Mixed precision accumulation for neural network inference guided by componentwise forward error analysis

El-Mehdi El ARAR

mehdi.elarar@lip6.fr

RAIM25:

Joint work with **Silviu-Ioan Filip**, **Theo Mary** and **Elisa Riccietti**

06/11/2025

Neural network inference

- We consider the computation of the forward pass on a neural network with L layers as:

$$h_\ell = \phi_\ell(W_\ell h_{\ell-1}), \quad \ell = 1, \dots, L$$

  where

  - W_ℓ are the weight matrices
  - ϕ_ℓ are the activation functions
  - $h_0 = x$ is the input and h_L is the output
- The deployment of large scale models motivates the use of reduced precision arithmetic for accelerating both training and inference
- Yet, need to preserve model accuracy $\Rightarrow$ mixed precision strategies

Reduced/mixed precision quantization

- Quantization stores the weights (W_ℓ) in reduced precision
- Many mixed precision variants have been proposed [Lin et al., 2016, Dong et al., 2020, Dong et al., 2019, Yao et al., 2021, Gong et al., 2019, Uhlich et al., 2019, Wang et al., 2019, Yang et al., 2021]
- Accumulation is often kept in high precision
  - Because some specialized hardware provides efficient high precision accumulators (e.g., NVIDIA tensor cores)
  - Because model accuracy is much more sensitive to accumulation precision than storage precision
- However, reducing accumulation precision can significantly improve performance, especially in resource-limited environments

Reduced precision accumulation

Several ideas to allow for reduced precision accumulation:

- Stochastic rounding: [Gupta et al., 2015, El Arar et al., 2025]
- Blocked summation: [Wang et al., 2018]
- Scaling (overflow prevention): [Sakr et al., 2019, Xie et al., 2021, Ni et al., 2021, Colbert et al., 2023, Colbert et al., 2024].

However, all these ideas only consider uniform precision accumulation

Mixed precision accumulation

Mixed precision accumulation has been surprisingly little investigated.
Difficulties/questions:

- Is it meaningful to accumulate different inner products in different precisions?
- If so, can we derive a criterion to decide which inner product should be accumulated in which precision?
- If so, can we leverage this criterion into a practical algorithm?

Mixed precision accumulation

Mixed precision accumulation has been surprisingly little investigated.
Difficulties/questions:

- Is it meaningful to accumulate different inner products in different precisions?
  ⇒ YES!
- If so, can we derive a criterion to decide which inner product should be accumulated in which precision?
  ⇒ YES!
- If so, can we leverage this criterion into a practical algorithm?
  ⇒ Depends. . . but in many cases, YES!

Componentwise forward error analysis

We are interested in identifying mixed precision opportunities. Therefore:

- We seek forward error bounds on $\|\widehat{h}_L - h_L\|_\infty$ to directly relate the precisions used to the accuracy of the final output
- We seek componentwise error bounds to track the effect of each inner product precision to the final accuracy

Componentwise error model

We will use the following componentwise error model:

$$\widehat{h}_\ell = \phi_\ell\Big((W_\ell \circ (\mathbf{1} + \Delta W_\ell))\widehat{h}_{\ell-1}\Big)\circ(\mathbf{1} + \Delta\phi_\ell), \quad |\Delta W_\ell| \leqslant \varepsilon_\ell^W, \quad |\Delta\phi_\ell| \leqslant \varepsilon_\ell^\phi,$$

where

- $\circ$ denotes the Hadamard (componentwise) product
- $\Delta W_\ell \in \mathbb{R}^{n_\ell \times n_{\ell-1}}$ and $\Delta\phi_\ell \in \mathbb{R}^{n_\ell}$ are the errors incurred in the matrix–vector product and in the activation **(≡ the errors)**
- $\varepsilon_\ell^W \in \mathbb{R}^{n_\ell}$ and $\varepsilon_\ell^\phi \in \mathbb{R}^{n_\ell}$ are nonnegative vectors whose components bound the corresponding errors **(≡ the precisions)**
- Note that $(\varepsilon_\ell^W)_i = \max_{j=1:\, n_{\ell-1}} |(\Delta W_\ell)_{ij}|$ for $i = 1:\, n_\ell$

