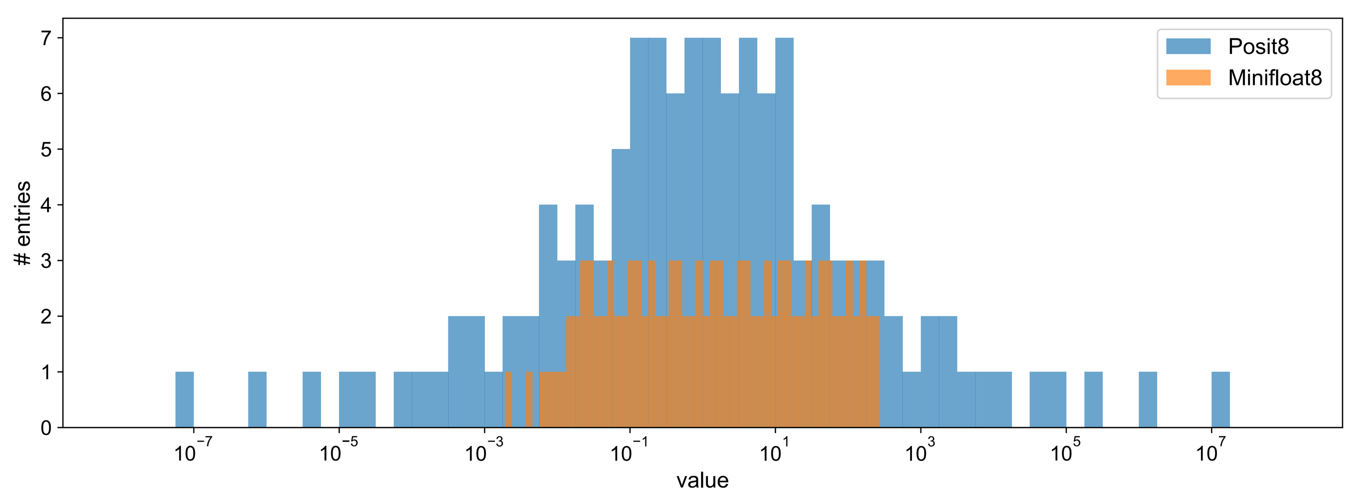

Fig. Distributions of a 8-bit posit (blue) and a 8-bit floating-point (orange).

Posit (*n* bits)	Sign (1 bit)	Regime (variable)	Exponent ({0..*es*} bits)	Fraction (remaining)

Fig. Format encoding of a posit with *n* bits and *es* exponent size.

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

DEEP LEARNING POSIT FRAMEWORK

- New open source framework for neural networks
- Training and inference using posits of any precision
- Support for mixed precision configurations
- Implemented in C++ and with a similar API to PyTorch

