PocketNN: Integer-only Training and Inference of Neural Networks via Direct Feedback Alignment and Pocket Activations in Pure C++

Jaewoo Song and Fangzhen Lin

Department of Computer Science and Engineering, The Hong Kong University of Science and Technology, Kowloon, Hong Kong, China

On-device training and inference is important for tinyML

On-device training and inference of neural network models can solve important problems including data privacy issues, high electric power usage, longer time delay caused by online communication and higher device price due to online communication modules.

Integer arithmetic is useful

For on-device training and inference, it is important to make DNN models faster and smaller because tinyML devices are slow and their storage is tiny. Integer arithmetic is useful for the purpose.

- Speed: Integer arithmetic is faster than floating-point arithmetic on many tinyML devices which do not have floating-point units (FPUs).
- Storage: Compared to 32-bit floating-point numbers, 8-bit integers can make model sizes 75% smaller.

PocketNN directly operates on integers without quantization

Quantization is a conventional approach for integer-only DNNs. However, existing quantization algorithms have several drawbacks:

1. Many are for inference only; require floating-point operations on training.
2. Others support both training and inference. However, they involve complex operations such as bit shifting, scaling, and deterministic and stochastic roundings.
3. Customized fixed-point notations are often used to implement floating-point real numbers with integers. Conceptually, they are not using integers but still using floating-point numbers.
4. They often suffer from overflow during training.

PocketNN solves these problems by directly operating on integers without any explicit quantization.

Backpropagation (BP) causes integer overflow

Backpropagation (BP), a de facto standard DNN training algorithm, can easily suffer from overflow in integer-only DNN training. Consider the kth layer of a fully connected neural network:

$$\boldsymbol{h}^{[k]} = \boldsymbol{a}^{[k-1]}W^{[k]} + \boldsymbol{b}^{[k]} \quad \text{and} \quad \boldsymbol{a}^{[k]} = \text{actv}(\boldsymbol{h}^{[k]})$$

BP updates the weight matrices and the bias vectors as below.

$$\delta_{BP}^{[k]} = \frac{\partial J}{\partial \boldsymbol{h}^{[k]}} = \delta_{BP}^{[k+1]} W^{[k+1]^T} \odot \text{actv}'^{[k]}(\boldsymbol{h}^{[k]})$$

$$\frac{\partial J}{\partial W^{[k]}} = \boldsymbol{a}^{[k-1]^T} \delta_{BP}^{[k]} \quad \text{and} \quad \frac{\partial J}{\partial \boldsymbol{b}^{[k]}} = \delta_{BP}^{[k]}$$

The recursive multiplication of δ_{BP}s causes integer overflow.

Integer direct feedback alignment (DFA) can prevent overflow

Direct feedback alignment (DFA) is a new emerging DNN training algorithm. DFA trains hidden layers independently from other layers by propagating error directly from the output layer to each individual hidden layer via fixed random feedback matrices. Again, consider the kth layer of a fully connected neural network:

$$\boldsymbol{h}^{[k]} = \boldsymbol{a}^{[k-1]}W^{[k]} + \boldsymbol{b}^{[k]} \quad \text{and} \quad \boldsymbol{a}^{[k]} = \text{actv}(\boldsymbol{h}^{[k]})$$

Instead of weights, appropriately sized random matrices $R^{[k]}$ are used to define $\delta_{DFA}^{[k]}$ such that

$$\delta_{DFA}^{[k]} = \left((\hat{y} - y)R^{[k]}\right) \odot \text{actv}'^{[k]}(\boldsymbol{h}^{[k]})$$

$$\frac{\partial J}{\partial W^{[k]}} = \boldsymbol{a}^{[k-1]^T} \delta_{DFA}^{[k]} \quad \text{and} \quad \frac{\partial J}{\partial \boldsymbol{b}^{[k]}} = \delta_{DFA}^{[k]}$$

PocketNN is for **integer-only training and inference of neural networks**. It directly operates on integers without quantization.

Direct feedback alignment (DFA) was used instead of backpropagation (BP) for training; a family of new activation functions was designed for integer-only DNNs and named **Pocket Activations**.

