OPTIMIZING FOR IMPERFECTIONS IN ANALOG NEURAL COMPUTATIONS ON BRAINSCALES-2

Eric Kern, Hendrik Borras, Bernhard Klein, Holger Fröning

27. September 2023, FastML Workshop

BRAINSCALES-2

Purpose:

- Energy Efficiency
- Scalability

Mixed Signal Chip:

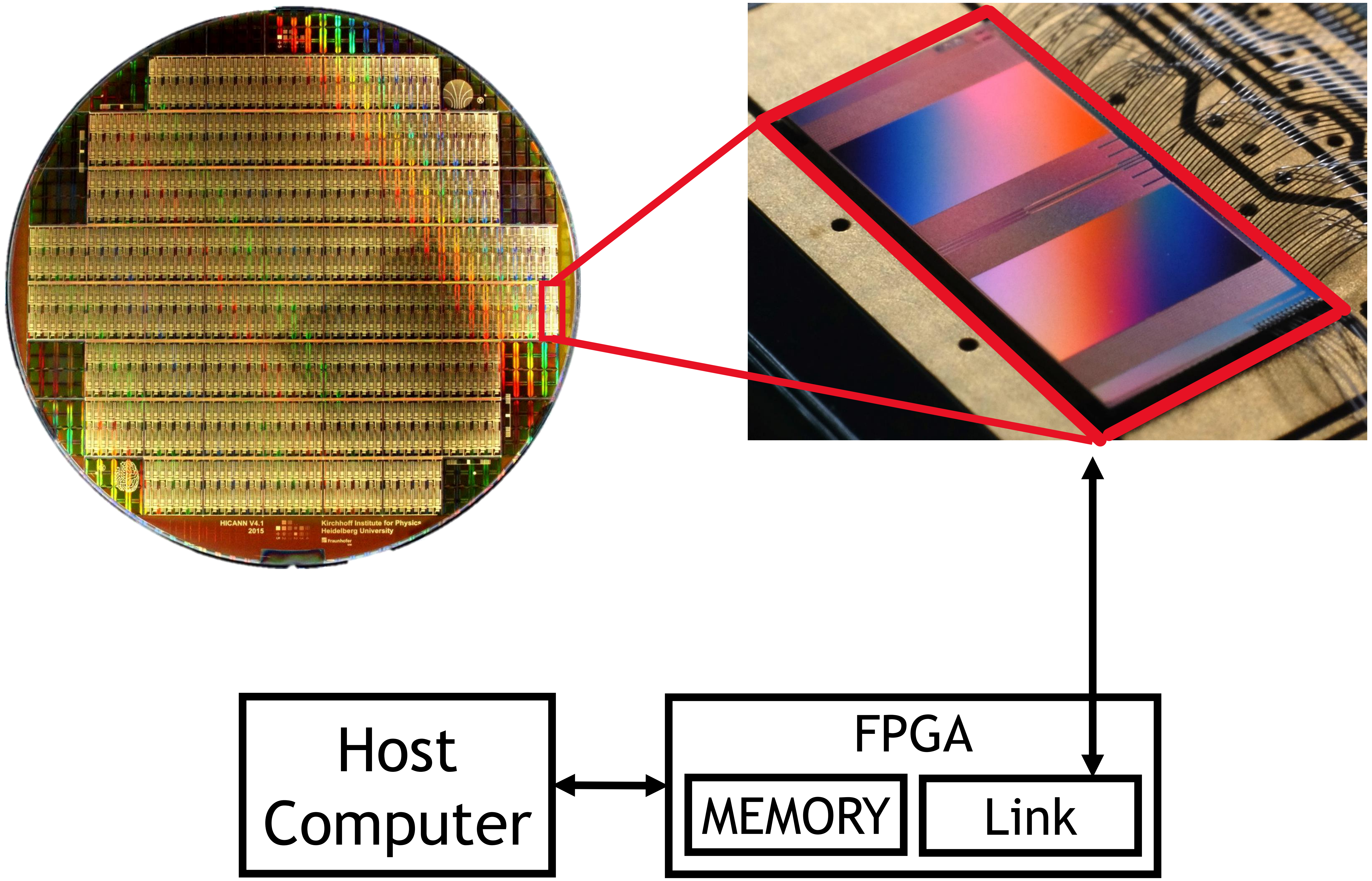
- Digital I/O
- Analog Core

Applications:

- Neuromorphic Computing
- Spiking Neural Networks
- Artificial Neural Networks

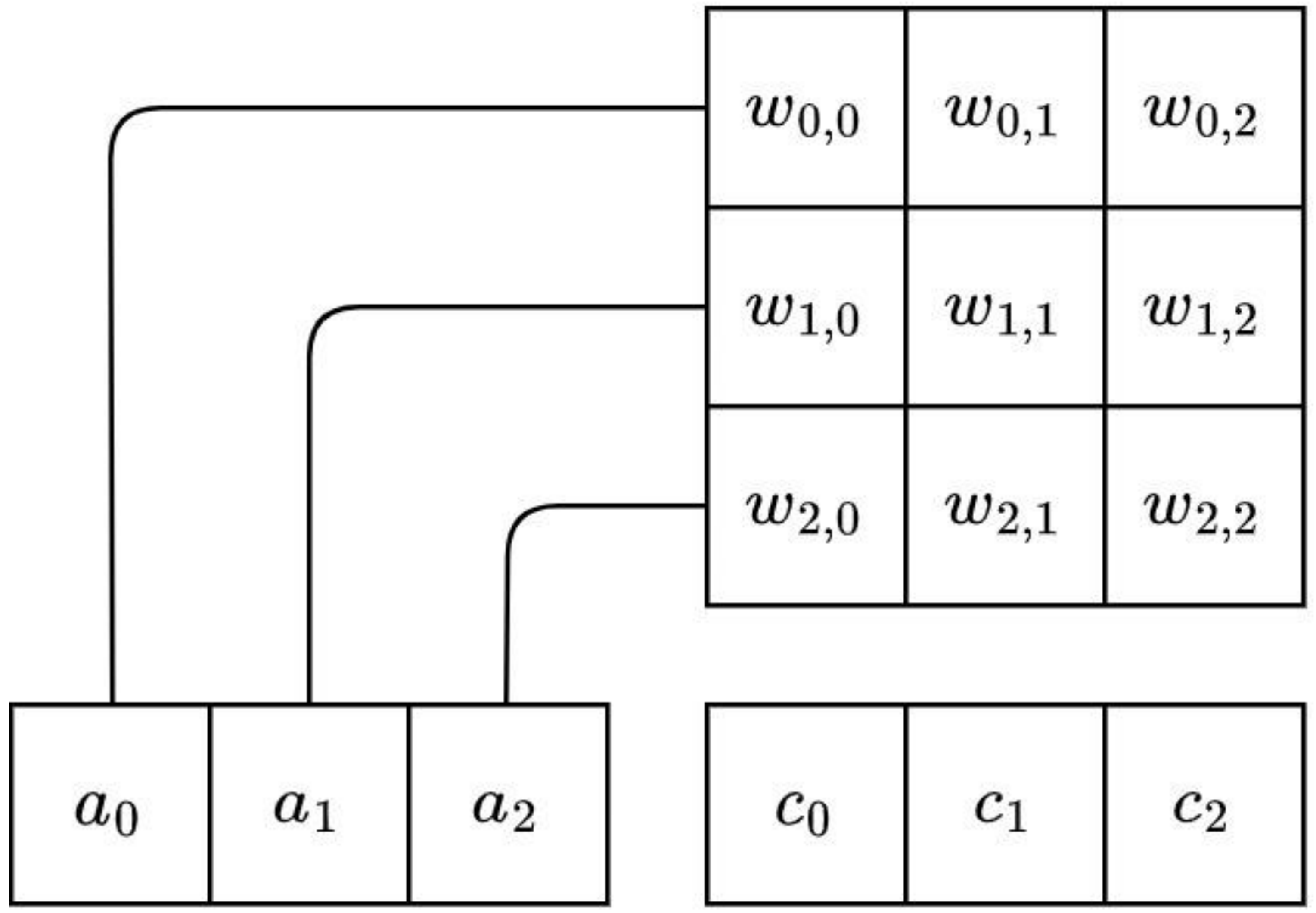
Wafer-Scale Integration

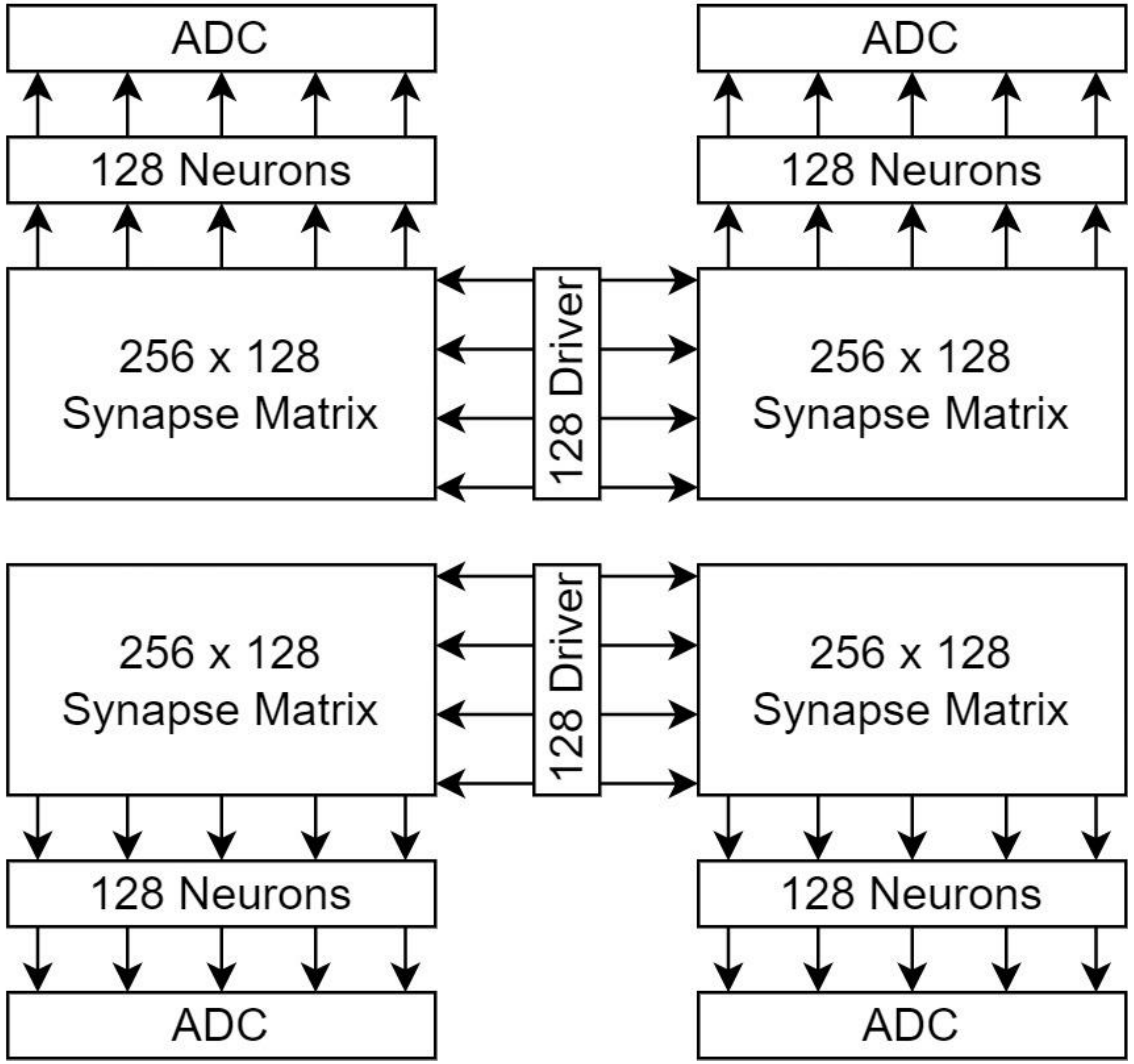
Single Chip



MATRIX MULTIPLICATION

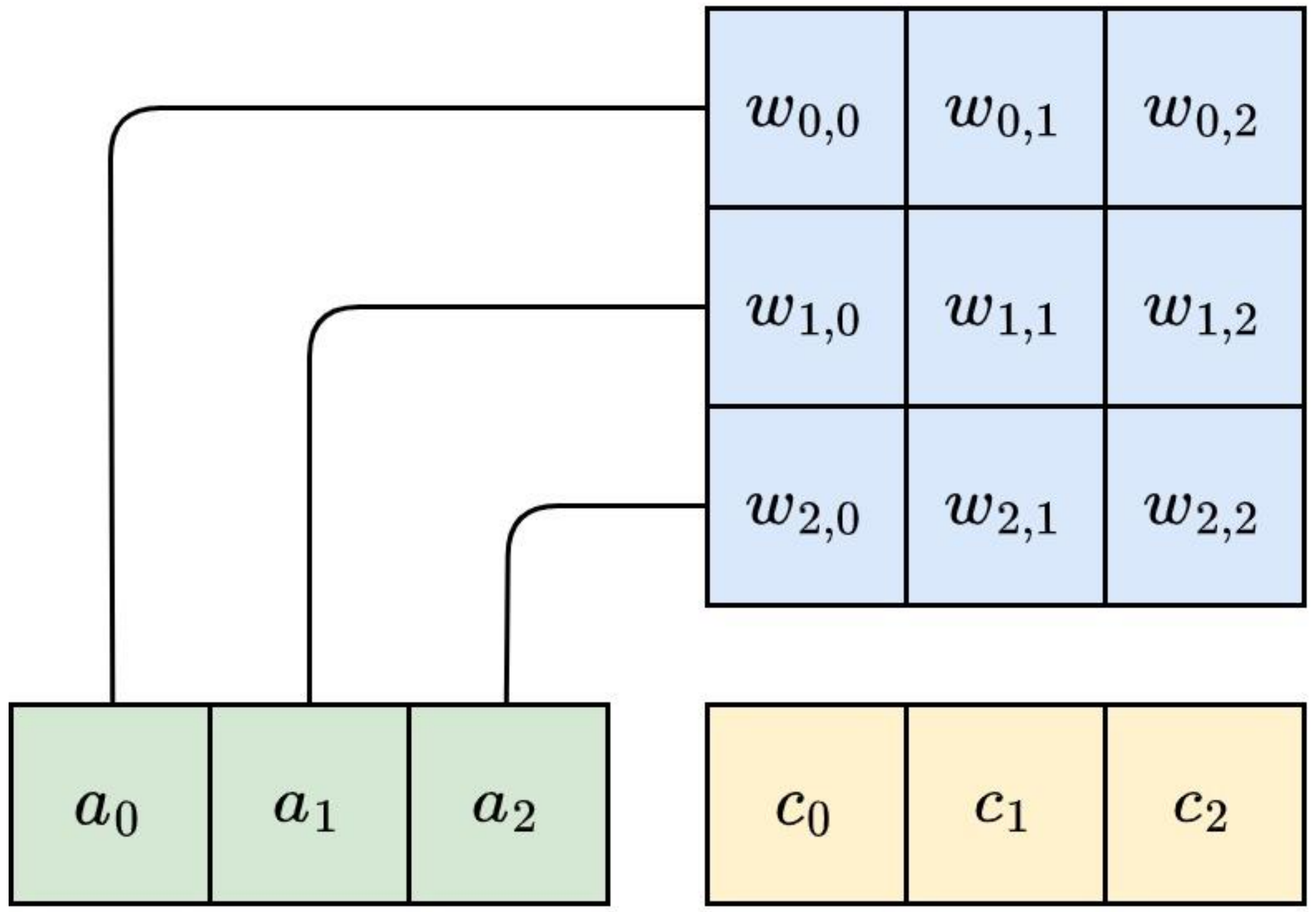
$$A \cdot W = C$$

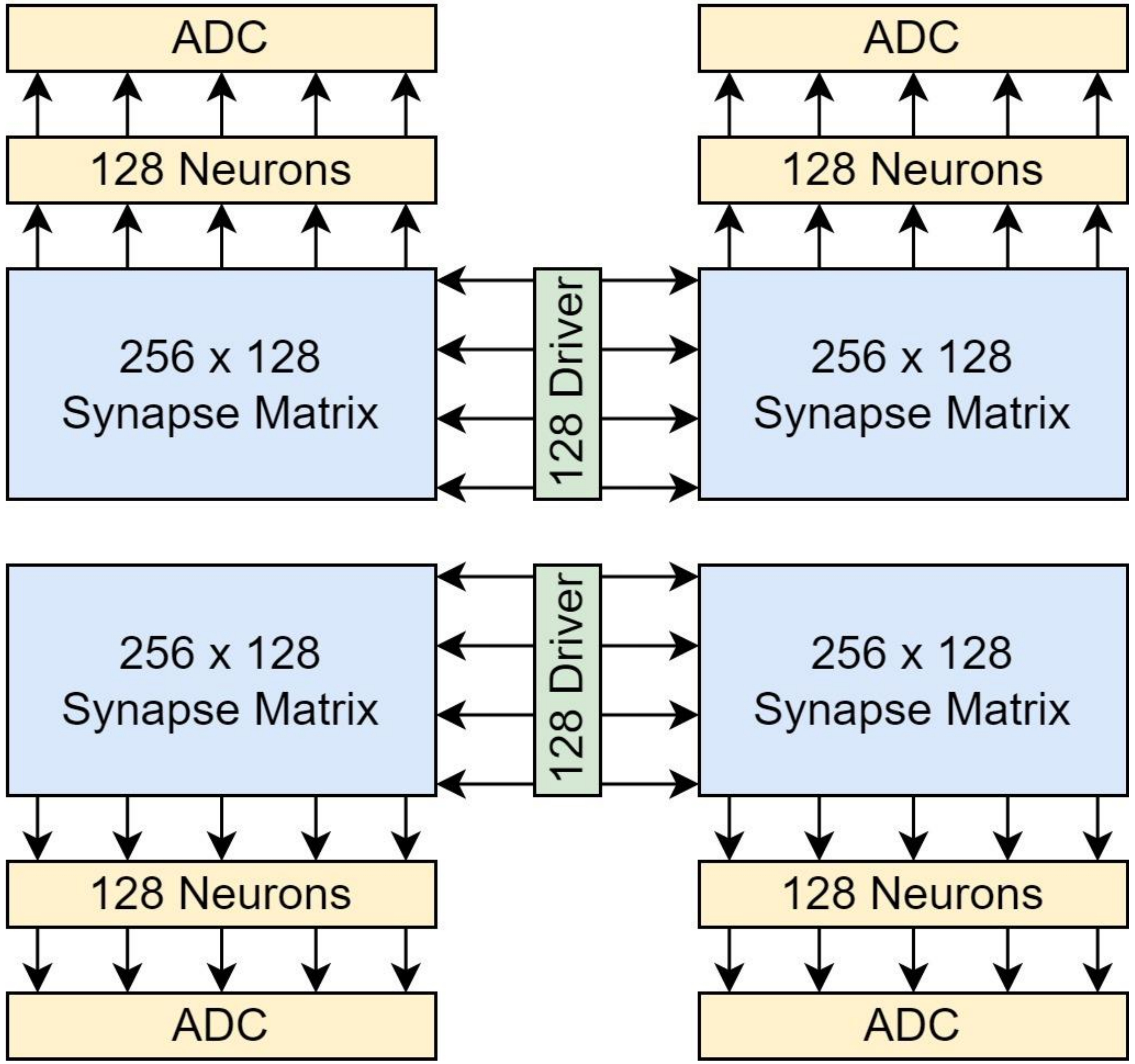




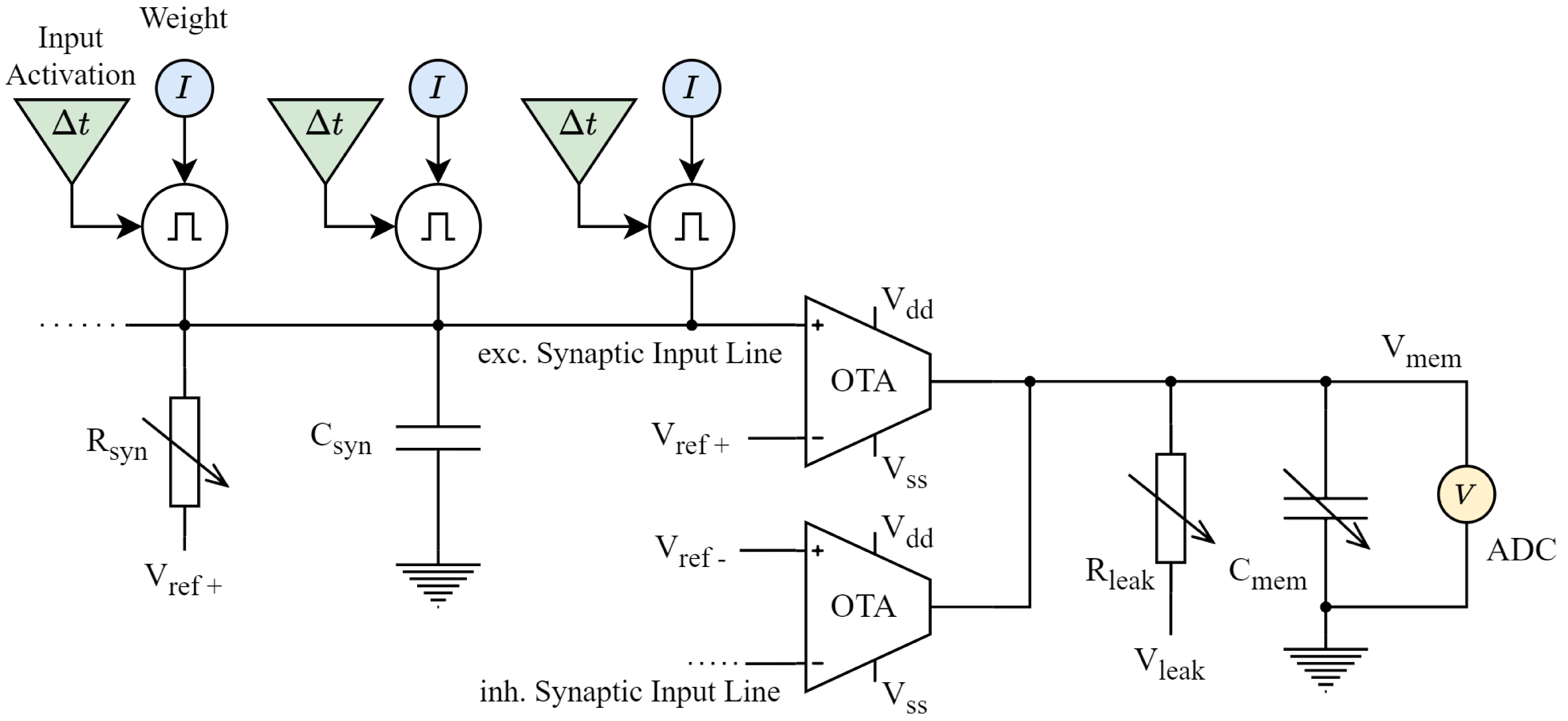
MATRIX MULTIPLICATION

$$A \cdot W = C$$





ANALOG MATRIX MULTIPLICATION



ANALOG MATRIX MULTIPLICATION

ANALOG MATRIX MULTIPLICATION

ANALOG MATRIX MULTIPLICATION

ANALOG MATRIX MULTIPLICATION

ANALOG MATRIX MULTIPLICATION

PROGRAMMING INTERFACE

Pytorch Extension “hxtorch”

