A Differentiable Neural Computer for Logic Reasoning with Scalable Near-Memory Computing and Sparsity Based Enhancement

Yuhao Ju
Electrical and Computer Engineering
Northwestern University
Evanston, IL, United States
Yuhaoju2017@u.northwestern.edu

Shiyu Guo
Electrical and Computer Engineering
Northwestern University
Evanston, IL, United States
ShiyuGuo2021@u.northwestern.edu

Zixuan Liu
Electrical and Computer Engineering
Northwestern University
Evanston, IL, United States
ZixuanLiu2021@u.northwestern.edu

Tianyu Jia
School of Integrated Circuits
Peking University
Beijing, China
tianyuj@pku.edu.cn

Jie Gu
Electrical and Computer Engineering
Northwestern University
Evanston, IL, United States
jgu@northwestern.edu

***Abstract*—Logic reasoning represents a new class of artificial intelligence. This work presents the first hardware implementation of the Differentiable Neural Computer accelerator based on brain inspired "working memory" concept for reasoning tasks. A special near-memory computing architecture is developed achieving high scalability and over 90% utilization of computing resources. Sparsity based enhancements such as zero skipping and data compression are applied with 30% speedup of the computing latency. A 65nm test chip was fabricated with demonstrations on a variety of logic reasoning tasks showing 700X and 46X speedup compared with CPU and GPU and up to 1.28TOPS/W energy efficiency.**

I. Introduction

Despite the recent success in image and voice recognition applications, a missing capability from the current deep learning based artificial intelligence (AI) is realizing human like logic reasoning. Fig. 1 shows several common cognitive reasoning tasks such as deductive/abstract/sequential reasoning, algorithm deduction, graphic traverse, etc. where sequential relationships are being inferred from context of graphs or texts. While exhaustive or sophisticated heuristic search algorithms are traditionally used to solve such problems, applying deep neural networks (DNN) to reasoning tasks allows a differentiable solution, e.g. learning through back-propagation without human intervention. However, existing convolutional neural networks (CNN) or long short-term memory (LSTM) architectures suffer from limited memory space due to the entanglement of computing and memory elements leading to poor performance in long sequential reasoning tasks. Recently, models of differentiable neural computer (DNC) or Memory-augmented Neural Networks (MANN) were developed for reasoning tasks [1-2]. As shown in Fig. 1, DNC incorporates content memory operations through special "read/write heads" to infer logical information from content memory contents overcoming limited memory space issues of CNN or LSTM. Such a capability resembles human brain's "working memory" which uses an "attention" based controller to access vocal or visual memory of the brain [3]. This work implemented an end-to-end logical inference processor based on DNC algorithm with offline trained models [4]. As highlighted in Fig. 1, the challenges of ASIC acceleration of DNC include (1) large amount of memory access from the attention mechanism with 10.6X more memory request than conventional CNN, (2) highly sparse input and memory contents and (3) complex model with eight operating phases making the ASIC acceleration very challenging. In this work, for the first time, an ASIC logic reasoning processor was designed to accelerate cognitive reasoning tasks with 700X/46X improvement over commercial CPU/GPU. The contributions include (1) A scalable near-memory architecture is developed to overcome the memory bandwidth challenges of the algorithm; (2) Special input zero skipping and data compression techniques are applied to exploit sparsity of the data; (3) Efficient transpose multiplication is introduced to avoid large data exchange among computing tiles; (4) Reconfigurable multiplier-accumulator units (MAC) are designed to support the eight operating phases with above 90% processing element (PE) utilization rate for the challenging mapping of the software model.

