Stitch-X: An Accelerator Architecture for Exploiting Unstructured Sparsity in Deep Neural Networks

Ching-En Lee[†] Yakun Sophia Shao[‡] Jie-Fang Zhang[†]
Angshuman Parashar[‡] Joel Emer[‡◇] Stephen W. Keckler[‡] Zhengya Zhang[†]

University of Michigan[†] NVIDIA[‡] Massachusetts Institute of Technology[◇]

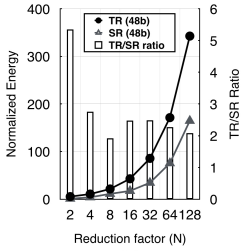
ABSTRACT

Sparse deep neural network (DNN) accelerators exploit the intrinsic redundancy in data representation to achieve high performance and energy efficiency. However, sparse weight and input activation arrays are unstructured, and their processing cannot take advantage of the regular data-access patterns offered by dense arrays, thus the processing incurs increased complexities in dataflow orchestration and resource management. In this work, we first present the importance of the data reduction mechanism, *i.e.*, how partial sums are accumulated *spatially* or *temporally*, a perspective that has not been fully explored in current literature. Motivated by the reduction analysis, we propose Stitch-X, a novel DNN inference accelerator architecture that can *stitch* together sparse weights and input activations for parallel execution. Specifically, Stitch-X employs a novel dataflow that leverages both spatial and temporal reduction to balance energy efficiency and dataflow control complexity. Moreover, Stitch-X adopts a new runtime Parallelism Discovery Unit (PDU) to efficiently extract fine-grained parallelizable operations from irregular sparse data arrays to enable a higher performance over a wide range of input data densities and for a variety of DNN layers. Our evaluations show that Stitch-X consistently achieves a 3.8× speedup and improves energy-delay-squared-product (ED^2P) by a factor of 10.3× over an efficient, dense DNN accelerator. Compared to a state-of-the-art sparse DNN accelerator, Stitch-X delivers 1.6× better performance. A silicon prototype of the Stitch-X architecture is scheduled for 2018.

1 INTRODUCTION

Deep learning [1] has emerged to be a key approach to solving complex cognition and learning problems. Deep neural networks (DNNs) in particular have become pervasive due to their successes across a variety of applications, including image recognition [2–6], object detection [7, 8], semantic segmentation [9–11], language translation [12], audio synthesis [13] and autonomous driving [14]. State-of-the-art DNNs [2–11] require up to billions of operations and hundreds of megabytes to store activations and weights, presenting substantial workloads and memory traffic.

The core computation behind DNN processing is the dot-product of input activations and weights. To obtain substantial performance and energy efficiency gains [15–20], many prior works, from research prototypes to industrial products, proposed specialized hardware to accelerate DNN processing [21–34]. Algorithmic techniques such as quantization and pruning [35, 36] can be used to sparsify data, which could be exploited to further enhance the performance of efficiency DNN processing [37–40]. A dot-product involves computing products followed by partial-sum reduction. To better understand the design complexities associated with partial-sum reduction



(a)

Dataflow Taxonomy		Reduction Mechanism: *Spatial*	*Temporal*	*Hybrid*
Data Reuse	*Output Stationary*		ShiDianNao [26] DnnWeaver [29]	HR
	Input (IA/W) Stationary	NVDLA [23] BrainWave [22]	NeuFlow [30] **SCNN [41]** **EIE [38]**	HR **(Stitch-X)**
	No Local Reuse	DianNao [24] DaDianNao [25] **Cambricon-X [39]** **Cnvlutin [40]**	TPU [21] Minerva [31]	
	Row Stationary		Eyeriss [27]	

(b)

Figure 1: (a) SR versus TR energy. (b) Taxonomy of DNN accelerators based on data reuse and reduction mechanism. Non-existent dataflows are grayed out, and sparse accelerators are annotated using bold font.

for sparse data, we analyze the reduction mechanisms in existing DNN accelerator designs. Despite being overlooked in the well-known dataflow analysis [27], the reduction mechanism plays a key role in constructing efficient dataflows and can lead to more than 3× difference in the overall energy efficiency. Motivated by this insight, we propose Stitch-X, a sparse DNN accelerator architecture that leverages a hybrid, spatial-temporal reduction to balance energy efficiency and dataflow control complexity. To maximize the degree of parallelism for reduction, Stitch-X utilizes a novel and scalable Parallelism Discovery Unit (PDU) that dynamically stitches together input activation and weight pairs to produce readily reducible partial sums.