- Global chip initialization with static functions

```
hxtorch.init_hardware(calib_path)
```

- Replacements for Linear, Conv1d, Conv2d, …

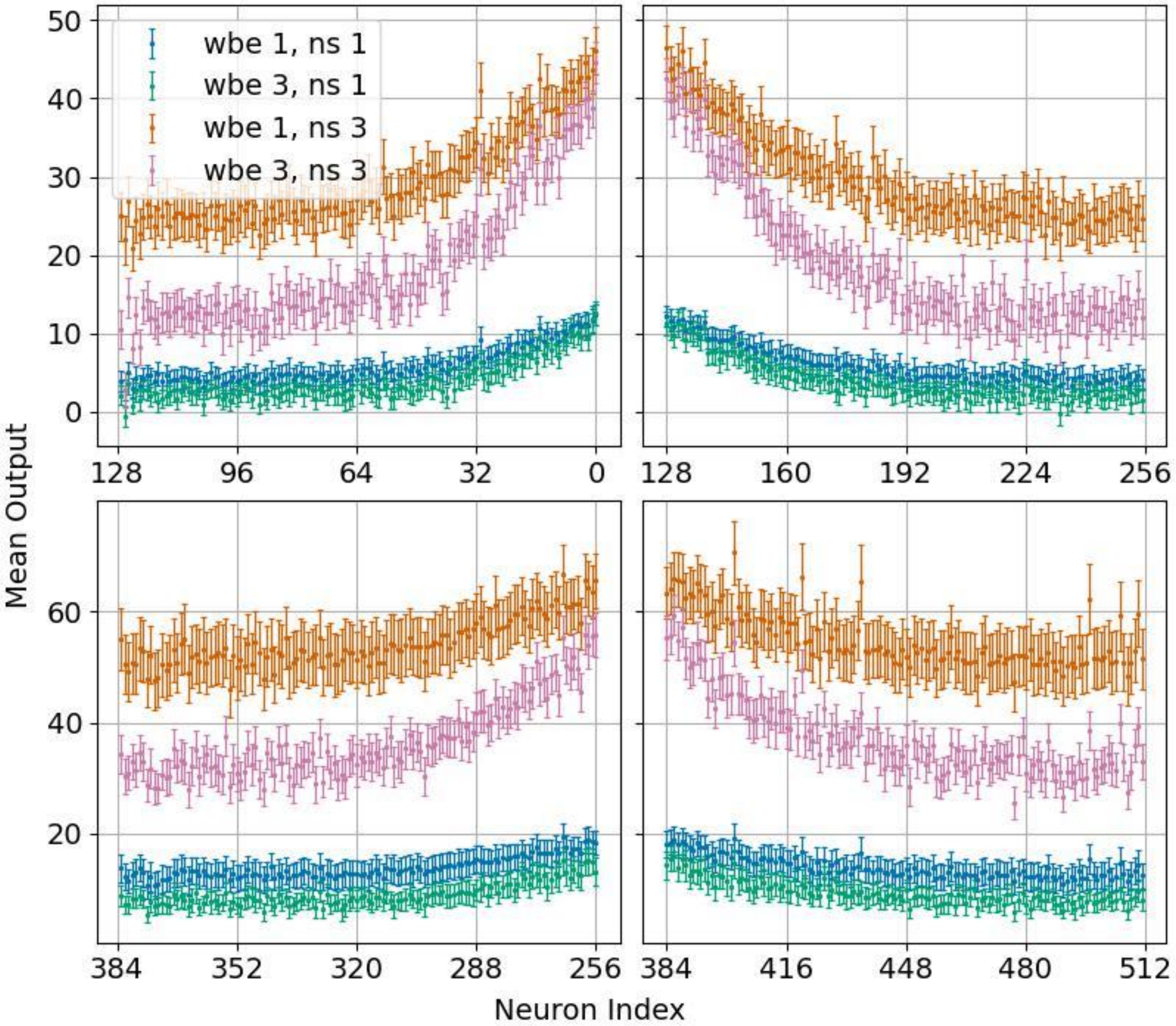
```
hxtorch.Linear( in_features, out_features, bias,
                num_sends, wait_between_events)
```

Python Library “calix”

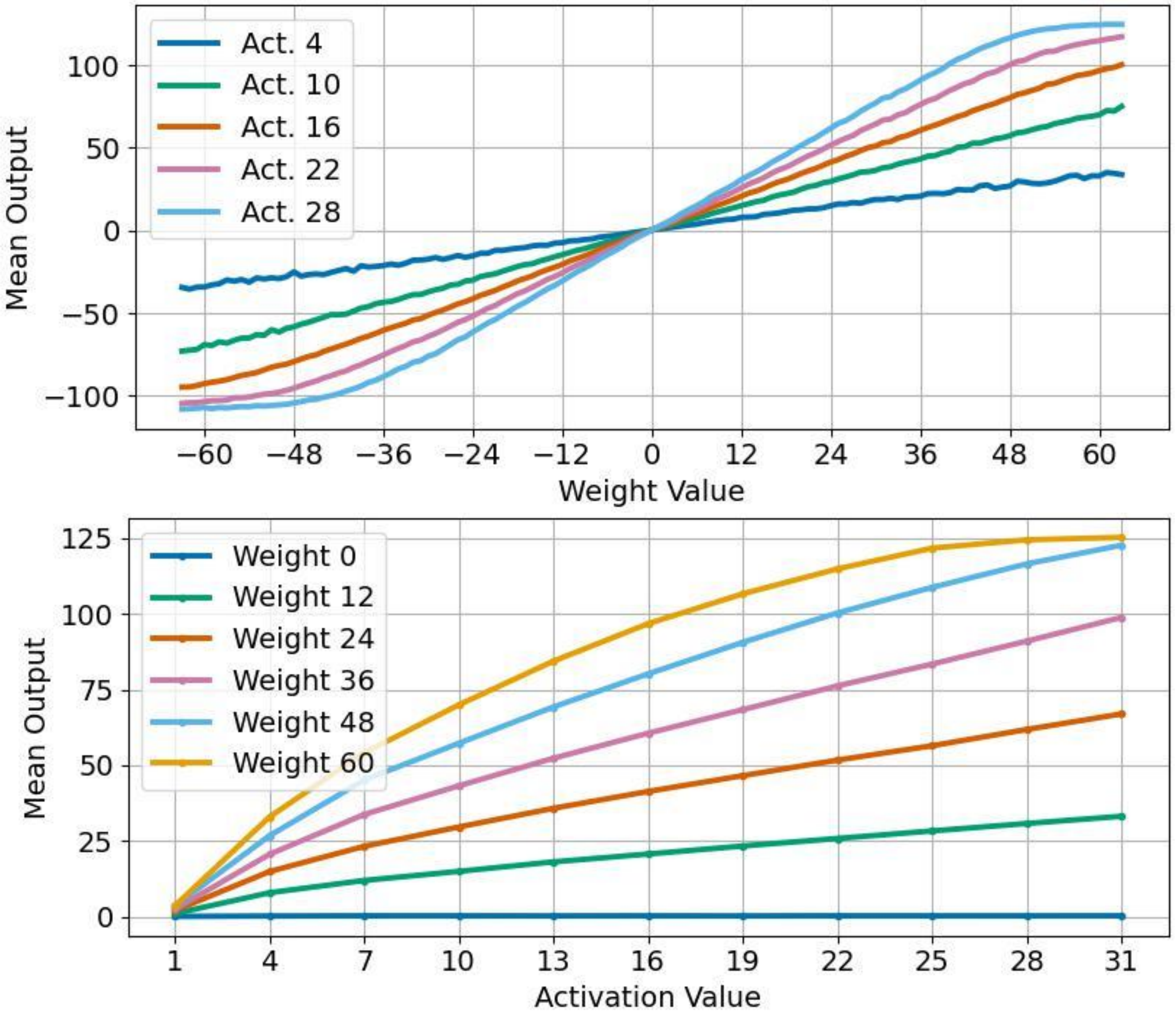
- Default calibration routines
- Allows custom parameter targets