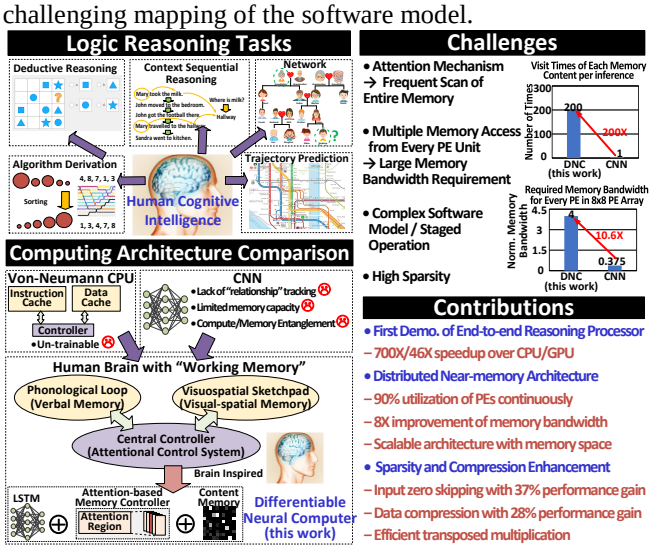


Fig. 1. Logic reasoning tasks with different computing architectures and main contributions of this work.

II. Architecture and Algorithm

Fig. 2 shows the top-level DNC algorithm. A LSTM serves as a central controller which preprocesses sequential input data and manages the access of various memory banks through memory controllers. The memory controllers include "write head" and "read head" which realize an "attention"

This work is supported in part by NSF grant CCF-2008906.

mechanism to select a region of "focus" from the large content memory, similar to human brain's memory retrieval mechanism. For "read head", the "attention" includes cosine similarity and matrix/dot operations where the entire contents of content memory are scanned for "related" information. A special "attention memory" is used to keep the "linkage" information, i.e. logical/sequential relationship among the contents of content memory. For "write head", a usage memory is added to keep track of the content memory usage for efficient recall, e.g. allocation of new memory for incoming information. Each iteration passes through a sequence of control/update/attention/recall/result operations and after hundreds of iterations, the final result is obtained from a fully connected network (FCN).

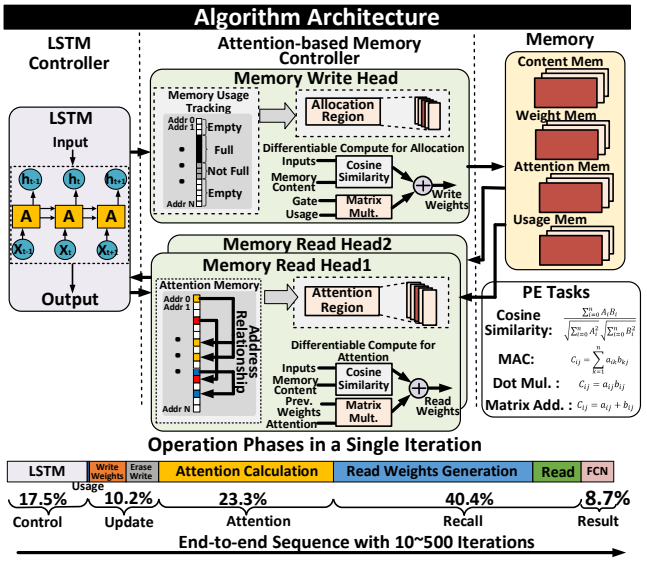


Fig. 2. Differentiable Neural Computer Algorithm.

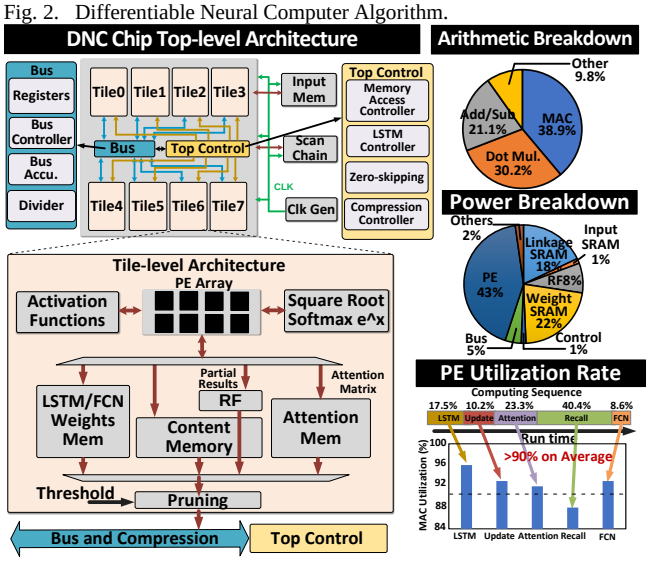


Fig. 3. DNC Chip Top-level Architecture, PE utilization, arithmetic and power breakdown.

