The Dark Side of DNN Pruning

Reza Yazdani
Jose-Maria Arnau
Marc Riera
Antonio González
DNN Pruning

• Efficient reduction of DNN size

✔ Higher performance
✔ Significant energy-saving
✔ Ultra-low power
✔ Lower area
Side-Effect of DNN Pruning

- Lack of confidence in DNN classification
  - Speech network of acoustic modeling

![Probability Distribution Graph]

The Dark Side of DNN Pruning, Session 9A, Wednesday June 6th, ISCA'18
Confidence Issue

- DNN dependent applications
  - Automatic Speech Recognition (ASR)
  - Machine Translation

- Example: ASR evaluation for pruned DNN
Outline

• Motivation

• DNN pruning & Confidence loss

• ASR using pruned DNN

• Accelerator's baseline

• Efficient design with DNN pruning

• Experimental results

• Conclusions
DNN Pruning: Accuracy

- Maintaining top-5 accuracy
Loss of Confidence

- The more the pruning rate in DNNs, the lower the classification probability
Outline

- Motivation
- DNN pruning & Confidence loss
- ASR using pruned DNN
- Accelerator's baseline
- Efficient design with DNN pruning
- Experimental results
- Conclusions
ASR

- ASR systems include two phases
  - DNN: computes probabilities of different phonemes at each frame

Frame i
\( \{X_1, X_2, \ldots, X_n\} \)
ASR

- ASR systems include two phases
  - DNN: computes probabilities of different phonemes at each frame
ASR

- ASR systems include two phases
  - DNN: computes probabilities of different phonemes at each frame
ASR

- ASR systems include two phases
  - DNN: computes probabilities of different phonemes at each frame
  - Viterbi search: explores WFST based on DNN scores
ASR Evaluation

- Viterbi search under pruned DNN model

Frame 2

DNN Scores of Frame 2

Cost of Best Path

Beam

The Dark Side of DNN Pruning, Session 9A, Wednesday June 6th, ISCA'18
ASR Evaluation

- Viterbi search under pruned DNN model
Viterbi Workload

- Increase in Viterbi's search activity
Outline

- Motivation
- DNN pruning & Confidence loss
- ASR using pruned DNN
- Accelerator's baseline
- Efficient design with DNN pruning
- Experimental results
- Conclusions
Hardware Baseline

- UNFOLD: state-of-the-art Viterbi accelerator
Hardware Baseline

- UNFOLD: state-of-the-art Viterbi accelerator
Hardware Baseline

- UNFOLD: state-of-the-art Viterbi accelerator
Hardware Baseline

• UNFOLD: state-of-the-art Viterbi accelerator
Hardware Baseline

- UNFOLD: state-of-the-art Viterbi accelerator
Hardware Baseline

- **UNFOLD**: state-of-the-art Viterbi accelerator
Hardware Baseline

- **UNFOLD**: state-of-the-art Viterbi accelerator

### Hash Bottlenecks
- Collision handling
  - Backup buffer
- Overflows
  - Overflow buffer
- Access delay
  - Backup
  - Overflow

---

**On-the-fly Accelerator**
- Hash 1 (current frame)
- Hash 2 (next frame)
- State Cache
- Arc Cache
- Offset Lookup Table
- State Issuer
- Arc Issuer
- Acoustic-likelihood Issuer
- Hypothesis Issuer
- Likelihood Evaluation
- Acoustic Likelihood Buffer
- Word Lattice Cache

---

**Main Memory**
- WFST States
- WFST Arcs
- Word Lattice
- Overflow Buffer
Outline

- Motivation
- DNN pruning & Confidence loss
- ASR using pruned DNN
- Accelerator's baseline
- Efficient design with DNN pruning
- Experimental results
- Conclusions
Efficient Hash Design

• Keeping the best N hypotheses at each frame
  – Known as Histogram Pruning
Efficient Hash Design

• Keeping the best N hypotheses at each frame
  – Known as Histogram Pruning

• Implementation issue
  – Sorting tokens at every frame
  – Expensive: $O(m\times\log(m))$ for $m$ hypotheses
Efficient Hash Design