CAVEATS OF ANALOG HARDWARE

Static Variations



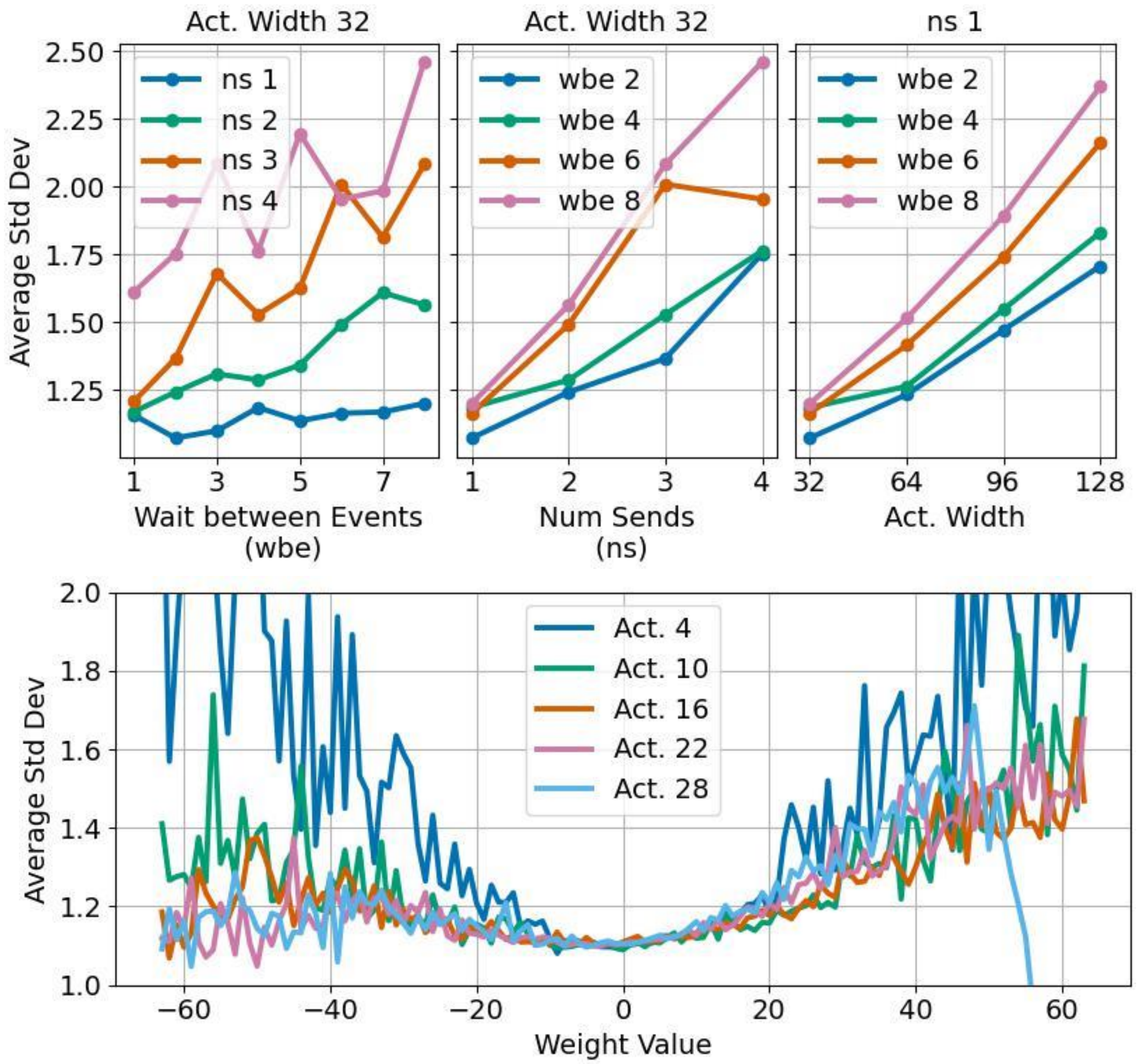
Non-Linearities



NOISE

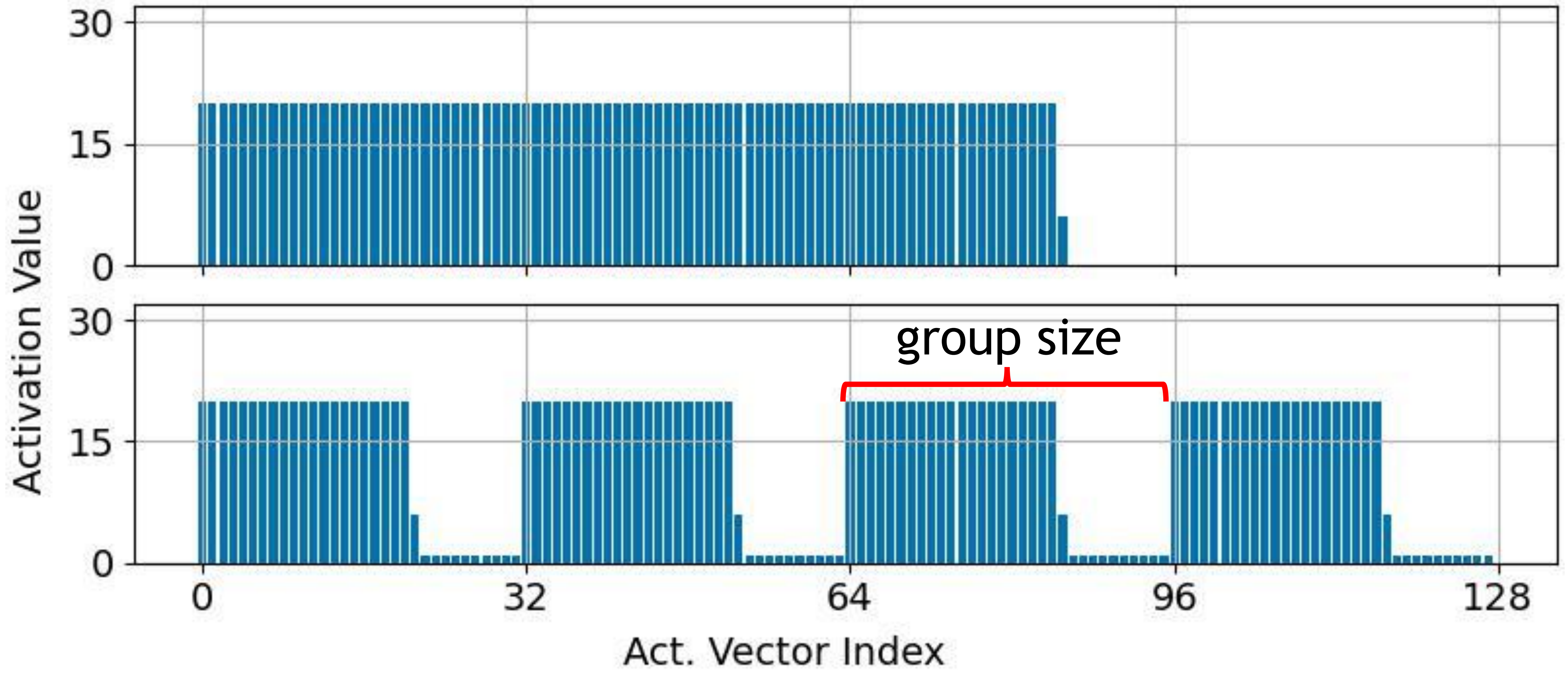
Many factors increase Noise:

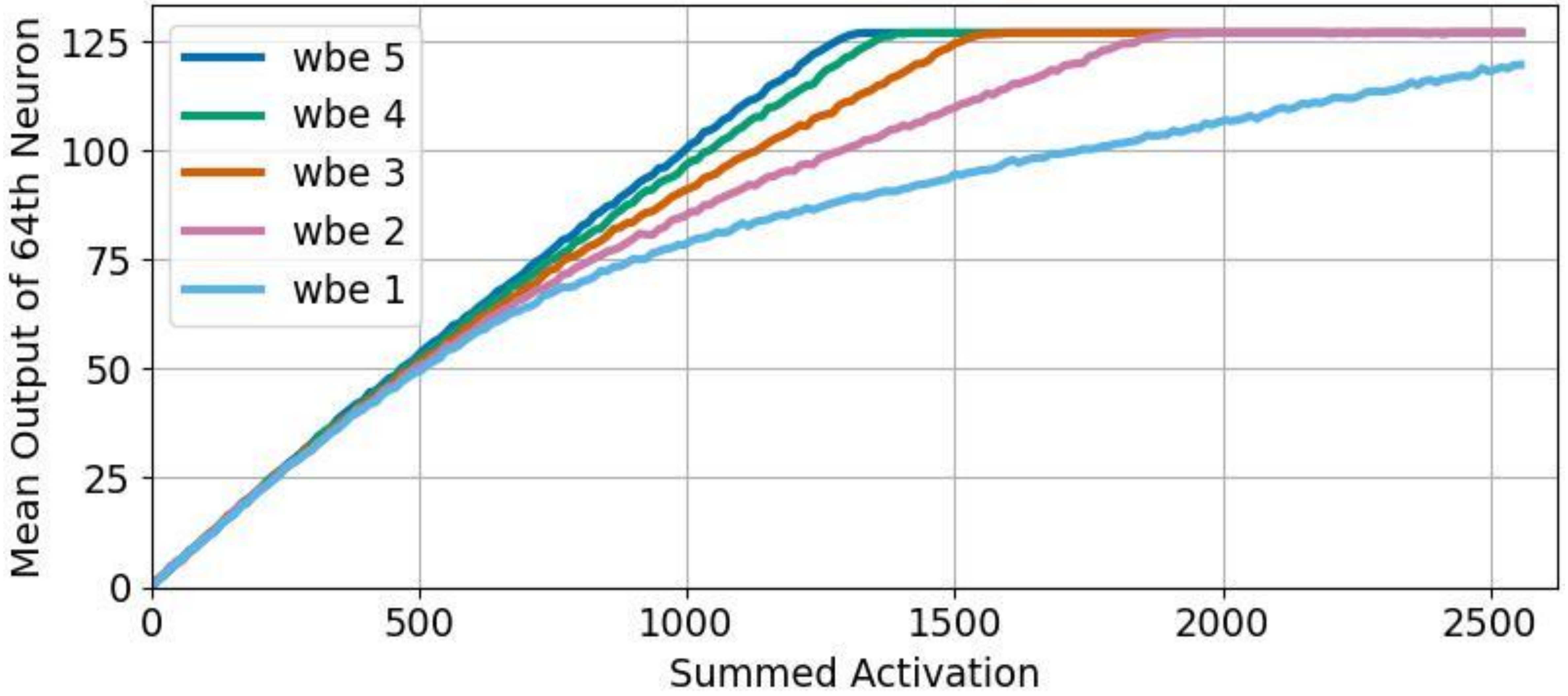
- Wait between Events
- Num Sends
- Number of input features
- Weight Magnitude
- OTA Gain
- Possibly more

