Bio-Inspired Edge Learning on the Akida Event-Based Neural Processor

S. Brüers, K. D. Carlson, M. Cheng, S. Crouzet, M. Devarajlu, H. Makki, D. McLelland, N. Oros, C. Wilson & K. Wu.

BrainChip

Contacts: kcarlson@brainchip.com rlevinson@brainchip.com

Summary

The Akida event-based neural processor is a high-performance, low-power SoC targeting edge applications, distinguishing itself from traditional deep learning accelerators (DLAs) through 2 key features:

Feature 1: Low-Power CNN Inference Using Event-Based Processing

Akida runs CNNs/DNNs in the event-domain, which enables:

- a reduction in computation (40-60%) when compared to non-event-based designs
- novel speed/power/accuracy trade-offs

Akida distributes network computation across ~80 small neural processing units (NPUs), each with its own collocated processing and memory, resulting in:

- a more granular distribution of computation to layers that need it most
- a reduction in data movement and SRAM reads
- eliminating the need for off-chip memory access or external host CPU (in many cases)

Akida utilizes primarily 1 to 4-bit weight and activity quantization and depthwise separable convolution, which:

- Reduces the total required memory
- Reduces the computational cost of each event processed

Feature 2: On-Chip Learning

Akida incorporates a bio-inspired learning algorithm, adapted from spike timing-dependent plasticity (STDP).

Combined with pre-trained feature extractor networks, this allows us to perform learning directly on the chip.

Akida SoC

Figure 1 below shows the Akida SoC, which includes 80 NPUs connected via a mesh network, pixel-spike converter, on-chip M-class CPU, and data input/external memory interfaces.

