Hardware Friendly Deep Neural Network Pruning

Shaoyi Huang Yueying Liang Qianying Ren

UCONN

Outline

- Introduction
- Background
- Methods
- Experiments
- Conclusion

Introduction

1.Large-scale deep neural networks (DNNs) have achieved great success in various fields, there are increasing demands to deploy DNNs on resource-constrained devices.

2.However, there are many challenges to accommodate DNN models on hardware:

limited storage

limited computational speed

high demand of real-time inference

3.The core idea of model compression is to generate a much sparse model thus we could get acceleration in computation and reduction on space.

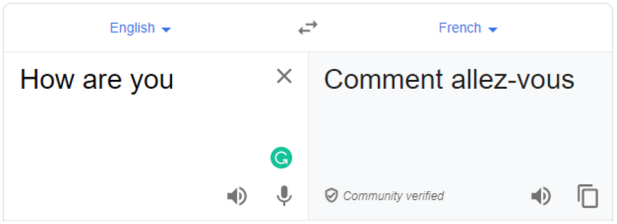
Introduction

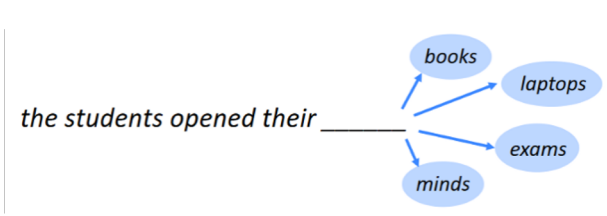
3.From the aspect of hardware friendly or not, weight pruning can be divided into two categories which are structured pruning (hardware friendly) and unstructured pruning (hardware unfriendly).

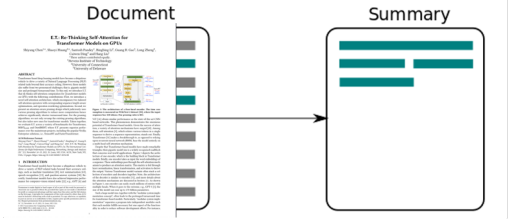
4.For structured pruning, we explored pruning methods such as row pruning, column pruning and whole block pruning while for the unstructured pruning, we conducted experiment on irregular pruning.

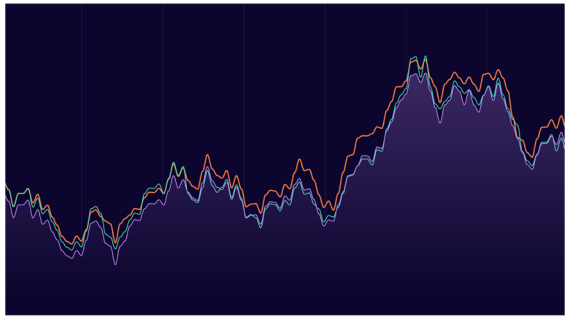
5.In this work, we will demonstrate the effectiveness of weight pruning in the fields of both CV and NLP.

Why software/hardware co-design









[1]. Girdhar, Rohit, et al. "Video action transformer network." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2019.

Why software/hardware co-design

Software:

- Increasing model size
- High demand of model performance

Hardware:

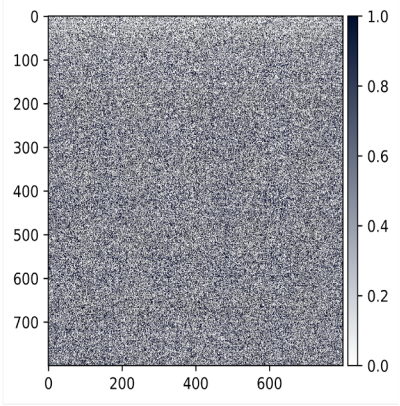
- Limited weighted storage
- Limited computational speed
- High demand of real-time inference

Hardware types:

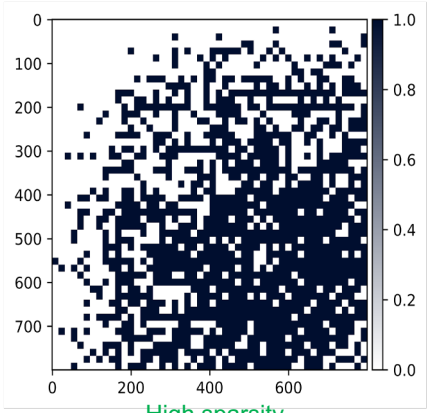
- FPGA
- GPU (Tensor core friendly pruning)
- ...