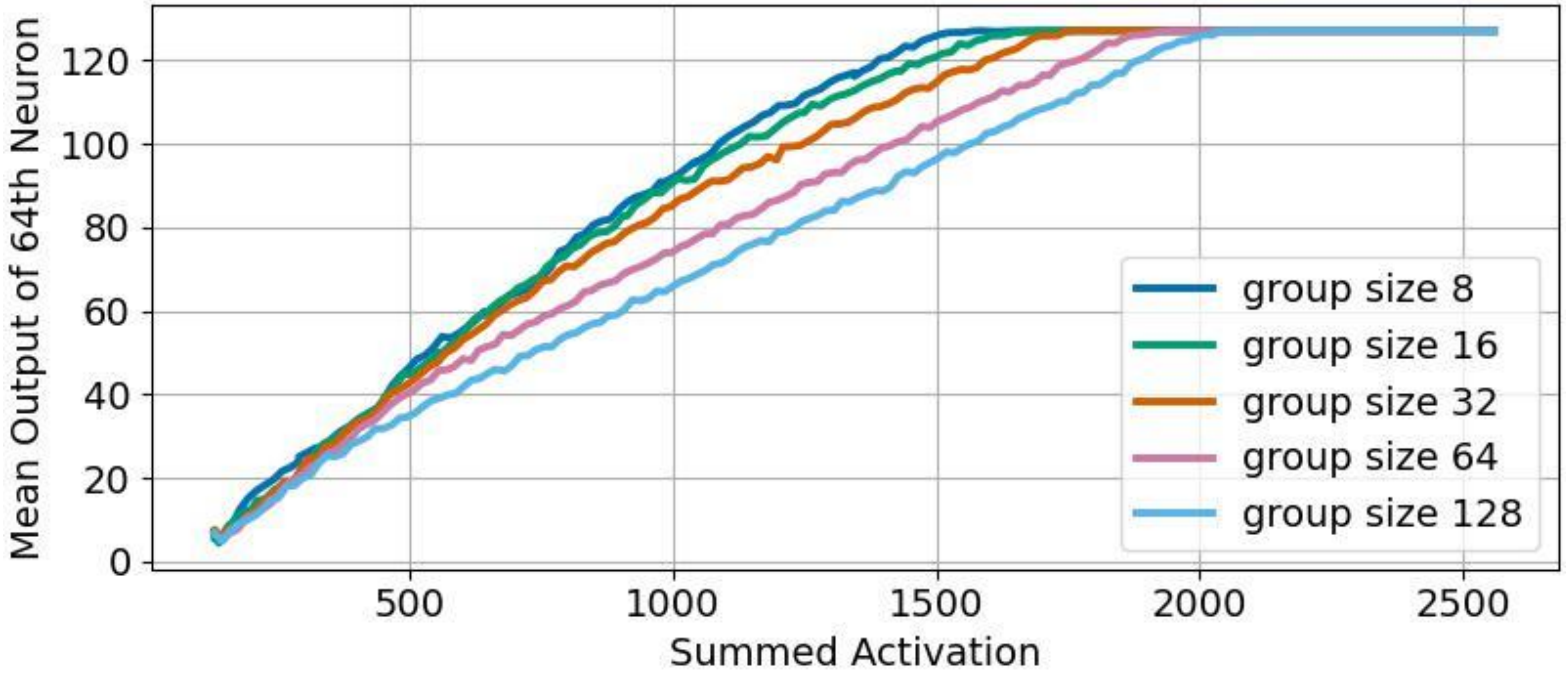


DYNAMIC SATURATION

Saturation depending on input magnitude and order

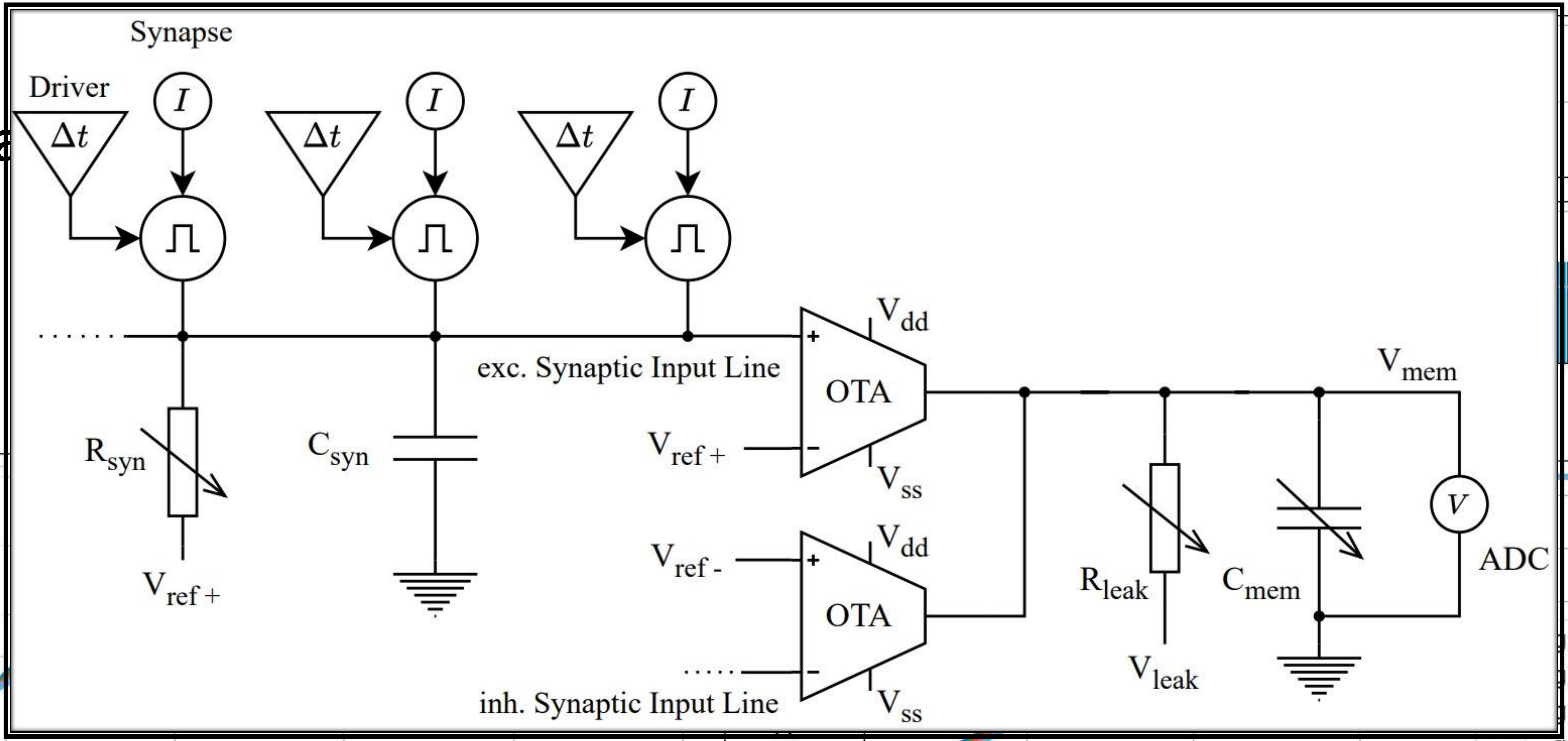


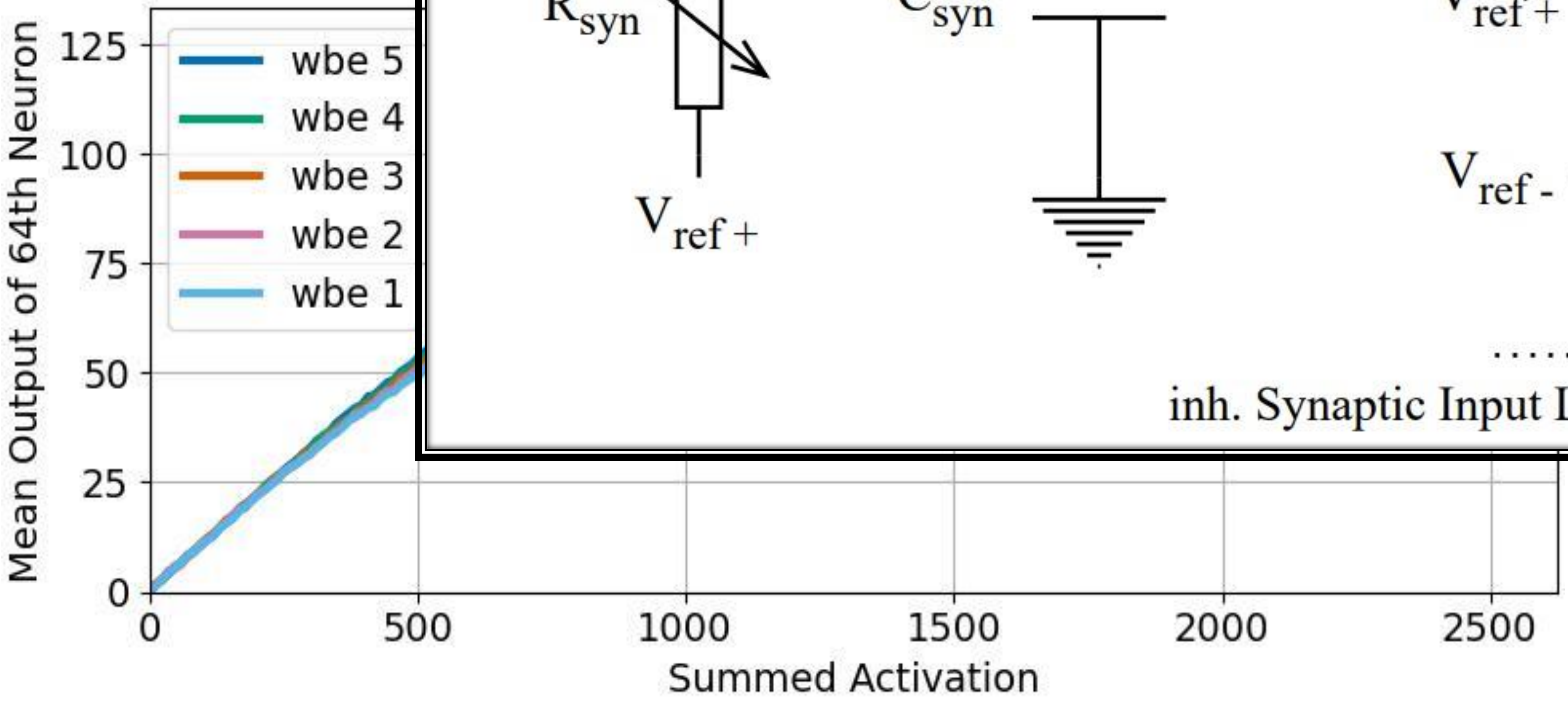


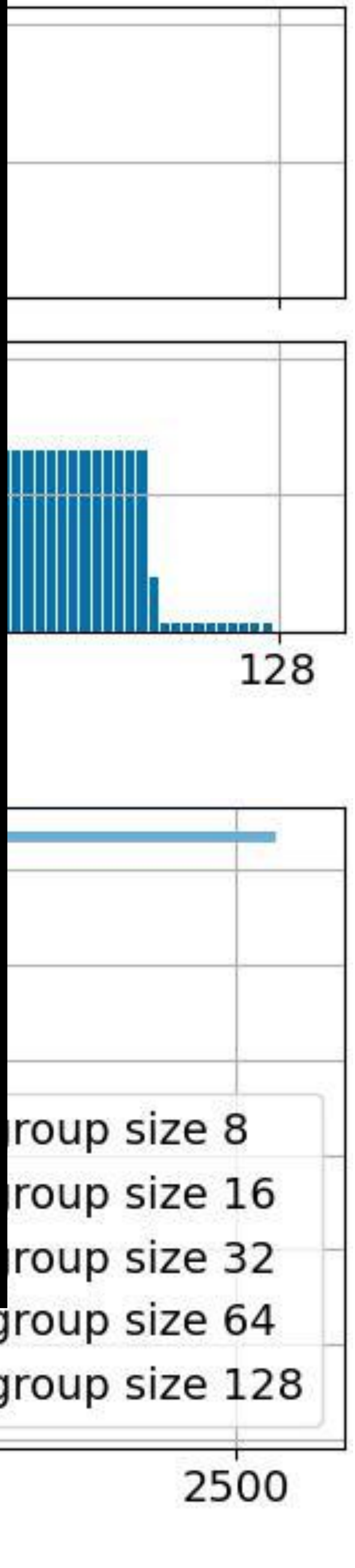


DYNAMIC SATURATION

Satura
input

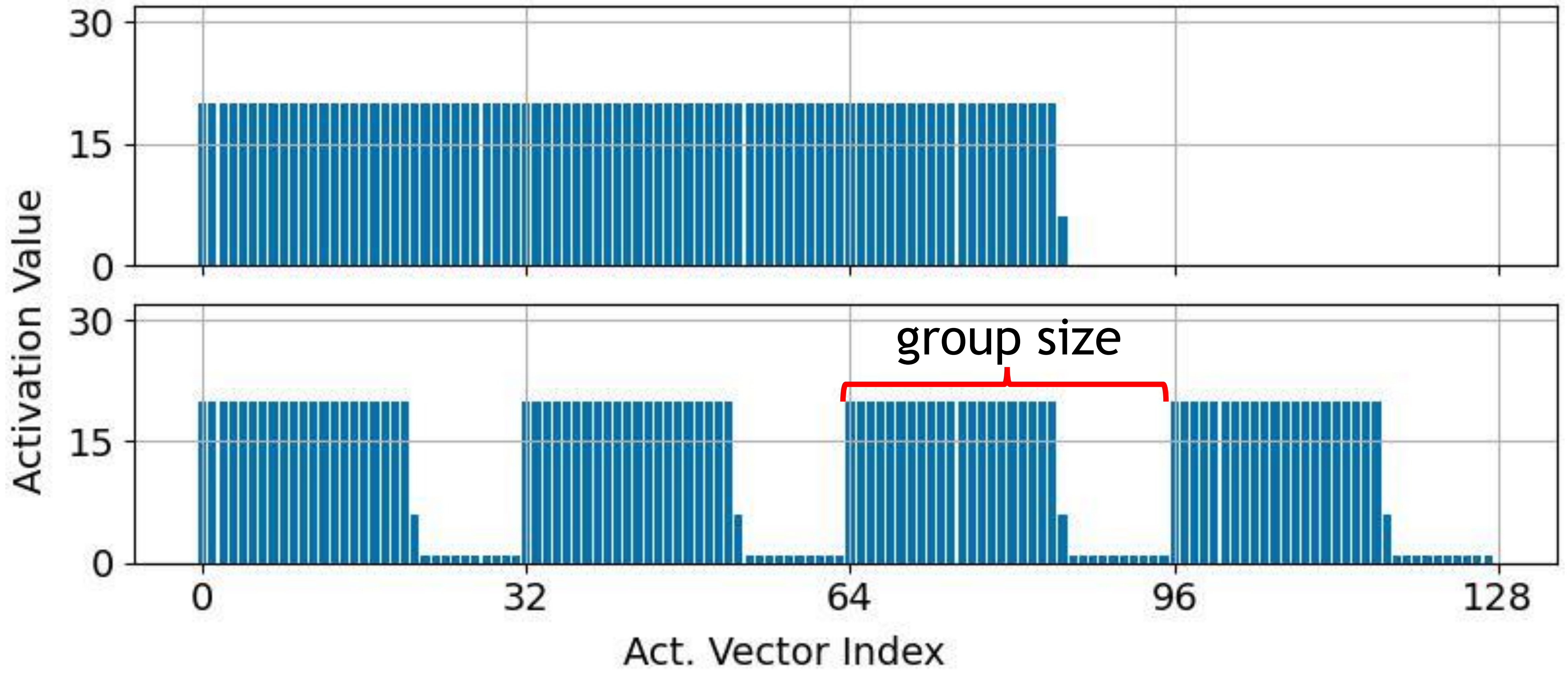


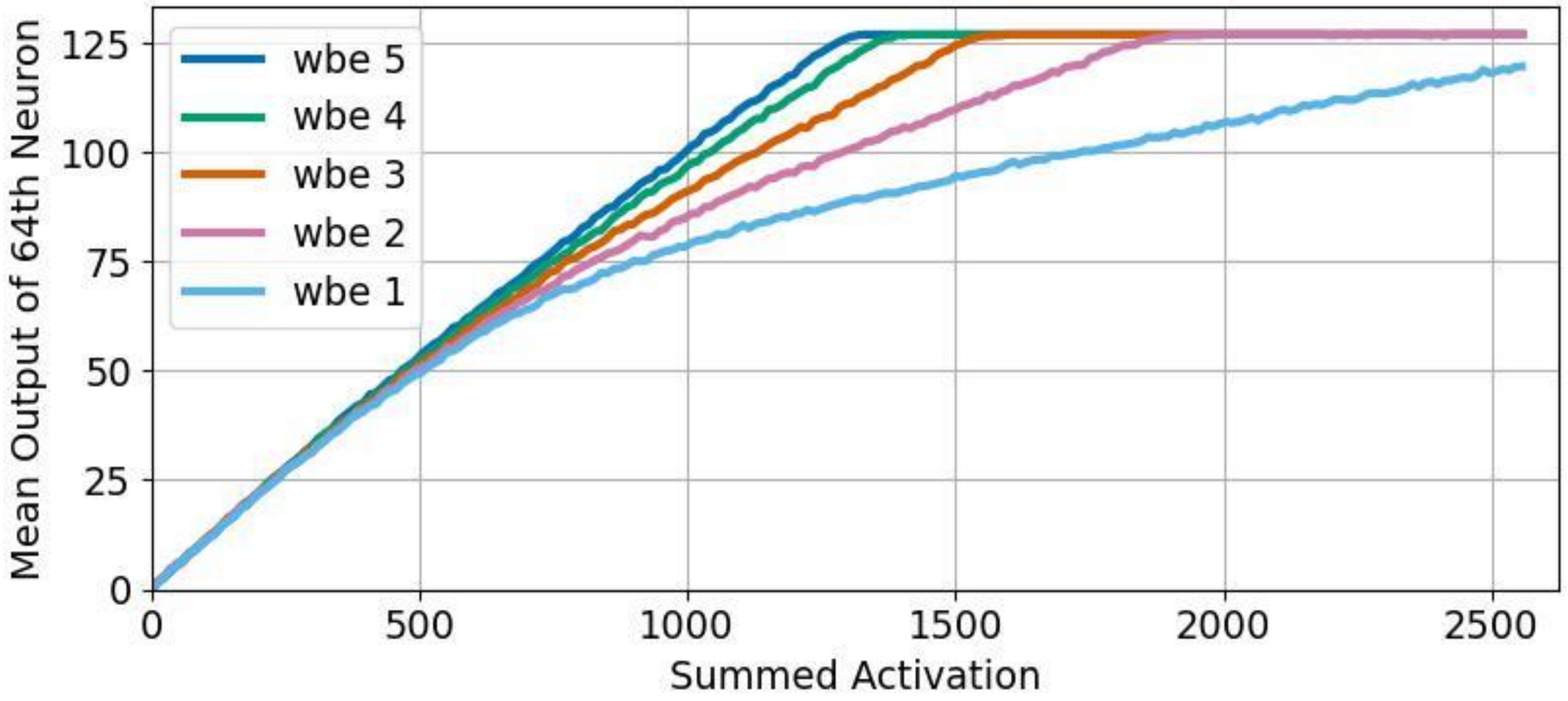


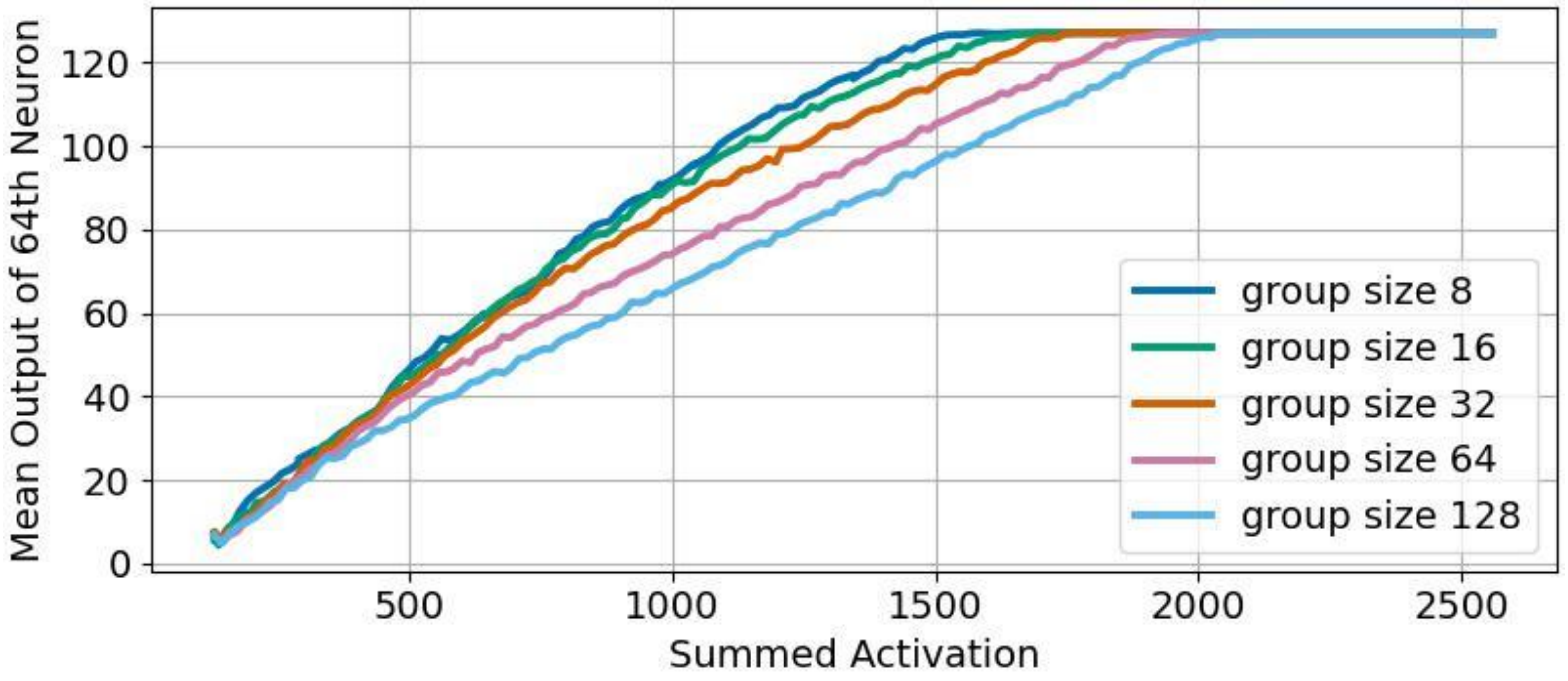


DYNAMIC SATURATION

Saturation depending on input magnitude and order







DEALING WITH IMPERFECTIONS

1. Retraining on the hardware (HW-in-the-Loop)
   Allows adjustment to offsets and gain factors
2. NNs don’t need linear components
3. Improve translation to the analog domain
4. Optimizing the calibration for a specific use-case
   noise vs. dynamic saturation vs. resolution vs. uniformity

DEALING WITH IMPERFECTIONS

1. Retraining on the hardware (HW-in-the-Loop)
   Allows adjustment to offsets and gain factors
2. NNs don't need linear components
3. Improve translation to the analog domain
4. Optimizing the calibration for a specific use-case
