Real-Time Mobile Acceleration of DNNs: From Computer Vision to Medical Applications

Hongjia Li, Geng Yuan, Wei Niu, Yuxuan Cai, Mengshu Sun, Zhengang Li, Bin Ren, Xue Lin, Yanzhi Wang
Outline

• Background and Motivation

• Proposed Framework
  • A Unified Fine-grained Block-based Pruning Scheme
  • Unified Regularization Pruning Algorithm
  • Compiler Optimizations

• Evaluation
  • Object Detection
  • Action Recognition
  • Medical Diagnosis
Background and Motivation

• Deep learning is everywhere
Background and Motivation

• Issues:
  • Model storage
  • Computation resource
    • Training and inference time
    • Power consumption

• Challenges and opportunities:
  • Mobile devices are rapidly becoming the central computer and carrier for deep learning tasks
  • Real-time execution requirements
Background and Motivation

• Model compression techniques

• Weight pruning
  • Unstructured pruning
  • Coarse-grained structured pruning
  • Pattern-based pruning
Background and Motivation

- Compiler-based DNN frameworks on mobile

Representative works
- TensorFlow-Lite (TFLite)
- Alibaba MNN (MNN)
- TVM
- PatDNN and PCONV

Sparse DNN models are not well supported

Only square and small convolution kernels used in $3 \times 3$ CONV layers are accelerated
Outline

• Background and Motivation
• Proposed Framework
  • A Unified Fine-grained Block-based Pruning Scheme
  • Unified Regularization Pruning Algorithm
  • Compiler Optimizations
• Evaluation
  • Object Detection
  • Action Recognition
  • Medical Diagnosis
A Unified Fine-grained Block-based Pruning Scheme

- Features:
  - Provide support for CONV layers with any kernel sizes and FC layers
  - Can be extended from 2D DNN models to 3D DNN models
A Unified Fine-grained Block-based Pruning Scheme

- Achieve high accuracy and hardware parallelism simultaneously

- High accuracy: fine-grained property of block-based pruning

- High hardware inference performance: proposed appropriate degree of structural regularity
Unified Regularization Pruning Algorithm

• The general reweighted pruning problem:

\[
\min_{W,b} \quad f(W; b) + \lambda \sum_{i=1}^{N} R(\alpha_i^{(t)}, W_i),
\]

• \(f(W; b)\) is the loss function of DNN.
• \(R(\cdot)\) denotes the regularization term used to generate model sparsity and the hyperparameter \(\lambda\) controls the trade-off between accuracy and sparsity.
• \(\alpha_i^{(t)}\) represents the collection of penalty values applied on the weights \(W_i\) for layer \(i\).
Unified Regularization Pruning Algorithm

• **2D DNNs:** the regularization term is

\[
R(\alpha_{ij}^{(t)}, W_i) = \sum_{j=1}^{K} \sum_{h=1}^{g^h_m} \sum_{w=1}^{g^h_n} \|\alpha_{ij}^{(t)} \circ [W_{ij}]_{h,w}\|_F^2,
\]

where \(\alpha_{ij}^{(t)}\) is updated by

\[
\alpha_{ij}^{(t)} = \frac{1}{\|W_{ij}\|_F^2 + \epsilon}.
\]

• **3D DNNs:** the regularization term is

\[
R(\alpha_{i,Gp,q}^{(t)}, W_i) = \sum_{p=1}^{P} \sum_{q=1}^{Q} \sum_{d=1}^{K_d} \sum_{h=1}^{K_h} \sum_{w=1}^{K_w} \|\alpha_{i,Gp,q}^{(t)} \circ [W_{i,Gp,q}]_{d,h,w}\|_F^2,
\]

where \(\alpha_{i,Gp,q}^{(t)}\) is updated by

\[
\alpha_{i,Gp,q}^{(t)} = \frac{1}{\|W_{i,Gp,q}\|_F^2 + \epsilon}.
\]
Compiler Optimizations

• Model weights is stored compactly.

• Computation reorder: reduce the branches within each thread and eliminate the load imbalance among threads.

• Incorporates auto-tuning approaches.
Outline

• Background and Motivation
• Proposed Framework
  • A Unified Fine-grained Block-based Pruning Scheme
  • Unified Regularization Pruning Algorithm
  • Compiler Optimizations
• Evaluation
  • Object Detection
  • Action Recognition
  • Medical Diagnosis
Object Detection

- YOLOv4 with 320×320 input size
- MS COCO dataset
- 64.36M weight parameters
- 35.8G FLOPs

Table 1: Accuracy and speed under different compression rates on COCO 2014 dataset.

<table>
<thead>
<tr>
<th>#Weight</th>
<th>#Weights Comp. Rate</th>
<th>#FLOPs</th>
<th>mAP</th>
<th>FPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>64.36M</td>
<td>1.0×</td>
<td>35.8G</td>
<td>57.3</td>
<td>3.5</td>
</tr>
<tr>
<td>16.11M</td>
<td>4.0×</td>
<td>10.48G</td>
<td>55.1</td>
<td>7.3</td>
</tr>
<tr>
<td>8.04M</td>
<td>8.1×</td>
<td>6.33G</td>
<td>51.4</td>
<td>11.5</td>
</tr>
<tr>
<td>6.37M</td>
<td>10.1×</td>
<td>5.48G</td>
<td>50.9</td>
<td>13</td>
</tr>
<tr>
<td>4.59M</td>
<td>14.0×</td>
<td>3.95G</td>
<td>49</td>
<td>17</td>
</tr>
</tbody>
</table>

Table 2: Comparison of different pruning schemes (objective detection).

