# Efficient and Effective Sparse LSTM on FPGA with Bank-Balanced Sparsity

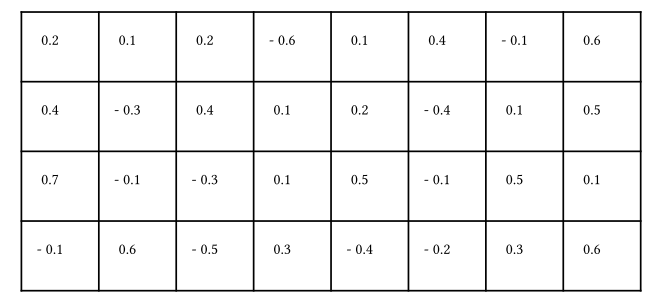
——FPGA’2019 Harbin Institute of Technology

## Contribution

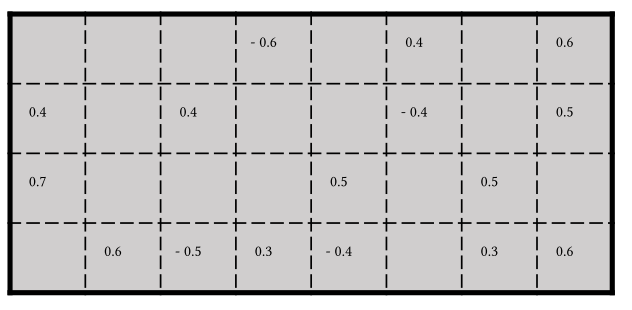
1. Bank-Balanced Sparsity(BBS): a novel sparsity pattern that can maintain model accuracy at a high sparsity level while still enable an efficient FPGA implementation.
2. A 3-step software-hardware cooptimization approach to apply BBS in real FPGA hardware.
3. Design an FPGA accelerator that takes advantage of BBS to eliminate irregular computation and memory accesses.

## Bank-Balanced Sparsity(BBS)

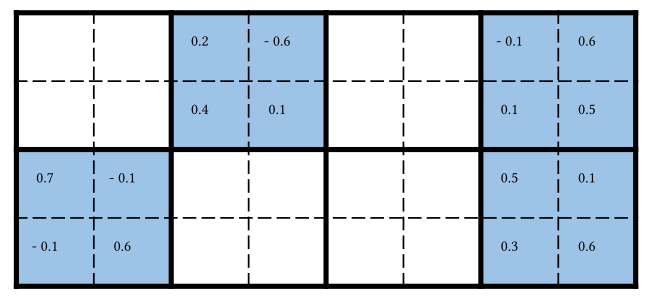
1. Each matrix row is split into multiple equal-sized banks, and each bank has the same number of non-zero values.