We evaluate Stitch-X over a suite of modern DNNs [2–4, 6] that have been pruned by methods demonstrated in [35, 36] with no accuracy loss when tested using ImageNet [41]. Stitch-X achieves up to 4.3× speedup (3.8× on average) and improves energy-delay-squared-product (ED^2P) by up to 13.2× (10.3× on average) over a state-of-the-art dense DNN accelerator. To compare sparse DNN accelerators, we use the metric of Proximity to Oracle Speedup (PTOS), *i.e.*, the achieved speedup benchmarked as a percentage of the oracle speedup[1]. Stitch-X achieves a PTOS of 77.4%, surpassing the state-of-the-art sparse DNN accelerator [40] by 1.6×.

2 DNN REDUCTION MECHANISM

We present a new perspective in DNN dataflow analysis: the method in which partial sums are accumulated, *i.e.*, reduced.

Spatial Reduction (SR) refers to spatial partial-sum accumulation without explicit storage during the reduction process. Given

[1]Oracle performance is defined as the total number of effectual multiplications, *i.e.*, both operands are non-zero, divided by the number of available multipliers.

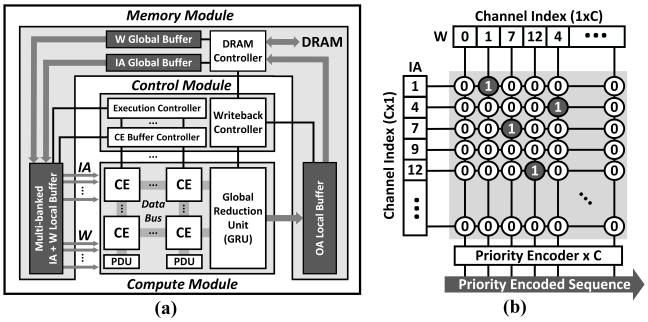


Figure 2: (a) Stitch-X microarchitecture diagram. (b) Parallelism Discovery Unit (PDU) hardware structure.

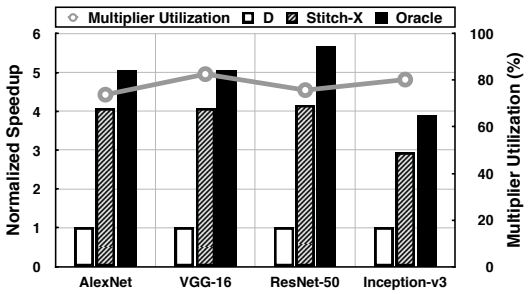


Figure 3: Overall performance of Stitch-X running modern DNNs: AlexNet, VGG-16, ResNet-50, and Inception-v3.

N partial sums, SR is realized using an $N:1$ *adder tree* to produce an output in *one* time step (a single clock cycle). DianNao [24] and NVDLA [23] are examples that adopted the SR approach.

Temporal Reduction (TR) refers to partial-sum accumulation over time by using a single adder to accumulate one partial sum per time step over N steps. TR's advantage is that there is less control dependence for partial-sum accumulation, since only one element is being accumulated at a time, instead of N in SR. However, from an energy perspective, TR incurs a register read and write cost each cycle, which can be significant especially when the register file is large. Examples of TR are found in TPU [21] and ShiDianNao [26].

Figure 1(a) demonstrates the energy of an $N:1$ reduction (normalized to a two-operand add) by sweeping the number of input operands (N) for the two types of reductions. The reduction factor is defined as the maximum number of input operands that can be reduced in a time step. The reduction factors for SR and TR are N and 1, respectively. The comparison between SR and TR leads to two conclusions. First, SR is *always* more energy-efficient than TR due to TRâĂŹs extra register access energy in every accumulation step. Second, the energy benefit of SR over TR (captured in the TR/SR ratio) decreases with increasing reduction factors. The ratio is the largest at small reduction factors (when $N = 2$ or 4) and decreases to about two at $N = 128$.