As shown in the arithmetic operation breakdown in Fig. 3, besides extensive memory operation, DNC needs to support a variety of operations using PEs including MAC operations (38.9%) from LSTM and FCN, dot and vector multiplication (30.2%) and matrix addition/subtraction (21.1%) for memory similarity calculation. This leads to challenges in utilization of PE and high memory bandwidth required from memory banks, i.e. weight memory for LSTM/FCN, main content memory, usage memory and attention memory. To overcome the challenges, as shown in Fig. 3, a distributed near-memory computing (NMC) architecture is developed where the chip is divided into computational tiles connected by a global bus. Each tile embeds a small PE array with 8 MACs, distributed memories, a register file and special function modules, e.g. SoftMax. For fitting into a small chip budget, 8 tiles were implemented and can be proportionally scaled up. Fig. 3 also shows the PE utilization of this design with over 90% on average. Power breakdown is also shown in Fig. 3 with 43% from PEs and 50% from memory.

III. Near-memory Computing Array

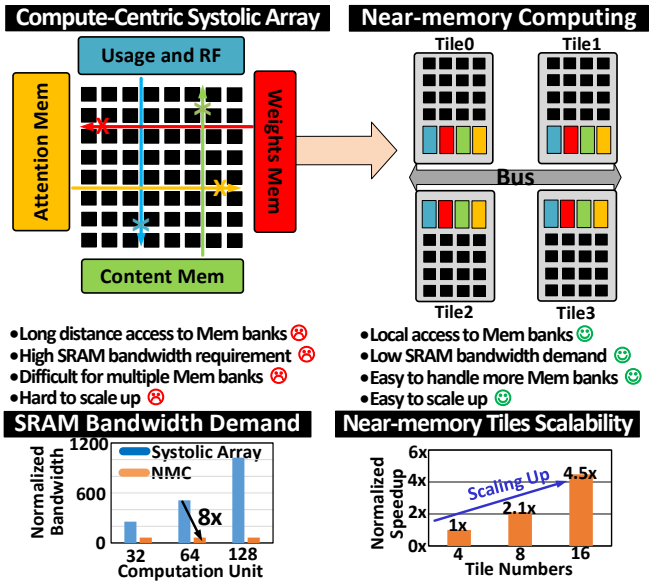


Fig. 4. Comparison between conventional systolic array and near-memory computing architectures.

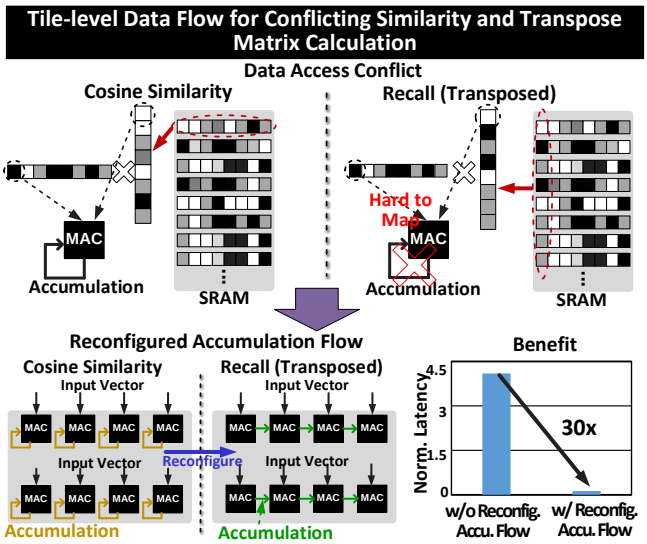


Fig. 5. Reconfigured dataflow for tile-level conflicts.

