INTERNATIONAL CONFERENCE ON PARALLEL PROCESSING
ICPP/2021/CHICAGO/USA
acm In-Cooperation
sighpc
AUGUST 9-12, 2021

Support Convolution of CNN with Compression Sparse Matrix Multiplication Flow in TVM

Hui-Hsin Liao[1], Chao-Lin Lee[1], Jenq-Kuen Lee[1], Wei-Chih Lai[2], Ming-Yu Hung[2], Chung-Wen Huang[2]

Department of Computer Science,

National Tsing Hua University, Hsinchu, Taiwan [1]

MediaTek Inc. [2]

Outline

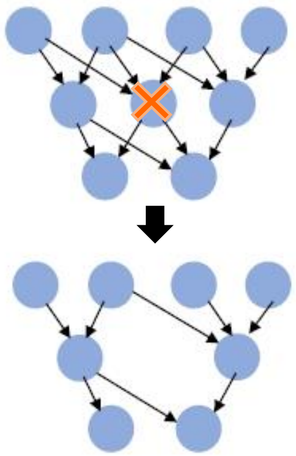
- Introduction and Background
- Im2col and GEMM
- Conversion Flow in TVM
- Experiments
- Conclusions

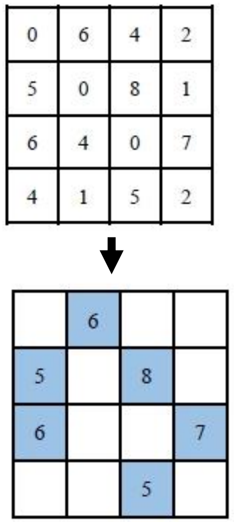
Introduction

- Deep learning is playing an important role nowadays.
- Lots previous researches tend to reduce convolutions overhead.
  - Direct convolution
  - Im2col and GEMM(General Matrix Multiplication)
  - Winograd
  - FFT(Fast Fourier Transformation)
- Our work try to build a general flow for different backends which target to sparse models.

Background – Sparse Model

- We collect sparse models from two different sources
  - ImageNet based model: SparseZoo
  - CIFAR10 based model: Prune with distiller from Intel AILab






```yaml
pruners:
  conv_pruner:
    class: 'AutomatedGradualPruner'
    initial_sparsity : 0.03
    final_sparsity: 0.82
    weights: ['features.module.3.weight',
              'features.module.13.weight',
              'features.module.18.weight']

policies:
  - pruner:
      instance_name : 'conv_pruner'
    starting_epoch: 1
    ending_epoch: 10
    frequency: 2
```

Example of pruning schedule

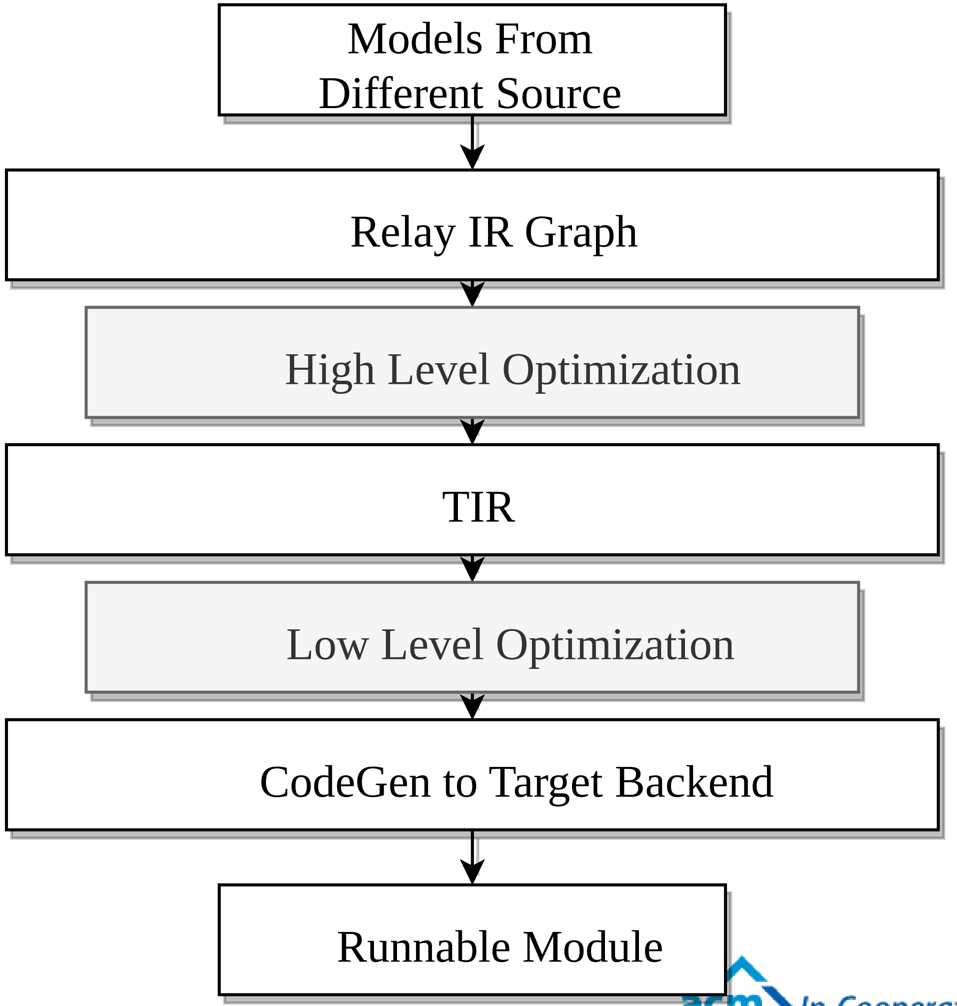
INTERNATIONAL CONFERENCE ON PARALLEL PROCESSING

acm In-Cooperation sighpc

Reference : distiller, Intel AILab, SparseZoo, Song Han et al., Learning both Weights and Connections for Efficient Neural Networks, NIPS, 2015