Componentwise error model

We will use the following componentwise error model:

$$\widehat{h}_\ell = \phi_\ell\Big(\big(W_\ell \circ (\mathbf{1} + \Delta W_\ell)\big)\widehat{h}_{\ell-1}\Big) \circ (\mathbf{1} + \Delta\phi_\ell), \quad |\Delta W_\ell| \leqslant \varepsilon_\ell^W, \quad |\Delta\phi_\ell| \leqslant \varepsilon_\ell^\phi,$$

where

- $\circ$ denotes the Hadamard (componentwise) product
- $\Delta W_\ell \in \mathbb{R}^{n_\ell \times n_{\ell-1}}$ and $\Delta\phi_\ell \in \mathbb{R}^{n_\ell}$ are the errors incurred in the matrix–vector product and in the activation **(≡ the errors)**
- $\varepsilon_\ell^W \in \mathbb{R}^{n_\ell}$ and $\varepsilon_\ell^\phi \in \mathbb{R}^{n_\ell}$ are nonnegative vectors whose components bound the corresponding errors **(≡ the precisions)**
- Note that $(\varepsilon_\ell^W)_i = \max_{j=1:\, n_{\ell-1}} |(\Delta W_\ell)_{ij}|$ for $i = 1\colon n_\ell$

This model is very generic since it allows different precisions for:

- inner products and activations (ε^W and ε^ϕ can be different)
- different layers (ε_ℓ can be different for $\ell = 1\colon L$)
- different components ($(\varepsilon_\ell)_i$ can be different for $i = 1\colon n_\ell$)

Condition numbers

The following key quantities will appear in our analysis:

- Condition number κ_{v_ℓ} of the matrix–vector products $v_\ell = W_\ell \widehat{h}_{\ell-1}$:

$$\Big(W_\ell \circ (\mathbf{1} + \Delta W_\ell)\Big)\widehat{h}_{\ell-1} = v_\ell \circ (\mathbf{1} + \Delta v_\ell), \quad |\Delta v_\ell| \leqslant \kappa_{v_\ell} \circ \varepsilon_\ell^W$$

  where

$$\kappa_{v_\ell} = (|W_\ell||\widehat{h}_{\ell-1}|) \oslash |v_\ell|$$

- Condition number κ_{ϕ_ℓ} of the activation functions:

$$\phi_\ell\Big(v_\ell \circ (\mathbf{1} + \Delta v_\ell)\Big) = \phi_\ell(v_\ell) \circ \Big(\mathbf{1} + \Delta \phi_\ell\Big), \quad |\Delta \phi_\ell| \leqslant \kappa_{\phi_\ell}(v_\ell) \circ |\Delta v_\ell|$$

  where

$$\kappa_{\phi_\ell}(v_\ell) = |v_\ell \circ \phi_\ell'(v_\ell) \oslash \phi_\ell(v_\ell)|$$

Main theorem

Theorem

The computed output $\widehat{h}_\ell$ of any layer ℓ satisfies

$$\widehat{h}_\ell = h_\ell \circ (\mathbf{1} + \Delta h_\ell), \quad |\Delta h_\ell| \leqslant \varepsilon_\ell^h,$$

where ε_ℓ^h satisfies the first-order recurrence relation

$$\varepsilon_\ell^h = \kappa_{\phi_\ell}(v_\ell) \circ \kappa_{v_\ell} \circ \left(\varepsilon_\ell^W + \|\varepsilon_{\ell-1}^h\|_\infty\right) + \varepsilon_\ell^\phi.$$