<table>
<thead>
<tr>
<th>Pruning Scheme</th>
<th>#Weight</th>
<th>#Weights Comp. Rate</th>
<th>mAP</th>
<th>FPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unpruned</td>
<td>64.36M</td>
<td>1×</td>
<td>57.3</td>
<td>3.5</td>
</tr>
<tr>
<td>Unstructured</td>
<td>8.04M</td>
<td>8.09×</td>
<td>53.9</td>
<td>6.4</td>
</tr>
<tr>
<td>Structured</td>
<td>8.04M</td>
<td>8.09×</td>
<td>38.6</td>
<td>12.7</td>
</tr>
<tr>
<td>Ours</td>
<td>8.04M</td>
<td>8.09×</td>
<td>51.4</td>
<td>11.5</td>
</tr>
</tbody>
</table>
Object Detection

- The end-to-end inference latency comparison of our framework with representative mobile DNN inference acceleration frameworks
Action Recognition

- Three representative 3D CNN models
  - C3D, R(2+1)D, and S3D
- UCF101 dataset

<table>
<thead>
<tr>
<th>Model (Size)</th>
<th>Pruning Scheme</th>
<th>#FLOPs</th>
<th>#FLOPs Comp. Rate</th>
<th>Top-1 Acc. (%)</th>
<th>Latency (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>C3D (299MB)</td>
<td>Unpruned</td>
<td>77.2G</td>
<td>1×</td>
<td>81.6</td>
<td>488</td>
</tr>
<tr>
<td></td>
<td>Structured</td>
<td>30.4G</td>
<td>2.6×</td>
<td>79.3</td>
<td>178</td>
</tr>
<tr>
<td></td>
<td>Ours</td>
<td>30.4G</td>
<td>2.6×</td>
<td>80.5</td>
<td>192</td>
</tr>
<tr>
<td></td>
<td>Ours</td>
<td>22.0G</td>
<td>3.6×</td>
<td>80.2</td>
<td>142</td>
</tr>
<tr>
<td></td>
<td>Unpruned</td>
<td>81.5G</td>
<td>1×</td>
<td>94.0</td>
<td>513</td>
</tr>
<tr>
<td>R(2+1)D (120MB)</td>
<td>Structured</td>
<td>31.8G</td>
<td>2.6×</td>
<td>90.5</td>
<td>180</td>
</tr>
<tr>
<td></td>
<td>Ours</td>
<td>31.8G</td>
<td>2.6×</td>
<td>92.5</td>
<td>201</td>
</tr>
<tr>
<td></td>
<td>Ours</td>
<td>26.2G</td>
<td>3.2×</td>
<td>92.0</td>
<td>141</td>
</tr>
<tr>
<td>S3D (31MB)</td>
<td>Unpruned</td>
<td>49.4G</td>
<td>1×</td>
<td>90.6</td>
<td>565</td>
</tr>
<tr>
<td></td>
<td>Structured</td>
<td>24.8G</td>
<td>2.1×</td>
<td>88.2</td>
<td>325</td>
</tr>
<tr>
<td></td>
<td>Ours</td>
<td>24.8G</td>
<td>2.1×</td>
<td>90.2</td>
<td>342</td>
</tr>
<tr>
<td></td>
<td>Ours</td>
<td>20.6G</td>
<td>2.4×</td>
<td>89.1</td>
<td>293</td>
</tr>
</tbody>
</table>
## Action Recognition

<table>
<thead>
<tr>
<th>Latency (ms)</th>
<th>PyTorch (dense CPU)</th>
<th>MNN (dense CPU)</th>
<th>Ours (dense CPU)</th>
<th>Ours (dense GPU)</th>
<th>Ours (3.6x CPU)</th>
<th>Ours (3.6x GPU)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2544</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

![Action Recognition](image)
Medical Diagnosis

- YOLOv3-based model with 320×320 input size
- RSNA Pneumonia Detection Dataset

Table 4: Comparison of different pruning schemes (medical diagnosis).

<table>
<thead>
<tr>
<th>Pruning Schemes</th>
<th>#Weights Comp. Rate</th>
<th>#FLOPs</th>
<th>mAP</th>
<th>FPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unpruned</td>
<td>1×</td>
<td>38.63G</td>
<td>71.2</td>
<td>4.2</td>
</tr>
<tr>
<td>Unstructured</td>
<td>11.6×</td>
<td>3.63G</td>
<td>70.5</td>
<td>10.8</td>
</tr>
<tr>
<td>Structured</td>
<td>11.6×</td>
<td>3.63G</td>
<td>69.7</td>
<td>16.7</td>
</tr>
<tr>
<td>Ours</td>
<td>11.6×</td>
<td>3.63G</td>
<td>70.2</td>
<td>15.5</td>
</tr>
</tbody>
</table>
Thank you!