[1]. https://paperswithcode.com/sota/question-answering-on-squad20-dev
[2]. https://paperswithcode.com/sota/semantic-textual-similarity-on-mrpc

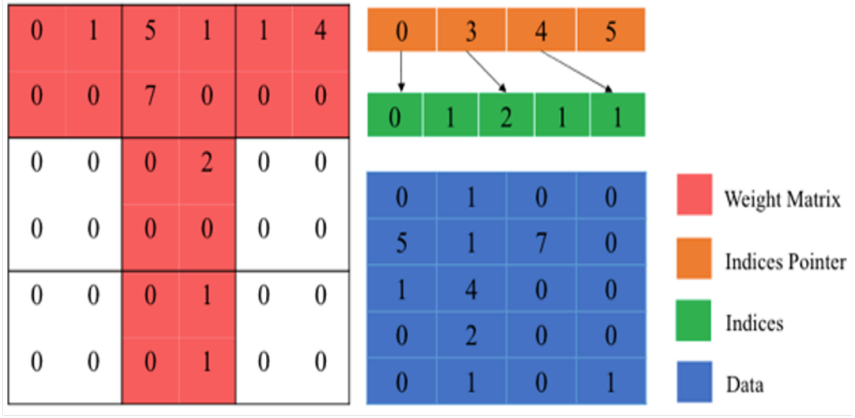
Why software/hardware co-design



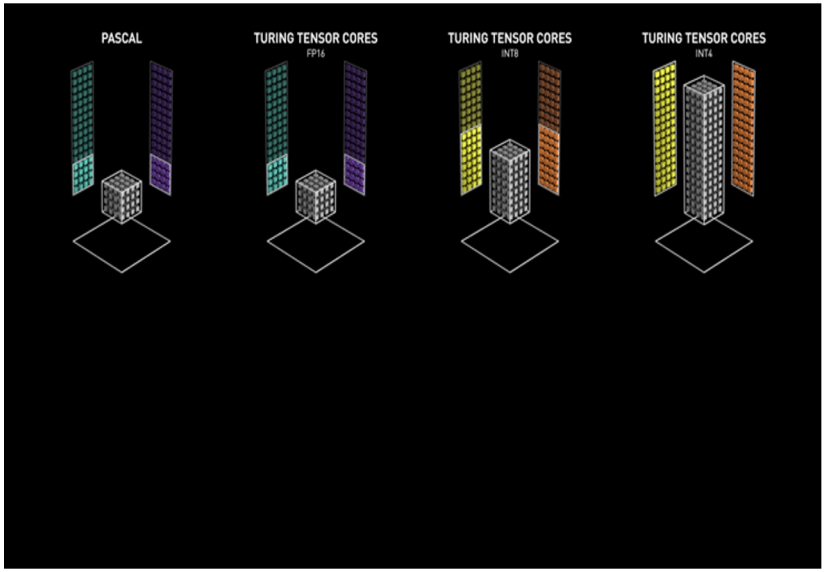
Very high sparsity
Hardware unfriendly

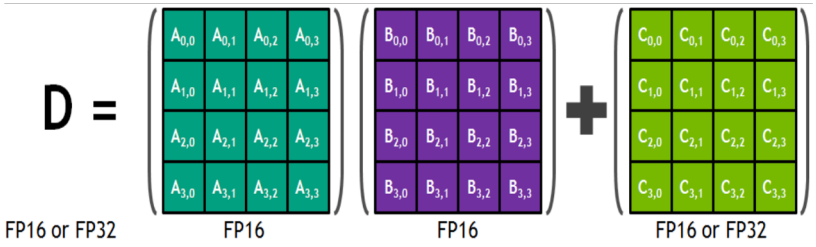


High sparsity
Hardware friendly



An example of BCSR format, where the block size is 2×2





D = **A** * **B** + **C**, where **A**, **B**, **C** and **D** are 4×4 matrices

Tensor core is 8× faster than the general cores.