This yields the scalar recurrence

$$\|\varepsilon_\ell^h\|_\infty = \|\kappa_{\phi_\ell}(v_\ell) \circ \kappa_{v_\ell} \circ \varepsilon_\ell^W\|_\infty + \|\kappa_{\phi_\ell}(v_\ell) \circ \kappa_{v_\ell}\|_\infty \|\varepsilon_{\ell-1}^h\|_\infty + \|\varepsilon_\ell^\phi\|_\infty$$

and hence the formula

$$\|\varepsilon_L^h\|_\infty = \sum_{\ell=1}^{L} \left[\left(\prod_{k=\ell+1}^{L} \|\kappa_{\phi_k}(v_k) \circ \kappa_{v_k}\|_\infty \right) \left(\|\kappa_{\phi_\ell}(v_\ell) \circ \kappa_{v_\ell} \circ \varepsilon_\ell^W\|_\infty + \|\varepsilon_\ell^\phi\|_\infty \right) \right]$$

Interpretation

$$\sum_{\ell=1}^{L} \left[\left(\prod_{k=\ell+1}^{L} \|\kappa_{\phi_k}(v_k) \circ \kappa_{v_k}\|_\infty \right) \left(\|\kappa_{\phi_\ell}(v_\ell) \circ \kappa_{v_\ell} \circ \varepsilon_\ell^W\|_\infty + \|\varepsilon_\ell^\phi\|_\infty \right) \right]$$

- Sum of terms $\Rightarrow$ balance them

Interpretation

$$\sum_{\ell=1}^{L}\left[\left(\prod_{k=\ell+1}^{L}\|\kappa_{\phi_k}(v_k)\circ\kappa_{v_k}\|_\infty\right)\left(\|\kappa_{\phi_\ell}(v_\ell)\circ\kappa_{v_\ell}\circ\varepsilon_\ell^W\|_\infty+\|\varepsilon_\ell^\phi\|_\infty\right)\right]$$

- Sum of terms $\Rightarrow$ balance them

$$\left(\prod_{k=\ell+1}^{L}\|\kappa_{\phi_k}(v_k)\circ\kappa_{v_k}\|_\infty\right)\left(\|\kappa_{\phi_\ell}(v_\ell)\circ\kappa_{v_\ell}\circ\varepsilon_\ell^W\|_\infty+\|\varepsilon_\ell^\phi\|_\infty\right)$$

- Product of condition numbers of subsequent layers: technically depends on ℓ, but
  - $\|\cdot\|_\infty$ likely to smudge most of the potential variations
  - not easy to estimate anyway

$\Rightarrow$ drop it

Interpretation

$$\sum_{\ell=1}^{L}\left[\left(\prod_{k=\ell+1}^{L}\|\kappa_{\phi_k}(v_k)\circ\kappa_{v_k}\|_\infty\right)\left(\|\kappa_{\phi_\ell}(v_\ell)\circ\kappa_{v_\ell}\circ\varepsilon_\ell^W\|_\infty+\|\varepsilon_\ell^\phi\|_\infty\right)\right]$$

- Sum of terms $\Rightarrow$ balance them

$$\left(\prod_{k=\ell+1}^{L}\|\kappa_{\phi_k}(v_k)\circ\kappa_{v_k}\|_\infty\right)\left(\|\kappa_{\phi_\ell}(v_\ell)\circ\kappa_{v_\ell}\circ\varepsilon_\ell^W\|_\infty+\|\varepsilon_\ell^\phi\|_\infty\right)$$

- Product of condition numbers of subsequent layers: technically depends on ℓ, but
  - $\|\cdot\|_\infty$ likely to smudge most of the potential variations
  - not easy to estimate anyway

