

Hardware Accelerators for Sparse Deep Learning

A Dissertation Presented to
The Academic Faculty

By

Marcus Johnson

In Partial Fulfillment
of the Requirements for the Degree Doctor of Philosophy in
Electrical and Computer Engineering
School of Electrical and Computer Engineering

Georgia Institute of Technology

May 2026

Copyright © Marcus Johnson 2026

Unofficial template, not affiliated with or endorsed by the Georgia Institute of Technology. The author, committee and results in this sample are fictional. Delete this line before you submit.

Hardware Accelerators for Sparse Deep Learning

Thesis committee:

Dr. Priya Raman, Advisor
School of Electrical and Computer
Engineering
Georgia Institute of Technology

Dr. Thomas A. Weller
School of Computer Science
Georgia Institute of Technology

Dr. Hana Kobayashi
School of Electrical and Computer
Engineering
Georgia Institute of Technology

Dr. Ioana Petrescu
School of Electrical and Computer
Engineering
Georgia Institute of Technology

Dr. Victor Lindqvist
Department of Computer Science
Vertex University

Date Approved: April 14, 2026

Acknowledgments

I thank my advisor, Dr. Priya Raman, for her guidance, and the members of my committee for their time and questions. I am grateful to my labmates for countless whiteboard sessions, and to my family for their patience.

Contents

Acknowledgments	iii
Summary	vii
1 Introduction	1
2 Background	2
3 A Two-Sided Zero-Skipping Processing Element	3
4 Evaluation	4
5 Conclusion	5
A Simulator Configuration	6

List of Tables

4.1 Speedup and energy relative to a dense baseline of equal area. 4

List of Figures

3.1	The processing element matches non-zero coordinates before multiplying. .	3
-----	---	---

Summary

Pruned neural networks keep most of their accuracy with a small fraction of their weights, but dense accelerators cannot turn that sparsity into speed. This dissertation designs hardware that does. It characterizes the sparsity patterns of pruned convolutional and transformer models, proposes a processing element that skips zero operands in both weights and activations, and evaluates the resulting accelerator in a cycle-accurate simulator. At 90% weight sparsity the design runs 3.4 times faster than a dense baseline of equal area and uses 2.7 times less energy per inference.

Chapter 1

Introduction

Deep networks are over-parameterized: pruning can remove most of their weights with little loss of accuracy [3]. On a dense accelerator, however, a multiplication by zero costs as much as any other, so the savings stay on paper [4]. Turning sparsity into performance requires hardware that skips ineffectual work without stalling on the irregular memory accesses that skipping creates.

This dissertation makes three contributions: a characterization of sparsity in pruned models, a zero-skipping processing element for weights and activations together, and an evaluation against dense and single-sided sparse baselines.

Chapter 2

Background

Dataflow accelerators organize an array of processing elements around a reuse pattern for weights, inputs or partial sums [1]. Sparse accelerators store tensors in a compressed format and match non-zero operands before multiplying. EIE targets sparse fully-connected layers [2], while SCNN handles convolutions by computing the Cartesian product of non-zero weights and activations [5]. If d_w and d_a are the densities of weights and activations, the fraction of effectual multiplications is

$$\rho = d_w d_a, \tag{2.1}$$

which bounds the speedup available from skipping.

Chapter 3

A Two-Sided Zero-Skipping Processing Element

Our processing element buffers compressed weight and activation fibres and uses a small intersection unit to emit only coordinate-matched pairs to the multiplier. A load-balancing network redistributes work between neighbouring elements when one runs out of effectual pairs.

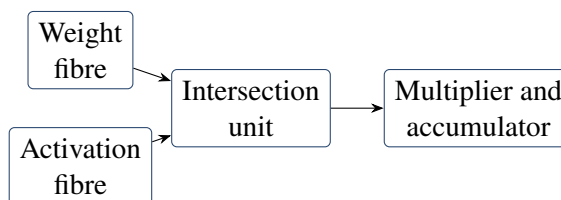


Figure 3.1: The processing element matches non-zero coordinates before multiplying.

Chapter 4

Evaluation

We compared designs of equal silicon area in a cycle-accurate simulator on five pruned models. Table 4.1 reports geometric-mean speedup and energy relative to the dense baseline.

Table 4.1: Speedup and energy relative to a dense baseline of equal area.

Weight sparsity	Speedup	Energy ratio	Utilization
50%	1.6	0.71	0.83
75%	2.4	0.52	0.79
90%	3.4	0.37	0.72

Utilization falls as sparsity rises because matched pairs become scarce, which is where the load-balancing network earns its area.

Chapter 5

Conclusion

Sparsity pays off in hardware only when both operands are skipped and the resulting imbalance is managed. The design studied here does both, and the remaining gap to the ideal speedup in Equation (2.1) comes mostly from memory bandwidth.

Appendix A

Simulator Configuration

The simulator models a 16-by-16 array at 1 GHz with 512 KiB of on-chip buffer and a 64 GB/s off-chip interface. Energy per access was taken from a 16 nm memory compiler.

Bibliography

- [1] Yu-Hsin Chen, Joel Emer, and Vivienne Sze. Eyeriss: A spatial architecture for energy-efficient dataflow for convolutional neural networks. In *Proceedings of the 43rd International Symposium on Computer Architecture*, pages 367–379, 2016.
- [2] Song Han, Xingyu Liu, Huizi Mao, Jing Pu, Ardavan Pedram, Mark A. Horowitz, and William J. Dally. EIE: Efficient inference engine on compressed deep neural network. In *Proceedings of the 43rd International Symposium on Computer Architecture*, pages 243–254, 2016.
- [3] Song Han, Huizi Mao, and William J. Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. In *International Conference on Learning Representations*, 2016.
- [4] Norman P. Jouppi et al. In-datacenter performance analysis of a tensor processing unit. In *Proceedings of the 44th Annual International Symposium on Computer Architecture*, pages 1–12, 2017.
- [5] Angshuman Parashar et al. SCNN: An accelerator for compressed-sparse convolutional neural networks. In *Proceedings of the 44th Annual International Symposium on Computer Architecture*, pages 27–40, 2017.