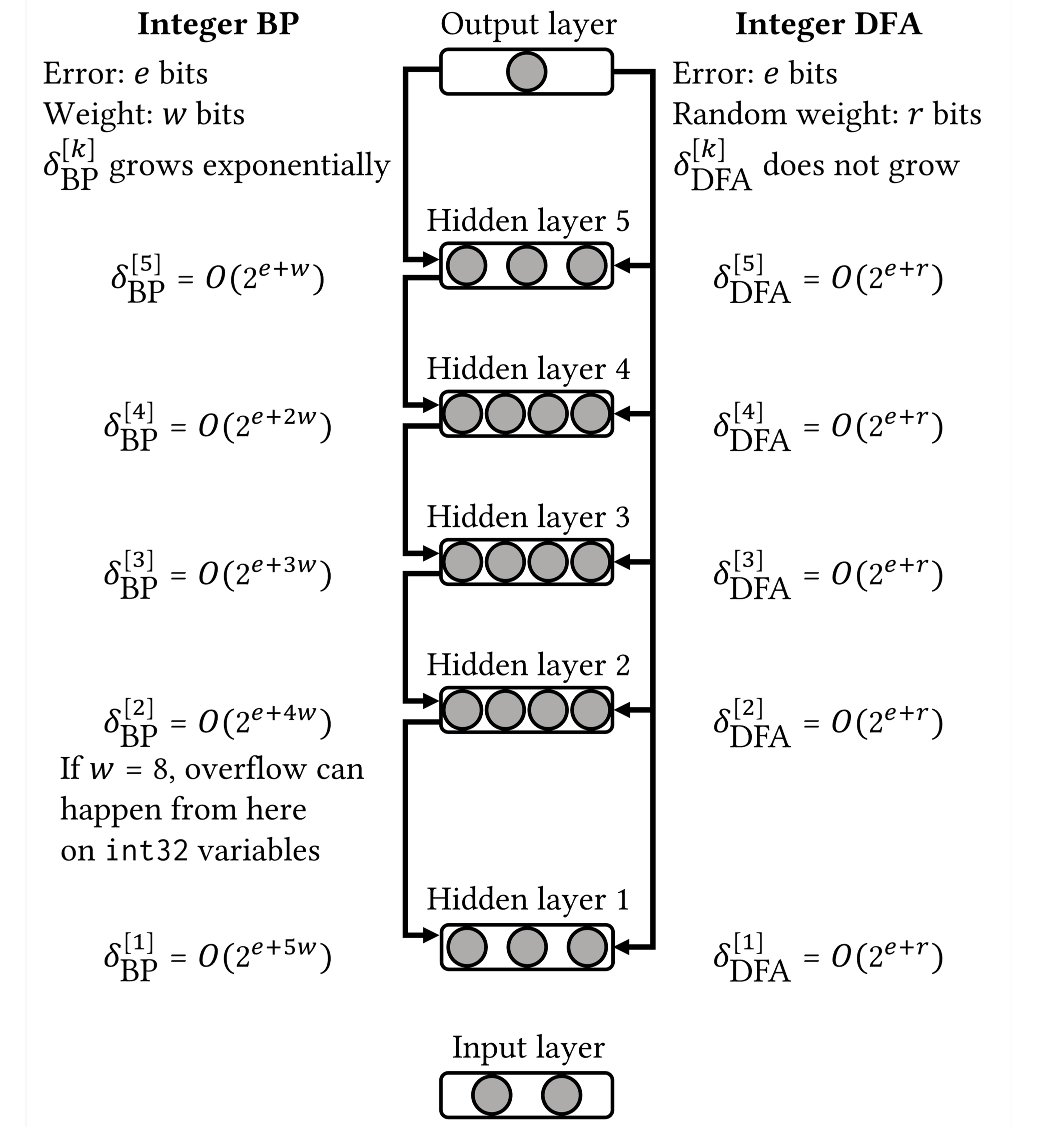
PocketNN was implemented in **pure C++** without any dependencies for maximum compatibility and portability

It will be a useful tool for tinyML researchers and developers working on integer-only DNNs

Paper

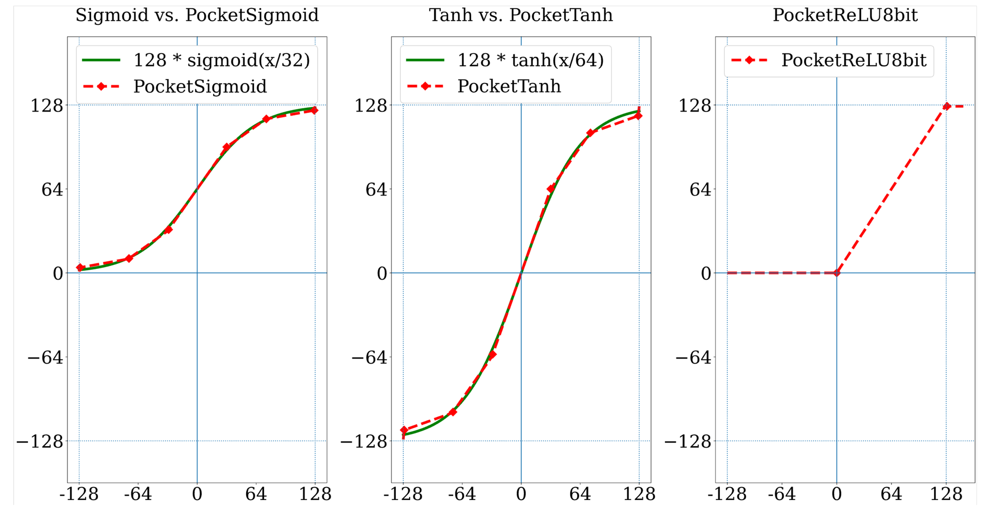
Code

Because $\delta_{DFA}^{[k]}$ is independent from δ_{DFA} of other layers, integer overflow is prevented in DFA.



Pocket activation functions

Pocket activations are a family of activation functions for integer-only DNNs. They are piecewise linear approximations of popular activation functions: PocketSigmoid, PocketTanh and PocketReLU8bit. They take 8-bit input values and generate 8-bit output values to ensure consistency and soundness of behavior.



Result

PocketNN achieved 96.98% and 87.7% accuracies on MNIST and Fashion-MNIST datasets, respectively, showing only marginal accuracy degradations compared to its floating-point BP counterparts.