- Keeping the best N hypotheses at each frame
  - Known as Histogram Pruning
- Implementation issue
  - Sorting tokens at every frame
  - Expensive: $O(m\times\log(m))$ for m hypotheses
- Our scheme
  - Loosely keeping N-best using hash mechanism
Efficient Hash Design

- Direct-mapped
Efficient Hash Design

- Direct-mapped
- Way-Associative
Efficient Hash Design

- Our scheme efficiency
Efficient Hash Design

• Way-associative main challenge
  – Replace when set is full
  – Finding hypothesis with max cost
Efficient Hash Design

• Way-associative main challenge
  – Replace when set is full
  – Finding hypothesis with max cost

• Our solution
  – Store index of each set based on max-heap
  – Replace with the root of tree
  – Updating max-heap fits in one cycle
Outline

- Motivation
- DNN pruning & Confidence loss
- ASR using pruned DNN
- Accelerator's baseline
- Efficient design with DNN pruning
- Experimental results
- Conclusions
Evaluation Methodology

- Cycle-accurate simulation of DNN and Viterbi
- Model accelerator's components in hardware
  - Verilog implementation of logic parts
  - Synthesized by design compiler
  - Cacti: Cache and memory components
  - Micron: main memory
- Combine simulation results with hardware models
  - Decoding time
  - Decoding power and energy consumption
  - Accelerator's area usage
## Accelerator's Parameters

- **DNN and Viterbi accelerators**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Tiles</td>
<td>4</td>
</tr>
<tr>
<td>Number of 32-bit multipliers</td>
<td>128</td>
</tr>
<tr>
<td>Number of 32-bit adders</td>
<td>128</td>
</tr>
<tr>
<td>Weights Buffer</td>
<td>18 MB</td>
</tr>
<tr>
<td>I/O Buffer</td>
<td>32KB, 64 Banks - 2RD and 1WR ports</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>State Cache</td>
<td>256 KB, 4-way, 64 B/line</td>
</tr>
<tr>
<td>Arc Cache</td>
<td>768 KB, 8-way, 64 B/line</td>
</tr>
<tr>
<td>Word Lattice Cache</td>
<td>128 KB, 2-way, 64 B/line</td>
</tr>
<tr>
<td>Acoustic Likelihood Buffer</td>
<td>64 Kbytes</td>
</tr>
<tr>
<td>Hash Table</td>
<td>100KB, 1K entries, 6 FP comparators</td>
</tr>
<tr>
<td>Memory Controller</td>
<td>32 in-flight requests</td>
</tr>
<tr>
<td>Likelihood Evaluation Unit</td>
<td>4 FP adders, 2 FP comparators</td>
</tr>
</tbody>
</table>

The Dark Side of DNN Pruning, Session 9A, Wednesday June 6th, ISCA'18
Experiment Configs

- **Viterbi Search:**
  - Baseline: Unfold's design
  - Beam: reduce beam without changing baseline
  - N-Best: our proposal

- **DNN:**
  - Non-pruned version
  - Pruned version: 70%, 80% and 90% pruning
Experimental Results

- Decoding time
Experimental Results

- Decoding time
- Energy consumption
Experimental Results

- Decoding time
- Energy consumption
- Area usage: 10.74 mm² (2x reduction)
Outline

• Motivation
• DNN pruning & Confidence loss
• ASR using pruned DNN
• Accelerator's baseline
• Efficient design with DNN pruning
• Experimental results
• Conclusions
Conclusions

- Major side effect of DNN pruning
  - Confidence loss: top-1's low likelihood
- DNN pruning in ASR systems
  - 20% confidence loss, 33% slowdown
- Our solution: A novel Viterbi accelerator
  - Resilient to DNN pruning
  - Less search activity while maintaining accuracy
- Compared to state-of-art ASR accelerated system
  - 9x energy-saving, 4.5x speedup, 2x area reduction
The Dark Side of DNN Pruning

Reza Yazdani
Jose-Maria Arnau

Marc Riera
Antonio González

Departament d'Arquitectura de Computadors

UNIVERSITAT POLITÈCNICA DE CATALUNYA