  $\Rightarrow$ drop it

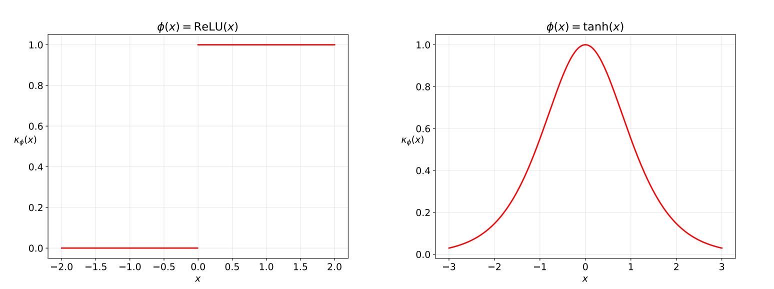
$$\|\kappa_{\phi_\ell}(v_\ell)\circ\kappa_{v_\ell}\circ\varepsilon_\ell^W\|_\infty+\|\varepsilon_\ell^\phi\|_\infty$$

- Activation error only plays a role in $\|\cdot\|_\infty \Rightarrow$ drop it

- We are left with

$$\|\kappa_{\phi_\ell}(v_\ell) \circ \kappa_{v_\ell} \circ \varepsilon_\ell^W\|_\infty$$

- Large potential variations of condition numbers:



⇒ The components $(\varepsilon_\ell^W)_i$ should be chosen to be inversely proportional to $(\kappa_\ell)_i := (\kappa_{\phi_\ell}(v_\ell) \circ \kappa_{v_\ell})_i$ ⇒ mixed precision opportunity!

Ideal conceptual algorithm

for each layer ℓ **do**
    Compute $\kappa_\ell = \kappa_{\phi_\ell}(v_\ell) \circ \kappa_{v_\ell}$
    for each component i **do**
        if $(\kappa_\ell)_i \leqslant \tau$ **then**
            Compute $(h_\ell)_i = \phi_\ell((W_\ell h_{\ell-1})_i)$ in precision u_{low}
        else
            Compute $(h_\ell)_i = \phi_\ell((W_\ell h_{\ell-1})_i)$ in precision u_{high}
        end if
    end for
end for

Ideal conceptual algorithm

for each layer ℓ **do**
  Compute $\kappa_\ell = \kappa_{\phi_\ell}(v_\ell) \circ \kappa_{v_\ell}$ $\rightarrow$ depends on $v_\ell = W_\ell h_{\ell-1}$!
  for each component i **do**
    if $(\kappa_\ell)_i \leqslant \tau$ **then**
      Compute $(h_\ell)_i = \phi_\ell((W_\ell h_{\ell-1})_i)$ in precision u_{low}
    else
      Compute $(h_\ell)_i = \phi_\ell((W_\ell h_{\ell-1})_i)$ in precision u_{high}
    end if
  end for
end for

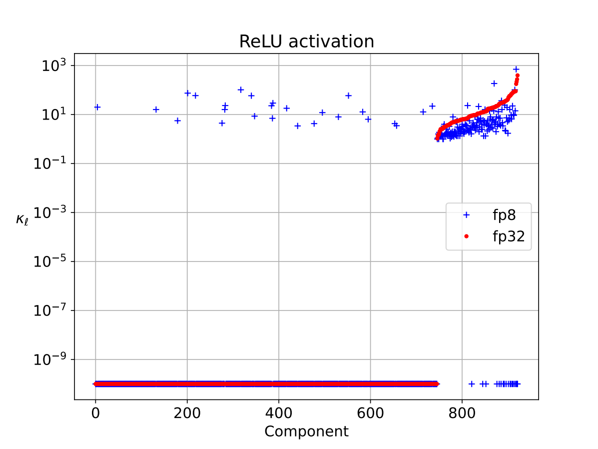
$\Rightarrow$ can we cheaply estimate κ_ℓ?

Estimating κ (part 1)

- We only need to know the order of magnitude of κ_ℓ $\Rightarrow$ can compute it in low precision?