Fig. 4 compares the proposed NMC architecture with a conventional systolic array (SA). The required access from different types of memory in DNC causes low efficiency, data collision, large travel distance and poor scalability from SA. NMC allows data to stay locally broadcasting only processed data with 8X reduction of memory bandwidth. In addition, NMC is scalable in throughput with computing tiles in contrast with SA. Optimization of dataflow for different computing phases, e.g. LSTM, attention, etc. are performed at tile level. As shown in Fig. 5, a data mapping conflict between similarity and recall operation with transposed matrix calculation is observed. A reconfigurable flow is used to pass accumulation results in different directions with significant latency enhancement.

IV. Zero-skipping and Data Compression

Fig. 6 shows a reconfigurable MAC which was developed to deal with a variety of operations including MAC, dot multiplication and addition/subtraction. In addition, extensive clock gating and a configurable hybrid precision of 8 bits (LSTM) and 16 bits (Read/Write Head) are used for PE array with minor accuracy loss (3~4% from 32 bits).

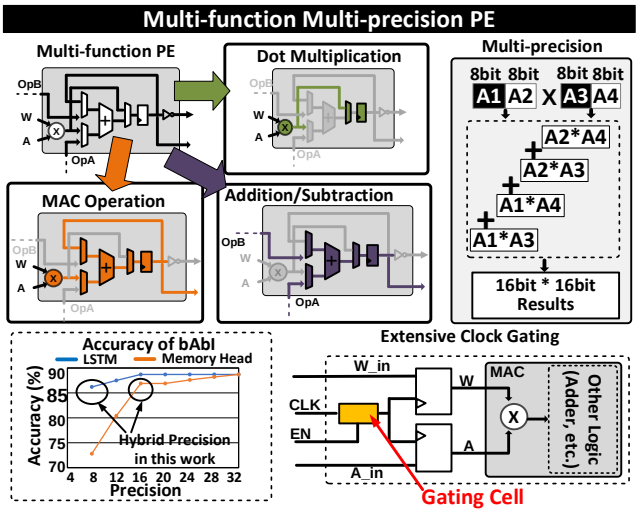


Fig. 6. PE Reconfiguration, multi-precision, clock-gating.

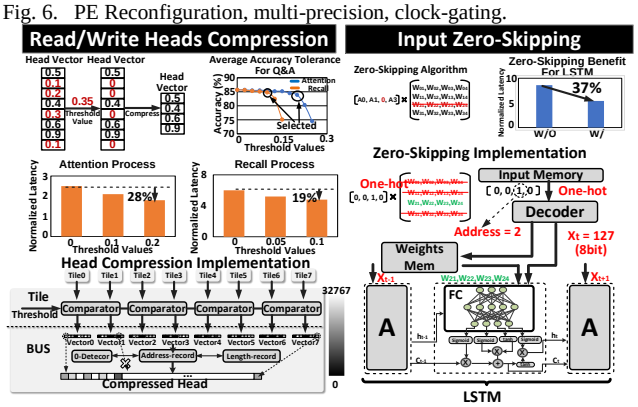


Fig. 7. Data compression and zero-skipping techniques used in this work leveraging sparsity of DNC accelerator.

Fig. 7 shows the sparsity and compression techniques used in this work. In write/read head operations, non-zero weights are compressed in a global bus before sending to each computing tile with preset threshold to prune the write/read weights to enhance sparsity. The data compression technique results in 28% speedup for the attention calculation with negligible overhead and minimal accuracy loss. Due to high sparsity of incoming data stream, input zero-skipping with associated detection and decoder logic as shown in Fig. 7 is also implemented to skip large amount of related MAC operations and weight loading in LSTM for frequent one-hot inputs. As a result, 96% of FCN operations or 37% of total LSTM operations are being bypassed.