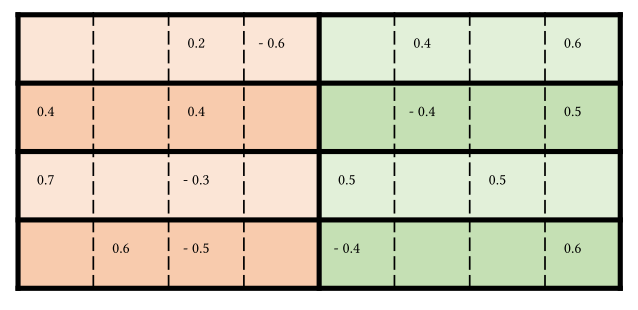
Dense matrix



Unstructured sparse matrix

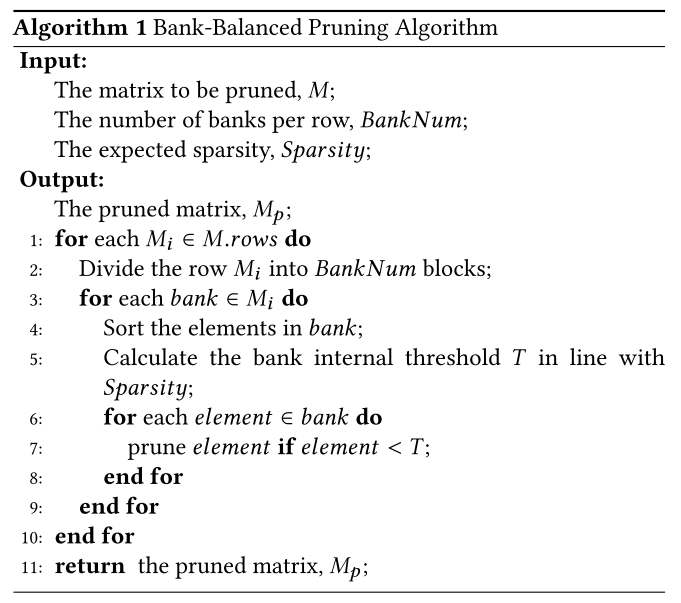


Block sparse matrix



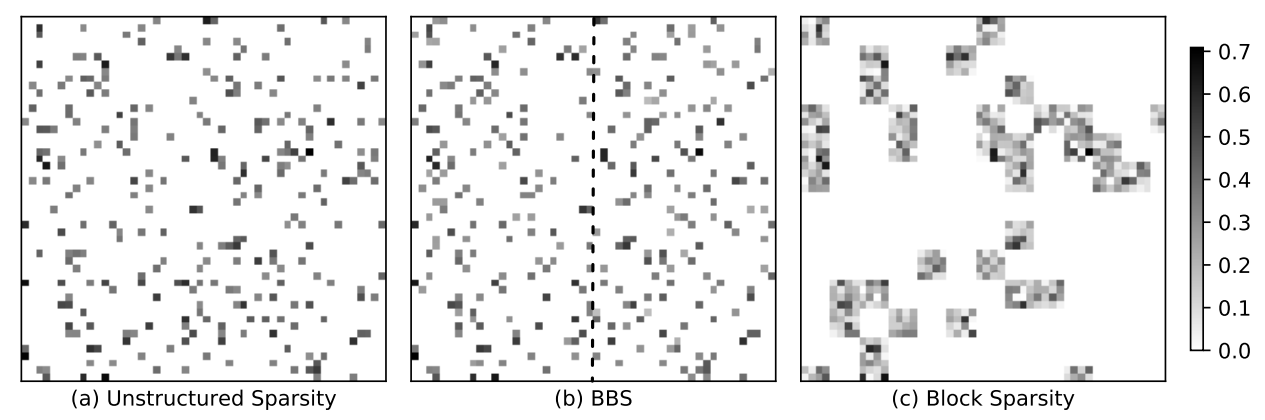
Bank-Balanced sparse matrix

1. Prune method:



adopt fine-grained pruning inside banks independently, so large weights inside each bank can be preserved.

1. Results:

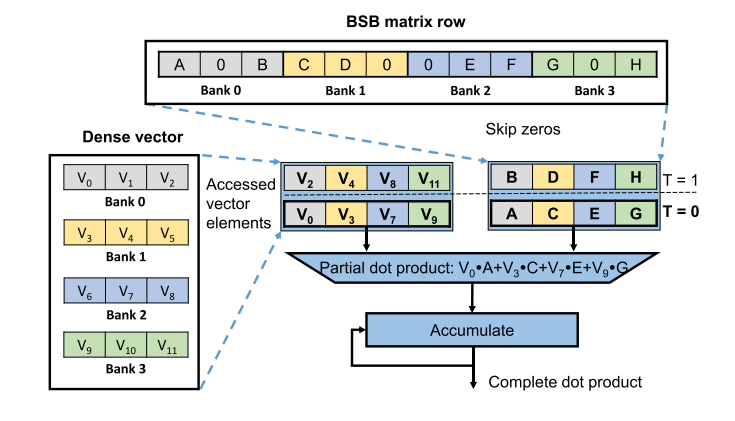


Weights visualization

BBS has almost the same effectiveness as unstructured sparsity and outperforms block sparsity, preserves more important waights than Block sparse method.

## Sparse matrix computation and format for BBS

1. Intra-row parallelism



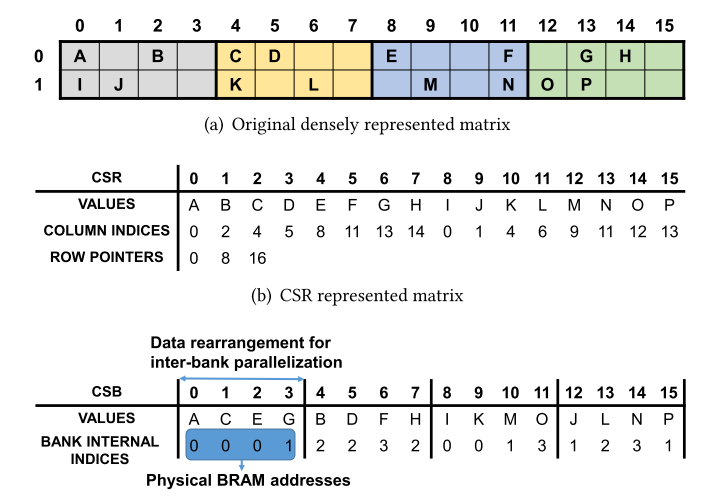
Exploiting inter-bank parallelism in dot product

The multiplications for the non-zero elements inside each bank are performed serially, while the multiplications in different banks are performed in parallel.