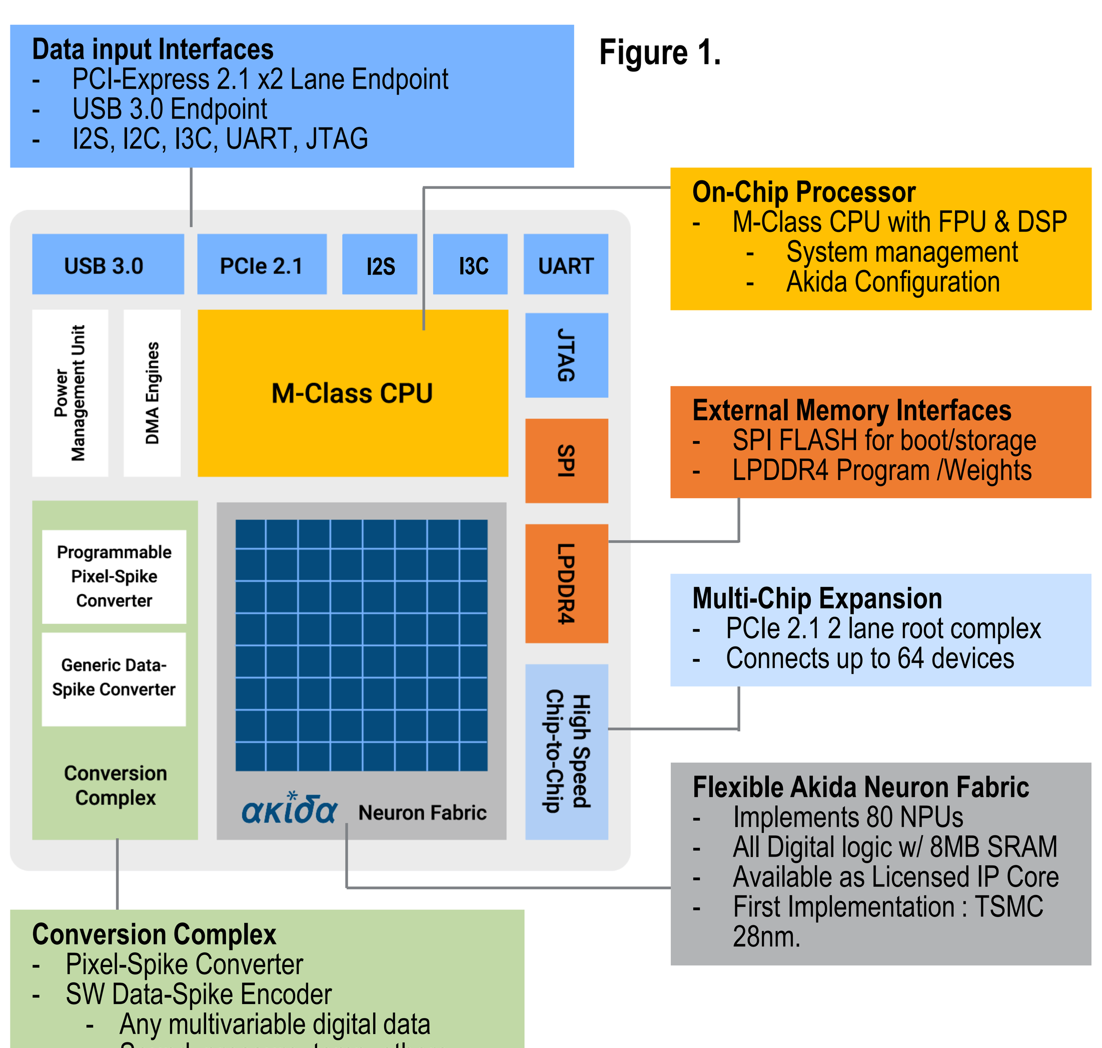


Figure 1.

Table 1.

Supported Ops	Kernel Size	Stride	Input Event	Weights
Image convolution (RGB888 or grayscale)	Conv: 3x3, 5x5, 7x7 Max Pool: 2x2	1,2,3 2	8-bit	8-bit
Standard convolution	1x1, 3x3, 5x5, 7x7	1	1, 2, 4-bit	1, 2-bit
Point wise convolution	1 × 1	1	N/A	2, 4-bit
Depth-wise convolution	3 × 3, 5x5, 7x7	1	1, 2, 4-bit	2, 4-bit
Max pooling	2 × 2, 3x3	1,2,3 (only for 3x3)	N/A	N/A
Global average pooling	$W_{Input} \times H_{Input}$	N/A	N/A	N/A
Fully Connected	N/A	N/A	1, 2-bit	1, 2, 3, 4-bit

The Akida Event-Based Neural Processor

The Akida SoC efficiently processes only events, which are non-zero activation outputs. Many DLAs utilize a single, large matrix multiplication engine (e.g. systolic array) to perform dense computational operations very efficiently. The Akida architecture distributes computation across ~80 NPUs. Each NPU sends only events to other NPUs for processing via the mesh network. Below we show some trade-offs between the Akida architecture and a traditional DLA architecture.

Akida Architecture

- Becomes more efficient as activation sparsity increases
- Computation can be more evenly distributed to reduce throughput and latency
- Each layer can have a different set of weight and activation bit widths (1 – 4)
- Clock speed runs low enough to utilize low-leakage memory
- Most suitable for small to moderately large CNNs that fit completely on-chip

Traditional DLA Architecture

- Cannot take advantage of activation sparsity
- Latency scales linearly with side length of systolic array
- im2col has additional memory overhead *O(CHWK)*
- Most suitable for moderate to very large CNNs/DNNs

The event-based computations (e.g. convolution) Akida performs are algorithmically identical but become increasingly efficient as the activation sparsity increases. **Figure 2** shows a comparison between frame-based and event-based convolution.