Background – TVM

- Deep learning inference framework
- Support models with different formats
- Support different target backends
- Two level optimization
  - Relay IR : Computation graph optimization
  - TIR : Decouple algorithm from schedule
    - Algorithm: What is computed
    - Schedule: Where and When and How it's computed

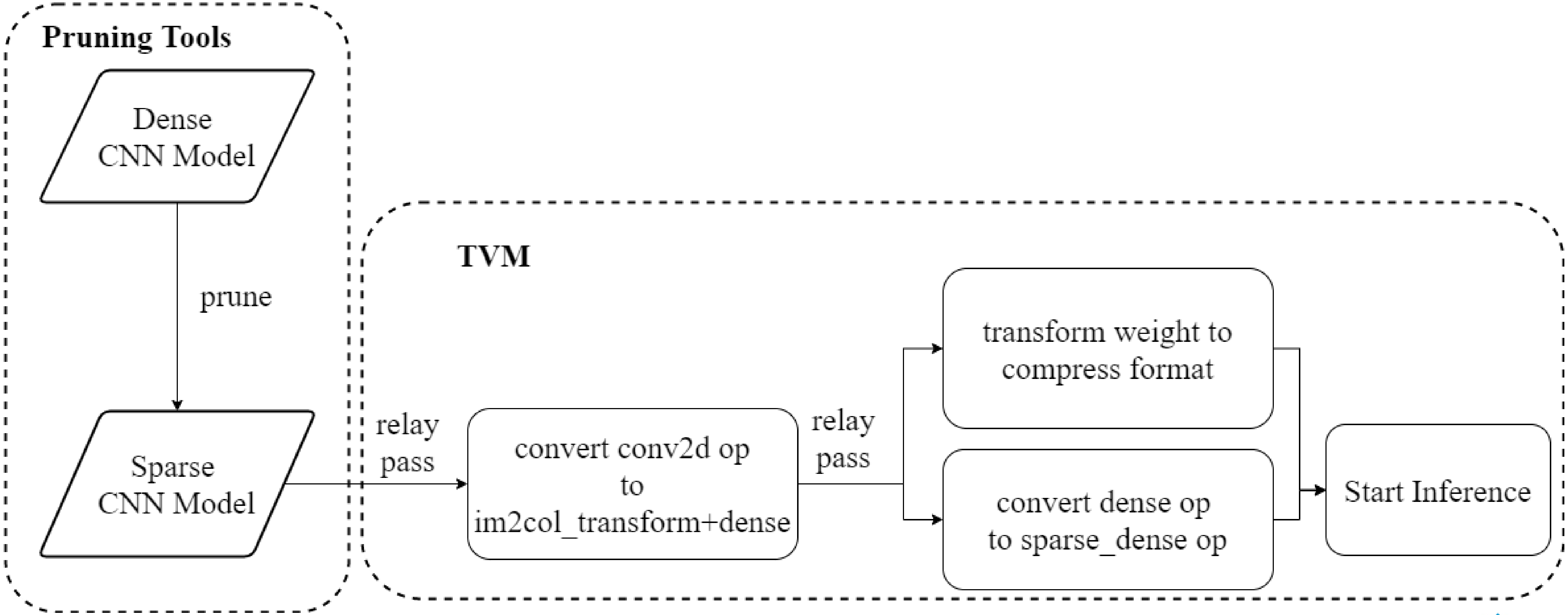


INTERNATIONAL CONFERENCE ON PARALLEL PROCESSING

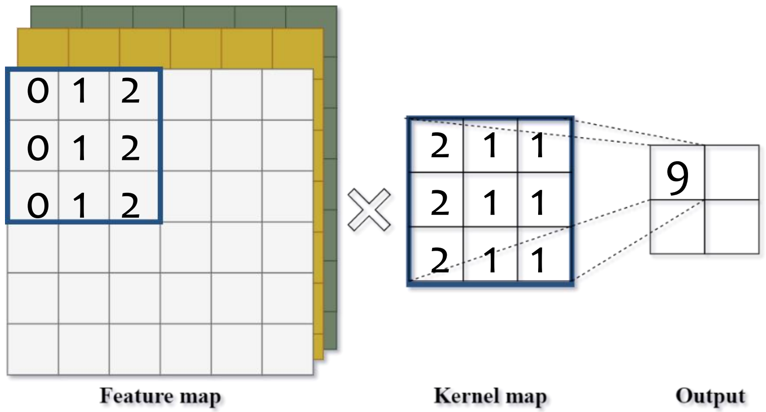
acm In-Cooperation sighpc

