CICC

IEEE Custom Integrated Circuits Conference

13-2: KeyRAM: A 0.34 μJ/decision 18 k decisions/s Recurrent Attention In-memory Processor for Keyword Spotting

Hassan Dbouk, Sujan K. Gonugondla, Charbel Sakr, and Naresh R. Shanbhag

University of Illinois at Urbana-Champaign

03/24/2020

IEEE SOLID-STATE CIRCUITS SOCIETY™

IEEE

Outline

- **Motivation and Background**
- Recurrent Attention Model for KWS
- Implementation
  - Chip Architecture
  - Sparsity-aware IMC Block
  - DM^{2}VM Digital Block
- Measurement Results
- Summary

Motivation

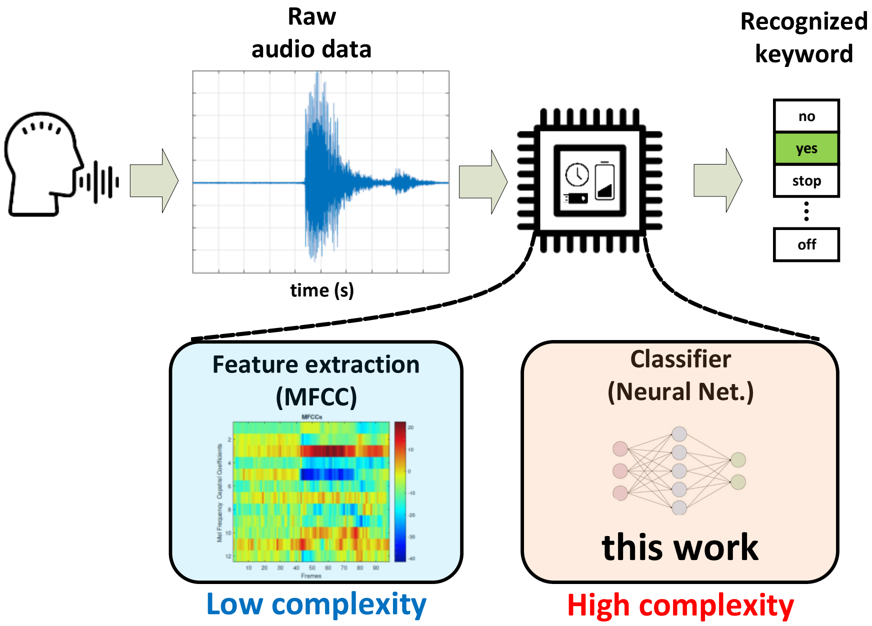
- Speech is a natural mode for humans to interact with intelligent Edge devices
- Edge devices are often constrained in terms of *storage*, *power*, and *compute resources*
- Keyword spotting (KWS) systems are used to detect specific wake-up words

Motivation

- Speech is a natural mode for humans to interact with intelligent Edge devices
- Edge devices are often constrained in terms of *storage*, *power*, and *compute resources*
- Keyword spotting (KWS) systems are used to detect specific wake-up words

Goal: An end-to-end **energy efficient** and **low latency** solution for keyword spotting

Typical KWS Pipeline



Raw audio data

time (s)

Recognized keyword

no

yes

stop

⋮

off

Feature extraction (MFCC)

Classifier (Neural Net.)

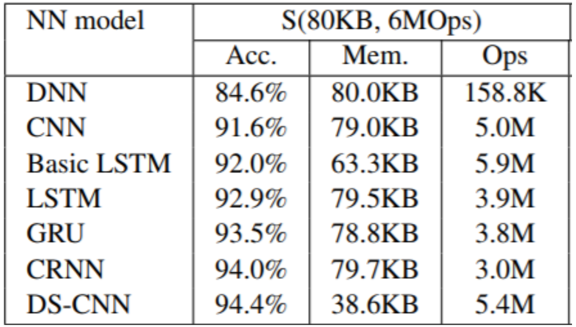
this work

Low complexity

High complexity

- Feature extraction: Mel-frequency Cepstral Coefficient (MFCC)
- What is a good classifier?

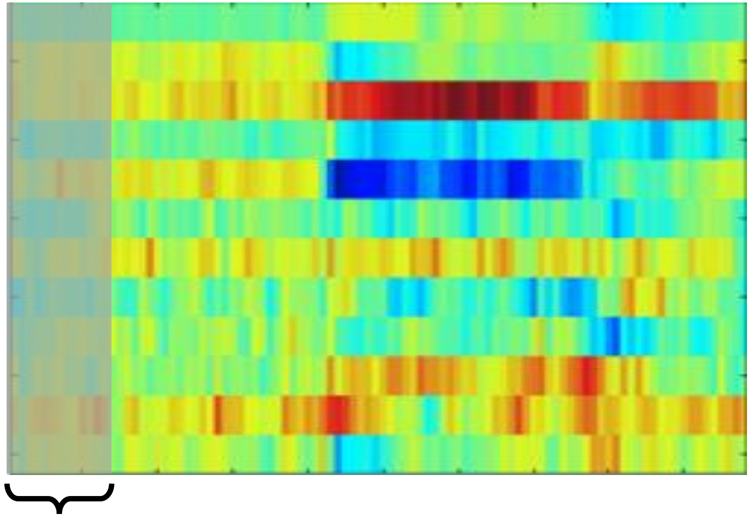
Prior works



NN model	S(80KB, 6MOps)		
	Acc.	Mem.	Ops
DNN	84.6%	80.0KB	158.8K
CNN	91.6%	79.0KB	5.0M
Basic LSTM	92.0%	63.3KB	5.9M
LSTM	92.9%	79.5KB	3.9M
GRU	93.5%	78.8KB	3.8M
CRNN	94.0%	79.7KB	3.0M
DS-CNN	94.4%	38.6KB	5.4M

Hello Edge [Zhang, arXiv 2018]

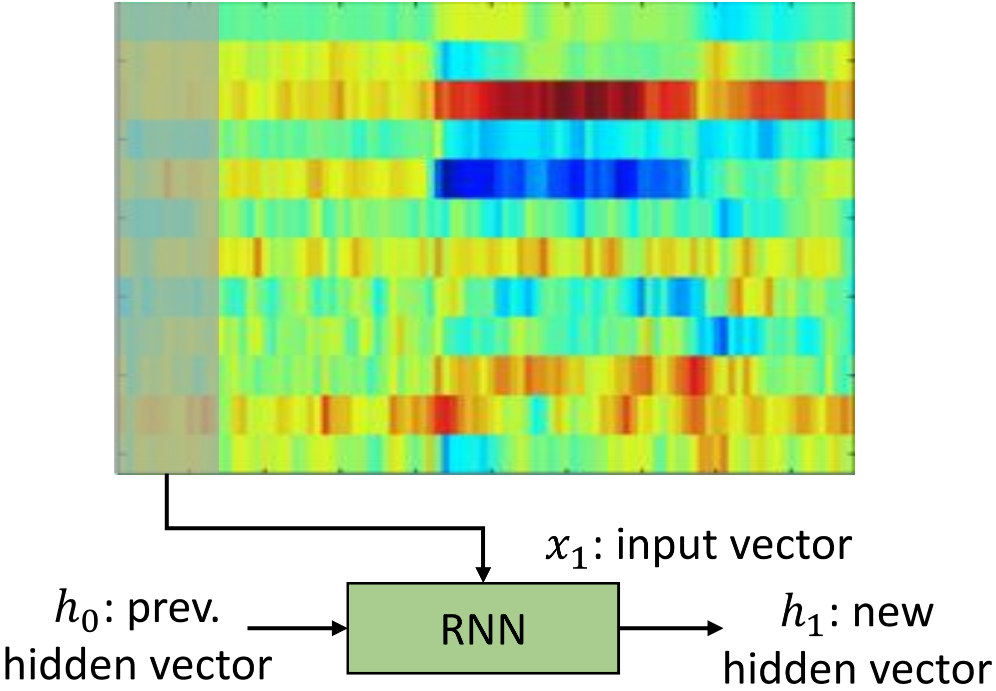
Vanilla RNN for KWS



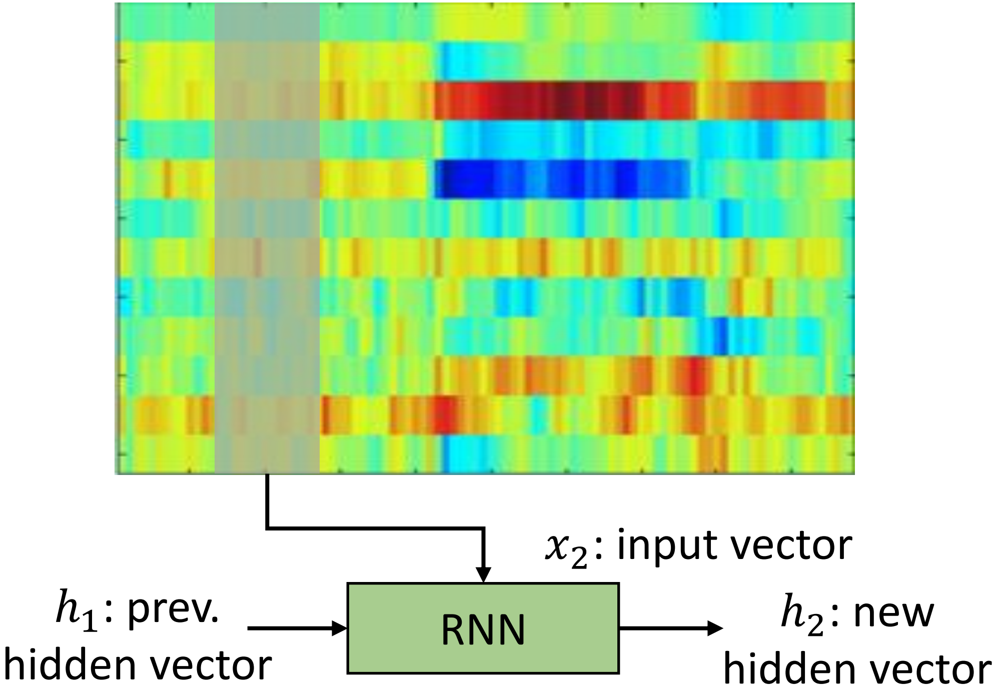
subset of features to be processed

- Sequential processing: must process the entire MFCC input features

Vanilla RNN for KWS (t=1)



Vanilla RNN for KWS (t=2)



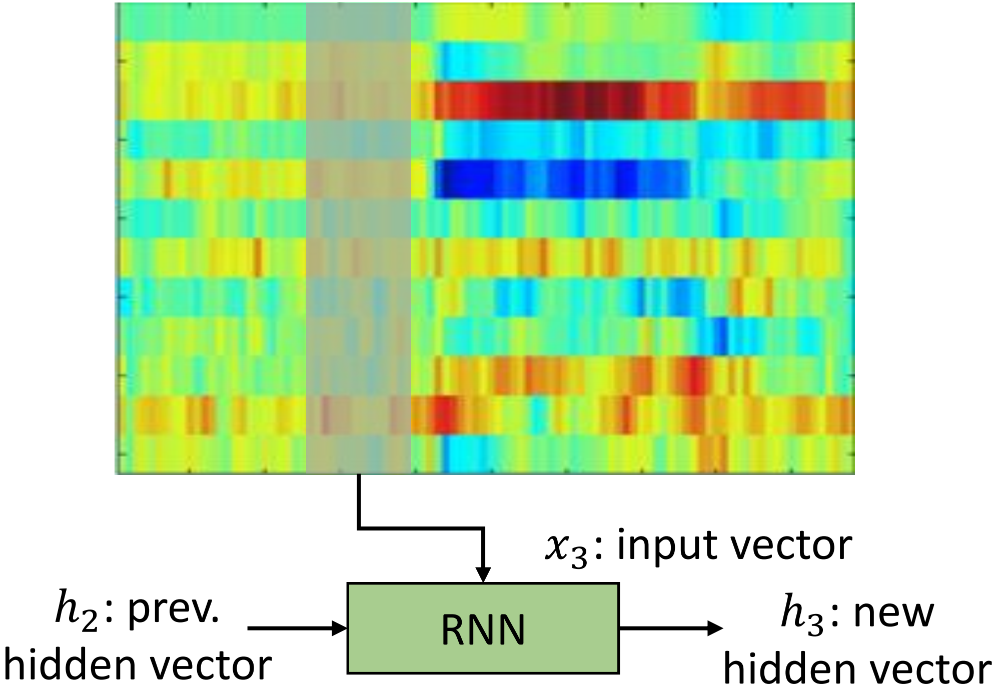
x_2: input vector

RNN

h_1: prev. hidden vector

h_2: new hidden vector

Vanilla RNN for KWS (t=3)



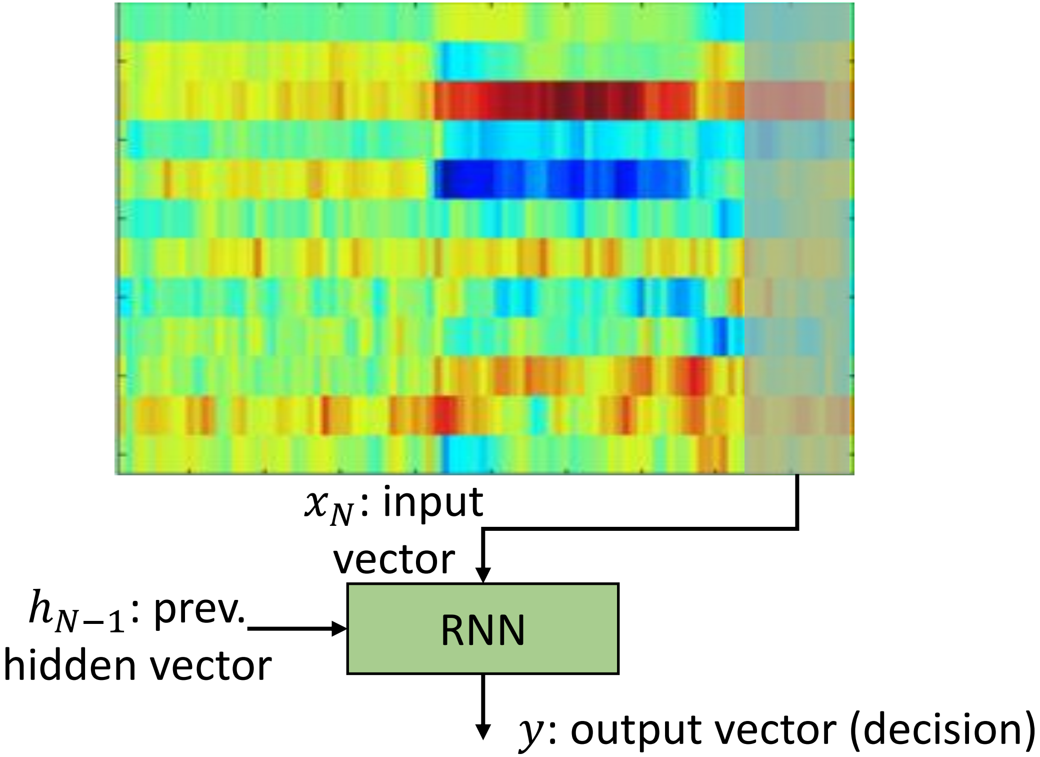
x_3: input vector

RNN

h_2: prev. hidden vector

h_3: new hidden vector

Vanilla RNN for KWS (t=N)



x_N: input vector

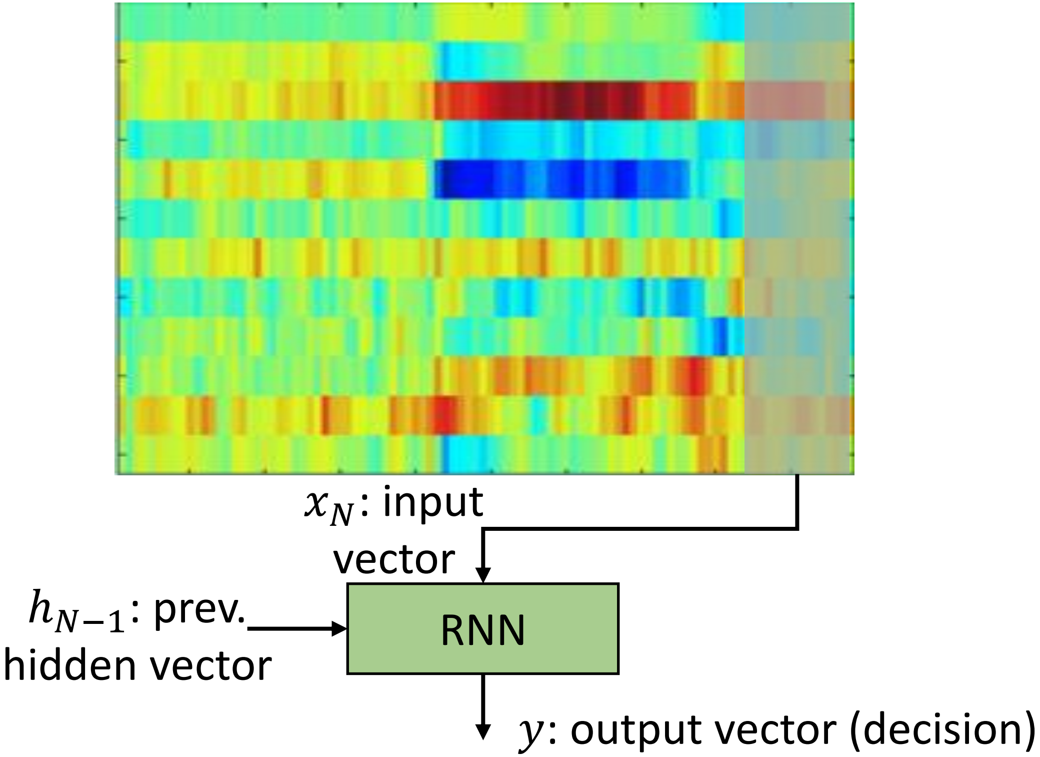
h_{N-1}: prev. hidden vector

RNN

y: output vector (decision)

Vanilla RNN for KWS (t=N)

Can we design a better classifier?



x_N: input vector

h_{N-1}: prev. hidden vector

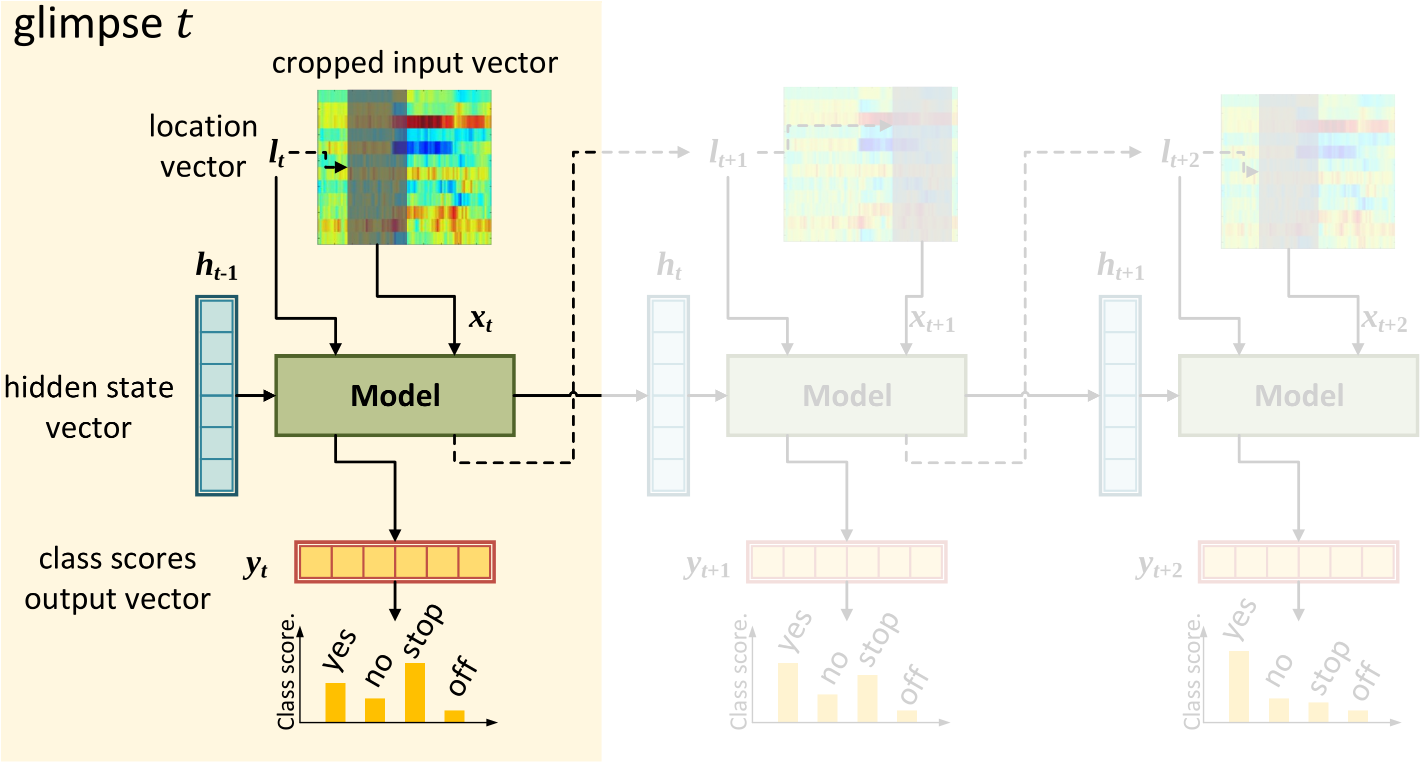
RNN

y: output vector (decision)

Outline

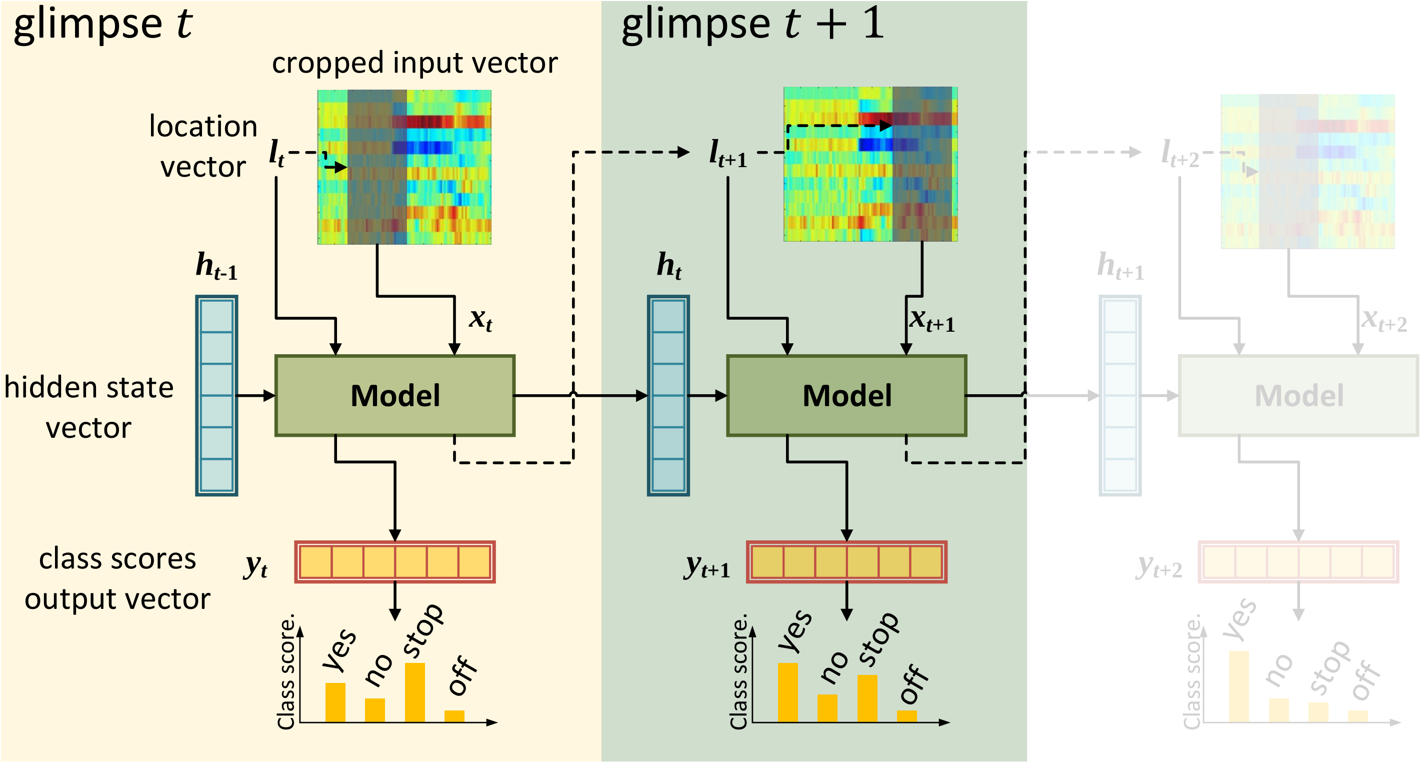
- Motivation and Background
- **Recurrent Attention Model for KWS**
- Implementation
  - Chip Architecture
  - Sparsity-aware IMC Block
  - DM^{2}VM Digital Block
- Measurement Results
- Summary

Recurrent Attention Model (RAM) for KWS



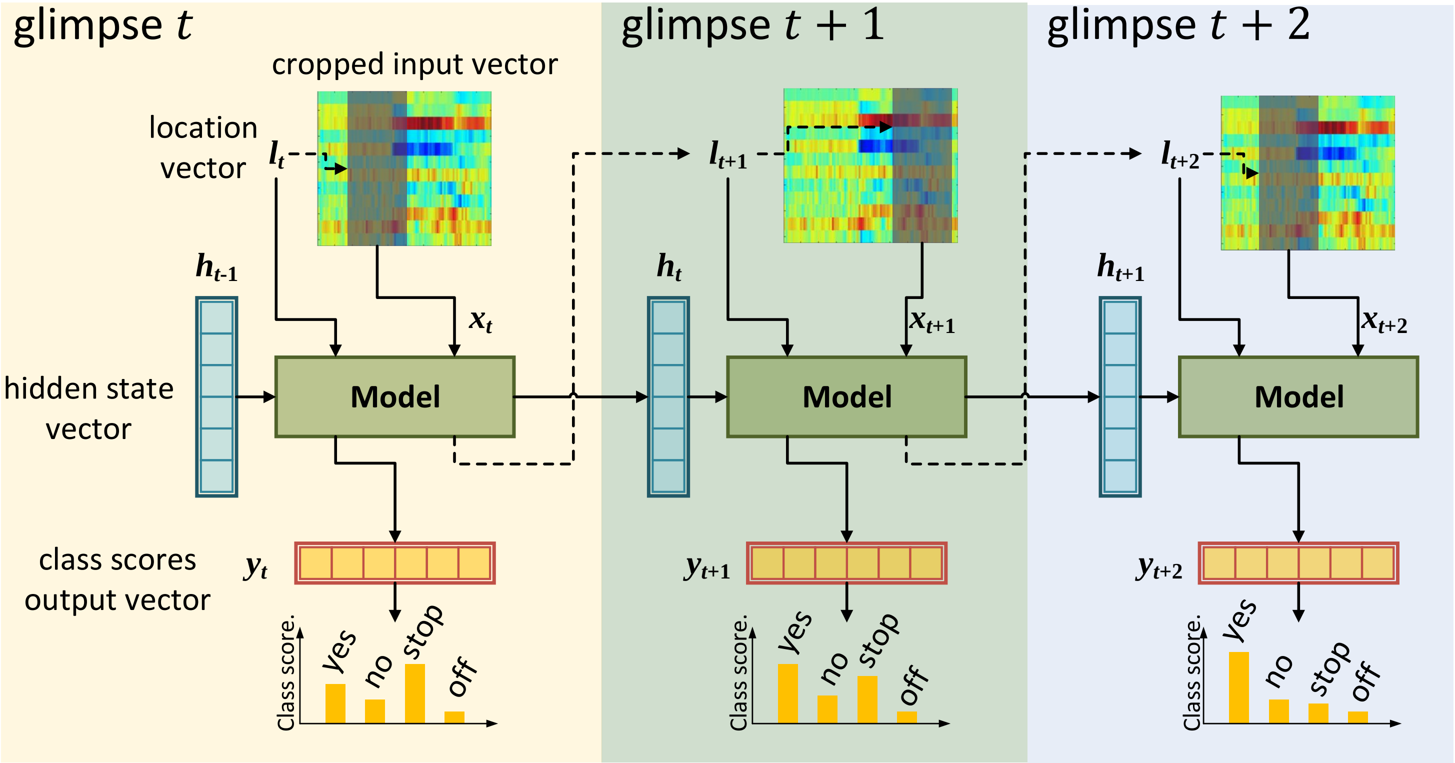
- Originally proposed for image classification [Mnih, NIPS'14]

Recurrent Attention Model (RAM) for KWS



- RAM: processes the input via glimpses, learns what glimpses to process

Recurrent Attention Model (RAM) for KWS

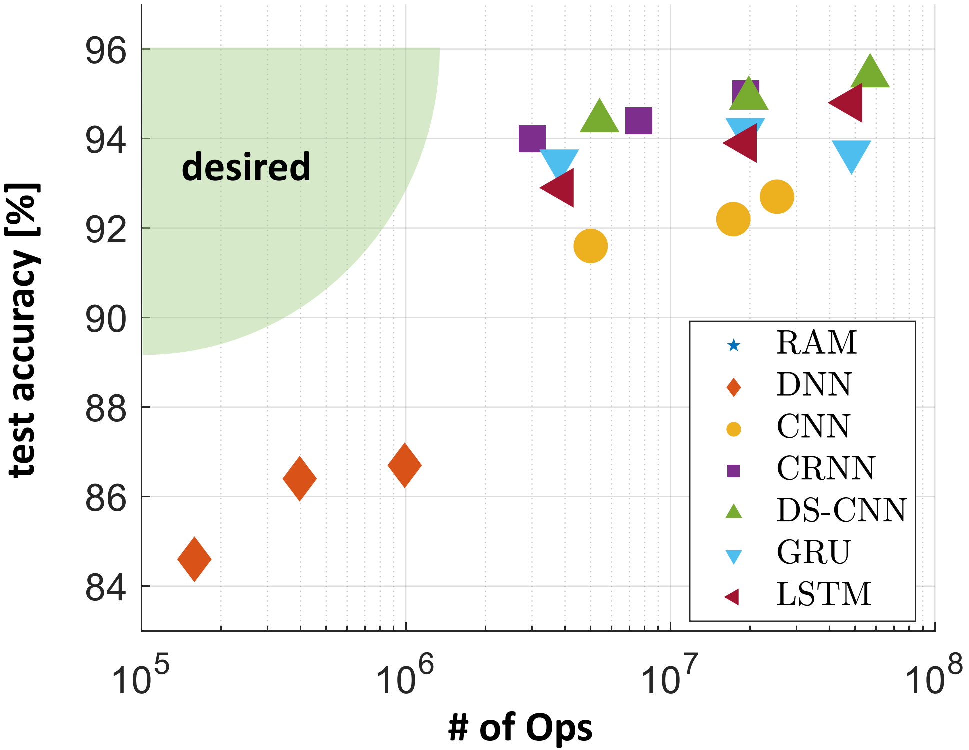


- More glimpses processed → more confident decisions
- Inherent accuracy-complexity (energy & latency) tradeoff

Efficiency of RAM for KWS

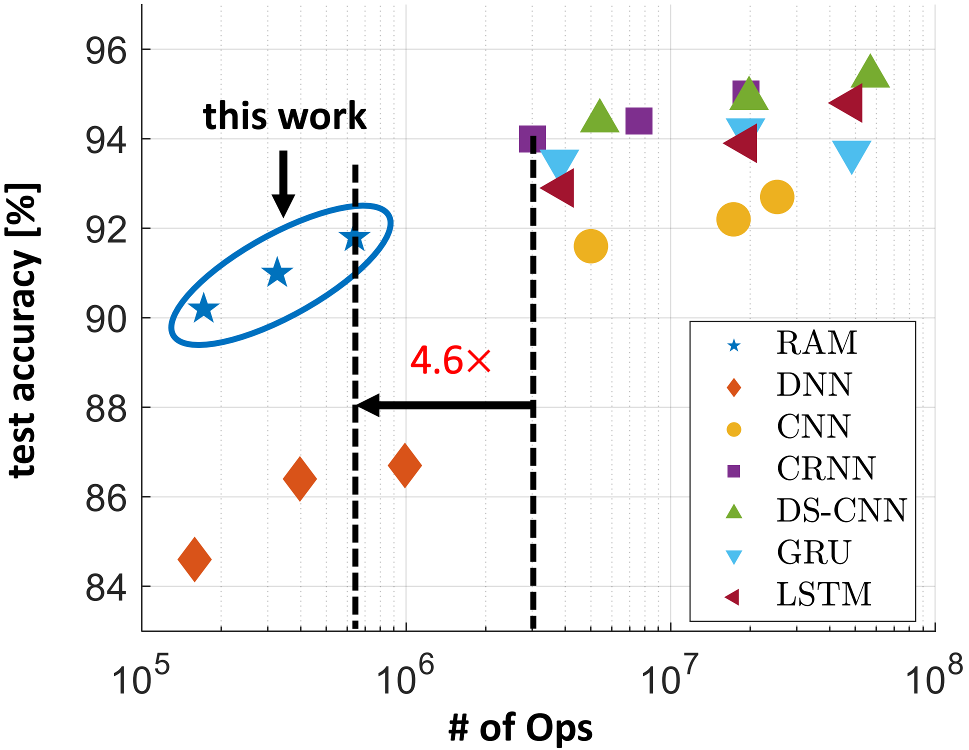
- KWS for 12 keywords using the Google Speech dataset

As reported by [Zhang, arXiv 2018]



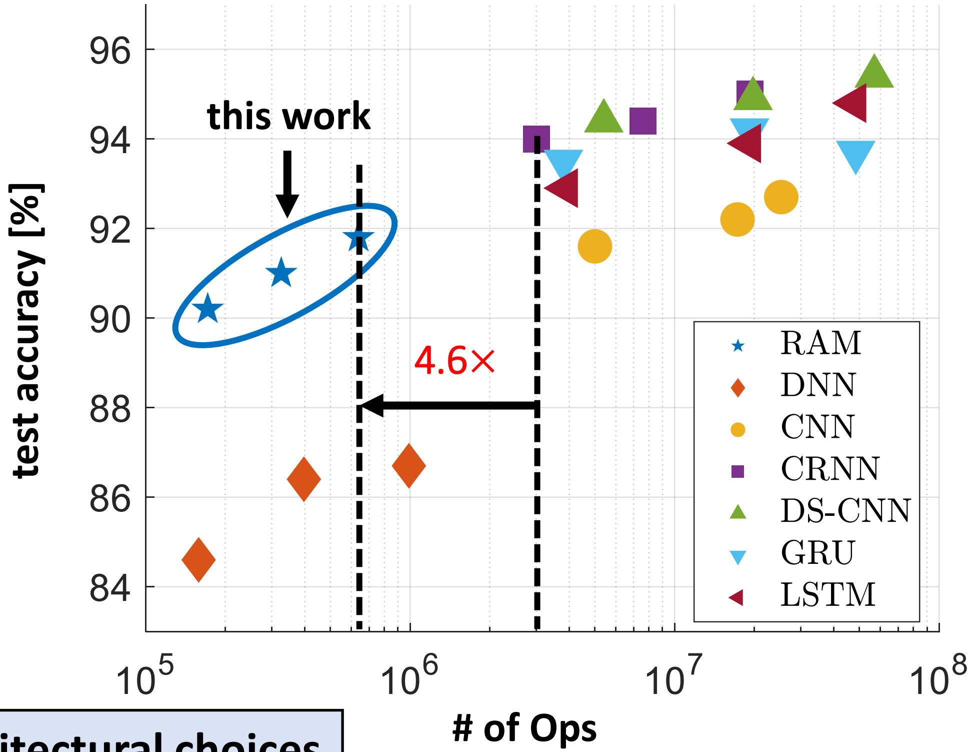
Efficiency of RAM for KWS

- KWS for 12 keywords using the Google Speech dataset
- RAM achieves a 4.6 × reduction in computational complexity at iso-accuracy



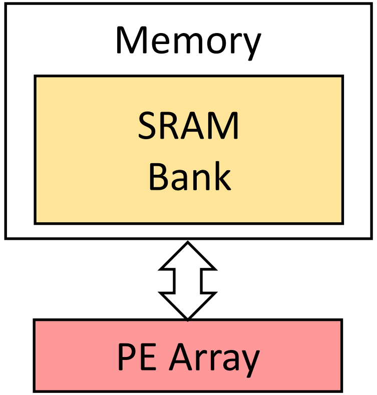
Efficiency of RAM for KWS

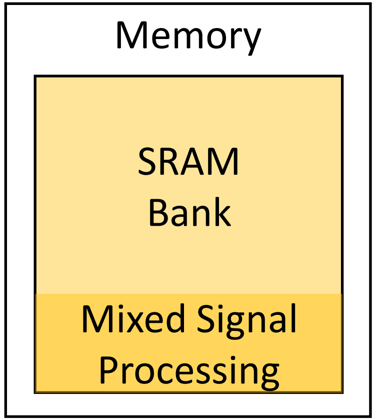
- KWS for 12 keywords using the Google Speech dataset
- RAM achieves a 4.6 × reduction in computational complexity at iso-accuracy



We have an **efficient** classifier, next: **architectural choices**

Architectural Choices

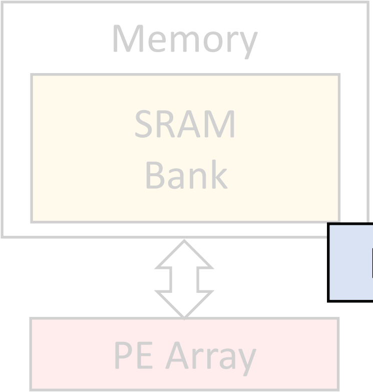




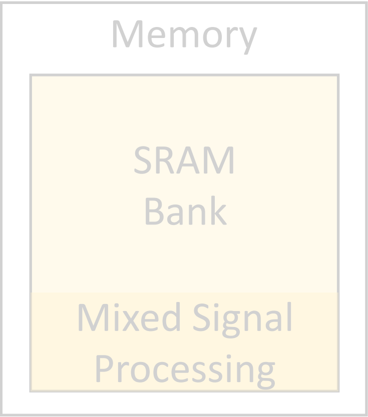
	Digital	In-Memory Compute (IMC)
Pros	reconfigurable – high precision	energy efficient – massive parallelism
Cons	high energy – limited parallelism	non reconfigurable – low precision

Architectural Choices

Digital



In-Memory Compute (IMC)



Hybrid design: choose both

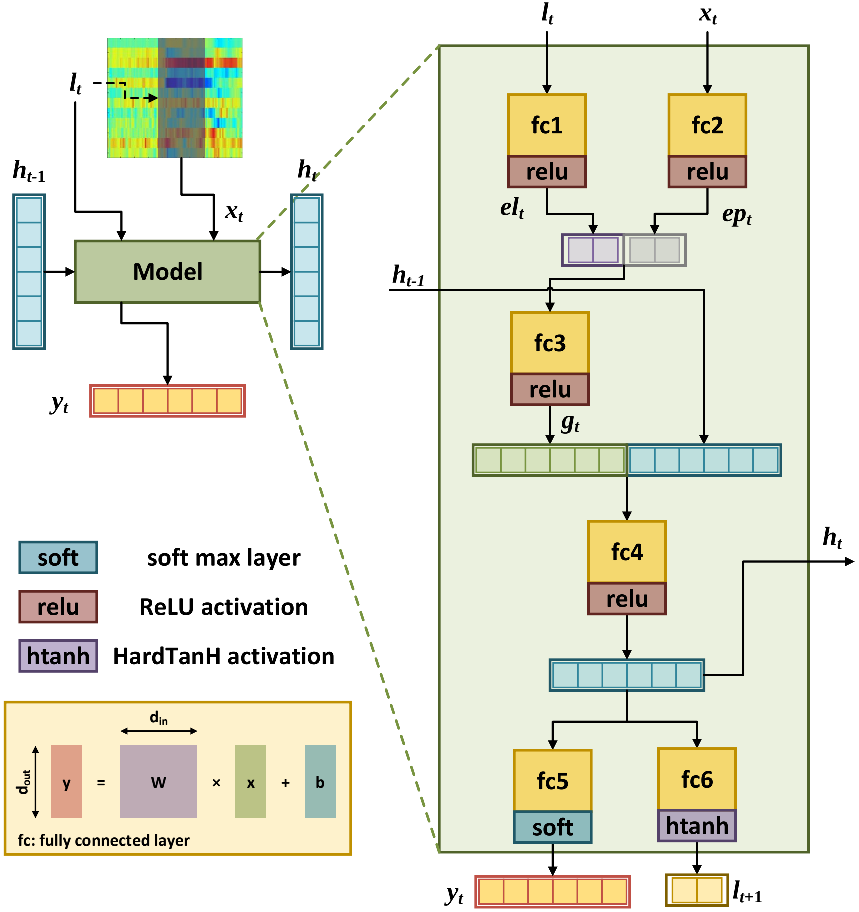
Pros	reconfigurable – high precision	energy efficient – massive parallelism
Cons	high energy – limited parallelism	non reconfigurable – low precision

Mapping of RAM

- 6 fully connected layers (fc1 to fc6)
- All weights on-chip

layer	d_{in}	d_{out}	B_x	B_w	#MACs	%MACs	Mapped to
fc1	2	63	8	8	189	0.35	DIGITAL
fc2	64	64	8	8	4160	7.63	DIGITAL
fc3	127	127	4	4	**16256**	**29.81**	**IMC**
fc4	254	127	4	4	**32385**	**59.39**	**IMC**
fc5	127	10	8	8	1280	2.35	DIGITAL
fc6	127	2	8	8	256	0.47	DIGITAL

fc3 & fc4: 89% of computations

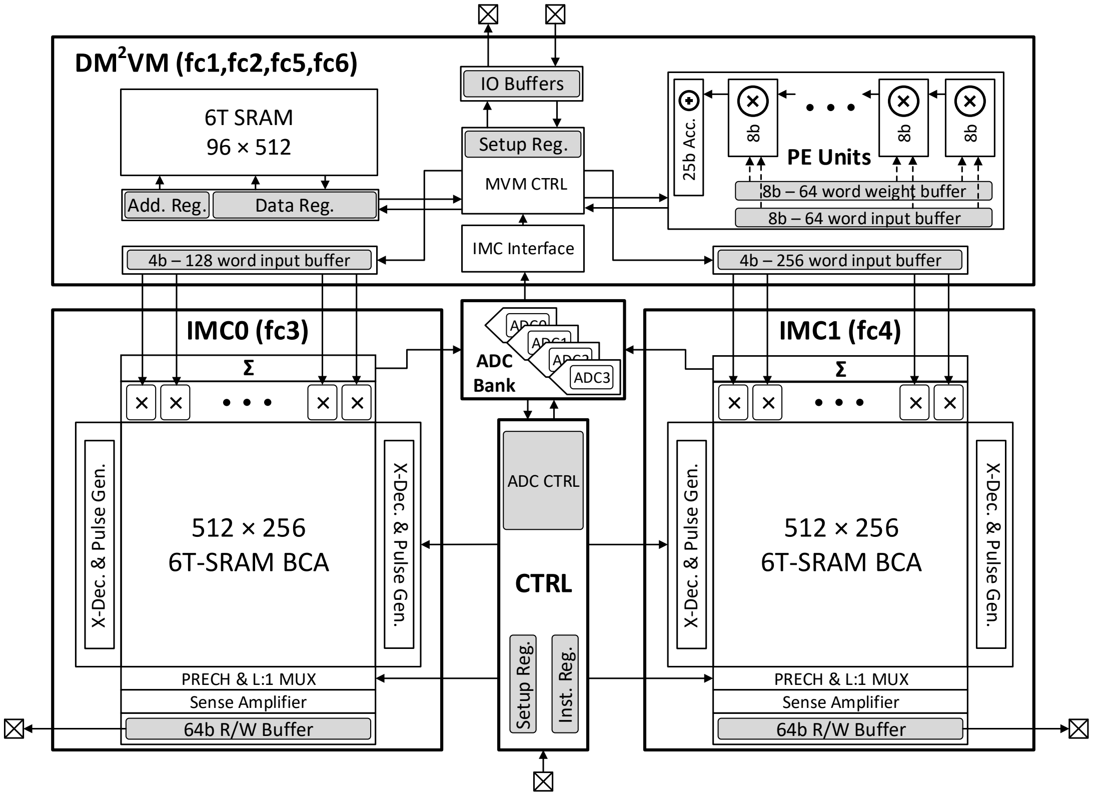


Outline

- Motivation and Background
- Recurrent Attention Model for KWS
- **Implementation**
  - **Chip Architecture**
  - Sparsity-aware IMC Block
  - DM^{2}VM Digital Block
- Measurement Results
- Summary

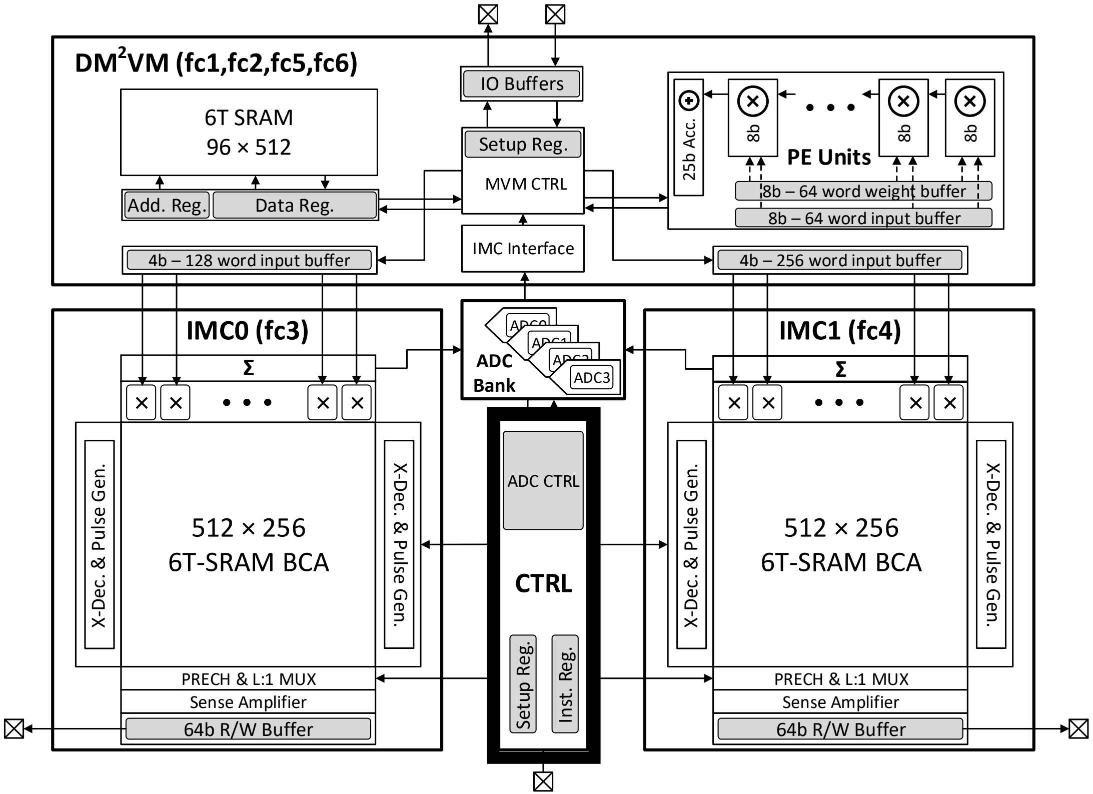
Chip Architecture

- Main controller
- Two IMC blocks
- Four single-slope ADCs
- Digital processor



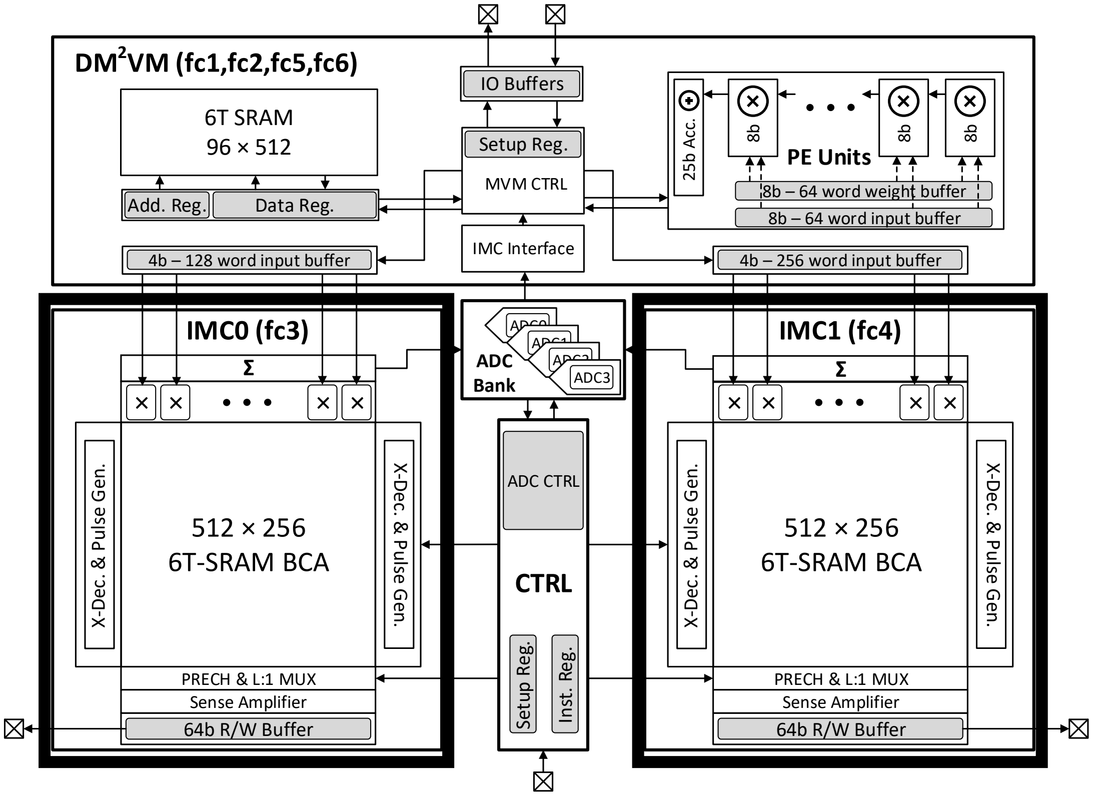
Chip Architecture

- **Main controller**
- Synchronizes all chip operations
- 6 main modes of operation
- Runs on a 1GHz external clock



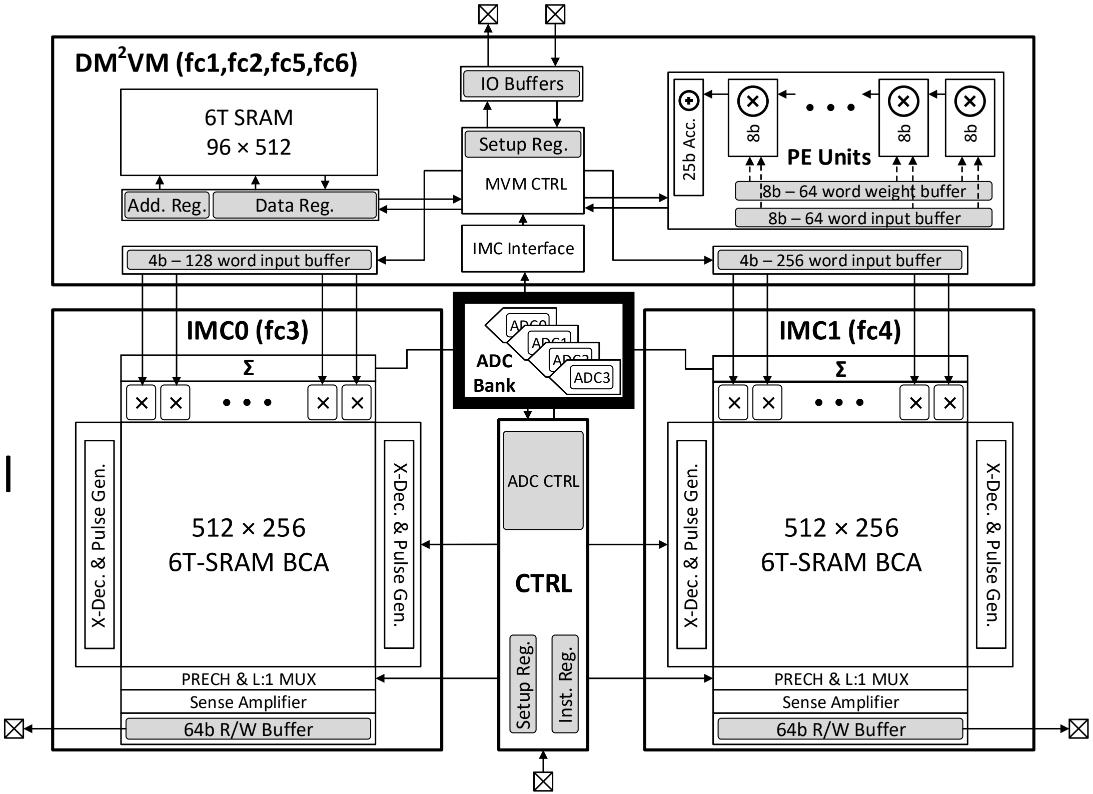
Chip Architecture

- Main controller
- **Two IMC blocks**
- 512 × 256 standard 6T SRAM banks
- Execute fc3 and fc4



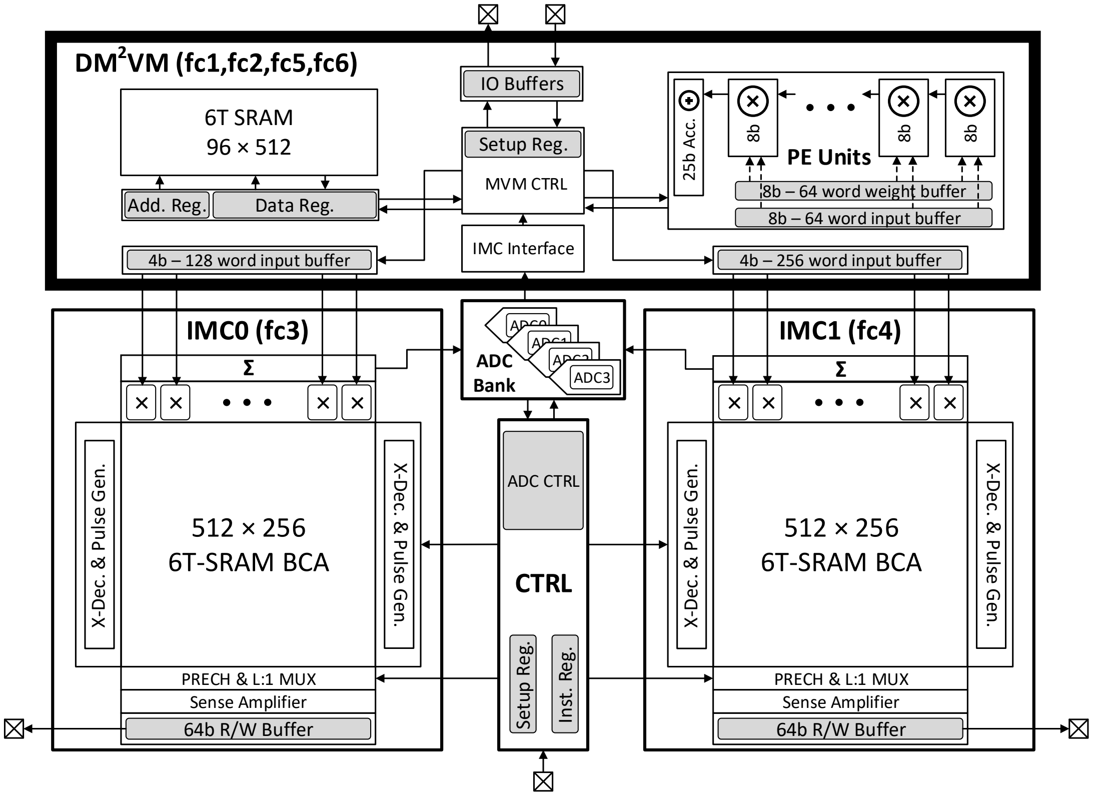
Chip Architecture

- Main controller
- Two IMC blocks
- **Four single-slope ADCs**
- Operate at 10 M Sample/s
- Two 6-b ADCs required per IMC dot product (differential design)



Chip Architecture

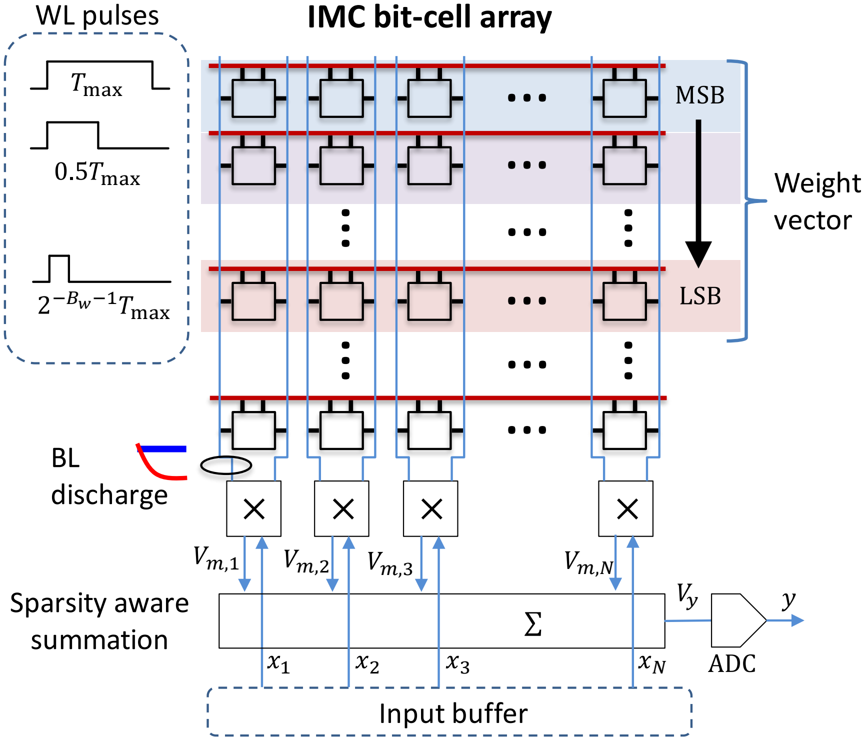
- Main controller
- Two IMC blocks
- Four single-slope ADCs
- **Digital processor**
- 6kB of SRAM + 64 8b MAC units
- Executes fc1, fc2, fc5, & fc6



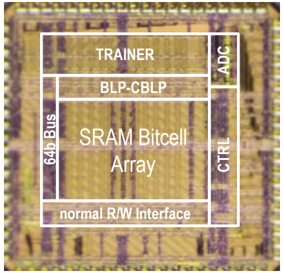
Outline

- Motivation and Background
- Recurrent Attention Model for KWS
- **Implementation**
  - Chip Architecture
  - **Sparsity-aware IMC Block**
  - DM^{2}VM Digital Block
- Measurement Results
- Summary

In-Memory Compute Block

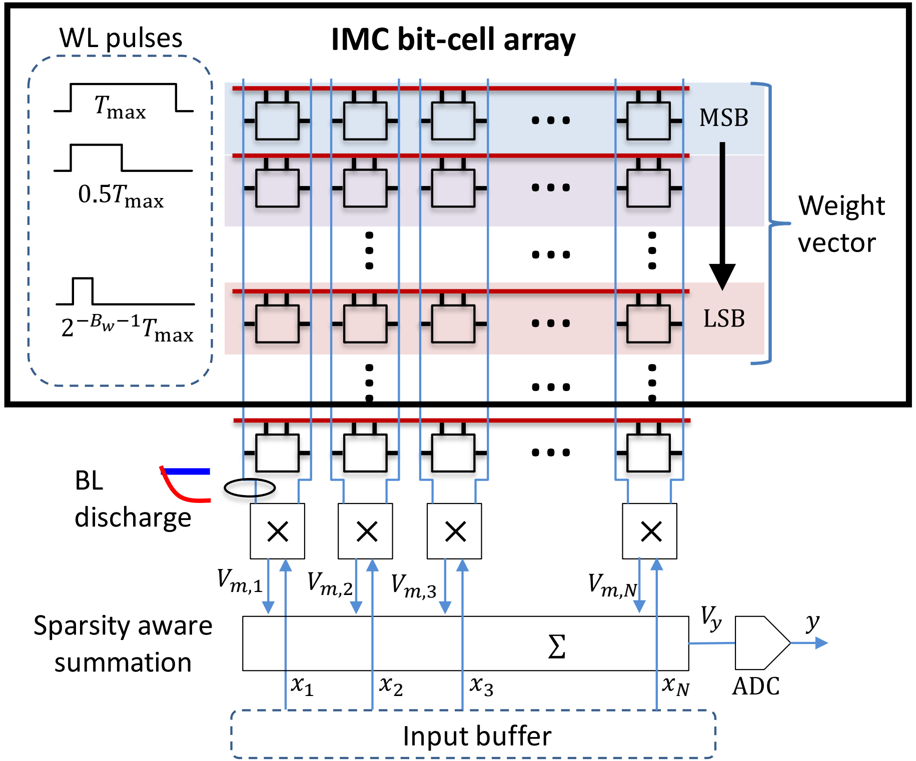


- Standard 6T SRAM bank
- Multi-bit dot products via four stages



Adapted from [Gonugondla, ISSCC'18]

In-Memory Compute Block

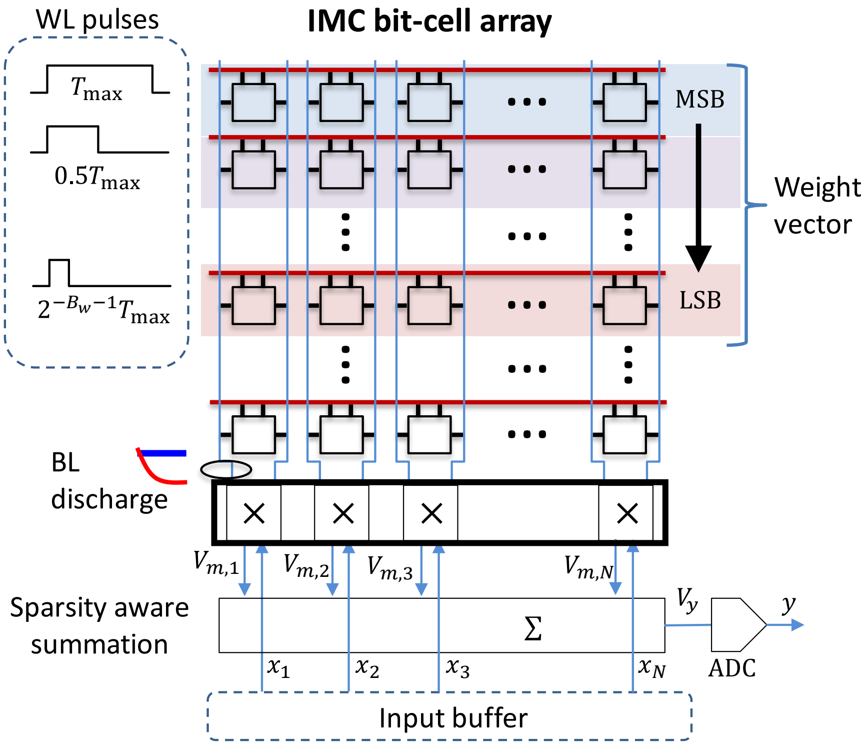


- Standard 6T SRAM bank
- Multi-bit dot products via four stages

①

Pulse-width modulated word-lines perform D2A conversion of weights on each bit-line (BL)

In-Memory Compute Block

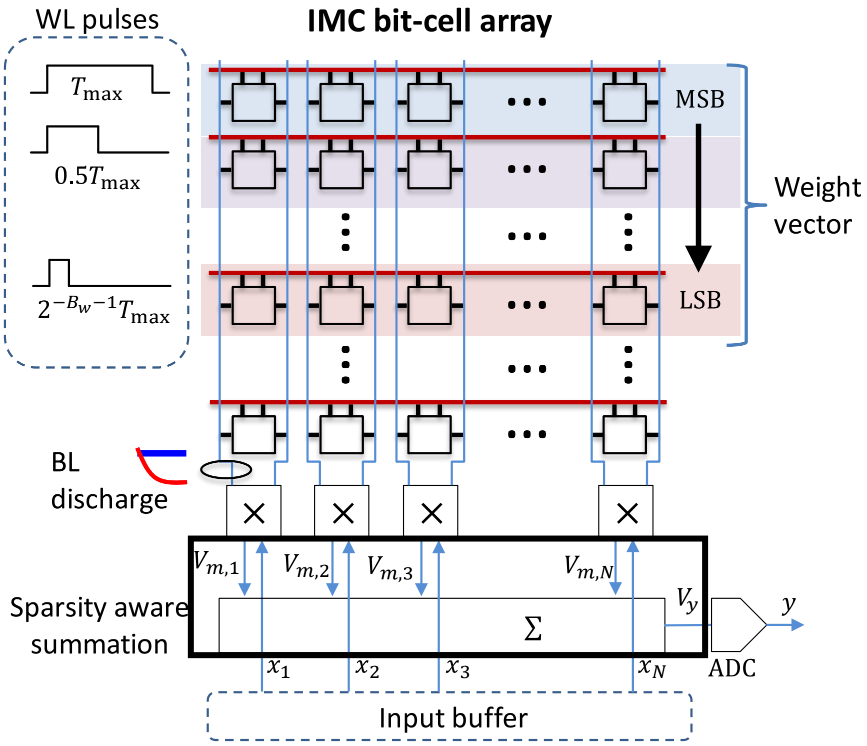


- Standard 6T SRAM bank
- Multi-bit dot products via four stages

2

BL discharges are multiplied with the corresponding input data from buffers via charge redistribution

In-Memory Compute Block

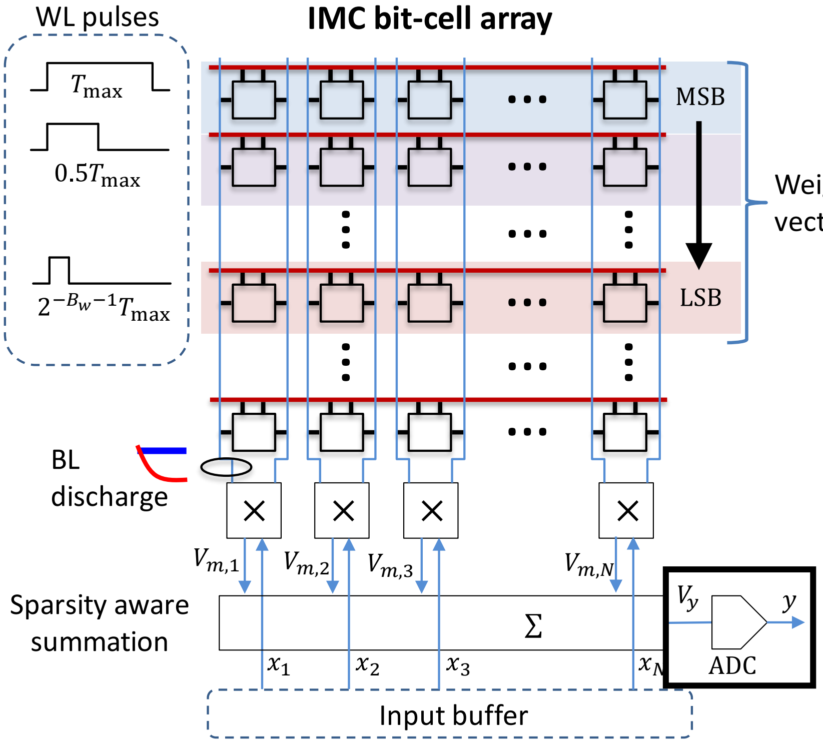


- Standard 6T SRAM bank
- Multi-bit dot products via four stages

3

Multiplier outputs are summed across the columns via charge sharing across BLs

In-Memory Compute Block

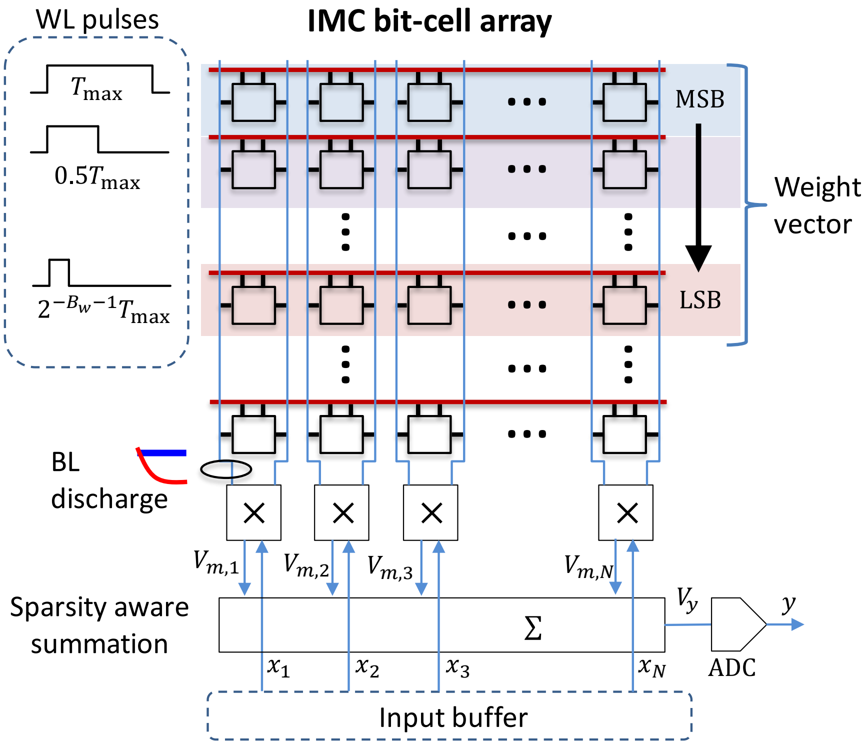


- Standard 6T SRAM bank
- Multi-bit dot products via four stages

4

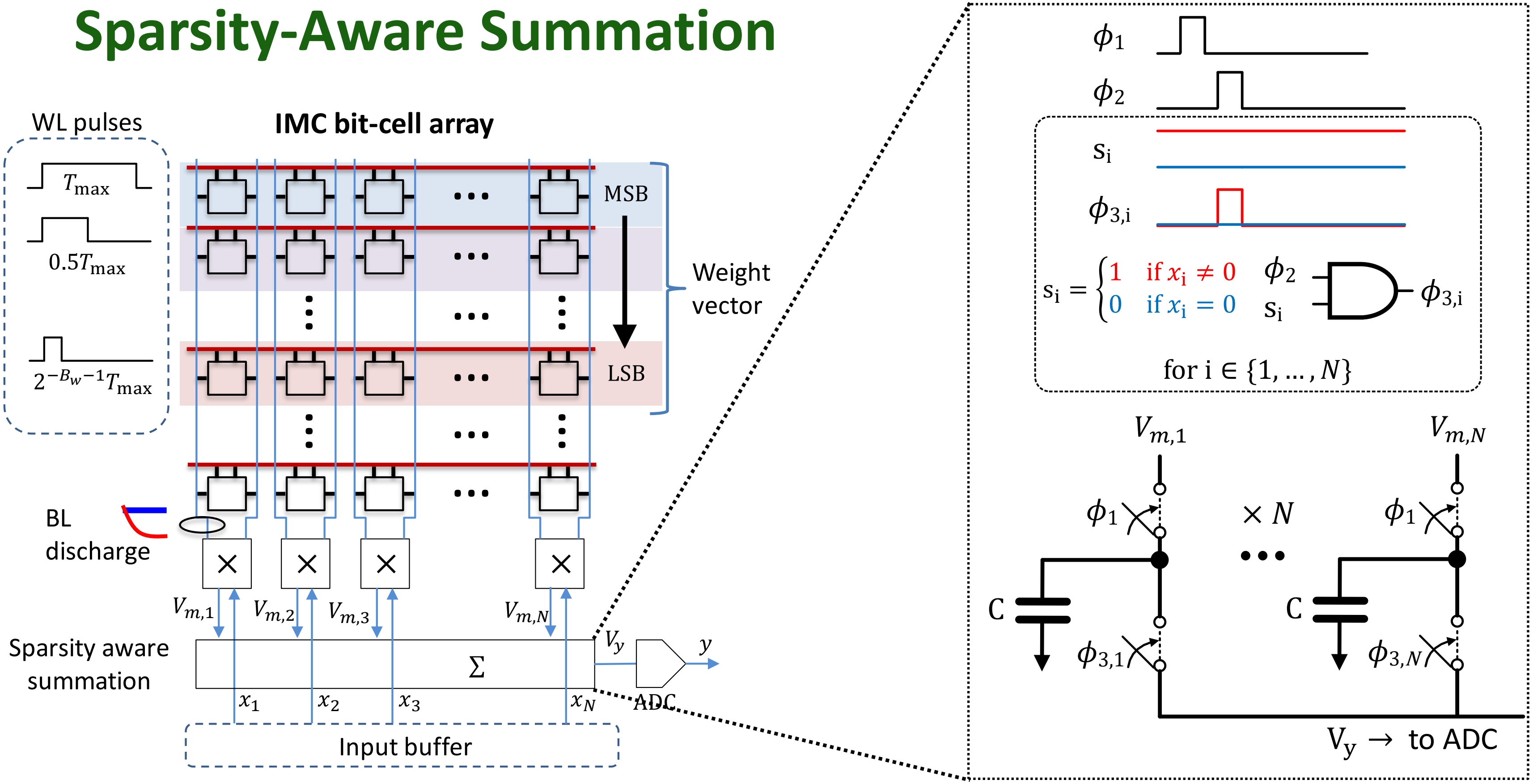
Final dot product voltage is converted to digital via ADCs

Input Sparsity Challenge



- ReLU activation functions cause sparse inputs ($\sim 50\% - 70\%$)
- Output voltage spread shrinks due to charge sharing

Sparsity-Aware Summation

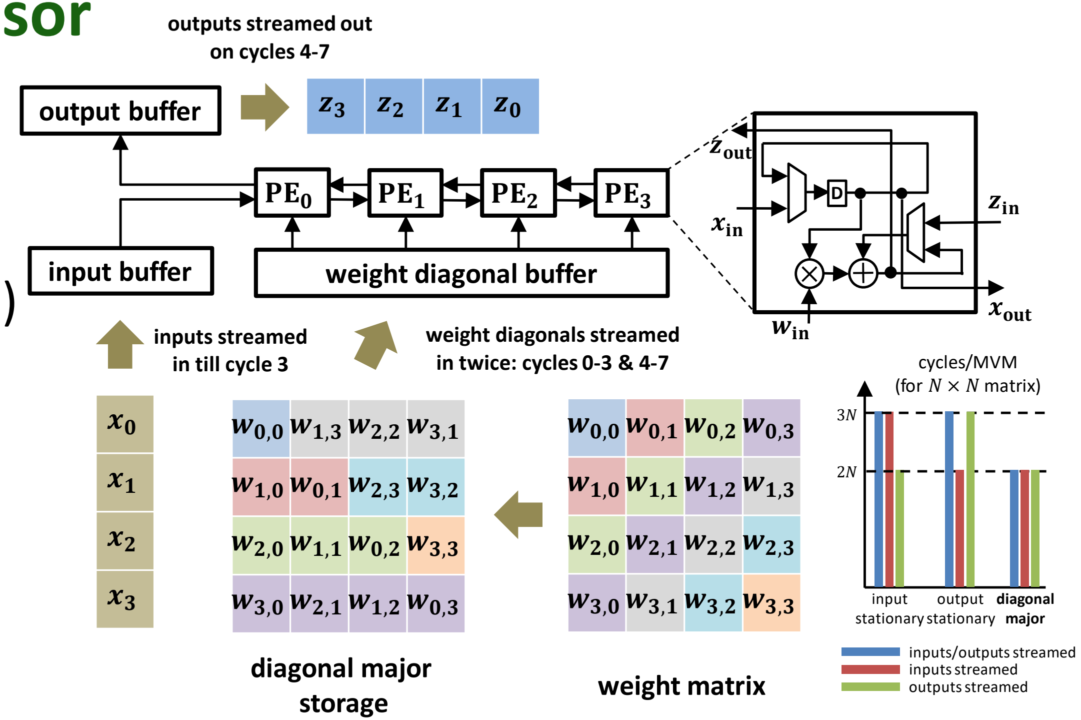


Outline

- Motivation and Background
- Recurrent Attention Model for KWS
- **Implementation**
  - Chip Architecture
  - Sparsity-aware IMC Block
  - **DM^{2}VM Digital Block**
- Measurement Results
- Summary

DM^{2}VM: Digital Processor

- Array of 64 8b MAC PEs
- 6kB of SRAM for weight storage
- Flexible support (fc1, fc2, fc5, fc6)
- Designed to minimize idle cycles when inputs/outputs are streamed in/out
- Completes an $N \times M$ MVM in a fixed number $N + M$ of cycles



outputs streamed out on cycles 4-7

inputs streamed in till cycle 3

weight diagonals streamed in twice: cycles 0-3 & 4-7

diagonal major storage

$w_{0,0}$	$w_{1,3}$	$w_{2,2}$	$w_{3,1}$
$w_{1,0}$	$w_{0,1}$	$w_{2,3}$	$w_{3,2}$
$w_{2,0}$	$w_{1,1}$	$w_{0,2}$	$w_{3,3}$
$w_{3,0}$	$w_{2,1}$	$w_{1,2}$	$w_{0,3}$

weight matrix

$w_{0,0}$	$w_{0,1}$	$w_{0,2}$	$w_{0,3}$
$w_{1,0}$	$w_{1,1}$	$w_{1,2}$	$w_{1,3}$
$w_{2,0}$	$w_{2,1}$	$w_{2,2}$	$w_{2,3}$
$w_{3,0}$	$w_{3,1}$	$w_{3,2}$	$w_{3,3}$

Principle of the diagonal major MVM (DM^{2}VM) processor for a 4×4 FC layer

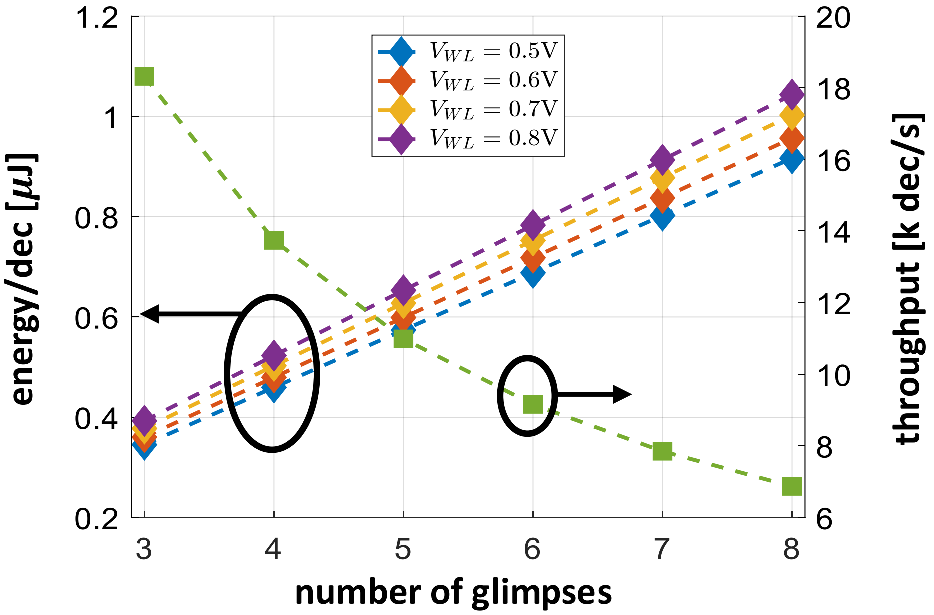
Outline

- Motivation and Background
- Recurrent Attention Model for KWS
- Implementation
  - Chip Architecture
  - Sparsity-aware IMC Block
  - DM^{2}VM Digital Block
- **Measurement Results**
- Summary

System Performance

- Energy/throughput tunable by varying V_{WL} and number of glimpses
- Measured results per glimpse:

Energy/glimpse	Latency/glimpse
0.11μJ	$18.2\mu s$

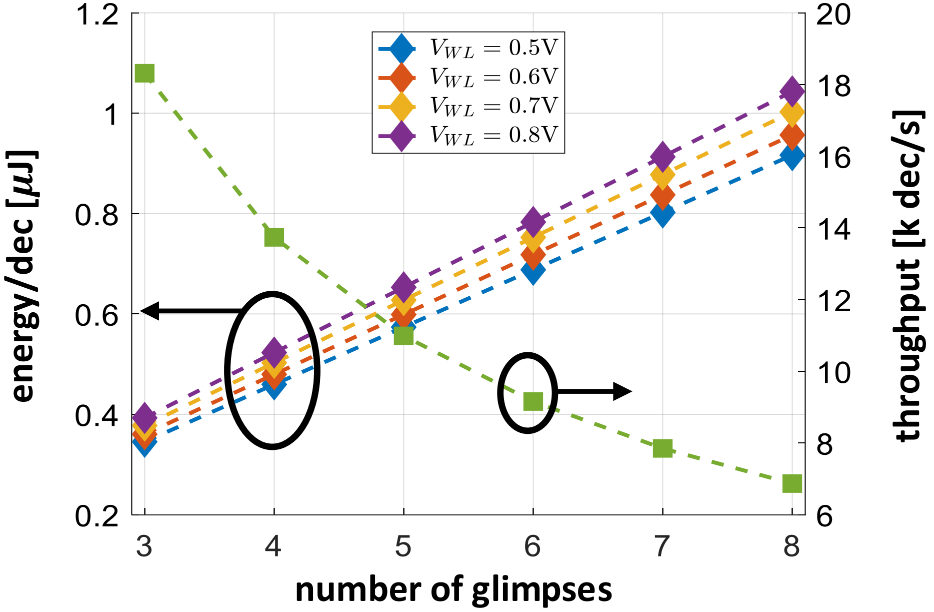


System Performance

- Energy/throughput tunable by varying V_{WL} and number of glimpses

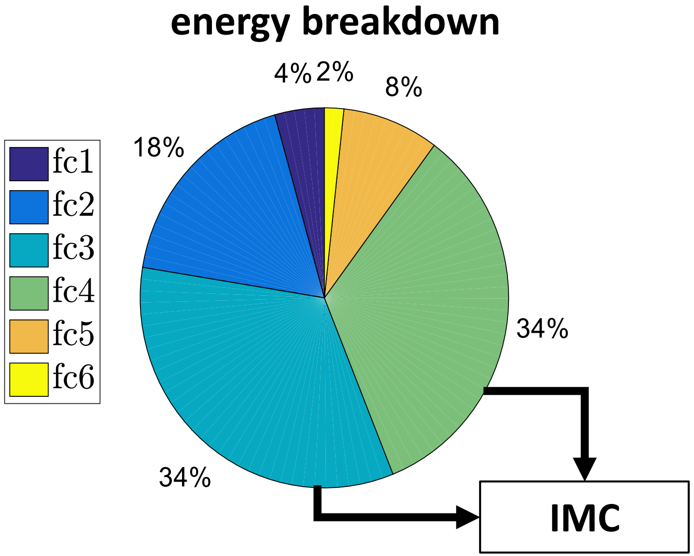
- Measured results per glimpse:

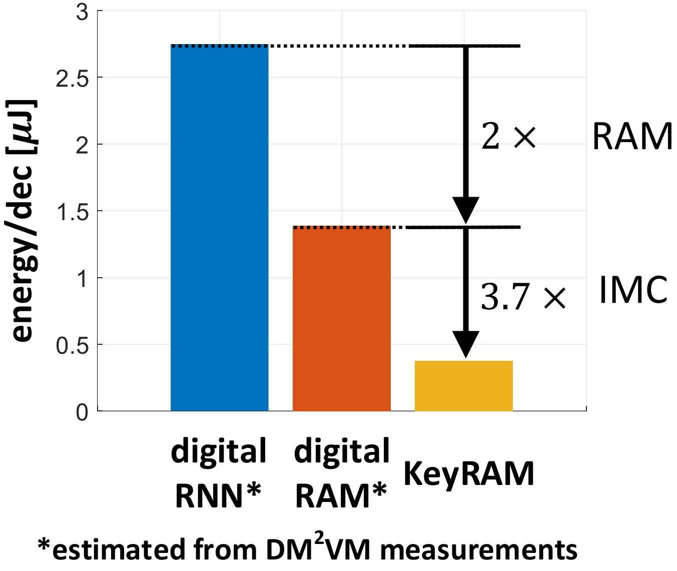
Energy/glimpse	Latency/glimpse
0.11μJ	$18.2\mu s$



1000× faster than a typical human reaction time

Energy Measurements

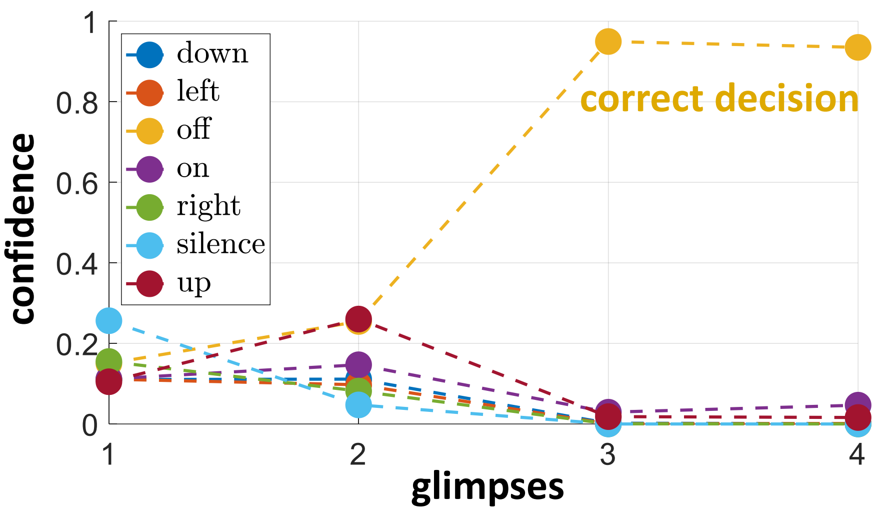




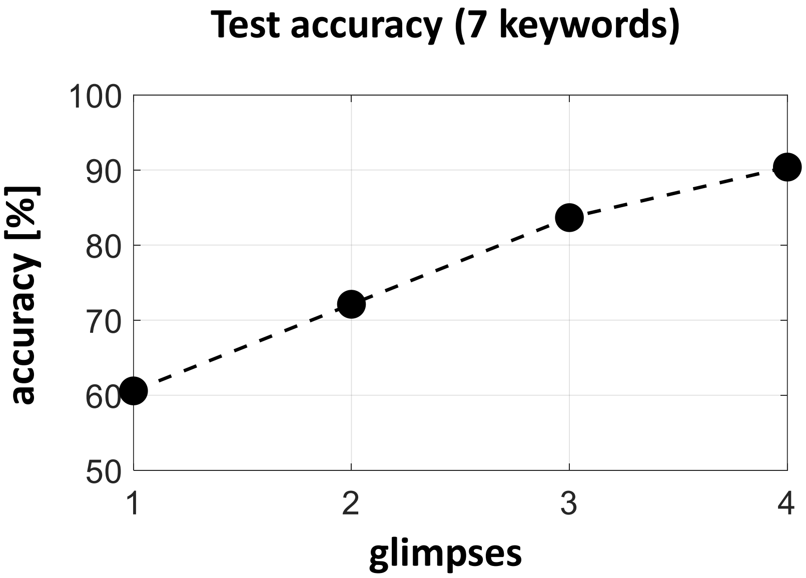
IMC: consumes 68% of the total energy, and implements 89% of computations

7.4 × better energy/dec compared to a digital RNN implementation

Measured Classification on Google Speech



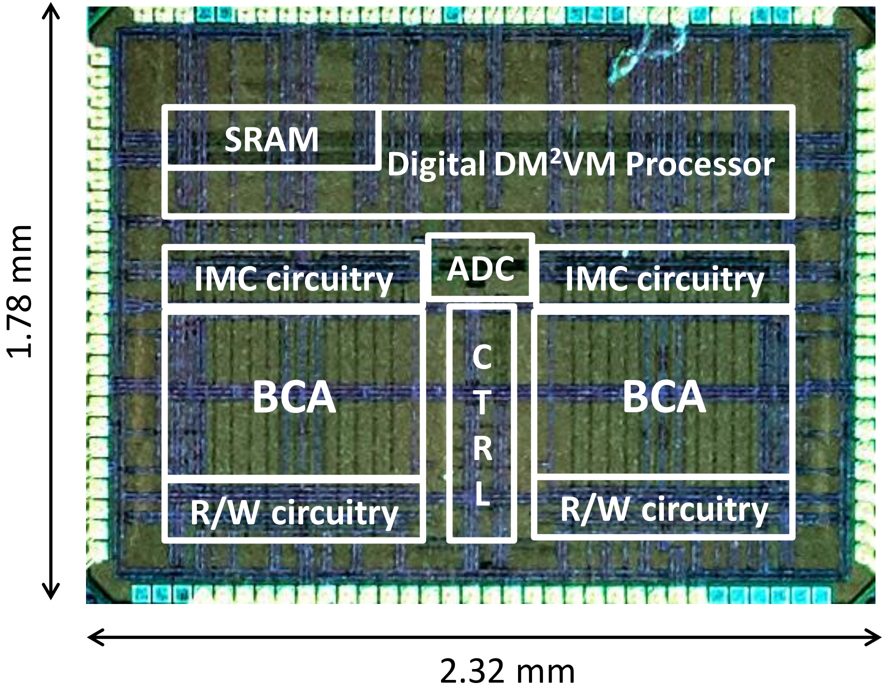
correct classification of one keyword after three glimpses