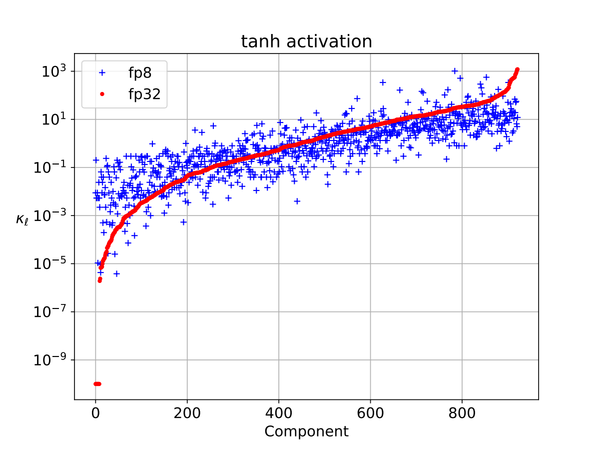
Estimating κ (part 1)

- We only need to know the order of magnitude of $\kappa_\ell \Rightarrow$ can compute it in low precision?



Estimating κ (part 1)

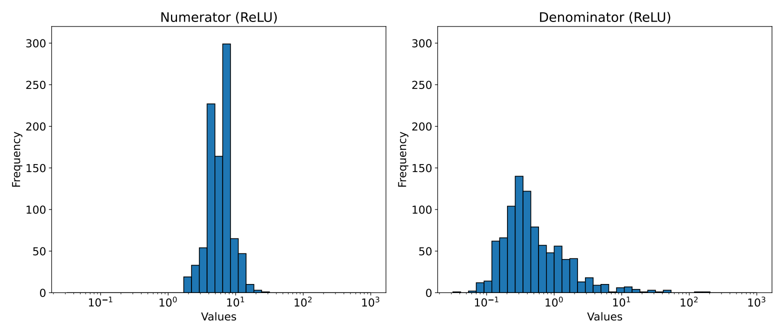
- We only need to know the order of magnitude of κ_ℓ $\Rightarrow$ can compute it in low precision?



Estimating κ (part 2)

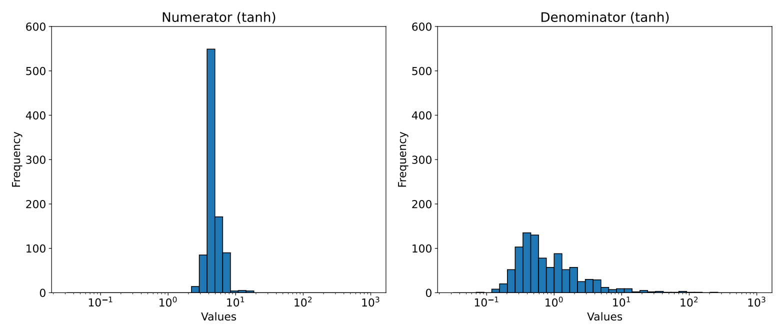
- $\kappa_{\phi_\ell}(v_\ell) = |v_\ell \circ \phi'_\ell(v_\ell) \oslash \phi_\ell(v_\ell)|$ $\rightarrow$ almost for free as by-product of computing $h_\ell = \phi_\ell(v_\ell)$ in low precision
- $\kappa_{v_\ell} = (|W_\ell||\widehat{h}_{\ell-1}|) \oslash |v_\ell|$ $\rightarrow$ denominator is a by-product but not numerator!

- $\kappa_{\phi_\ell}(v_\ell) = |v_\ell \circ \phi'_\ell(v_\ell) \oslash \phi_\ell(v_\ell)|$ $\rightarrow$ almost for free as by-product of computing $h_\ell = \phi_\ell(v_\ell)$ in low precision
- $\kappa_{v_\ell} = (|W_\ell||\widehat{h}_{\ell-1}|) \oslash |v_\ell|$ $\rightarrow$ denominator is a by-product but not numerator!