Figure 1(b) summarizes the data reuse and reduction mechanisms of existing DNN accelerators. Our proposed Stitch-X architecture adopts a hybrid reduction (HR) mechanism to combine the efficiency of SR with the flexibility of TR to handle the irregularity in the data access patterns of sparse DNNs. Stitch-X employs a PDU to facilitate SR to enable higher overall performance and efficiency.

3 THE STITCH-X ACCELERATOR

Figure 2(a) shows the top-level block diagram of the Stitch-X architecture including compute, control, and memory modules.

Compute module consists of an array of 8×3 compute elements (CEs), PDUs, and a Global Reduction Unit (GRU). Each CE contains three multipliers. At the local CE level, Stitch-X applies HR; and at the global level, Stitch-X applies SR across CEs along the diagonal, vertical, and horizontal directions with the help of the GRU to support different types of DNN layers.

Control module contains execution, CE buffer, and writeback controllers. The execution controller coordinates input operand streaming from the memory module to the compute module and the OA local buffer writeback to DRAM. The CE buffer controller fetches and delivers operands for computation. The writeback controller determines the writeback address upon completion of a reduction.

Memory module includes on-chip SRAM that serves as buffers for storing input activations (IA), weights (W), and output activations (OA), as well as a DRAM controller.

Figure 2(b) shows a scalable PDU design that performs parallel search on IA and W arrays to find reducible IA and W pairs. The PDU operates on the channel index vectors of IA and W of size C at the same time. The PDU uses a $C \times C$ comparator matrix to search for matching IA and W channel indices in *parallel*. The comparator matrix produces a binary output at each junction, 1 for match and 0 for mismatch. Priority encoding is applied to each column of comparison outputs to obtain an ordered index sequence. Finally, iterative leading-one detection is used to locate the addresses for fetching W operands from the W register file (RF). A similar approach is used to locate the addresses for fetching IA operands from the IA RF. To target a higher degree of data parallelism, the decode width can be increased accordingly.

4 EVALUATION

A prototype Stitch-X accelerator architecture was synthesized at a 1.0 GHz clock frequency in a commercial 40 nm CMOS technology. The core area is 2.7 mm^2 and the overhead of having PDU and decoder to stitch input operands accounts for less than 8% of the area, significantly lower than the decoding overhead reported in the previous sparse DNN accelerator [38].

Figure 3 shows the overall performance of Stitch-X running a range of modern DNN workloads, including AlexNet [2], VGG-16 [3], ResNet-50 [4], and Inception-v3 [6]. For each network, we evaluated the speedup and CE utilization of Stitch-X over an input-stationary SR baseline DNN accelerator, denoted by D. Across all the networks evaluated, Stitch-X achieves a 3.8× speedup over the dense DNN accelerator while maintaining an average CE utilization of 74%. Compared to the oracle, Stitch-X achieves a PTOS of 77%, 1.6× better than the state-of-the-art sparse accelerator that operates on both sparse W and IA operands [40]. In addition, Stitch-X also achieves at least 70% CE utilization when processing fully-connected layers and 1×1 convolution layers, both of which are challenging to accelerate with existing DNN accelerator designs.

REFERENCES

[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, pp. 1097–1105, 2012.

[3] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[4] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[5] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9, 2015.

[6] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2818–2826, 2016.

[7] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 580–587, 2014.

[8] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 779–788, 2016.

[9] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3431–3440, 2015.

[10] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *arXiv preprint arXiv:1511.00561*, 2015.

[11] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *arXiv preprint arXiv:1606.00915*, 2016.

[12] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, pp. 3104–3112, 2014.

[13] A. v. d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio," *arXiv preprint arXiv:1609.03499*, 2016.

[14] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, *et al.*, "End to end learning for self-driving cars," *arXiv preprint arXiv:1604.07316*, 2016.

[15] R. Hameed, W. Qadeer, M. Wachs, O. Azizi, A. Solomatnikov, B. C. Lee, S. Richardson, C. Kozyrakis, and M. Horowitz, "Understanding sources of inefficiency in general-purpose chips," in *ACM SIGARCH Computer Architecture News*, vol. 38, pp. 37–47, ACM, 2010.

