STICKER: An Energy-Efficient Multi-Sparsity Compatible Accelerator in 65-nm CMOS

Abstract

STICKER is an energy-efficient convolutional neural network (NN) processor.

It mainly improves energy efficiency by making full use of sparsity.

It can automatically switch the rocessor among nine sparsity modes for higher energy efficiency

SYSTEM ARCHITECTURE OVERVIEW

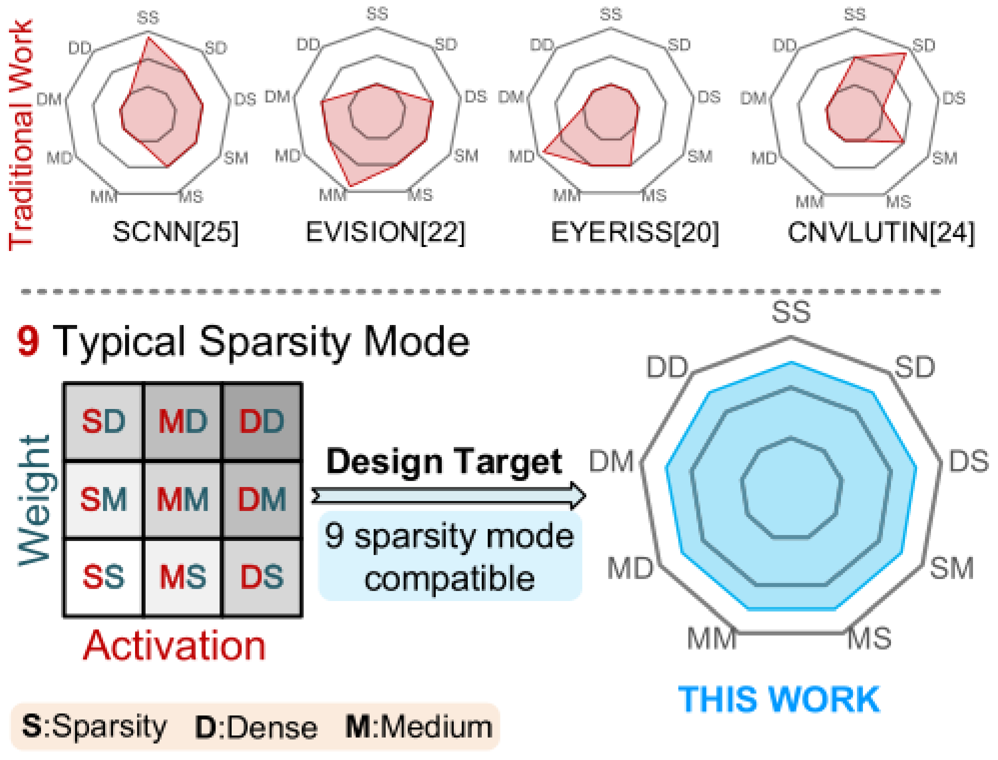
Motivation

One key problem is that the **sparsity of CNN has a wide distribution** even after pruning.

Therefore,hardware accelerators need to have sufficient flexibility.

Optimization of previous works:

1. PEs are **powered off** when processing zero
2. Networks are stored in a **sparse format**

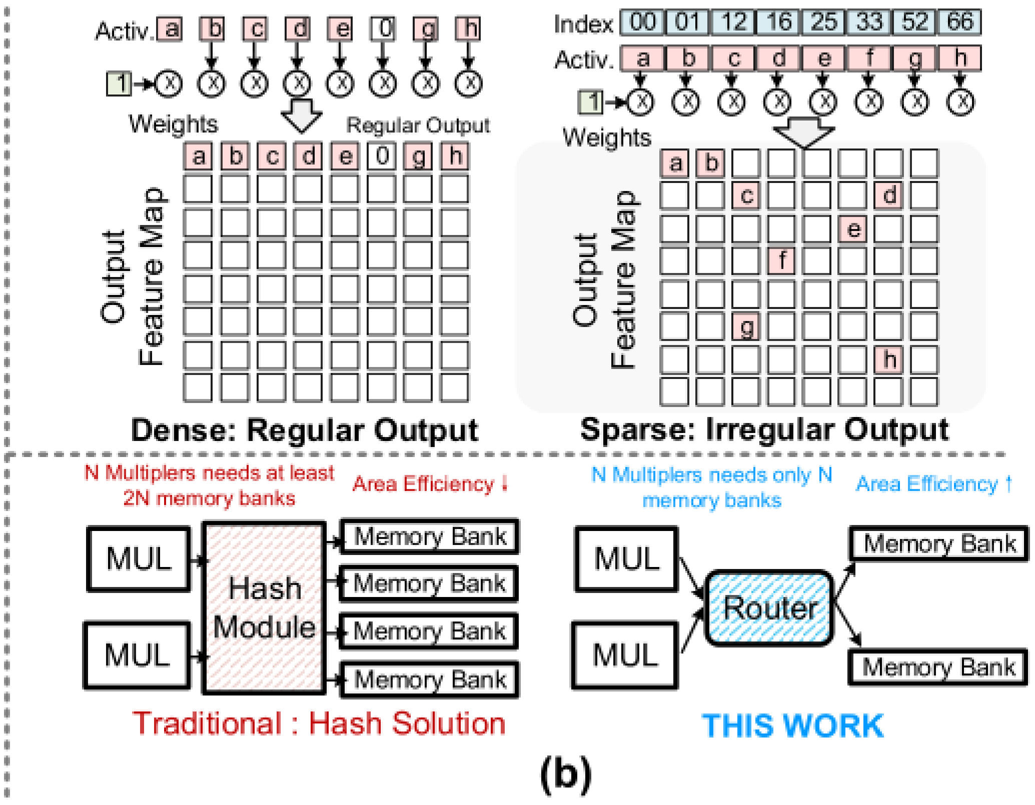


(a)

The **second key problem** is the **irregularity of the output data** when both activations and weights are in sparse mode.

Standard SRAM macros used in silicon chips only have one or two ports. Therefore, these irregular data need to be written into different memory banks

Therefore, the second motivation of this article is to design a simpler PE architecture to process irregular
output data of sparse convolutions efficiently.