Estimating κ (part 2)

- $\kappa_{\phi_\ell}(v_\ell) = |v_\ell \circ \phi'_\ell(v_\ell) \oslash \phi_\ell(v_\ell)|$ $\rightarrow$ almost for free as by-product of computing $h_\ell = \phi_\ell(v_\ell)$ in low precision
- $\kappa_{v_\ell} = (|W_\ell||\widehat{h}_{\ell-1}|) \oslash |v_\ell|$ $\rightarrow$ denominator is a by-product but not numerator!

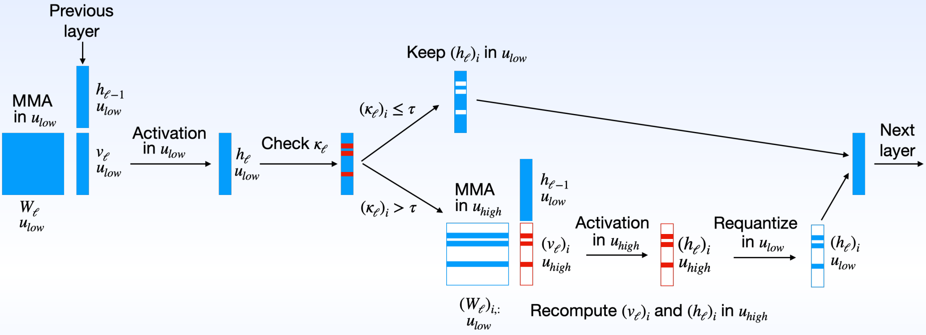


$\Rightarrow$ Approximate $\kappa_{v_\ell} \approx c\mathbf{1} \oslash |v_\ell|$

Practical algorithm

Input: $W_1, \dots, W_L$, the weight matrices; $h_0 = x$, the input vector; τ, a tolerance controlling the precision choice; $u_{\mathrm{low}}, u_{\mathrm{high}}$, the precisions.
Output: h_L, the output of the network.

for $\ell = 1, \dots, L$ **do**
  Compute $v_\ell = W_\ell h_{\ell-1}$ in precision u_{low}.
  Compute $h_\ell = \phi_\ell(v_\ell)$ in precision u_{low}.
  Compute $\kappa_{\phi_\ell}(v_\ell) = |v_\ell \circ \phi'_\ell(v_\ell)| \oslash |\phi_\ell(v_\ell)|$ in precision u_{low}.
  Compute $\kappa_\ell = \kappa_{\phi_\ell} \oslash |v_\ell|$ in precision u_{low}.
  for every component $(\kappa_\ell)_i$ **do**
    if $(\kappa_\ell)_i > \tau$ **then**
      Recompute $(v_\ell)_i = (W_\ell h_{\ell-1})_i$ in precision u_{high}.
      Recompute $(h_\ell)_i = \phi_\ell((v_\ell)_i)$ in precision u_{high}.
      Requantize $(h_\ell)_i$ back to precision u_{low}.
    end if
  end for
end for


Previous
layer
MMA
in u_{low}
$h_{\ell-1}$
u_{low}
v_ℓ
u_{low}
W_ℓ
u_{low}
Activation
in u_{low}
h_ℓ
u_{low}
Check κ_ℓ
$(\kappa_\ell)_i \le \tau$
$(\kappa_\ell)_i > \tau$
Keep $(h_\ell)_i$ in u_{low}
MMA
in u_{high}
$h_{\ell-1}$
u_{low}
$(v_\ell)_i$
u_{high}
$(W_\ell)_{i,:}$
u_{low}
Activation
in u_{high}
$(h_\ell)_i$
u_{high}
Requantize
in u_{low}
$(h_\ell)_i$
u_{low}
Next
layer
Recompute $(v_\ell)_i$ and $(h_\ell)_i$ in u_{high}

Cost model

- This mixed precision algorithm will only be cost-effective if the number of inner products that need to be recomputed is small
- This can be modelled as follows:

$$c_{\text{mixed}} = c_{\text{low}} + \rho c_{\text{high}} = \left(\frac{c_{\text{low}}}{c_{\text{high}}} + \rho \right) c_{\text{high}},$$