Reference : Apache TVM

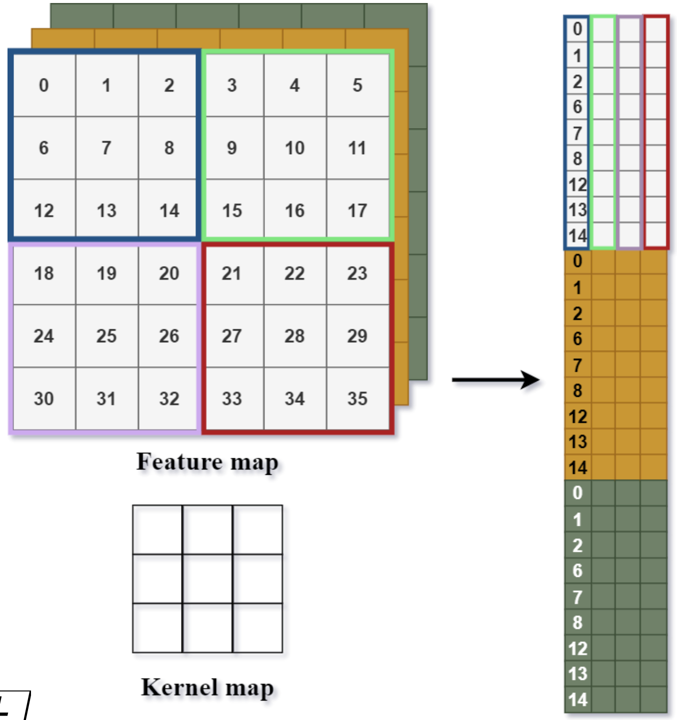
Overview of Our Flow

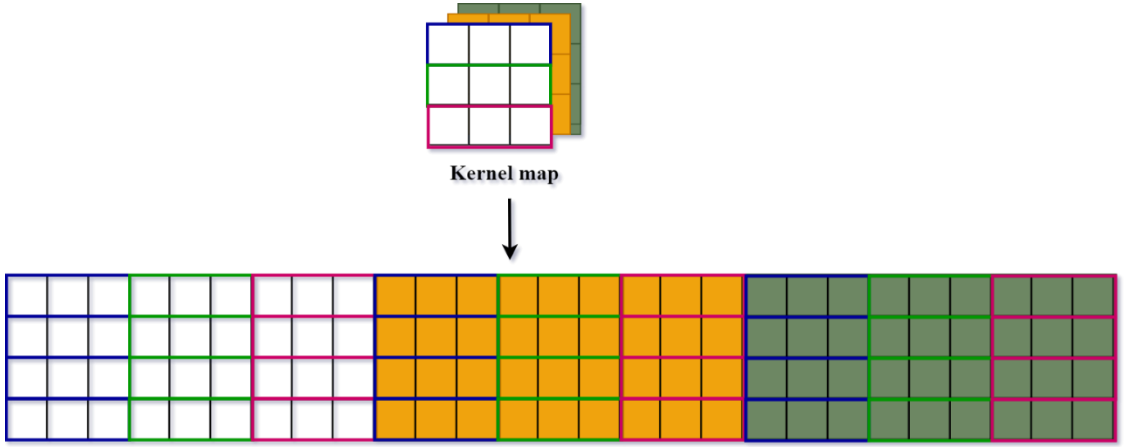


Im2col and GEMM



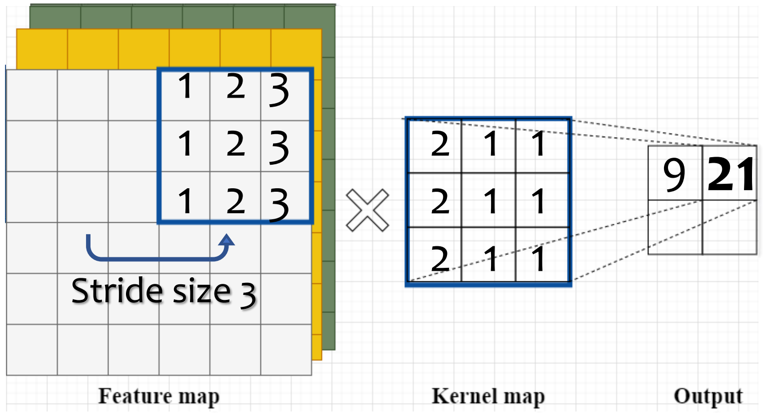
Direct convolution



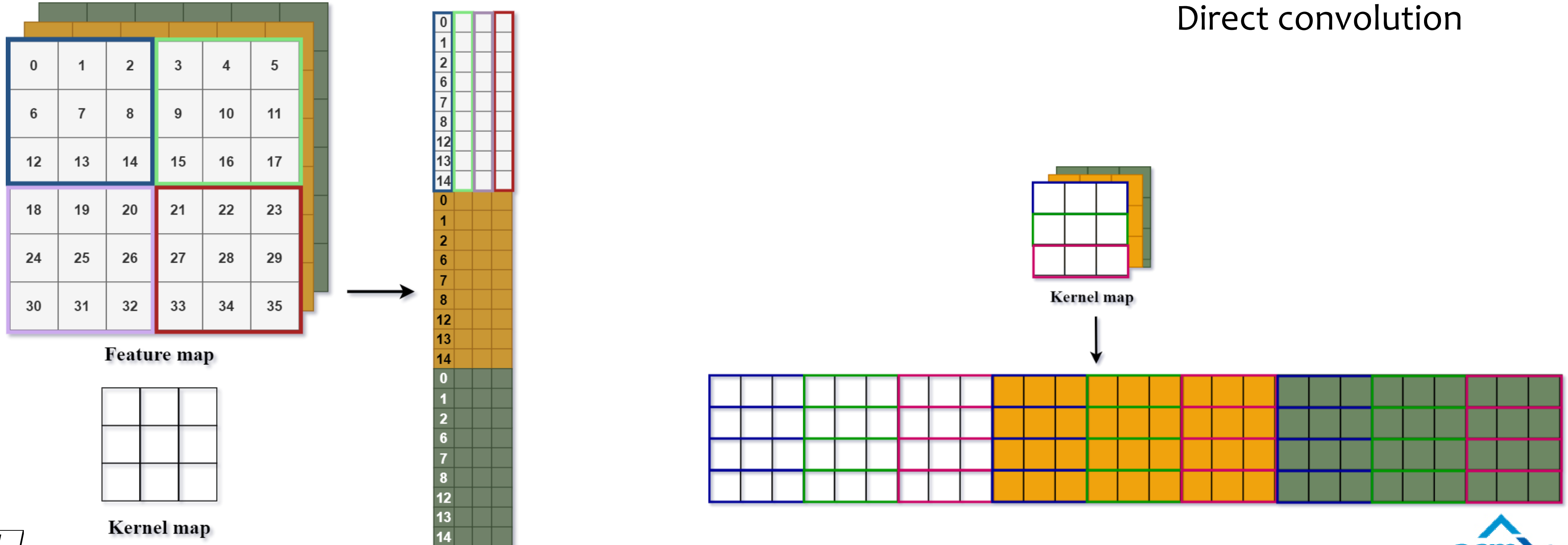


Im2col

Im2col and GEMM



Direct convolution



Im2col

Im2col and GEMM – Im2col Operator

- The new shape could be inferred by parameters of convolution.
- Transferred matrix shape

$$transIn_h = channel_{in} * k_h * k_w$$