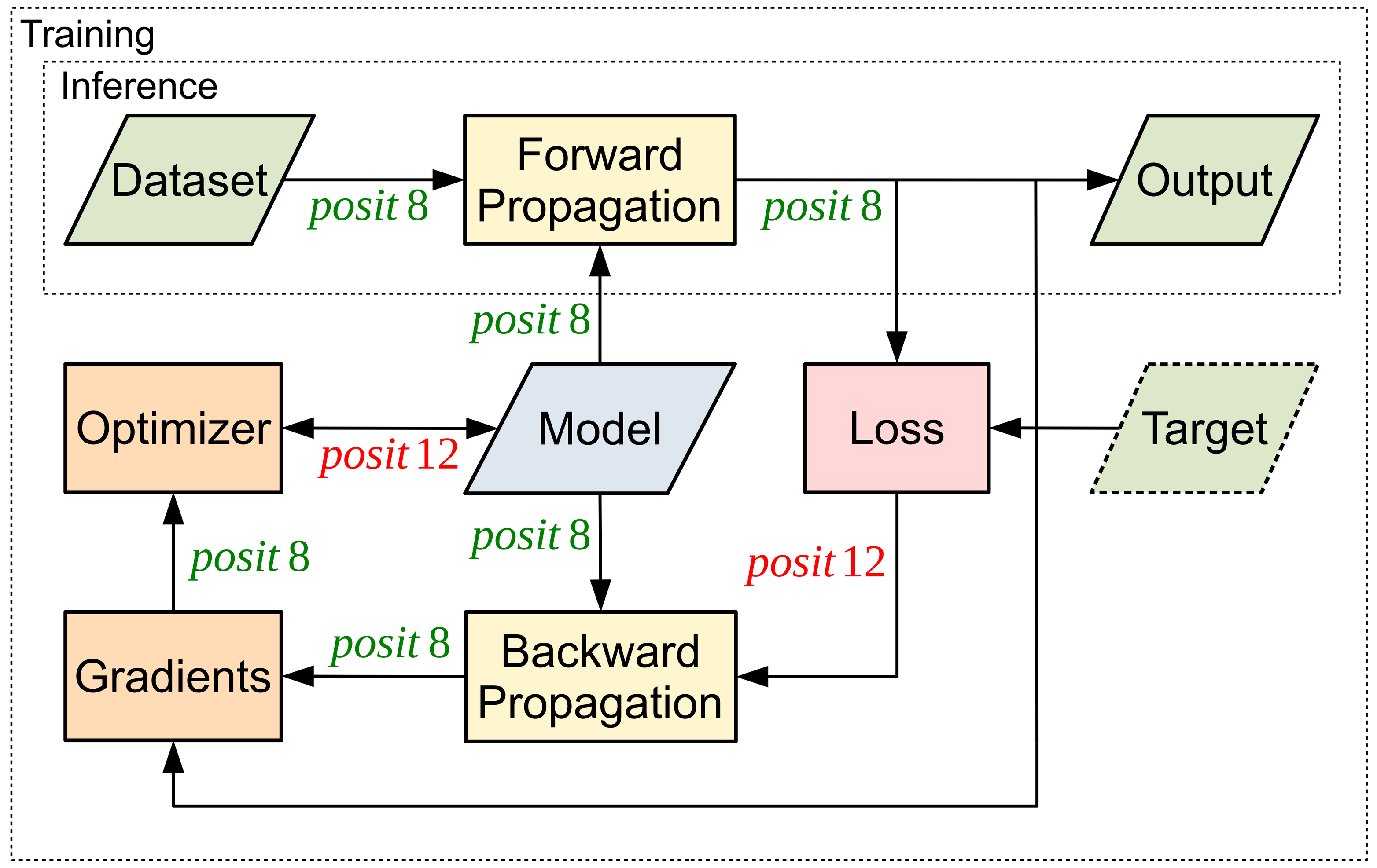

Fig. Block diagram of a possible mixed precision configuration for DNN training and inference.

- The gradients decrease as the model converges – vanishing gradient problem
- Insufficient dynamic range and resolution with narrow posit precisions for the optimizer and loss function

Tab. 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

EXPERIMENTAL EVALUATION

Tab. Accuracy of CNNs trained and tested with posits (accumulating with quires). Everything with 8-bit posit except optimizer (O) and loss (L). Compared against 32-bit float.

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 O16-L16$_q$	99.19%	90.46%	71.30%	92.65%	35.41%	67.00%
Posit8 O16-L12$_q$	99.17%	90.14%	71.09%	92.83%	35.27%	66.57%
Posit8 O12-L12$_q$	99.20%	90.07%	68.28%	91.22%	25.85%	57.77%
Posit8 O12-L10$_q$	99.17%	90.13%	68.41%	91.41%	25.37%	56.21%

CONCLUSION

- 8-bit posits can replace 32-bit floats in a mixed precision configuration for DNN training (accuracy degradation < 1%)
- Optimizer and loss function require higher precision
- 85 – 95% of the computation were performed with 8-bit posits (~ 4x less memory)
- Future work shall evaluate these results in a hardware implementation of a posit unit, namely, its critical path (time) and energy consumption (ongoing)

REACH OUT

E-mail: goncalo.cascalho.raposo@tecnico.ulisboa.pt

GitHub: https://github.com/hpc-ulisboa/posit-neuralnet