V. Measurement Results and Demonstration Cases

A 65nm test chip was fabricated running at 350MHz at nominal 1V. Different reasoning tasks using DNC models trained offline was sent into the chip for evaluation with end-to-end operations. Fig. 8 to 10 show detailed descriptions of four examples of reasoning tasks implemented in the test chip including copy task, finding family relationship based on family tree, graph traversal task for traversing London underground stations within a given number of steps and context-based Q&A using bAbI database [5]. Attention mechanism from attention memory is highlighted to show the sequential relationship discovered by the chip. Performance comparison with CPU and GPU and accuracy comparison with floating point model are also shown.

As in the copy task in Fig. 8, DNC receives a sequence of vectors as input data and generate the same vector pattern in as the output. The attention memory is used to store the relationship, i.e. the sequence, between the different addresses of the content memory. For example, the large value in the coordinate (1,2) of the attention memory represents the address 1 and 2 (blue circle) are highly related in relationship enabling "copying" the sequence of the vectors. As shown in Fig. 9 for the family tree example, relationships for immediate family, i.e. father, mother, son, daughter are encoded as vectors sent into DNC to build the family tree graph. The attention memory represents family relationship. As an example, "Amy, David, Father" and "Mary, Amy, Mother" are inputs that include family relationships to be recorded by the attention memory. By performing inference, DNC can generate any relationship between 2 people in the family tree. In this task, the DNC accelerator can achieve about 693X speedup than CPU (Ryzen 5 2600X) with 4% accuracy loss.

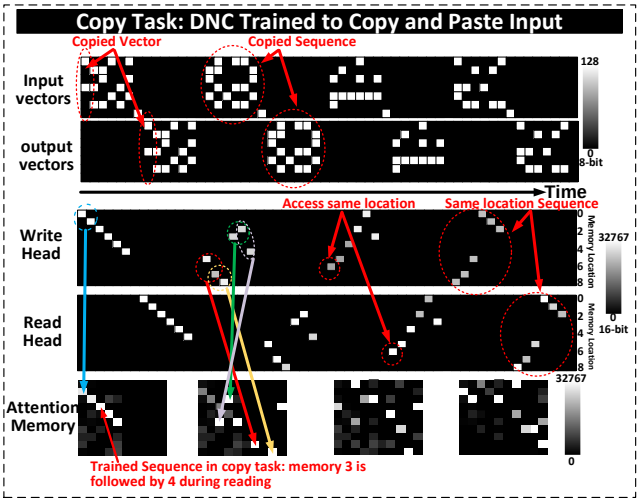


Fig. 8. Detailed demonstration of copy task.

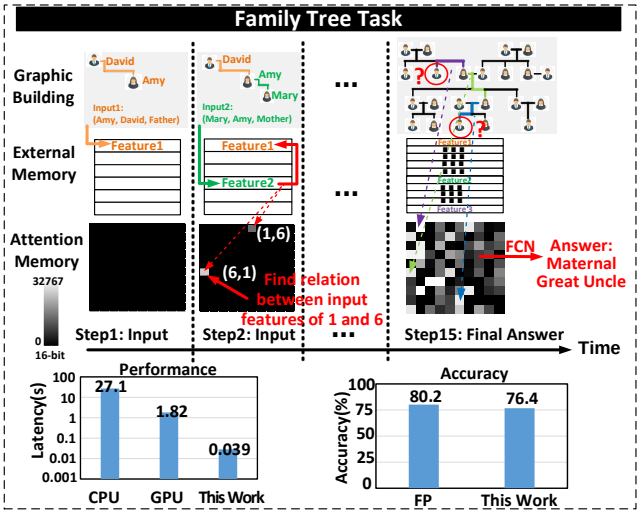


Fig. 9. Detailed demonstration of family tree task.