   noise vs. dynamic saturation vs. resolution vs. uniformity

General (items 1–2)

DEALING WITH IMPERFECTIONS

1. Retraining on the hardware (HW-in-the-Loop)
   Allows adjustment to offsets and gain factors
2. NNs don’t need linear components

General

3. Improve translation to the analog domain
4. Optimizing the calibration for a specific use-case
   noise vs. dynamic saturation vs. resolution vs. uniformity

Hardware Specific

DEALING WITH IMPERFECTIONS

1. Retraining on the hardware (HW-in-the-Loop)
   Allows adjustment to offsets and gain factors
2. NNs don't need linear components
3. Improve translation to the analog domain
4. Optimizing the calibration for a specific use-case
   noise vs. dynamic saturation vs. resolution vs. uniformity

(1–2: General; 3–4: Hardware Specific)

NNs can tolerate static imperfections:
-reduced uniformity
-static non-linearities

NNs are sensitive to dynamic imperfections:
- Noise
- dyn. saturation

TRANSLATION APPROACHES

Uniform symmetric quantization:

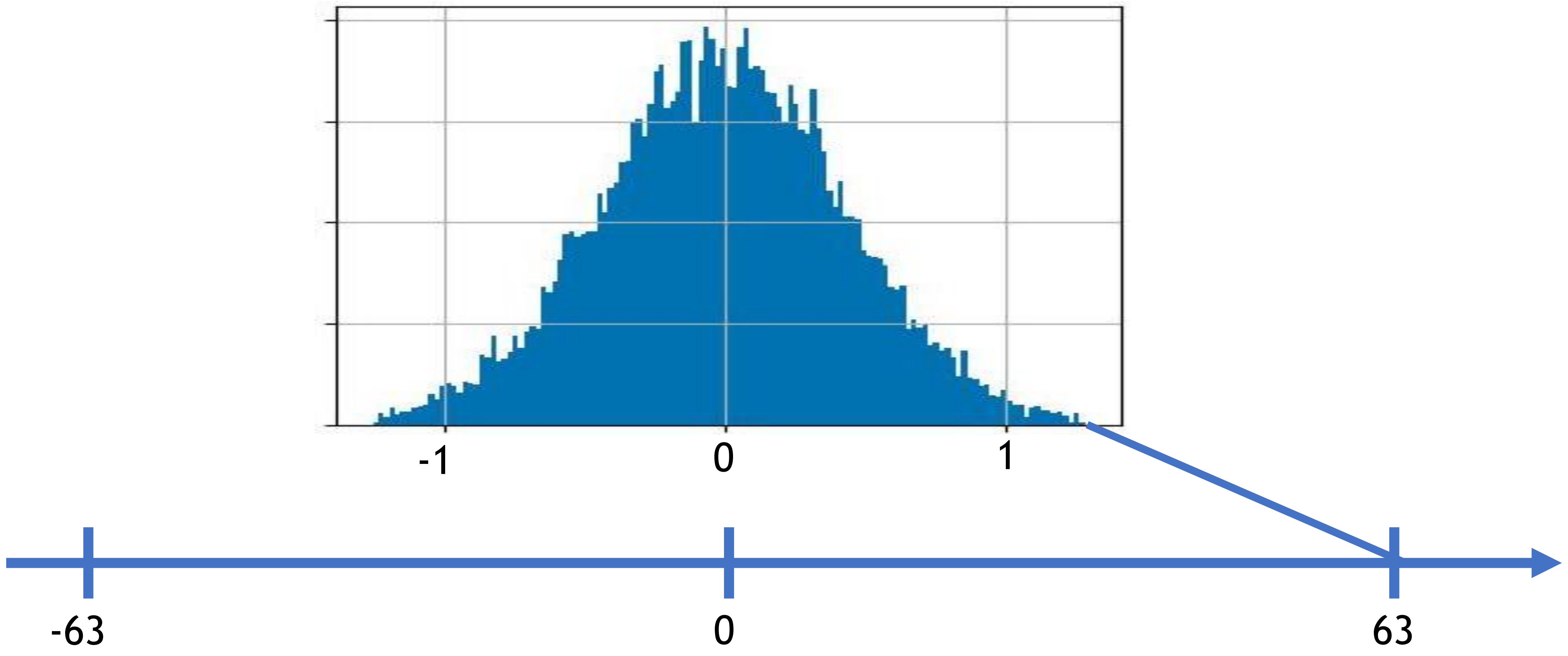
$$y = \text{quantize}(x) = \text{clip}\big(\text{round}(x \cdot s)\big)$$