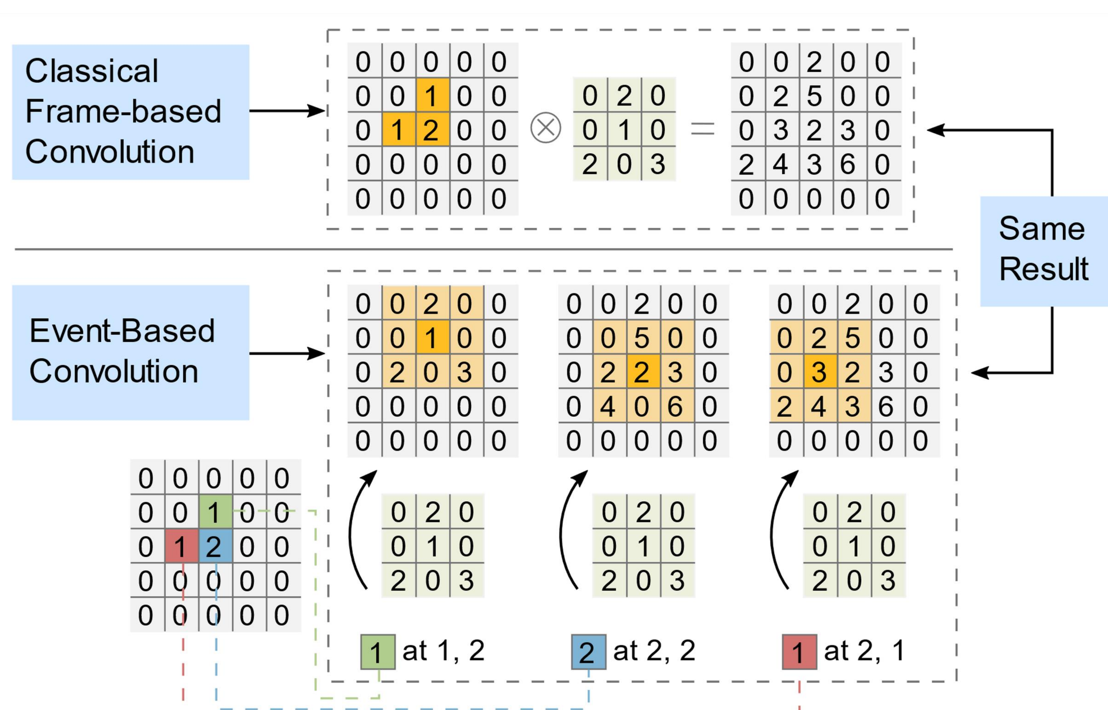


Figure 2. Comparison of frame-based convolution and event-based convolution.

Many CNNs utilize Rectified Linear Units (ReLUs) as the network activation function shown as the blue line in **Figure 3** below. Because ReLUs produce zero output for input values less than zero, many popular CNN models have a mean activation of sparsity of 40-50% (Albericio et al, 2016).

Akida takes advantage of this activity sparsity by processing and sending *only* non-zero activations (events). Instead of processing/sending 32-bit floating point activations, we use 1, 2, and 4-bit quantized ReLUs (QReLUs).

We have observed activation sparsity ranges between ~40% and ~80% in our CNNs and have successfully integrated activation sparsity into our training loss functions in TensorFlow Keras.