Finding the logic relationship between two objects in the sequential context or graph is another important application of DNC. As the Q&A task shown in Fig. 10, DNC receives the text contents and is able to give the answer about the questions

on the relationship of the objects. The speedup compared with CPU is around 709X. The London underground traversal task is also shown in Fig. 10, DNC can find the traversal path back to the starting station after receiving the information from London underground map in advance. The speedup compared with CPU is 697X with about 4% accuracy degradation.

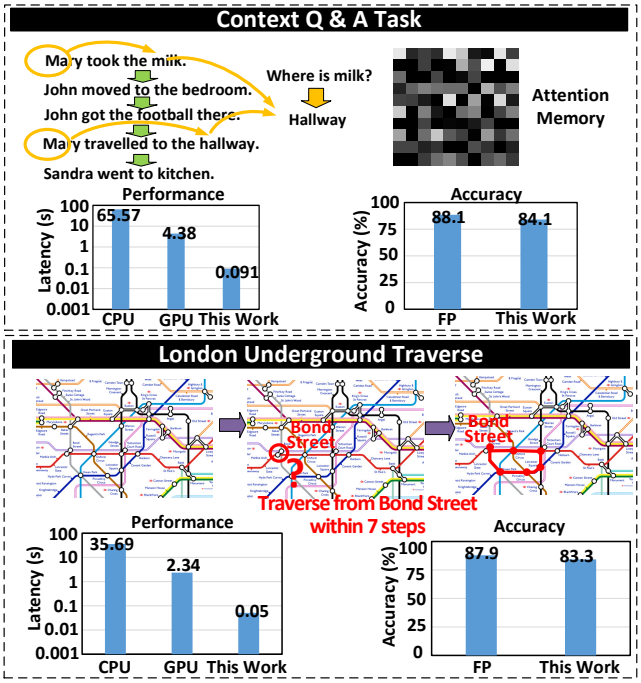


Fig. 10. Details of Q&A task and London underground traversal task.

In total, eight different logic reasoning tasks spanning across diversified jobs including sorting, copying, repeated copy, recall, sort, context Q&A, graphic traversal and shortest path, were tested to verify the functionality and performance against commercial CPU and GPU. Fig. 11 shows the measurement results on power and latency. An average of above 90% utilization has been observed among computing phases. The test chip achieved 700X and 46X speedup over CPU and GPU processors across the eight test cases. End-to-end speedup of 30% was also achieved from the applied sparsity enhancement techniques. The comparison table was shown in Fig. 12. As this work is the first implementation of a reasoning processor, comparison was made mainly to prior DNN accelerators specially for LSTM/FCN in similar technology. A maximum efficiency of 1.28TOPS/W is observed for 8-bit LSTM. Compared with a prior simulation-based work using a related but different computation model of MANN [6], a 21X improvement of efficiency is observed from this work. Fig. 13 shows the chip micrograph.

VI. Conclusion

A 65nm test chip using Differentiable Neural Computer model was implemented to perform logic reasoning tasks for the first time. A special NMC architecture was developed rendering lower requirement of SRAM bandwidth with better scalability. Input zero-skipping and data compression techniques are applied to achieve 28% reduction on attention calculation and 37% reduction of LSTM operations. Eight different logic reasoning tasks are demonstrated using the test chip. 700X and 46X speedup compared with commercial CPU and GPU are observed on the logic reasoning tasks with a power efficiency of up to 1.28TOPS/W and over 90% utilization of PE units in all the eight computing phases.

References

[1] J. Rae, et al., “Scaling Memory-Augmented Neural Networks with Sparse Reads and Writes,” NIPS, 2016.

[2] T. Munkhdalai, et al., “Meta Networks”, ICML, 2017.

[3] A. Baddeley, et al., “Working Memory: Theories, Models, and Controversies,” Annual Review of Psychology, Jan. 2012.

[4] Graves, A., Wayne, G., Reynolds, M. et al., “Hybrid computing using a neural network with dynamic ex-ternal memory,” Nature, 2016.