$$transIn_w = batch_{in} * o_h * o_w$$

- Output shape

$$o_h = \lfloor (in_h + pad_h - ((k_h - 1) * dilation_h + 1))/stride_h \rfloor + 1$$

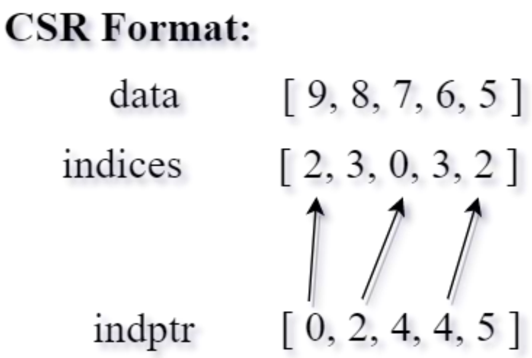
$$o_w = \lfloor (in_w + pad_w - ((k_w - 1) * dilation_w + 1))/stride_w \rfloor + 1$$

INTERNATIONAL CONFERENCE ON PARALLEL PROCESSING

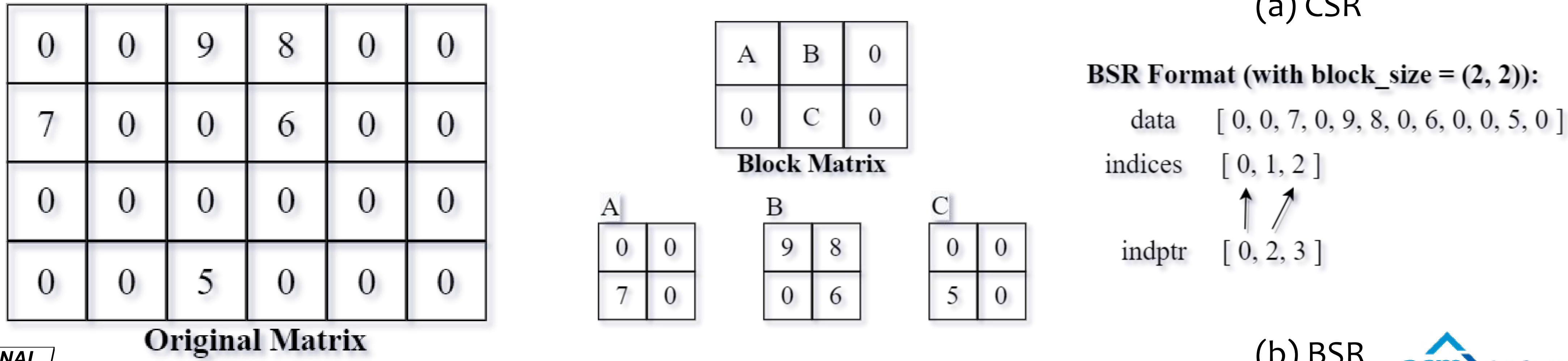
acm In-Cooperation sighpc

Im2col and GEMM – Compression Format

- A matrix in which lots of the elements are **zero.**
- Compression Formats :