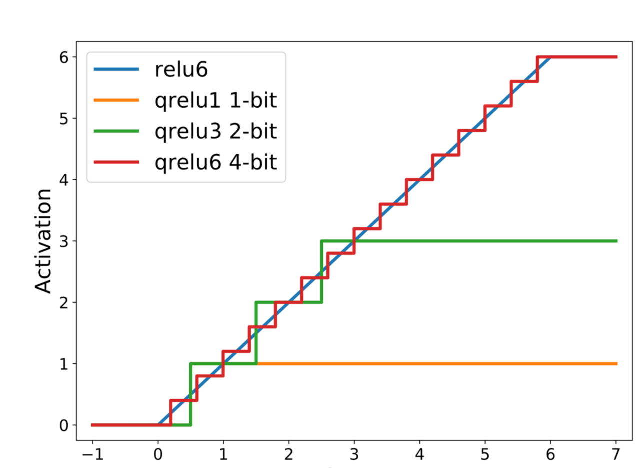
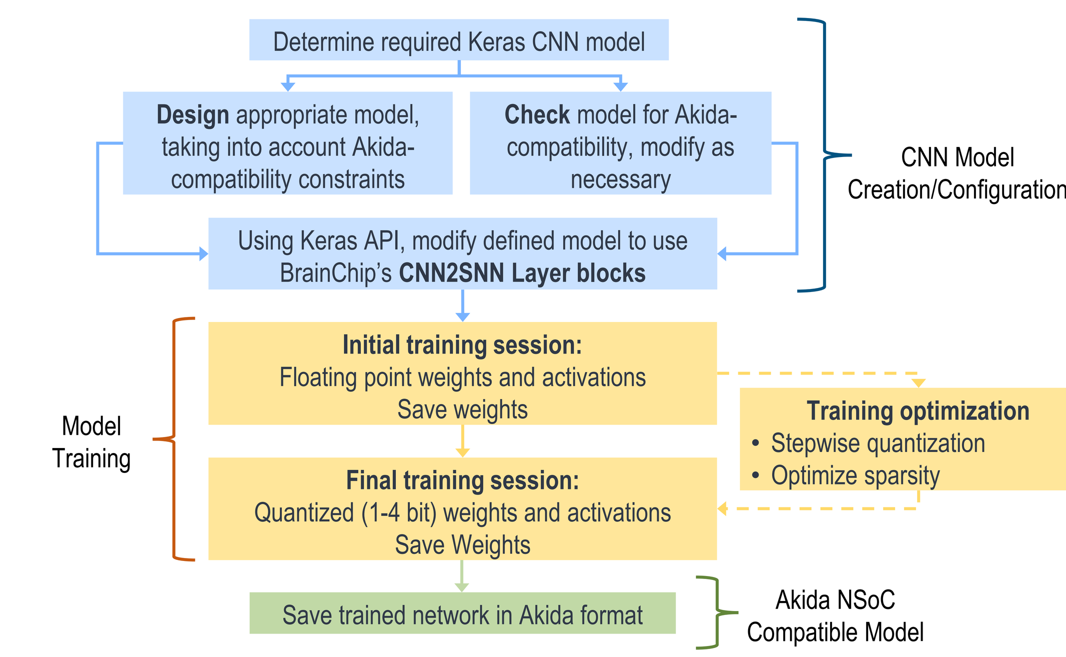


Figure 3. ReLU and QReLU Activations.

Low-Power Pretrained Networks for Inference

Our 'cnn2snn' toolkit enables easy preparation of trained networks for low-power inference tasks. The cnn2snn flow is shown below.



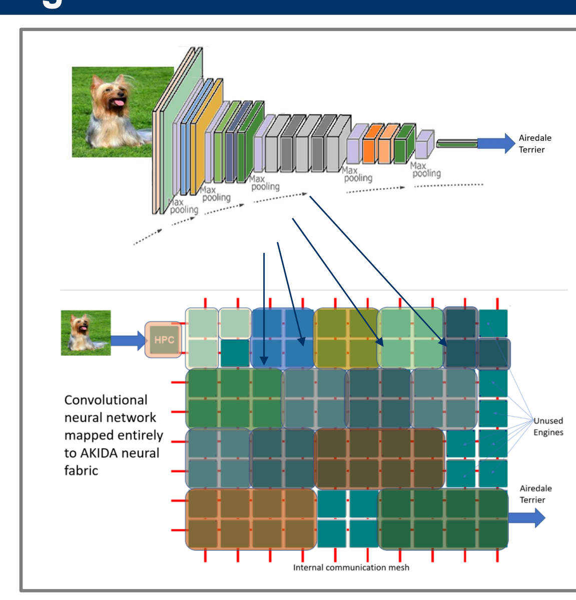
Test Case 1: Image Classification

Model:

- MobileNet V1 (minimally modified)
- Quantized to
  - 4-bit activations (QReLu)
  - 4-bit weights (layer 1: 8-bit)

Dataset:

- ImageNet (ILSVRC2012)



Training:

- From scratch
- Standard data augmentation
- Progressively quantized (in steps)

Performance:

- Top 1: 69% (vs 71% for original, full-precision MobileNet V1)
- Activity sparsity: 42%
- 30 inferences/sec at ~150 mW (simulated estimate)

Test Case 2: Audio Keyword Classification

Model:

- 6-layer CNN (primarily depthwise separable blocks; modified from Zhang et al, 2018)
- MFCC (Mel-frequency cepstral coefficients) generated as preprocessing step
- Quantized to
  - 4-bit activations
  - 4-bit weights (layer 1: 8-bit)

Dataset:

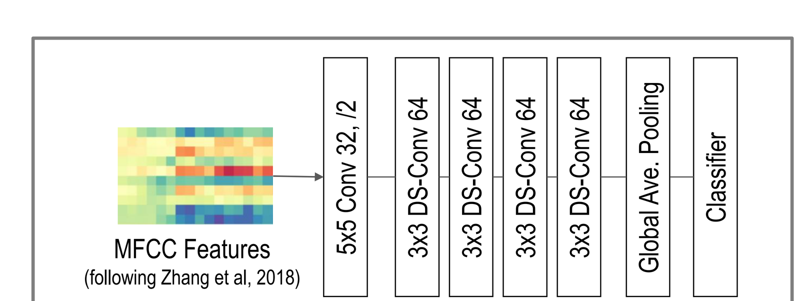
- Google Speech Commands (V1; Warden, 2018)
- 10-word subset, plus silence and 'unknown' classes

Training:

- From scratch, with progressive quantization
- Data augmentation: temporal jitter, background noise

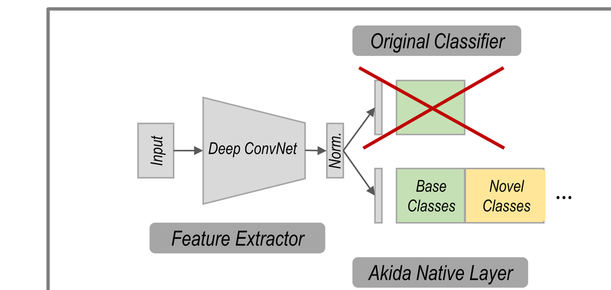
Performance:

- Accuracy: 93% (vs 94% for the reference model with 8-bit weights/activations)
- Activity sparsity: 72%
- 7 inferences/sec at ~200 μW (simulated estimate)



Learning On the Edge

- Base CNN Feature Extractor (pre-trained)
- CNN top layer replaced with Akida Native Learning Layer
- Enables:
  - Few-Shot Learning
  - Continuous Learning
- Easily incorporate many state-of-the-art innovations in CNN training, e.g. *dense classification*, Lifchitz *et al* (2019).



Edge Learning with Akida. *Modified from Qi et al (2018). Learning with the Akida Native layer is similar to their weight imprinting method, but rather than average the embedding activations over multiple samples, a neuroinspired weight update rule is applied instead.*

Mini-Imagenet on the Edge

Model Preparation:

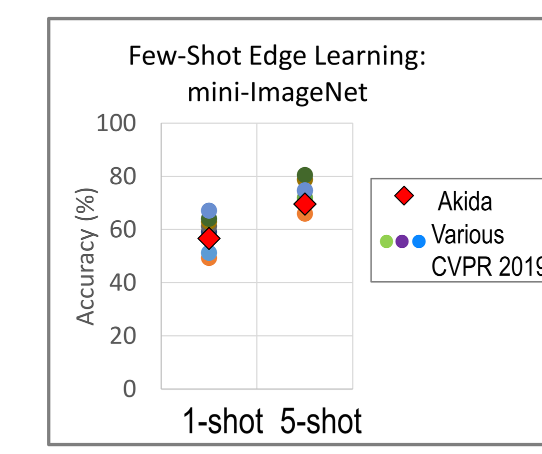
- Quantized MobileNet V1
- Trained on the miniImageNet training set (64 classes, see Vinyals et al, 2016)
- Top layer replaced with Akida Native Learning Layer