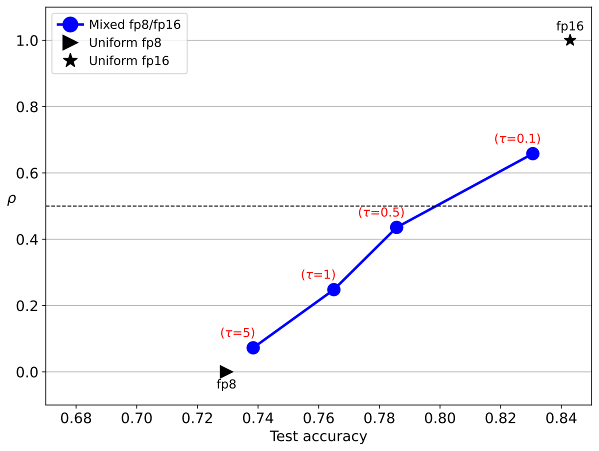
  where

  - c_{low} and c_{high} are the inference costs in uniform (low and high) precision
  - $\rho \in [0, 1]$ is the fraction of inner products that need to be recomputed

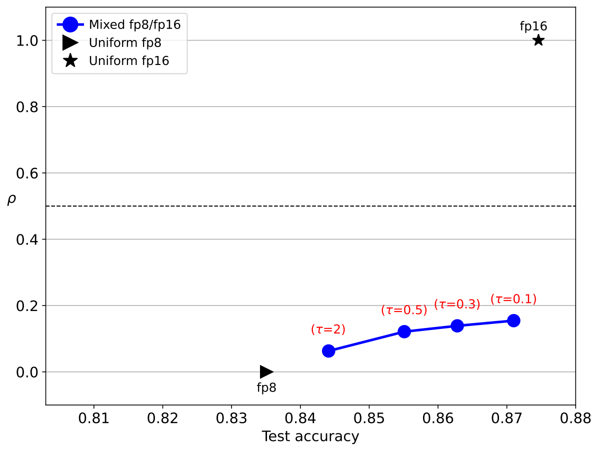
Experimental setting

- We test multilayer perceptron networks
  - with 3, 5, or 8 layers
  - with either tanh or ReLU activations
  - trained on either the MNIST or FMNIST datasets (using FP32 precision)
  - we will show a sample of these tests, see paper for full results
- We use FP16 as u_{high} and FP8 (E4M3) as u_{low} $\Rightarrow$ assuming $c_{\text{high}} = 2c_{\text{low}}$, we want $\rho < 0.5$
- We report the test accuracy on 10,000 inputs

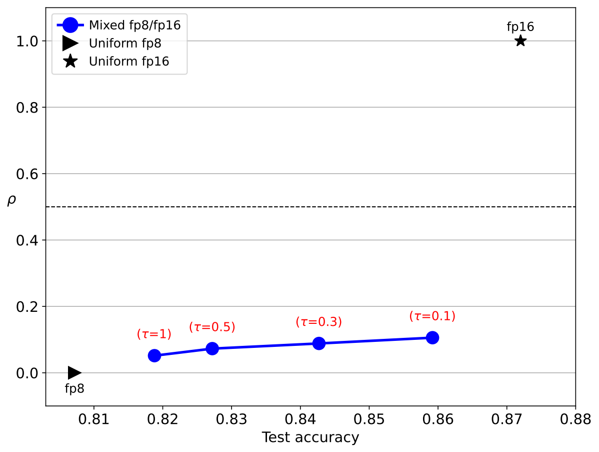
Experimental results: tanh, 5 layers, FMNIST



Experimental results: ReLU, 5 layers, FMNIST



Experimental results: ReLU, 3 layers, FMNIST



Conclusion

- Accumulation errors are proportional to the condition numbers of the inner products and activation functions, componentwise
- This observation can be leveraged into a practical mixed precision algorithm
- With ReLU activations, can reach FP16-equivalent model accuracy while computing $> 80\%$ of the inner products in FP8 $\Rightarrow$ 40% expected time reduction

Thanks!
Questions?

https://hal.science/hal-04995708

Colbert, I., Grob, F., Franco, G., Zhang, J., and Saab, R. (2024).
Accumulator-aware post-training quantization.
arXiv preprint arXiv:2409.17092.

Colbert, I., Pappalardo, A., and Petri-Koenig, J. (2023).
Quantized neural networks for low-precision accumulation with guaranteed overflow avoidance.
arXiv preprint arXiv:2301.13376.

Dong, Z., Yao, Z., Arfeen, D., Gholami, A., Mahoney, M. W., and Keutzer, K. (2020).
HAWQ-v2: Hessian aware trace-weighted quantization of neural networks.
Advances in neural information processing systems, 33:18518–18529.

Dong, Z., Yao, Z., Gholami, A., Mahoney, M. W., and Keutzer, K. (2019).
HAWQ: Hessian aware quantization of neural networks with mixed-precision.
In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 293–302.

El Arar, E.-M., Fasi, M., Filip, S.-I., and Mikaitis, M. (2025).
Probabilistic error analysis of limited-precision stochastic rounding.

Gong, C., Jiang, Z., Wang, D., Lin, Y., Liu, Q., and Pan, D. Z. (2019).
Mixed precision neural architecture search for energy efficient deep learning.
In *2019 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 1–7. IEEE.

Gupta, S., Agrawal, A., Gopalakrishnan, K., and Narayanan, P. (2015).
Deep learning with limited numerical precision.
In *International conference on machine learning*, pages 1737–1746. PMLR.

Lin, D., Talathi, S., and Annapureddy, S. (2016).
Fixed point quantization of deep convolutional networks.
In *International conference on machine learning*, pages 2849–2858. PMLR.

Ni, R., Chu, H., O., C. F., Chiang, P., Studer, C., and Goldstein, T. (2021).
Wrapnet: Neural net inference with ultra-low-precision arithmetic.
In *International Conference on Learning Representations ICLR 2021*. OpenReview.

Sakr, C., Wang, N., Chen, C., Choi, J., Agrawal, A., Shanbhag, N., and Gopalakrishnan, K. (2019).
Accumulation bit-width scaling for ultra-low precision training of deep networks.
arXiv preprint arXiv:1901.06588.

Uhlich, S., Mauch, L., Cardinaux, F., Yoshiyama, K., Garcia, J. A., Tiedemann, S., Kemp, T., and Nakamura, A. (2019).
Mixed precision DNNs: All you need is a good parametrization.
arXiv preprint arXiv:1905.11452.

Wang, K., Liu, Z., Lin, Y., Lin, J., and Han, S. (2019).
HAQ: Hardware-aware automated quantization with mixed precision.
In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8612–8620.

Wang, N., Choi, J., Brand, D., Chen, C., and Gopalakrishnan, K. (2018).
Training deep neural networks with 8-bit floating point numbers.
In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, pages 7686–7695, Red Hook, NY, USA. Curran Associates Inc.

Xie, H., Song, Y., Cai, L., and Li, M. (2021).
Overflow aware quantization: Accelerating neural network inference by low-bit multiply-accumulate operations.
In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pages 868–875.

Yang, H., Duan, L., Chen, Y., and Li, H. (2021).
BSQ: Exploring bit-level sparsity for mixed-precision neural network quantization.

Yao, Z., Dong, Z., Zheng, Z., Gholami, A., Yu, J., Tan, E., Wang, L., Huang, Q., Wang, Y., Mahoney, M. W., and Keutzer, K. (2021).
HAWQ-v3: Dyadic neural network quantization.
In *International Conference on Machine Learning*, pages 11875–11886. PMLR.