https://developer.nvidia.com/blog/programming-tensor-cores-cuda-9/
https://www.nvidia.com/en-us/data-center/tensor-cores/

Efficient computing on sparse models

Irregular

0	0	6	0
0	7	0	2
0	8	0	0
5	2	0	0

Row

4	3	6	9
0	0	0	0
4	8	3	2
0	0	0	0

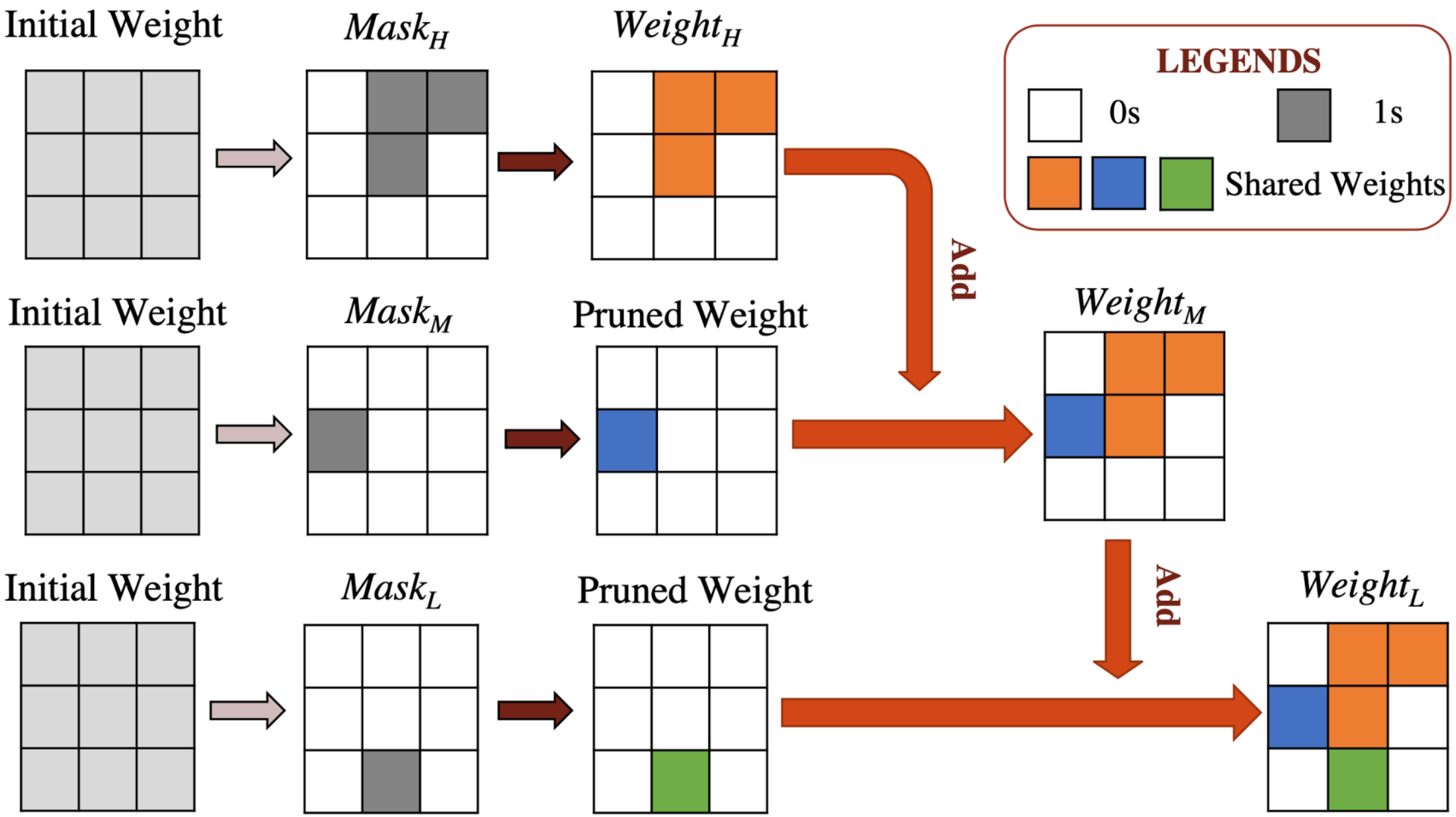
Column pruning

4	0	6	0
8	0	6	0
4	0	3	0
5	0	4	0

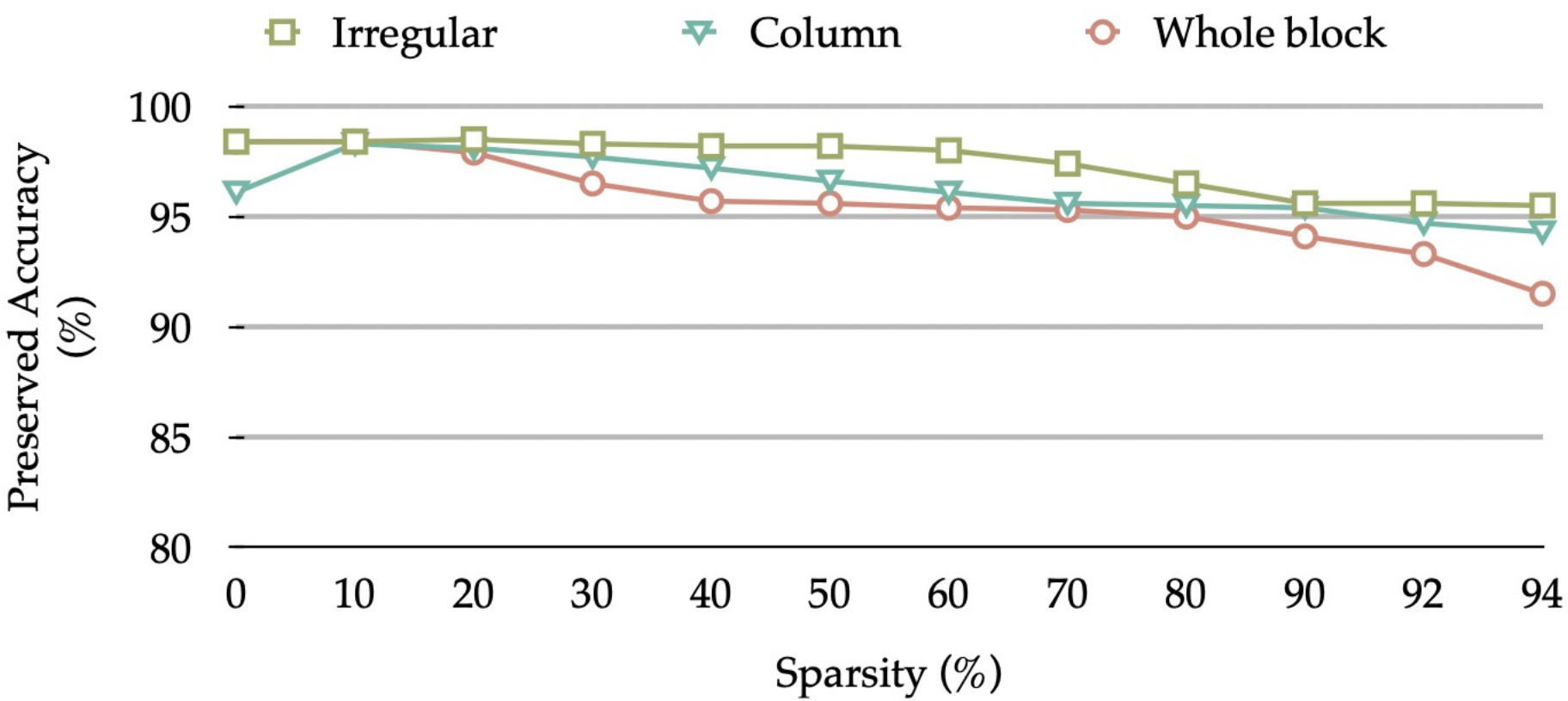
Whole block Pruning

0	0	6	9
0	0	6	2
4	8	0	0
5	2	0	0

Method


Initial Weight
$Mask_H$
$Weight_H$
LEGENDS
0s
1s
Shared Weights
Add
Initial Weight
$Mask_M$
Pruned Weight
$Weight_M$
Add
Initial Weight
$Mask_L$
Pruned Weight
$Weight_L$

Experiments



Conclusion

Pruning strategy has benefits on multiple-tasks in both hardware and software level, by accelerating inference process and saving computation costs.