Edge Learning:

- 5-way (5/20 classes, cross-validated x 100)
- 1- to 20-shot learning tested
- Simple data augmentation only

Conclusions:

- Competitive Performance despite small network
- Fully Edge-compatible learning



Audio Keywords on the Edge

Model Preparation:

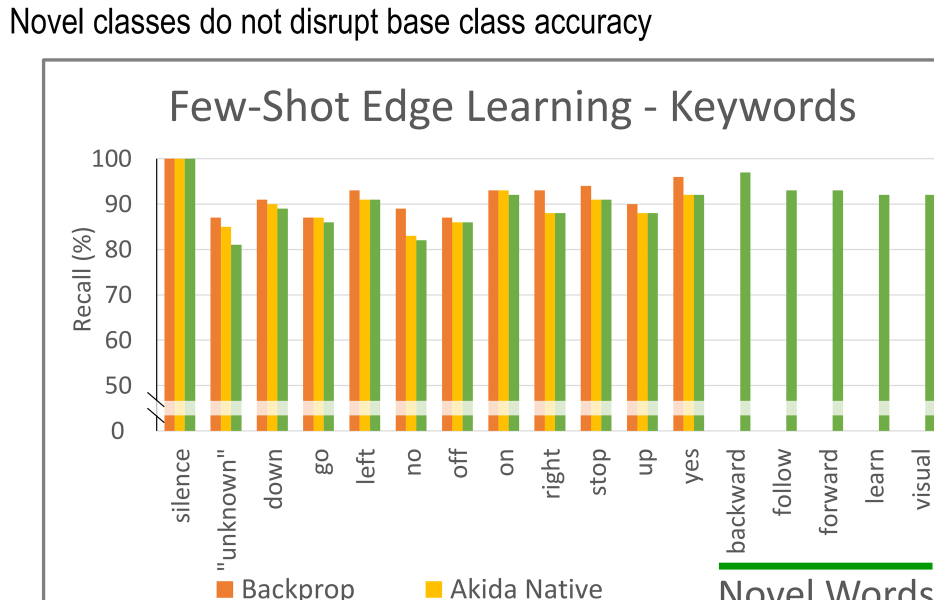
- 6-layer CNN (*see left: Test Case 2*)
- Pre-training on base 10 keywords (*see Figure below*)
- Top layer replaced with Akida Native Learning Layer

Edge Learning:

- Novel classes learned and tested per individual
- 5 novel classes selected based on sufficient repeats per subject
- 4-shot, 3-way learning

Conclusions:

- Quantized edge-compatible Akida layer can be as good as pre-trained CNN
- Edge-learned subject-specific novel classes classified with similar accuracy
- Novel classes do not disrupt base class accuracy



References

- Albericio, J., Judd, P., et al. (2016). Cnvlutin: Ineffectual-neuron-free deep neural network computing. ACM SIGARCH Computer Architecture News, 44(3), 1-13.
- Warden, P. (2018) 'Speech commands: A dataset for limited-vocabulary speech recognition'. arXiv:1804.03209v1 [cs.CL]
- Vinyals, O., Blundell, C., Lillicrap, T., Kavukcuoglu, K. & Wierstra, D. (2018) 'Matching Networks for One Shot Learning'. arXiv:1606.04080 [cs.LG]
- Zhang, Y., Suda, N., Lai, L. & Chandra, V. (2018) 'Hello Edge: Keyword Spotting on Microcontrollers'. arXiv:1711.07128 [cs.SD]