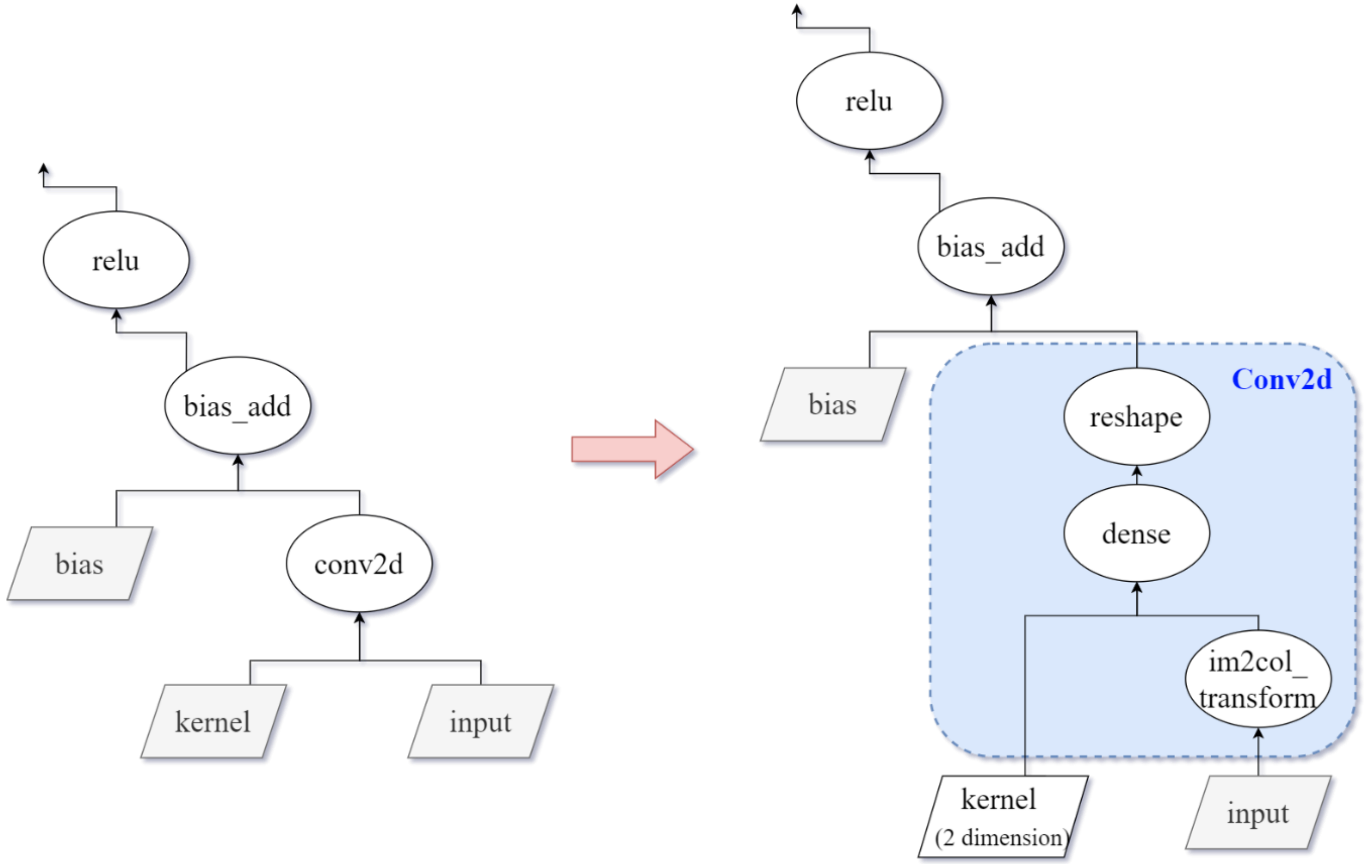
(a) CSR



(b) BSR

INTERNATIONAL CONFERENCE ON PARALLEL PROCESSING

Conversion Flow in TVM



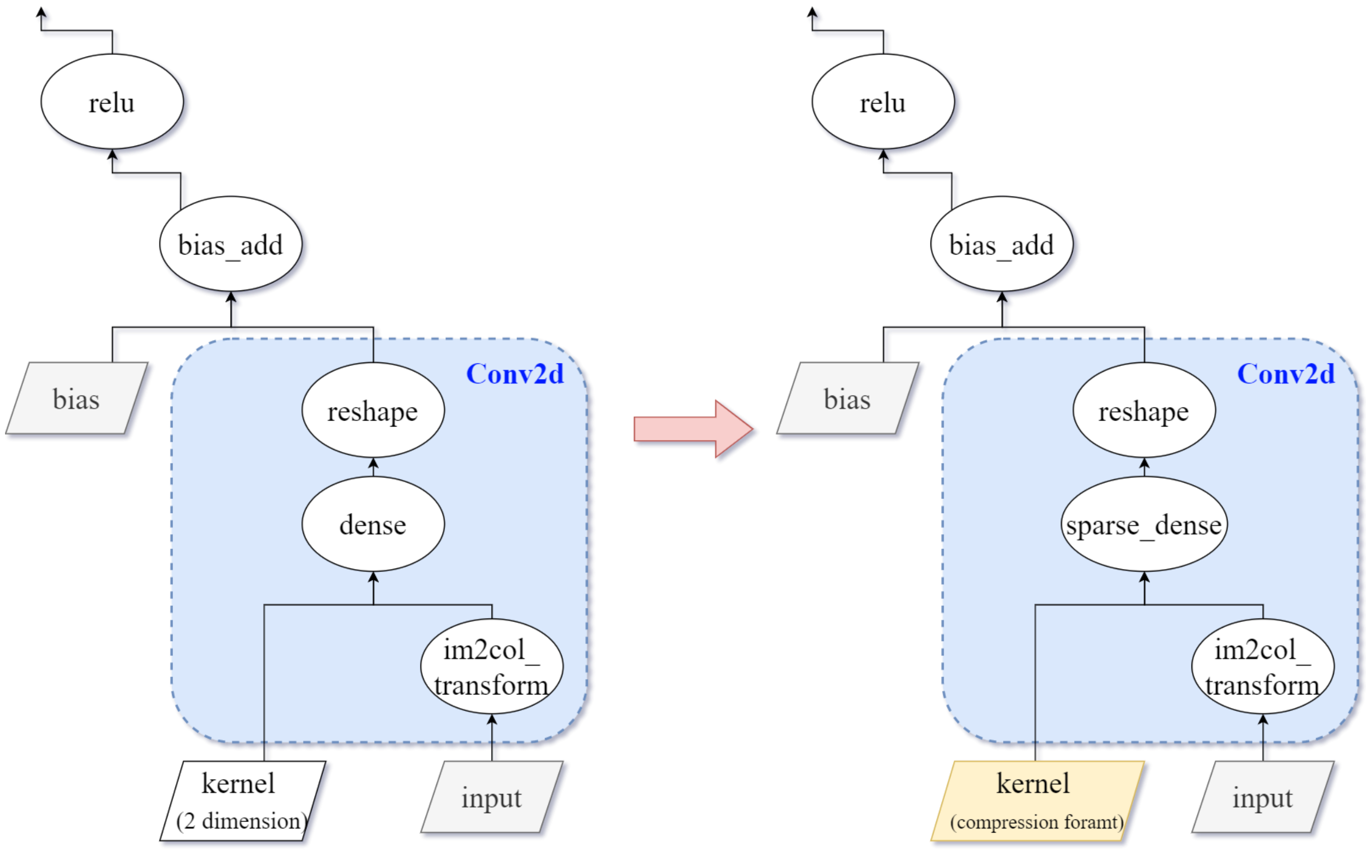
Algorithm 1: First conversion of relay graph

Input: Original relay graph
Output: Transformed relay graph
compress_list = []
while *Node != NULL* **do**
  if *Node type == Var && weight belongs to conv2d* **then**
    transform(weight)
    compress_list.append(weight.name)
    // replace with the new Var node in transformed weight shape.
  else if *Node type == Call* **then**
    if *Call.op == conv2d() && weight.name in compress_list* **then**
      new_Call = []
      new_Call.append(im2col())
      new_Call.append(dense())
      new_Call.append(reshape())
      // replace Call node with new_Call.

INTERNATIONAL CONFERENCE ON PARALLEL PROCESSING

acm In-Cooperation sighpc

Conversion Flow in TVM

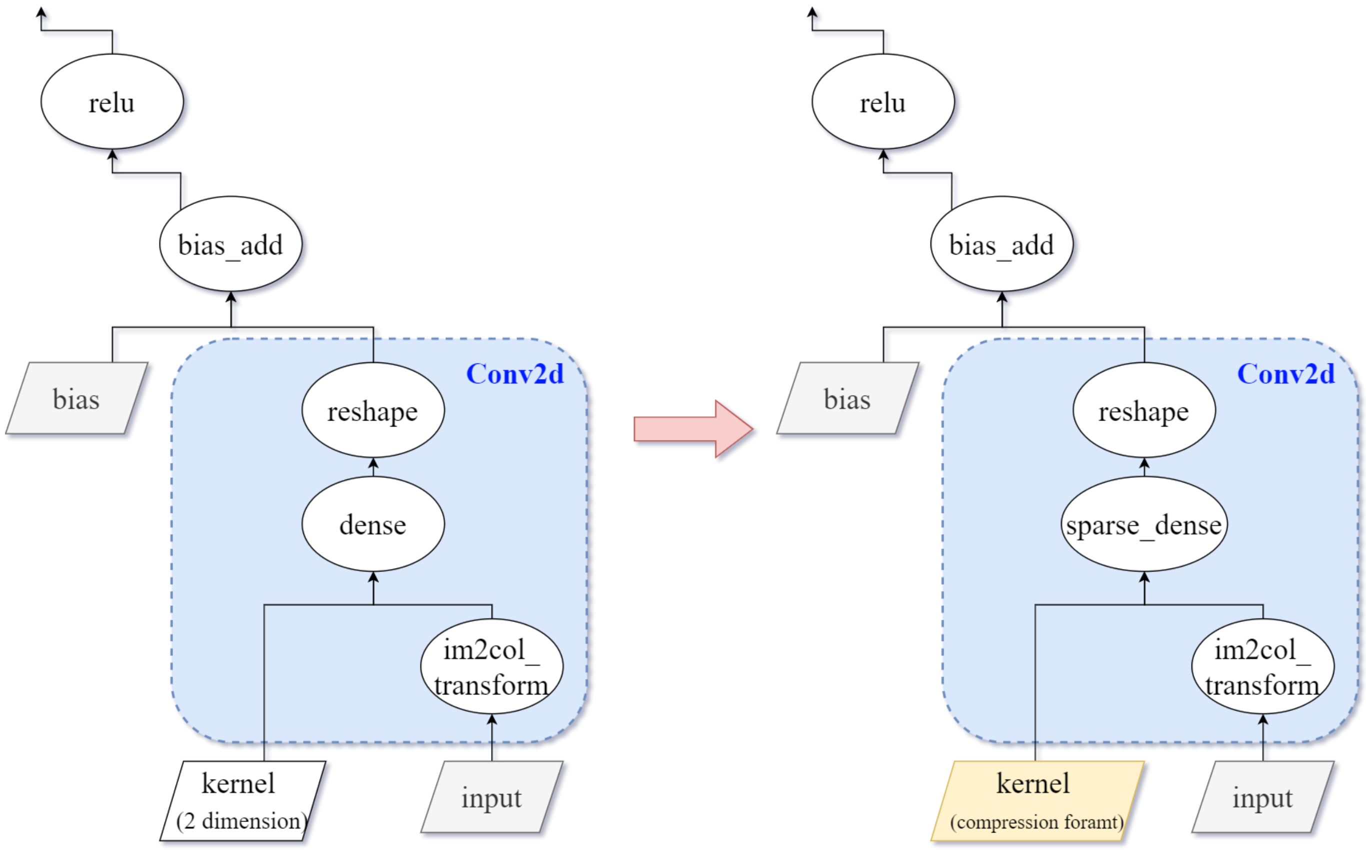


Algorithm 2: Second conversion of relay graph

Input: Params, compress_list, compression_format
Output: New params and relay graph

```
weight_info = []
for weight in compress_list do
    if compression_format == "CSR" then
        data, indices, indptr = params[weight].to_csr()
    end
    else if compression_format == "BSR" then
        data, indices, indptr = params[weight].to_bsr()
    end
    del params[weight]
    params[weight+"data"] = data
    params[weight+"indices"] = indices
    params[weight+"indptr"] = indptr
    weight_info.append((weight, data.shape, indices.shape,
      indptr.shape))
end
return weight_info
```

Conversion Flow in TVM



Algorithm 2: Second conversion of relay graph

```
Input: Params, compress_list, compression_format
Output: New params and relay graph
weight_info = []
for weight in compress_list do
    if compression_format == "CSR" then
        data, indices, indptr = params[weight].to_csr()
    end
    else if compression_format == "BSR" then
        data, indices, indptr = params[weight].to_bsr()
    end
    del params[weight]
    params[weight+"data"] = data
    params[weight+"indices"] = indices
    params[weight+"indptr"] = indptr
    weight_info.append((weight, data.shape, indices.shape,
      indptr.shape))
end
return weight_info
```

INTERNATIONAL CONFERENCE ON PARALLEL PROCESSING

Reference : Apache TVM

Experiments

- Environment
  - CPU : x86_64 Intel i7-9700K CPU
  - OS : Linux
  - LLVM : 9.0.1
  - TVM : 0.7.0
- Test Model Preparation
  - ImageNet based : SparseZoo
  - Cifar10 based : Pruning with distiller
    - Pruning for every layers of convolution weights
    - Using AGP Pruner
    - Retraining for 100 epochs with 0.006 lr.

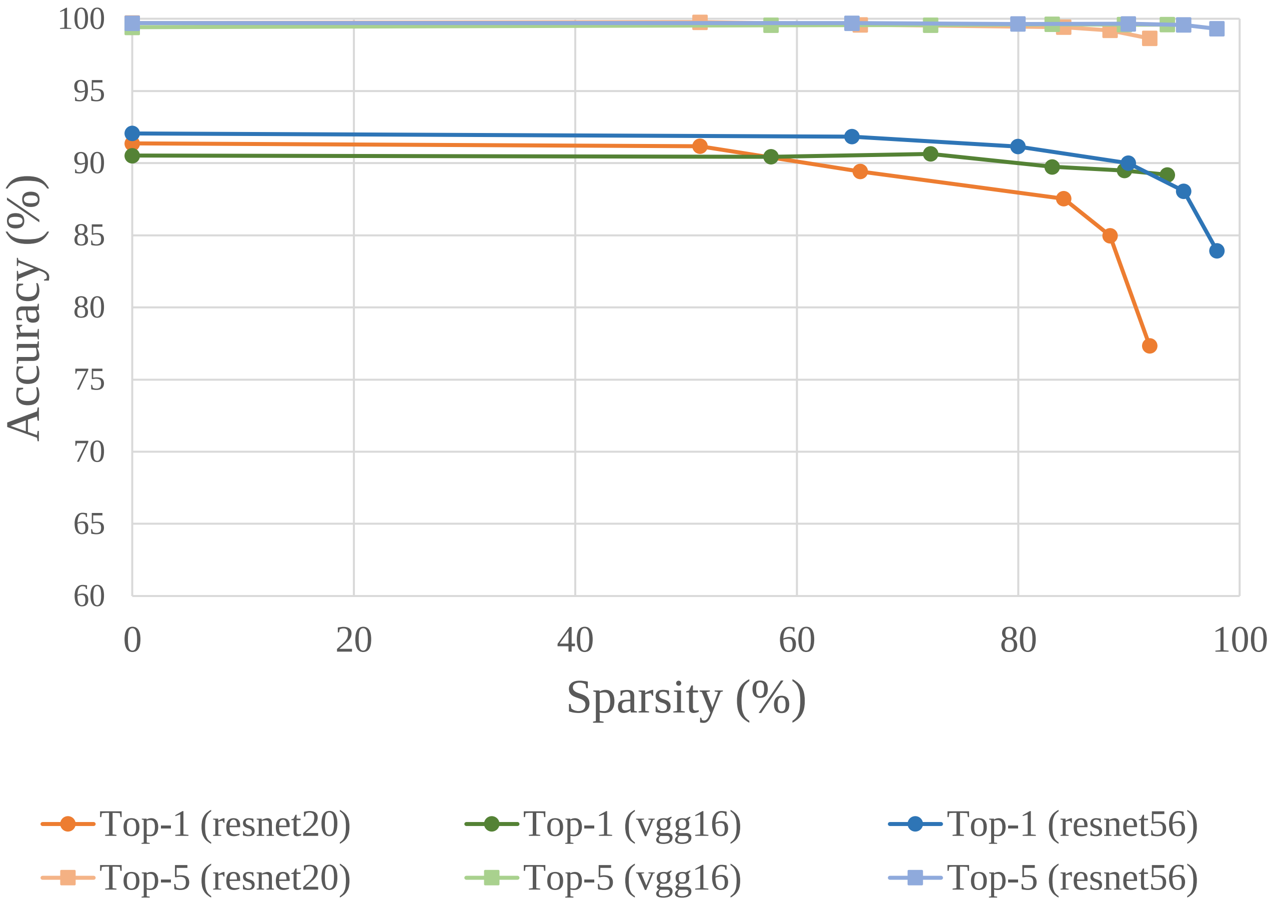
INTERNATIONAL CONFERENCE ON PARALLEL PROCESSING

acm In-Cooperation sighpc

Experiments – Accuracy Influence

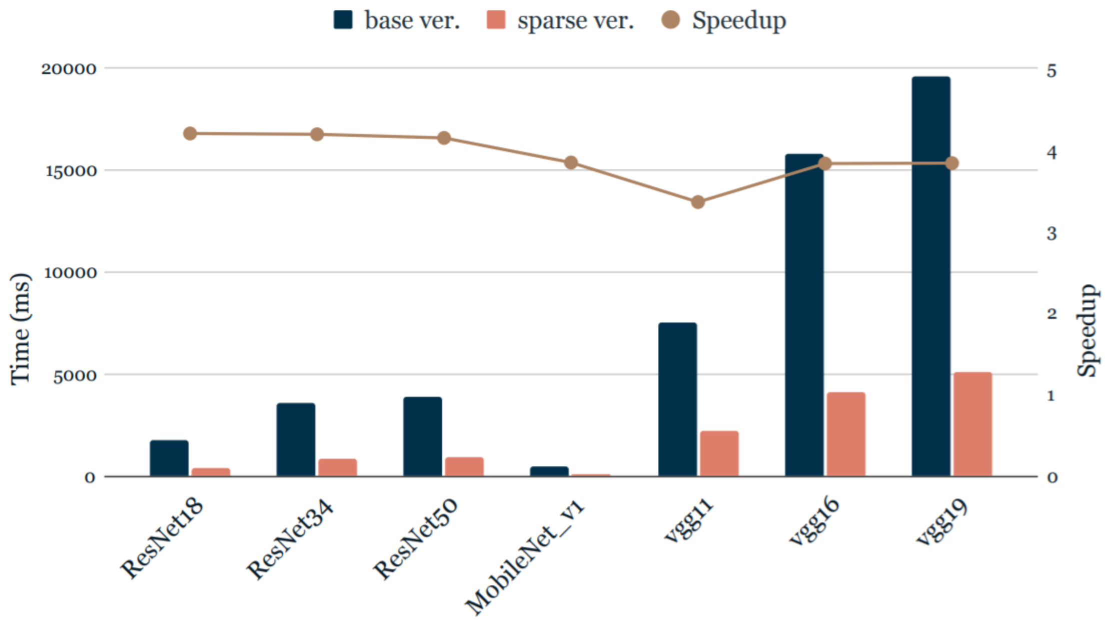
- Model : Cifar-10 based

ResNet20						
Sparsity	0	51.26	65.74	84.11	88.28	91.87
Top-1	91.36	91.18	89.42	87.54	84.98	77.35
Top-5	99.72	99.70	99.59	99.43	99.21	98.65
Vgg16						
Sparsity	0	57.67	72.10	83.07	89.59	94.23
Top-1	90.52	90.44	90.64	89.75	89.49	88.52
Top-5	99.42	99.56	99.57	99.63	99.61	99.56
ResNet56						
Sparsity	0	64.97	79.96	89.94	94.94	97.94
Top-1	92.06	91.84	91.15	90.00	88.06	83.92
Top-5	99.70	99.70	99.65	99.66	99.58	99.32



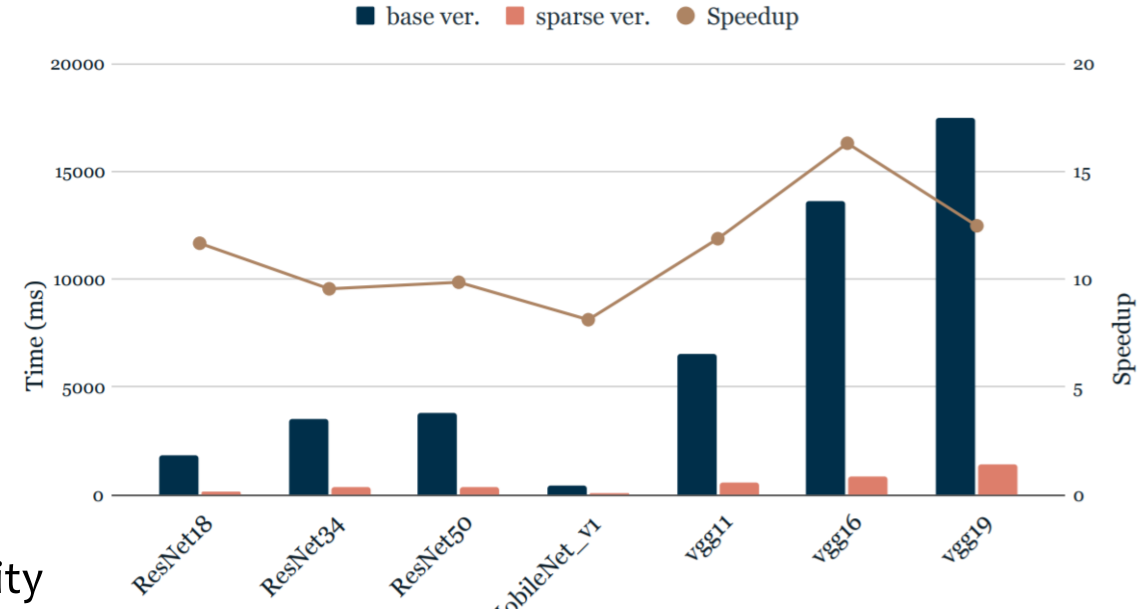
Experiments – Performance

- Model : ImageNet (SparseZoo)
- Speedups compared dense models (sparsity=0)
  - Base version : run with conv2d()
  - Sparse version: run with im2col() + sparse_dense()



Dense model

- Speedups compared sparse models



Sparsity (average): 87% 84% 88% 80% 85% 86% 86.5%

Sparse model

CONFERENCE ON PARALLEL PROCESSING

sighpc

Conclusion

- We design a flow for sparse convolution in TVM.
- We create a new operator, im2col, in TVM to support sparse convolution.
- We design a visitor to replace target conv2d to im2col+GEMM.
- The performance of sparse models with compression scheme have at most 16.3x of speedup compared with direct convolution, both are without any TVM optimization.

Thank you for listening.

Q&A