High-Level Architecture Overview

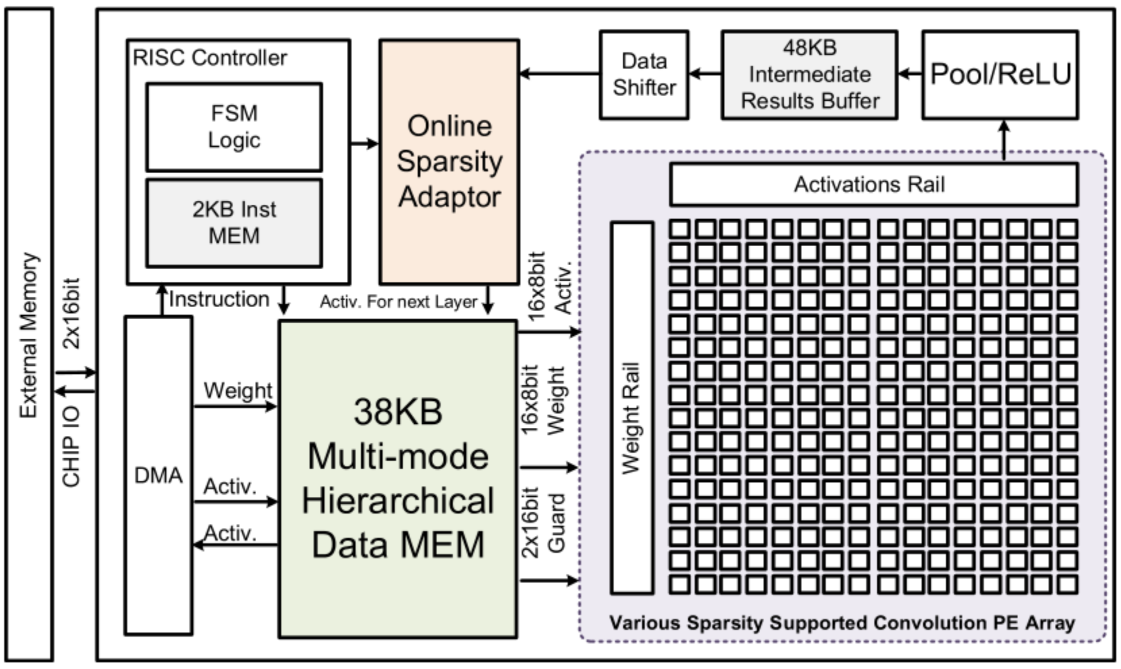


Fig. 4. High-level architecture overview of STICKER. It contains an online sparsity adaptor, a multi-mode hierarchical data memory, and a various sparsity-supported convolution PE Arrays.

store data in 8 bit.

use 24 bit to accumulate the temporary results. The data shifter is designed to cut data from 24 to 8 bit based on the quantization scale of each layer.

Three key innovative modules:

1. **online sparsity adaptor** lets STICKER treat different sparsity situations in different modes
2. **multi-mode data memory** is configurable to store activations/weights with different sparsity modes in the most efficient format.
3. **arious sparsity-supported convolution PE array**

MULTI-SPARSITY CONTROL AND DATA FLOW

Sparsity Adaptive Control and Data Flow

get activations mode (AM) from the sparsity adaptor and the weights mode (WM) from the offline analysis.

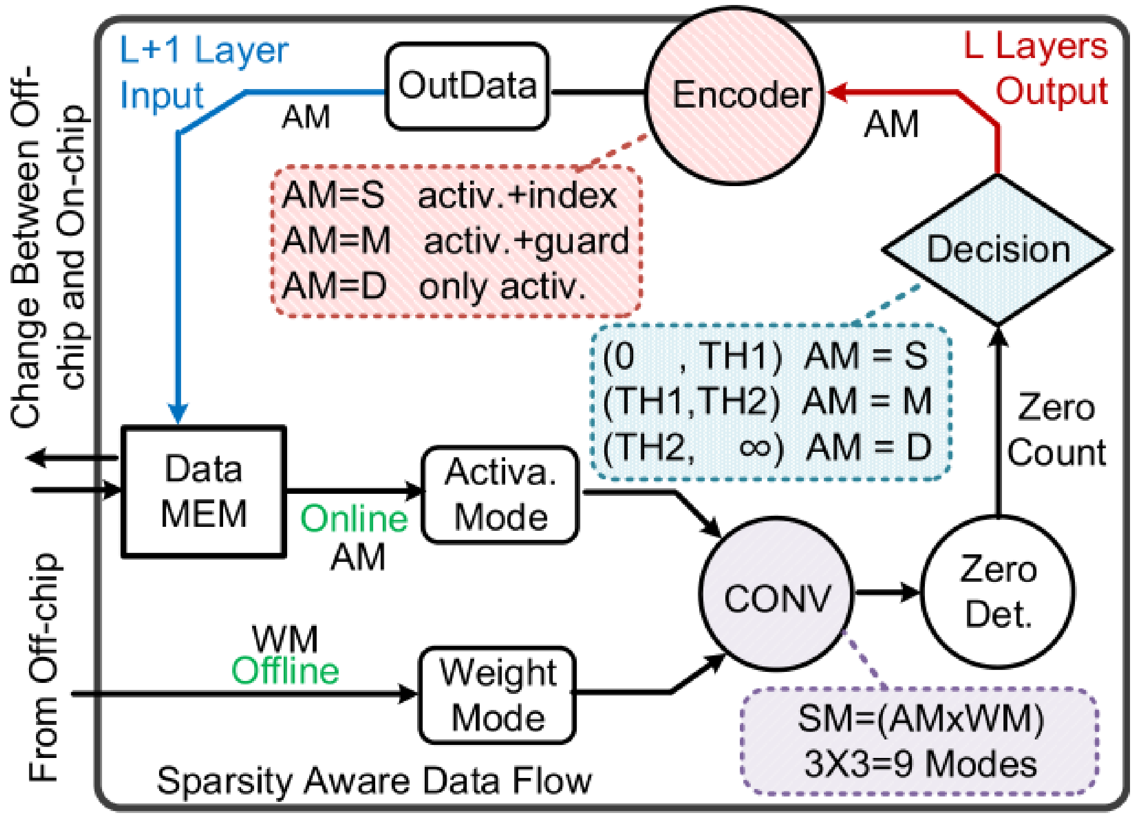


Fig. 5. Multi-sparsity control and data flow changes the chip in nine different modes. Modes of activations come from sparsity-aware adaptor, while the modes of weights come from off-chip analysis.