[16] E. S. Chung, P. A. Milder, J. C. Hoe, and K. Mai, "Single-chip heterogeneous computing: Does the future include custom logic, fpgas, and gpgpus?," in *Proceedings of the 2010 43rd Annual IEEE/ACM International Symposium on Microarchitecture*, IEEE Computer Society, 2010.

[17] G. Venkatesh, J. Sampson, N. Goulding, S. Garcia, V. Bryksin, J. Lugo-Martinez, S. Swanson, and M. B. Taylor, "Conservation cores: reducing the energy of mature computations," in *ACM SIGARCH Computer Architecture News*, 2010.

[18] I. Magaki, M. Khazraee, L. V. Gutierrez, and M. B. Taylor, "Asic clouds: specializing the datacenter," in *Proceedings of the 43rd International Symposium on Computer Architecture*, pp. 178–190, IEEE Press, 2016.

[19] L. Wu, A. Lottarini, T. K. Paine, M. A. Kim, and K. A. Ross, "Q100: The architecture and design of a database processing unit," in *Proceedings of the 19th International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS '14, 2014.

[20] S. Kumar, N. Vedula, A. Shriraman, and V. Srinivasan, "Dasx: Hardware accelerator for software data structures," in *Proceedings of the 29th ACM on International Conference on Supercomputing*, ICS '15, 2015.

[21] N. P. Jouppi, C. Young, N. Patil, D. Patterson, G. Agrawal, R. Bajwa, S. Bates, S. Bhatia, N. Boden, A. Borchers, R. Boyle, P. luc Cantin, C. Chao, C. Clark, J. Coriell, M. Daley, M. Dau, J. Dean, B. Gelb, T. V. Ghaemmaghami, R. Gottipati, W. Gulland, R. Hagmann, C. R. Ho, D. Hogberg, J. Hu, R. Hundt, D. Hurt, J. Ibarz, A. Jaffey, A. Jaworski, A. Kaplan, H. Khaitan, A. Koch, N. Kumar, S. Lacy, J. Laudon, J. Law, D. Le, C. Leary, Z. Liu, K. Lucke, A. Lundin, G. MacKean, A. Maggiore, M. Mahony, K. Miller, R. Nagarajan, R. Narayanaswami, R. Ni, K. Nix, T. Norrie, M. Omernick, N. Penukonda, A. Phelps, and J. Ross, "In-datacenter performance analysis of a tensor processing unit," 2017.

[22] E. Chung, J. Fowers, K. Ovtcharov, M. Papamichael, A. Caulfield, T. Massengil, M. Liu, D. Lo, S. Alkalay, M. Haselman, C. Boehn, O. Firestein, A. Forin, K. S. Gatlin, M. Ghandi, S. Keil, K. Holohan, T. juhasz, R. K. Kovvuri, S. Lanka, F. van Megen, D. Mukhortov, P. Patel, S. Reinhardt, A. Sapek, R. Seera, B. Sridharan, L. Woods, P. Yi-Xiao, R. Zhao, and D. Burger, "Accelerating persistentneural networks at datacenter scale," in *HotChips*, 2017.

[23] NVIDIA, "Nvidia deep learning accelerator (nvdla)." https://github.com/nvdla/, 2017.

[24] T. Chen, Z. Du, N. Sun, J. Wang, C. Wu, Y. Chen, and O. Temam, "Diannao: A small-footprint high-throughput accelerator for ubiquitous machine-learning," in *ACM Sigplan Notices*, vol. 49, pp. 269–284, ACM, 2014.

[25] Y. Chen, T. Luo, S. Liu, S. Zhang, L. He, J. Wang, L. Li, T. Chen, Z. Xu, N. Sun, *et al.*, "Dadiannao: A machine-learning supercomputer," in *Proceedings of the 47th Annual IEEE/ACM International Symposium on Microarchitecture*, pp. 609–622, IEEE Computer Society, 2014.

[26] Z. Du, R. Fasthuber, T. Chen, P. Ienne, L. Li, T. Luo, X. Feng, Y. Chen, and O. Temam, "Shidiannao: Shifting vision processing closer to the sensor," in *ACM SIGARCH Computer Architecture News*, vol. 43, pp. 92–104, ACM, 2015.