In BBS matrices, every row (and every bank) has the same number of elements which automatically guarantees the load balance across rows and banks.

Store each vector bank in an independently accessible block RAM can supply vector elements simultaneously with high bandwidth and without memory access conflicts.

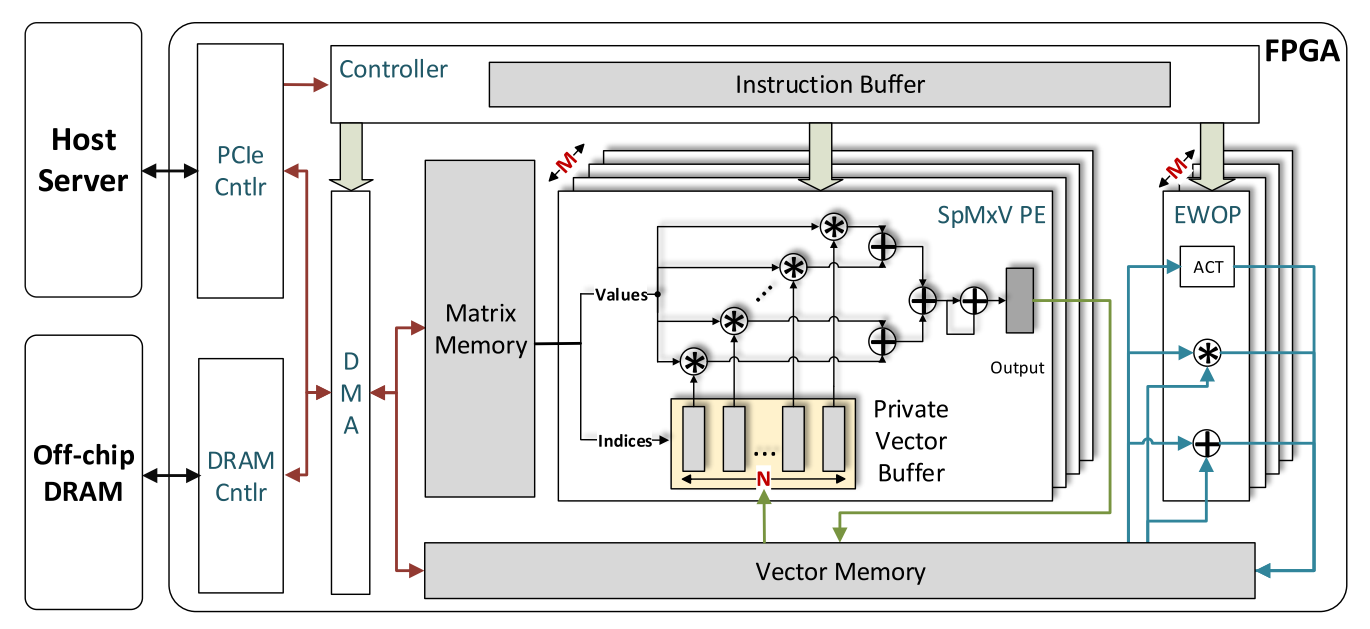
1. Decoding-free sparse matrix format



The comparison between CSR and CSB

## LSTM ACCELERATOR

1. Overall architecture



The BBS accelerator is implemented as an accelerator on the PCIe I/O bus to serve LSTM inference requests from the host server.

The controller receives and stores instructions from the host server in the instruction buffer and dispatches them to their corresponding modules to execute.

1. Sparse matrix-vector multiplication unit

The SpMxV unit consists of M parallel processing elements (PEs) that compute dot products of distinct matrix rows in order to realize inter-row parallelism.

Each PE contains a private vector buffer (PVB) to buffer the dense vector being multiplied, because vector elements are randomly accessed multiple times for all matrix rows in SpMxV.

Every PE contains N multipliers.

In order to support random vector accesses at a high bandwidth

without replicas inside a PE, we adopt the banking approach to buffer vectors

1. Results:

