MASR: A Modular Accelerator for Sparse RNNs

Udit Gupta, Brandon Reagen,
Lillian Pentecost, Marco Donato, Thierry Tambe
Alexander M. Rush, Gu-Yeon Wei, David Brooks

Harvard University
In this talk

Parallelism

Sparsity
In this talk

Parallelism

Sparsity

Custom sparse encoding
In this talk

Parallelism

Sparsity

Sparsity + Parallelism

Custom sparse encoding

Area  Energy  Perf.
In this talk

Sparsity + Parallelism

Sparsity

Parallelism

Custom sparse encoding

Area  Energy  Perf.
RNNs can revolutionize interactions with tech

Applications → Model

Recurrent Neural Networks

Text To Speech
Must deploy RNNs onto resource constrained HW
RNNs levy high inference cost

Large memory footprint

High compute footprint

High energy footprint

Tens of MBs for ASR

Tens of GFLOPs for ASR

Billions of memory accesses for ASR
Inference cost of ASR RNNs

Inputs are represent as a sequence of waveforms

DeepSpeech 2 Topology
Inference cost of ASR RNNs

Inputs are represent as a sequence of waveforms
Inference cost of ASR RNNs

Inputs are represent as a sequence of waveforms

Speech Inputs

DeepSpeech 2 Topology
Inference cost of ASR RNNs

Inputs are represent as a sequence of waveforms

DeepSpeech 2 Topology
Inference cost of ASR RNNs

Transcribed Speech

FC (29)

RNN (800)

RNN (800)

RNN (800)

RNN (800)

Conv2D

Speech Inputs

DeepSpeech 2 Topology

Over 98% of parameters found in recurrent layers (over 30 million, 50MB)

Inputs are represented as a sequence of waveforms
RNNs cost scales with input length (timesteps)
RNNs cost scales with input length (timesteps)

With increasing number of timesteps:
- Number of matrix-vector operations increases (FLOPs)
- Activation storage increases (area)
RNNs pose unique challenges

Solutions for CNNs optimize static weights

AlexNet

VGG16

ResNet34

Memory consumption (%)
RNNs pose unique challenges

Solutions for CNNs optimize static weights

RNN memory consumption dominated by dynamic activation requiring unique solutions
Limitations of current sparse DNN accelerators

**EIE: Efficient Inference Engine on Compressed DNNs**
Song Han, et. al.
ISCA, 2016, citation count: 909

Reduces weight footprint by \(3x\)

Does not compress activations (up to \(3x\) savings)
Limitations of current sparse DNN accelerators

EIE: Efficient Inference Engine on Compressed DNNs
Song Han, et. al.
ISCA, 2016, citation count: 909

Reduces weight footprint by 3x
Does not compress activations (up to 3x savings)

Does not scale (over 2x savings at high parallelism)
Proposed solution: MASR

<table>
<thead>
<tr>
<th>Problem</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Large memory footprint – static weights and dynamic activations</td>
<td>Logic centric sparse encoding</td>
</tr>
</tbody>
</table>
Proposed solution: MASR

<table>
<thead>
<tr>
<th>Problem</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Large memory footprint – static weights and dynamic activations</td>
<td>Logic centric sparse encoding</td>
</tr>
<tr>
<td>Irregular, sparse computation makes parallelism hard</td>
<td>Scalable sparse encoding architecture</td>
</tr>
<tr>
<td></td>
<td>Accelerator to exploit parallelism</td>
</tr>
</tbody>
</table>
Proposed solution: MASR

<table>
<thead>
<tr>
<th>Problem</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Large memory footprint – static weights and dynamic activations</td>
<td>Logic centric sparse encoding</td>
</tr>
<tr>
<td>Irregular, sparse computation makes parallelism hard</td>
<td>Scalable sparse encoding architecture</td>
</tr>
<tr>
<td>High performance with parallelism and irregularity is hard</td>
<td>Accelerator to exploit parallelism</td>
</tr>
<tr>
<td></td>
<td>Work stealing for load balancing</td>
</tr>
</tbody>
</table>
Proposed solution: MASR

<table>
<thead>
<tr>
<th>Problem</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Large memory footprint – static weights and dynamic activations</td>
<td>Logic centric sparse encoding</td>
</tr>
<tr>
<td>Irregular, sparse computation makes parallelism hard</td>
<td>Scalable sparse encoding architecture</td>
</tr>
<tr>
<td>High performance with parallelism and irregularity is hard</td>
<td>Work stealing for load balancing</td>
</tr>
</tbody>
</table>
Encoding techniques to exploit sparsity

Weight Matrix

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Compressed Sparse Row

<table>
<thead>
<tr>
<th>Row pointers</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 2 1</td>
</tr>
<tr>
<td>2 3 0 1 2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Col pointers</th>
<th>Weights</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Current state-of-the-art
Song Han, et. al. ISCA, 2016
Encoding techniques to exploit sparsity

Pressures memory system (2 pointers/1 weight)