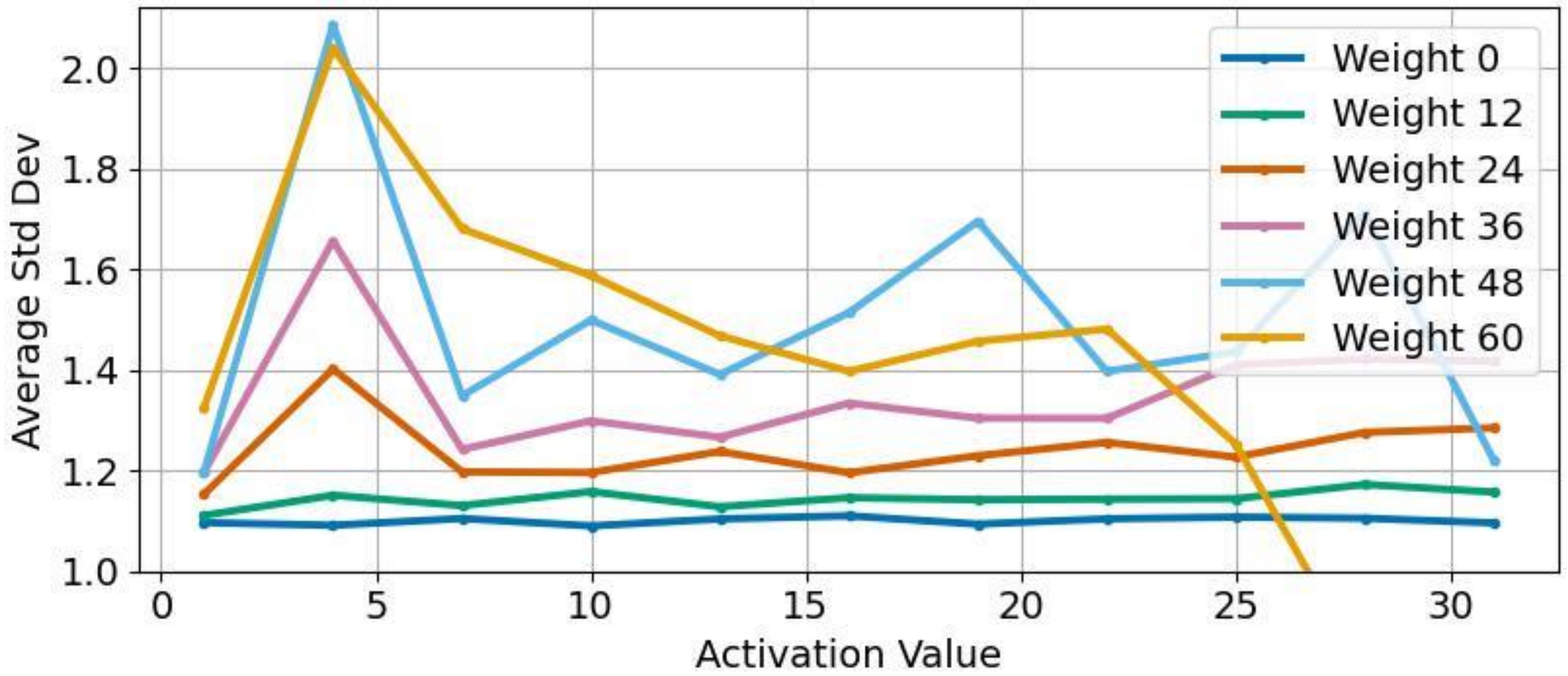
But how to choose the scaling factor during training?

1. Use a static scaling factor
2. Dynamically adjust the scaling factor for each batch
3. Use an exponential moving average

Can we clip small noisy input activations?