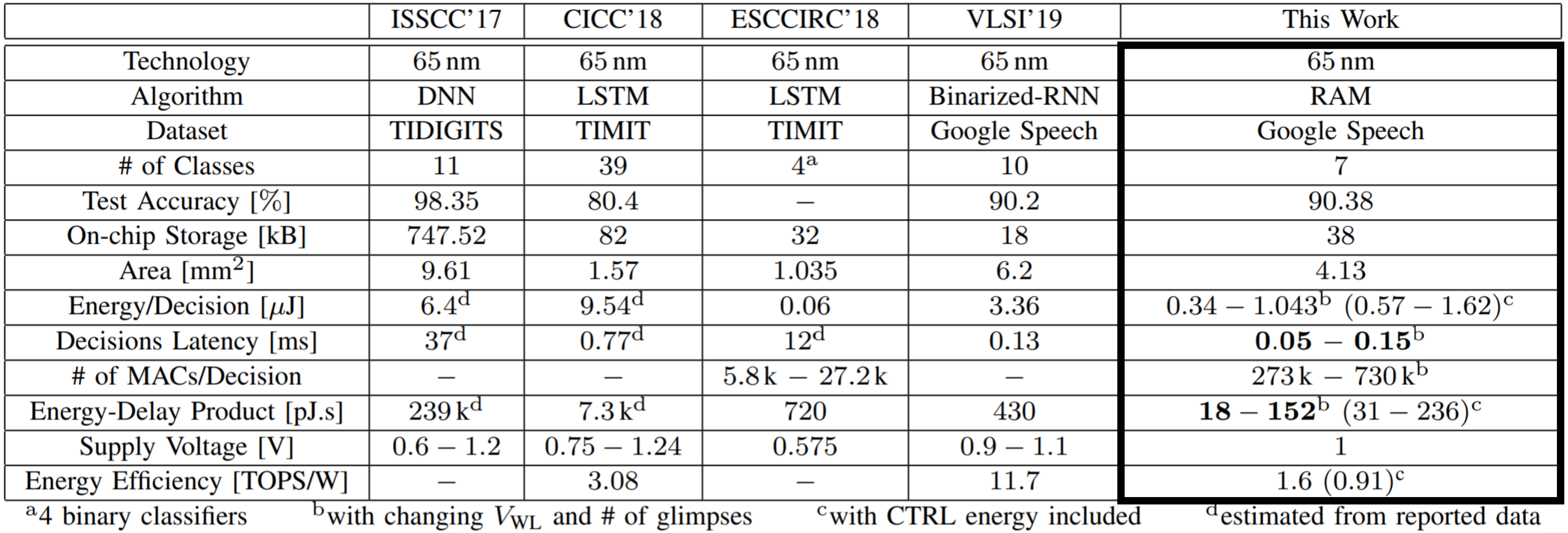
test set accuracy increases with number of glimpses

Chip Micrograph

Technology	65nm
Die Size	1.78mm × 2.32 mm
Memory Capacity	38kB
Nominal Supply	1.0V
CTRL Frequency	1GHz
Latency	0.05ms – 0.15ms
Energy/dec	0.34μJ – 1.043μJ
Algorithm	RAM



Comparison with State-of-the-art

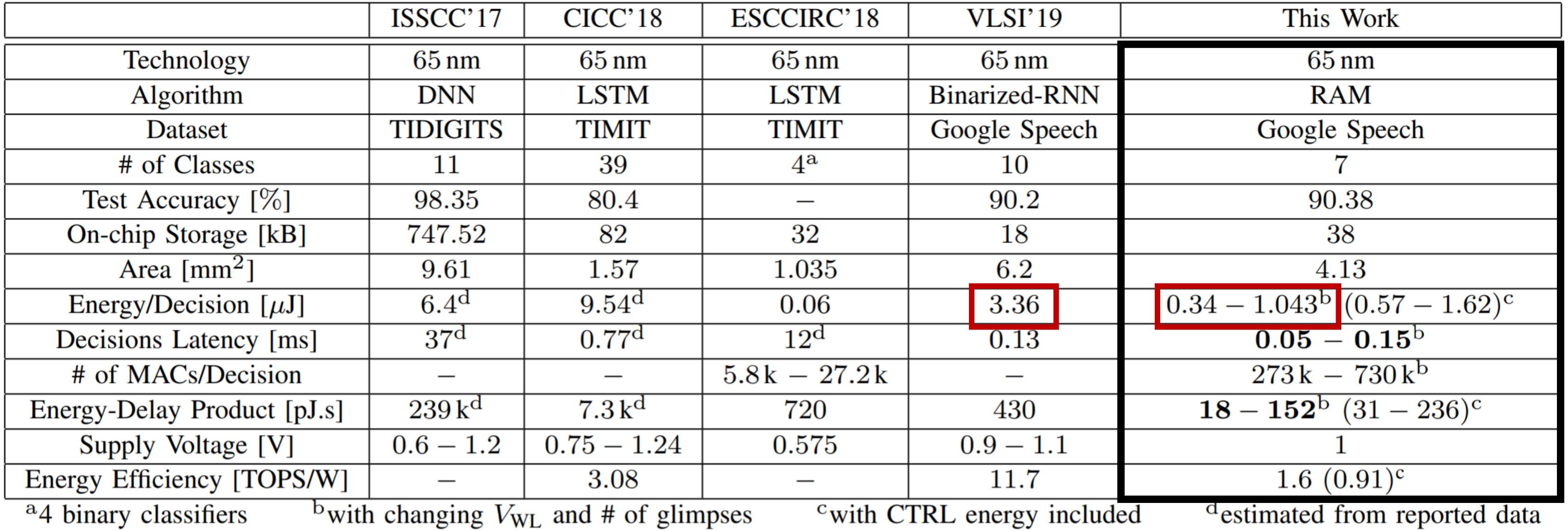


	ISSCC'17	CICC'18	ESCCIRC'18	VLSI'19	This Work
Technology	65 nm	65 nm	65 nm	65 nm	65 nm
Algorithm	DNN	LSTM	LSTM	Binarized-RNN	RAM
Dataset	TIDIGITS	TIMIT	TIMIT	Google Speech	Google Speech
# of Classes	11	39	4[a]	10	7
Test Accuracy [%]	98.35	80.4	–	90.2	90.38
On-chip Storage [kB]	747.52	82	32	18	38
Area [mm^2]	9.61	1.57	1.035	6.2	4.13
Energy/Decision [μJ]	6.4[d]	9.54[d]	0.06	3.36	0.34 – 1.043[b] (0.57 – 1.62)[c]
Decisions Latency [ms]	37[d]	0.77[d]	12[d]	0.13	**0.05 – 0.15**[b]
# of MACs/Decision	–	–	5.8 k – 27.2 k	–	273 k – 730 k[b]
Energy-Delay Product [pJ.s]	239 k[d]	7.3 k[d]	720	430	**18 – 152**[b] (31 – 236)[c]
Supply Voltage [V]	0.6 – 1.2	0.75 – 1.24	0.575	0.9 – 1.1	1
Energy Efficiency [TOPS/W]	–	3.08	–	11.7	1.6 (0.91)[c]

[a] 4 binary classifiers [b] with changing V_{WL} and # of glimpses [c] with CTRL energy included [d] estimated from reported data

- **Lowest reported decision latency**
- **More than 23 × reduction in EDP**

Comparison with State-of-the-art



	ISSCC'17	CICC'18	ESCCIRC'18	VLSI'19	This Work
Technology	65 nm	65 nm	65 nm	65 nm	65 nm
Algorithm	DNN	LSTM	LSTM	Binarized-RNN	RAM
Dataset	TIDIGITS	TIMIT	TIMIT	Google Speech	Google Speech
# of Classes	11	39	4[a]	10	7
Test Accuracy [%]	98.35	80.4	–	90.2	90.38
On-chip Storage [kB]	747.52	82	32	18	38
Area [mm^2]	9.61	1.57	1.035	6.2	4.13
Energy/Decision [μJ]	6.4[d]	9.54[d]	0.06	3.36	0.34 – 1.043[b] (0.57 – 1.62)[c]
Decisions Latency [ms]	37[d]	0.77[d]	12[d]	0.13	**0.05 – 0.15**[b]
# of MACs/Decision	–	–	5.8 k – 27.2 k	–	273 k – 730 k[b]
Energy-Delay Product [pJ.s]	239 k[d]	7.3 k[d]	720	430	**18 – 152**[b] (31 – 236)[c]
Supply Voltage [V]	0.6 – 1.2	0.75 – 1.24	0.575	0.9 – 1.1	1
Energy Efficiency [TOPS/W]	–	3.08	–	11.7	1.6 (0.91)[c]

[a] 4 binary classifiers [b] with changing V_{WL} and # of glimpses [c] with CTRL energy included [d] estimated from reported data

- **Lowest reported decision latency**
- **More than 23 × reduction in EDP**

3 × – 10 × reduction in energy/decision

Outline

- Motivation and Background
- Recurrent Attention Model for KWS
- Implementation
  - Chip Architecture
  - Sparsity-aware IMC Block
  - DM^{2}VM Digital Block
- Measurement Results
- **Summary**

Summary

- Energy efficient and low latency KWS systems are of utmost importance

- We adopt an algorithm-hardware co-design approach by proposing:
    - Novel classification algorithm for KWS using RAM
    - Sparsity-aware IMC-based computations for energy efficient dot product operations

- KeyRAM: a classifier IC in 65nm for KWS achieving state-of-the-art decision latency of $50\mu s$ with $< 0.5\mu$J/decision

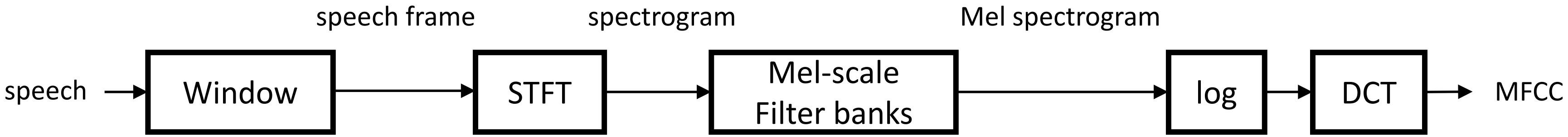
Acknowledgements

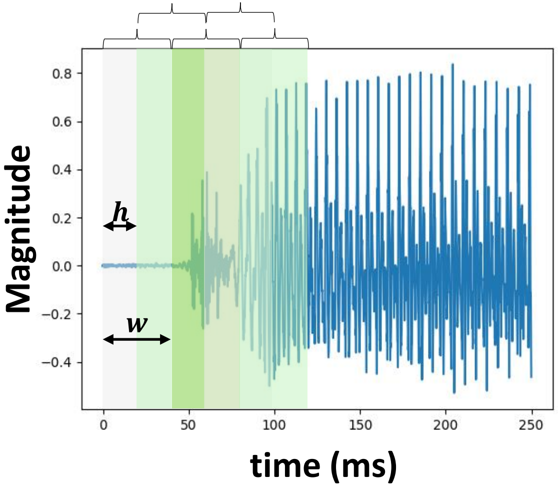
This work was sponsored by the AFRL and DARPA under agreement FA8650-18-2-7866 as part of the FRANC program.

Thank you

Backup Slides

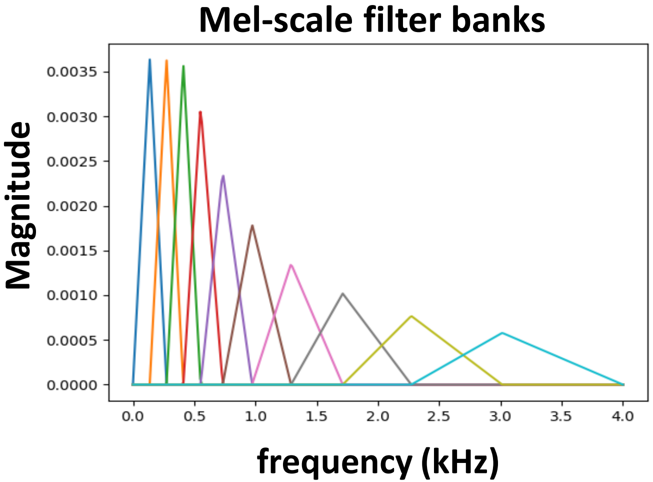
Mel-frequency Cepstral Coefficients (MFCC)





h: hop length (20ms)

w: window length(40ms)



10 Mel features

$f_{min} = 20\text{Hz}\ \ f_{max} = 2\text{kHz}$