Static weight encoding computed offline
Activations generated at run-time; uncompressed

Current state-of-the-art
Song Han, et. al. ISCA, 2016
Encoding techniques to exploit sparsity

Memory centric

Logic centric

Pressures memory system (2 pointers/1 weight)

Static weight encoding computed offline
Activations generated at run-time; uncompressed

Current state-of-the-art
Song Han, et. al. ISCA, 2016

Proposed solution
Encoding techniques to exploit sparsity

**Memory centric**

- Weight Matrix
- Compressed Sparse Row

Pressures memory system (2 pointers/1 weight)

**Static** weight encoding computed offline
Activations generated at run-time; uncompressed

**Logic centric**

- Sparse encoding binary mask

Relieves memory pressure (single pointer)

Compute sparse address at run-time!
Weights and activations are compressed

Current state-of-the-art
Song Han, et. al. ISCA, 2016

Proposed solution

```
0 0 1 1
1 1 0 0
0 0 1 0
```
MASR’s logic centric sparse encoding

Compute address of non-zero weight and activation stored in compressed format

<table>
<thead>
<tr>
<th>Weights</th>
<th>Mask</th>
<th>Compressed</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0.1 -0.3</td>
<td>0 0 1 1</td>
<td>0.1 -0.3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Weights</th>
<th>Mask</th>
<th>Compressed</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1 0.5 0 1.2</td>
<td>1 1 0 1</td>
<td>0.1 0.5 1.2</td>
</tr>
</tbody>
</table>

Activations
MASR’s logic centric sparse encoding

Compute address of non-zero weight and activation stored in compressed format

Weights

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th>0.1</th>
<th>-0.3</th>
</tr>
</thead>
</table>

Mask

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th>1</th>
<th>1</th>
</tr>
</thead>
</table>

Compressed

| 0.1 | -0.3 |

Activations

| 0.1 | 0.5 | 0   | 1.2  |

| 1    | 1    | 0   | 1    |

| 0.1 | 0.5 | 1.2 |

Mask

| 1    | 1    | 0   | 1    |
1. Compute when weight and activation are both non-zero

MASR’s logic centric sparse encoding

Compute address of non-zero weight and activation stored in compressed format
MASR’s logic centric sparse encoding

Compute address of non-zero weight and activation stored in compressed format

1. Compute when weight and activation are both non-zero

2. Find next non-zero pair (leading non-zero detect)
MASR’s logic centric sparse encoding

Compute address of non-zero weight and activation stored in compressed format

1. Compute when weight and activation are both non-zero

2. Find next non-zero pair (leading non-zero detect)

3. Evaluate address of non-zero weight and activation (population count)
MASR’s logic centric sparse encoding
Compute address of non-zero weight and activation stored in compressed format

1. Compute when weight and activation are both non-zero
2. Find next non-zero pair (leading non-zero detect)
3. Evaluate address of non-zero weight and activation (population count)
MASR’s logic centric sparse encoding

Can compute address of sparse weight/activation stored compactly in memory!

**Takeaways**

3x memory savings from weight compression

3x additional memory savings from activation compression

![Diagram showing sparse encoding process](diagram.png)
Proposed solution: MUSR

<table>
<thead>
<tr>
<th>Problem</th>
<th>Solution</th>
<th>Optimizes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Large memory footprint – static weights and dynamic activations</td>
<td>Logic centric sparse encoding</td>
<td>Area</td>
</tr>
<tr>
<td>Irregular, sparse computation makes parallelism hard</td>
<td>Scalable sparse encoding architecture</td>
<td></td>
</tr>
<tr>
<td>High performance with parallelism and irregularity is hard</td>
<td>Accelerator to exploit parallelism</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Work stealing for load balancing</td>
<td></td>
</tr>
</tbody>
</table>
Memory centric encodings do not scale

Song Han, et. al. ISCA, 2016
Memory centric encodings do not scale

Song Han, et. al. ISCA, 2016
Memory centric encodings do not scale

Row pointers

Row pointers

Row pointers

Duplicating memory is expensive!

Song Han, et. al. ISCA, 2016
Proposed: parallelize logic centric encoding

Single MASR Lane
Proposed: parallelize logic centric encoding
Proposed: parallelize logic centric encoding

Duplicating logic is cheap!
MASR’s sparse encoding improves scalability

Takeaways

Scalable sparse encoding and architecture

• Enables highly parallel execution with varying number of MACs/PEs
## Proposed solution: MARS

<table>
<thead>
<tr>
<th>Problem</th>
<th>Solution</th>
<th>Optimizes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Large memory footprint – static weights and dynamic activations</td>
<td>Logic centric sparse encoding</td>
<td>Area</td>
</tr>
<tr>
<td>Irregular, sparse computation makes parallelism hard</td>
<td>Scalable sparse encoding architecture</td>
<td></td>
</tr>
<tr>
<td>High performance with parallelism and irregularity is hard</td>
<td>Work stealing for load balancing</td>
<td></td>
</tr>
</tbody>
</table>
Parallelism within matrix-vector multiplication

Input activations

Weight matrix

Output activations

\[
\begin{bmatrix}
    x_0 \\
    \vdots \\
    x_7
\end{bmatrix}
\times
\begin{bmatrix}
    w_{0,1} & w_{0,2} & w_{0,3} & w_{0,4} \\
    w_{7,1} & w_{7,2} & w_{7,3} & w_{7,4} \\
    \vdots & \vdots & \vdots & \vdots \\
    w_{7,13} & w_{7,14} & w_{7,15} & w_{7,16}
\end{bmatrix}
= 
\begin{bmatrix}
    y_0 \\
    \vdots \\
    y_{16}
\end{bmatrix}
\]
MASR micro-architecture: Parallelizing across input and output neurons

- **Horizontal lanes** parallelize output neurons
- **Vertical lanes** parallelize input neurons

PEs composed of neighboring horizontal lanes:
- Share activation register file (**area, power, load time**)
- Private weight and mask SRAM within lane (decoupled to enable **high-bandwidth** access)
MASR: design space exploration
MASR: design space exploration
MASR: design space exploration

Recurrent Neural Network
MASR: design space exploration

Recurrent Neural Network

1GHz
Cycle accurate simulation

16nmFinFet
MASR: design space exploration

1GHz

16nmFinFet

Cycle accurate simulation

Design space exploration

Recurrent Neural Network
MASR: Performance, Energy, Area tradeoffs

MASR is 2 *orders* of magnitude faster than CPU

Fits our *on-chip area* target for mobile devices
MASR: Performance, Energy, Area tradeoffs

MASR is 2 orders of magnitude faster than CPU

Fits our on-chip area target for mobile devices
MASR: Performance, Energy, Area tradeoffs

MASR is 2 orders of magnitude faster than CPU

Fits our on-chip area target for mobile devices
MASR: Performance, Energy, Area tradeoffs

MASR is 2 orders of magnitude faster than CPU

Fits our on-chip area target for mobile devices

Takeaways

Parallelism can be configured to target:
- High-performance
- Energy-efficiency
- Area-efficiency
### Proposed solution: MASR

<table>
<thead>
<tr>
<th>Problem</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Large memory footprint – static weights and dynamic activations</td>
<td>Logic centric sparse encoding</td>
</tr>
<tr>
<td>High performance with parallelism and irregularity is hard</td>
<td>Scalable sparse encoding architecture Accelerator to exploit parallelism</td>
</tr>
<tr>
<td>Irregular, sparse computation makes parallelism hard</td>
<td>Work stealing for load balancing</td>
</tr>
</tbody>
</table>

- **Optimizes**
  - Area
  - Performance
  - Area
  - Energy
Further investigating sources of inefficiency

Increasing number of parallel MACs/lanes from 64 to 1024 (16x), improves performance by 5.5x
Increasing number of parallel MACs/lanes from 64 to 1024 (16x), improves performance by 5.5x

30% MAC utilization
Remainder spent on stalls/idles due to load imbalance
Further investigating sources of inefficiency

Increasing number of parallel MACs/lanes from 64 to 1024 (16x), improves performance by 5.5x

30% MAC utilization
Remainder spent on stalls/idles due to load imbalance

Some lanes get 1.5x more work (non-zeros) to process
Dynamic load balancing
Lanes that finish early can steal work from neighboring lanes that are straggling behind
Dynamic load balancing

Lanes that finish early can steal work from neighboring lanes that are straggling behind

Horizontal load balancing requires duplicating weights
Dynamic load balancing
Lanes that finish early can steal work from neighboring lanes that are straggling behind

Horizontal load balancing requires duplicating weights

Vertical load balancing requires duplicating weights and activation register files
Dynamic load balancing

Vertical load balancing better targets load imbalance in dynamic activations
Up to 1.9x speedup (LANESx1024)
Requires duplicating 10% weight storage and activation register files
Proposed solution: MASR

<table>
<thead>
<tr>
<th>Problem</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Large memory footprint – static weights and dynamic activations</td>
<td>Logic centric sparse encoding</td>
</tr>
<tr>
<td>Irregular, sparse computation makes parallelism hard</td>
<td>Scalable sparse encoding architecture Accelerator to exploit parallelism</td>
</tr>
<tr>
<td>High performance with parallelism and irregularity is hard</td>
<td>Work stealing for load balancing</td>
</tr>
</tbody>
</table>

Over state-of-the-art, MASR provides:

- 3x area
- 3x energy
- 2x perf
Scalable acceleration of sparse RNNs is possible!

Stay tuned...
MASR: A Modular Accelerator for Sparse RNNs

Udit Gupta, Brandon Reagen,
Lillian Pentecost, Marco Donato, Thierry Tambe
Alexander M. Rush, Gu-Yeon Wei, David Brooks

ugupta@g.harvard.edu

Thanks for listening!