Turns out ineffective ☹





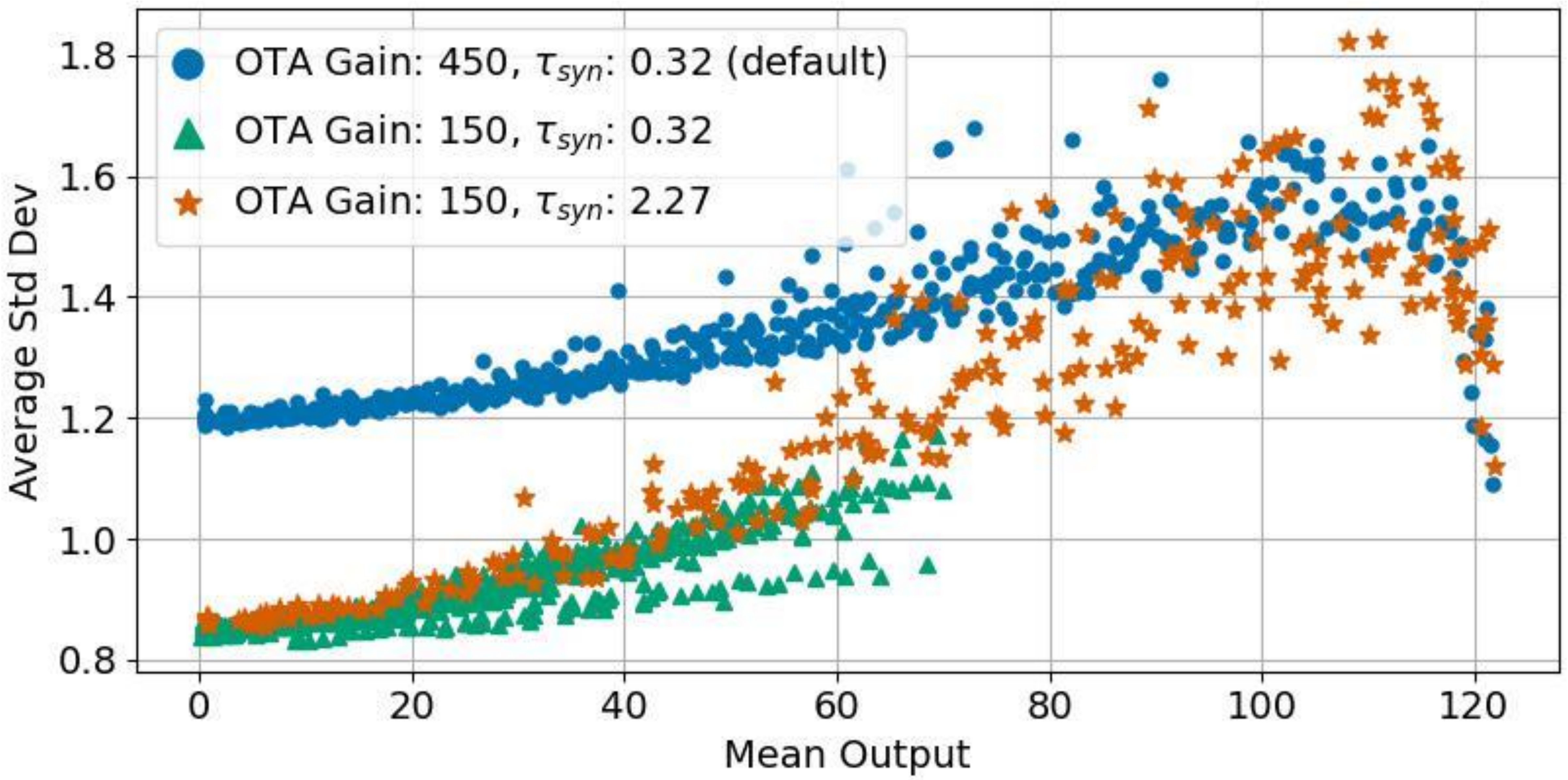
CUSTOM CALIBRATION

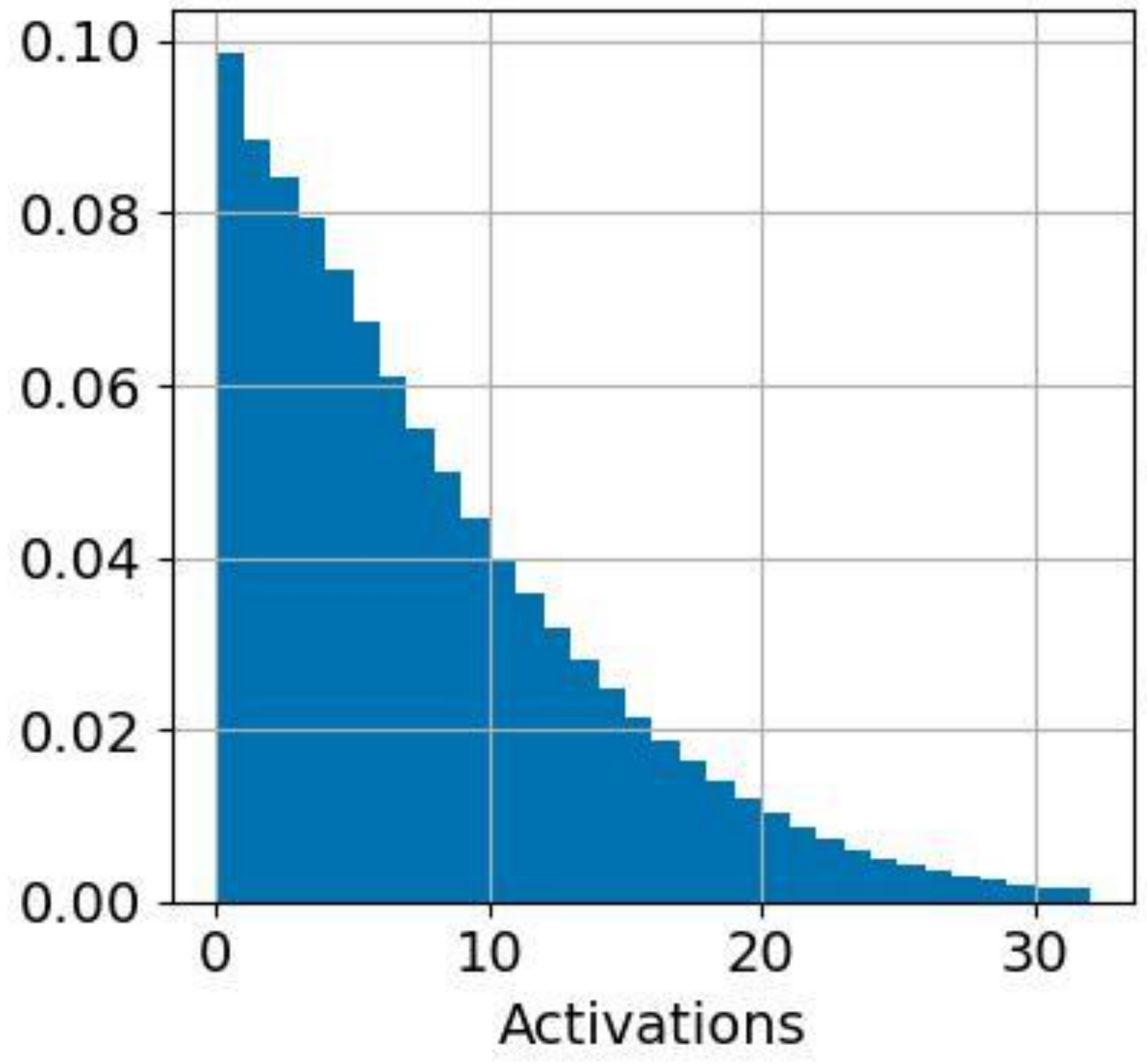
Reducing OTA Gain reduces Noise

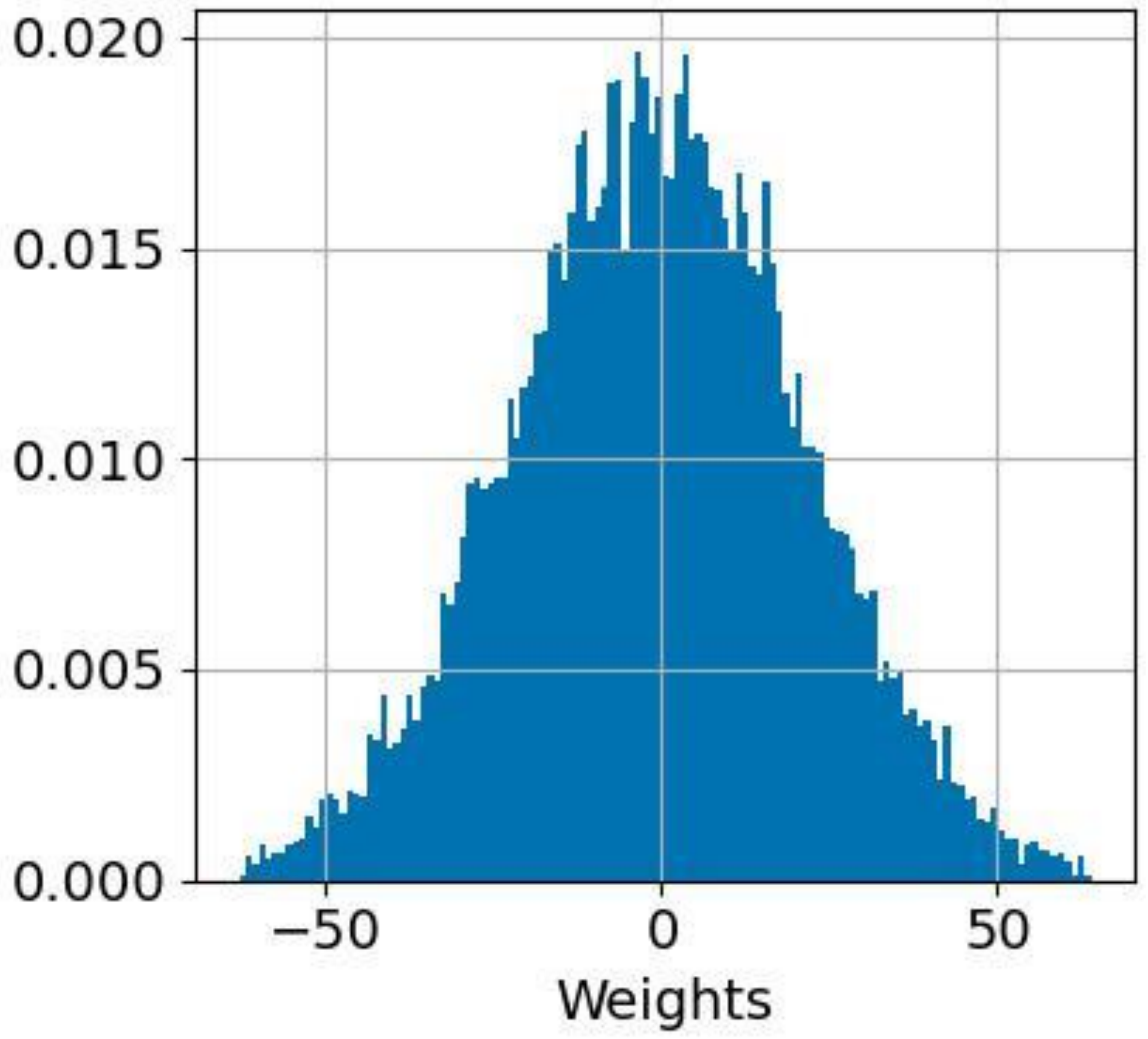
The synaptic input time constant increases gain
<u>but</u> also the risk of dynamic saturation

→ Increasing the synaptic input time constant restores gain with smaller noise

Typical distributions of NNs allow an increased time constant without dyn. saturation

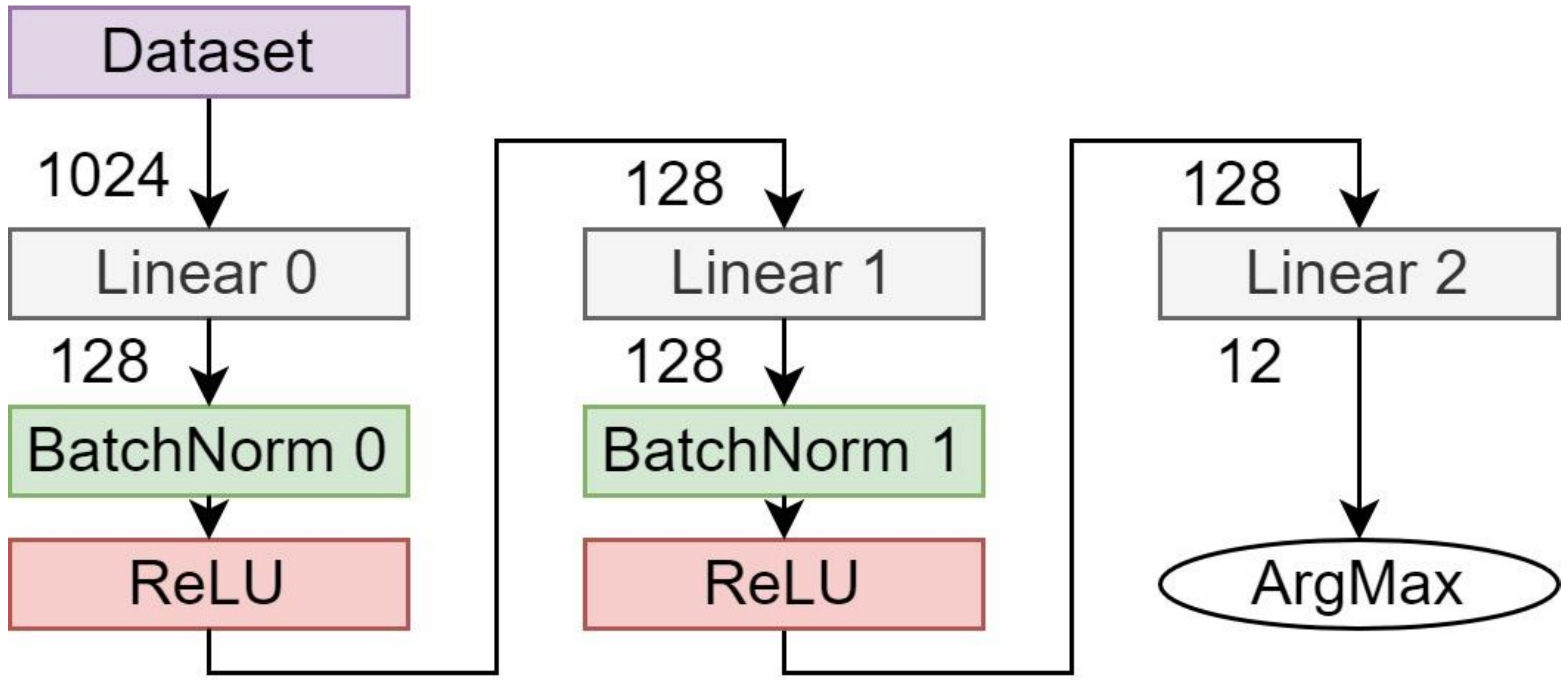


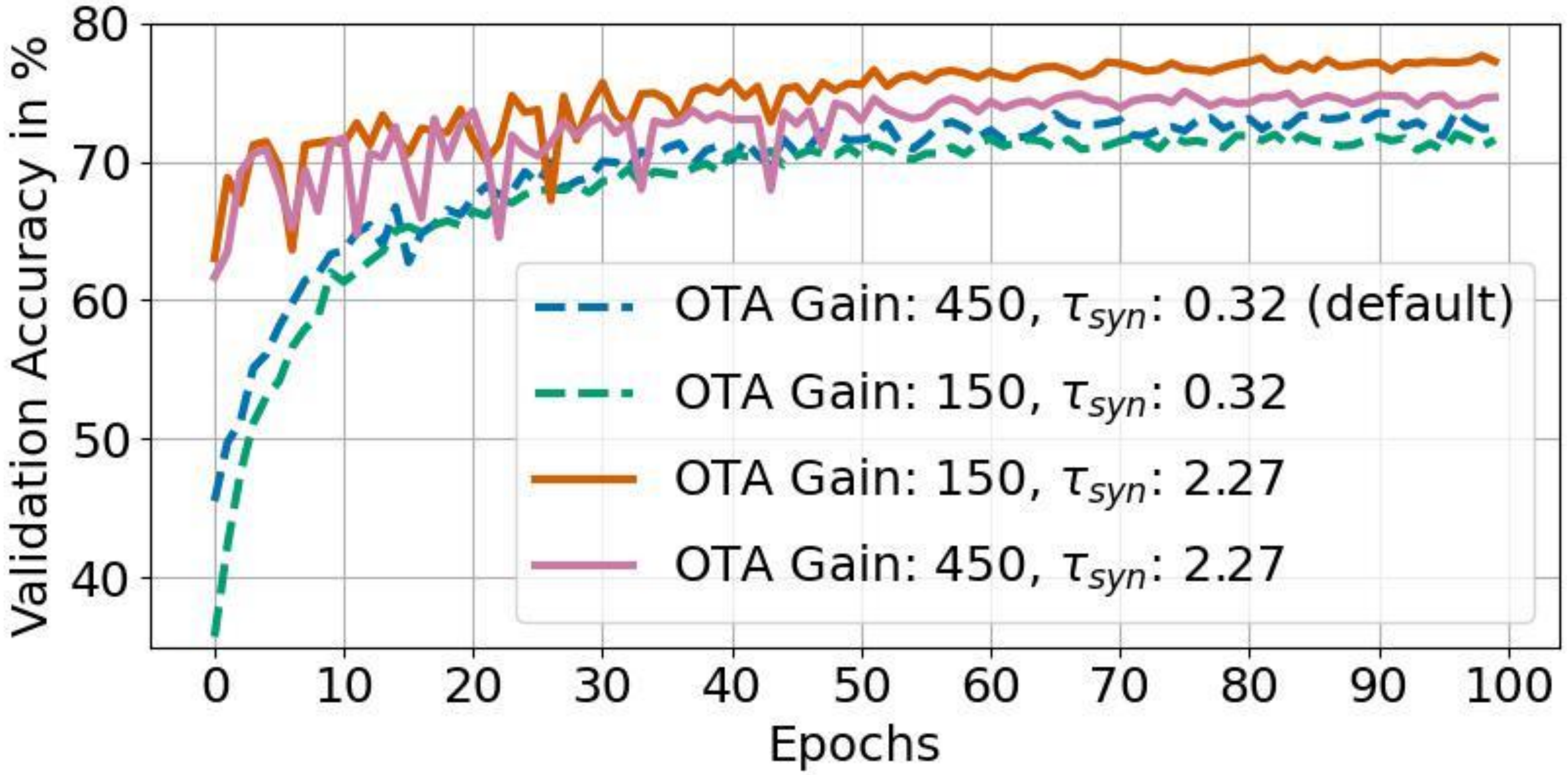




TRAINING RESULTS

- MLP with BatchNorm
- SpeechCommands V1
  Log Mel Spectrogram
- After full precision training:
  Retraining on analog Hardware





Initial accuracy increase

+

Final accuracy increase

CONCLUSION

We show:

- Factors influencing the analog imperfections
- Algorithmic adaptions to the imperfections
- Guidelines to improve the calibration
- Accuracy improvement of 7% with our custom calibration

HW Calibration	Default	Custom
Plain Transfer	17.73 %	57.93 %
HW-in-the-loop	68.74 %	**75.55 %**

CONCLUSION

We show:

- Factors influencing the analog imperfections
- Algorithmic adaptions to the imperfections
- Guidelines to improve the calibration
- Accuracy improvement of 7% with our custom calibration

HW Calibration	Default	Custom
Plain Transfer	17.73 %	57.93 %
HW-in-the-loop	68.74 %	**75.55 %**

Recommendations:

1. Static quantization scaling
2. Reduce OTA Gain to reduce base noise
3. Increase global gain for few input features
4. Reduce integration time

TRY IT YOURSELF

Test the BrainScaleS-2 system from your browser:

ebrains.eu