[27] Y.-H. Chen, J. Emer, and V. Sze, "Eyeriss: A spatial architecture for energy-efficient dataflow for convolutional neural networks," in *Computer Architecture (ISCA), 2016 ACM/IEEE 43rd Annual International Symposium on*, pp. 367–379, IEEE, 2016.

[28] H. Esmaeilzadeh, A. Sampson, L. Ceze, and D. Burger, "Neural acceleration for general-purpose approximate programs," in *Proceedings of the 2012 45th Annual IEEE/ACM International Symposium on Microarchitecture*, MICRO-45, 2012.

[29] H. Sharma, J. Park, D. Mahajan, E. Amaro, J. K. Kim, C. Shao, A. Mishra, and H. Esmaeilzadeh, "From high-level deep neural models to fpgas," in *Microarchitecture (MICRO), 2016 49th Annual IEEE/ACM International Symposium on*, pp. 1–12, IEEE, 2016.

[30] C. Farabet, B. Martini, B. Corda, P. Akselrod, E. Culurciello, and Y. LeCun, "Neuflow: A runtime reconfigurable dataflow processor for vision," in *Computer Vision and Pattern Recognition Workshops (CVPRW), 2011 IEEE Computer Society Conference on*, pp. 109–116, IEEE, 2011.

[31] B. Reagen, P. Whatmough, R. Adolf, S. Rama, H. Lee, S. K. Lee, J. M. Hernández-Lobato, G.-Y. Wei, and D. Brooks, "Minerva: Enabling low-power, highly-accurate deep neural network accelerators," in *Proceedings of the 43rd International Symposium on Computer Architecture*, pp. 267–278, IEEE Press, 2016.

[32] C. Zhang, P. Li, G. Sun, Y. Guan, B. Xiao, and J. Cong, "Optimizing fpga-based accelerator design for deep convolutional neural networks," in *Proceedings of the 2015 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, pp. 161–170, ACM, 2015.

[33] Y. Shen, M. Ferdman, and P. Milder, "Escher: A cnn accelerator with flexible buffering to minimize off-chip transfer," in *Proceedings of the 25th IEEE International Symposium on Field-Programmable Custom Computing Machines (FCCM). IEEE Computer Society, Los Alamitos, CA, USA*, 2017.

[34] M. Alwani, H. Chen, M. Ferdman, and P. Milder, "Fused-layer cnn accelerators," in *Microarchitecture (MICRO), 2016 49th Annual IEEE/ACM International Symposium on*, pp. 1–12, IEEE, 2016.

[35] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," *arXiv preprint arXiv:1510.00149*, 2015.

[36] S. Han, J. Pool, J. Tran, and W. Dally, "Learning both weights and connections for efficient neural network," in *Advances in Neural Information Processing Systems*, pp. 1135–1143, 2015.

[37] S. Han, X. Liu, H. Mao, J. Pu, A. Pedram, M. A. Horowitz, and W. J. Dally, "Eie: efficient inference engine on compressed deep neural network," in *Proceedings of the 43rd International Symposium on Computer Architecture*, pp. 243–254, IEEE Press, 2016.

[38] S. Zhang, Z. Du, L. Zhang, H. Lan, S. Liu, L. Li, Q. Guo, T. Chen, and Y. Chen, "Cambricon-x: An accelerator for sparse neural networks," in *Microarchitecture (MICRO), 2016 49th Annual IEEE/ACM International Symposium on*, pp. 1–12, IEEE, 2016.

[39] J. Albericio, P. Judd, T. Hetherington, T. Aamodt, N. E. Jerger, and A. Moshovos, "Cnvlutin: ineffectual-neuron-free deep neural network computing," in *Computer Architecture (ISCA), 2016 ACM/IEEE 43rd Annual International Symposium on*, pp. 1–13, IEEE, 2016.

[40] A. Parashar, M. Rhu, A. Mukkara, A. Puglielli, R. Venkatesan, B. Khailany, J. Emer, S. W. Keckler, and W. J. Dally, "Scnn: An accelerator for compressed-sparse convolutional neural networks," in *Proceedings of the 44th Annual International Symposium on Computer Architecture*, pp. 27–40, ACM, 2017.

[41] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pp. 248–255, IEEE, 2009.