1. In the sparse mode, only nonzero activations and their indexes are stored.
2. In the medium mode, activations and their zero guard data are stored to power off PEs to save energy.
3. In the dense mode, only activations are stored.

the sparsity adaptor contains four main parts: a sparsity detector, a 4-KB adapter buffer, an adaptive encoder, and an SAC.

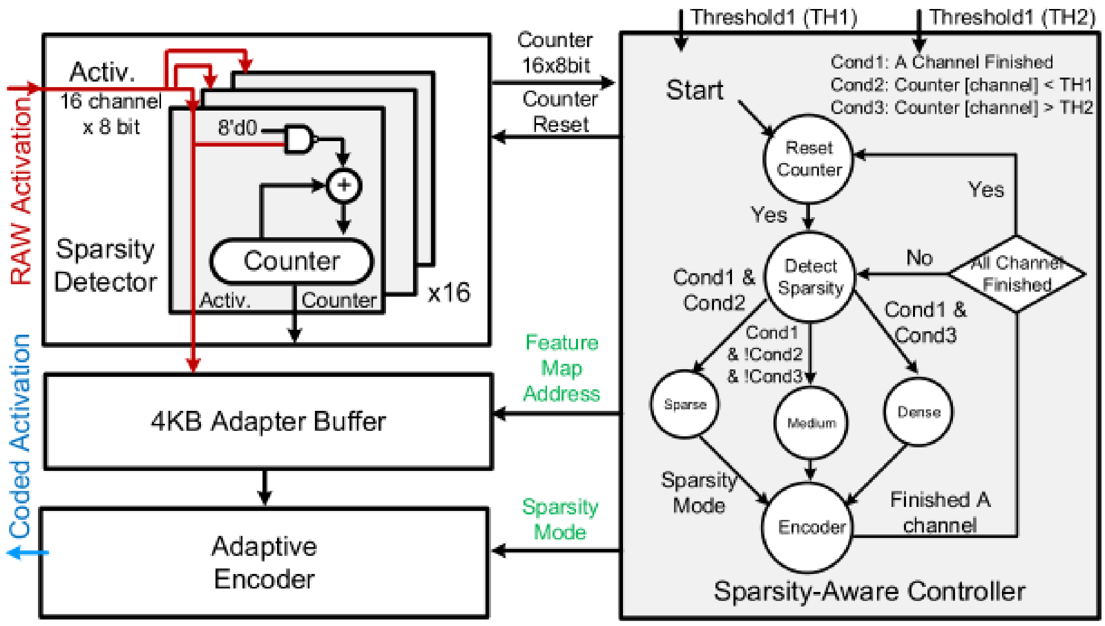


Fig. 6. Online sparsity adaptor is used for detecting activations sparsity, changing chip state, and encoding activations in different modes. It contains a sparsity detector, a 4-KB adaptor buffer, an adaptive encoder, and an SAC.

1. The SAC first resets the sparsity detector.
2. After that, it starts the detector to count zero numbers in the activations.
3. After 16 channel feature maps are written into the adapter buffer, it makes a decision based on two thresholds (TH1 and TH2) and chooses one sparsity mode for the feature maps in the adapter buffer.
4. Then, it gives this sparsity mode to the adaptive encoder.
5. After data in all 16 channels are encoded, it resets the detector and starts a new loop of this flow.

Storage Efficient Instruction Set

this article adopts a 32-bit storage efficient instruction set.

two types of instructions are designed: dynamic instructions and static instructions.