[5] Facebook bAbI, https://research.fb.com/downloads/babi

[6] J. Stevens, et al., “Manna: An Accelerator for Memory-Augmented Neural Networks,” MICRO, pp. 794–806, 2019.

[7] D. Shin, J. Lee, J. Lee and H. -J. Yoo, “14.2 DNPU: An 8.1TOPS/W reconfigurable CNN-RNN processor for general-purpose deep neural networks,” ISSCC, 2017.

[8] Y. -H. Chen, T. Krishna, J. Emer and V. Sze, "14.5 Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks," ISSCC, 2016.

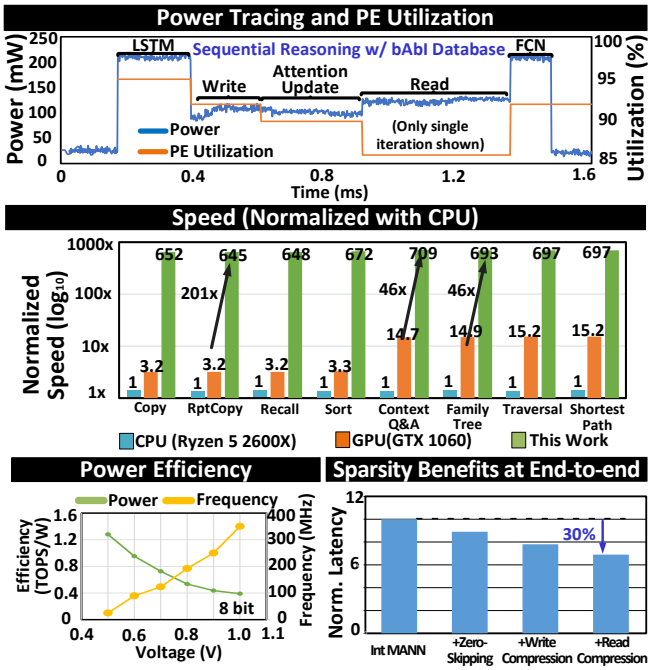
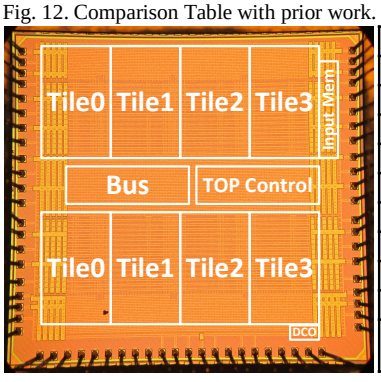


Fig. 11. Measurement results

TABLE I. Comparison Table

	MICRO2019[6]	DNPU[7]	Eyeriss[8]	This Work
Core	MANN	CNN,FC,LSTM	CNN	DNC
Num. of PE	3*256	768(16bit)	168	64
Process(nm)	15nm Nangate Open Cell Library	65nm	65nm	65nm
Area(mm2)	40	16	12.25	7.75
Supply Vdd	-	1.1V	1.0V	1.0V
Power	16W (TDP)	279mW	278mW	230mW
Freq. (MHz)	500	200	200	350
Data Type	FP32	INT1~16	INT16	INT8(LSTM) INT16(Read/Write Head)
Memory	39.8MB	-	181KB	200KB
Power Efficiency	18GOPS/W (Simulated Results)	3.9TOPS/W (4b) 1.0TOPS/W (16b)	0.241TOPS/W (1V,16b)	389.6GOPS/W (1V,8b) 1.28GOPS/W (0.5V,8b)

Fig. 12. Comparison Table with prior work.



Technology	65nm CMOS
Area	7.75mm^2
Power	230mW
PE Number	64
Tile Number	8
PE/Tile	8
Bit Precision	INT8, INT16
Frequency	350MHz
Supply Vdd	0.5~1V
SRAM	200KB
Efficiency (TOPS/W)	0.39(1V, 8bit) 1.28(0.5V, 8bit)

Fig. 13. Micrograph of the test chip.