NNBench-X: A Benchmarking Methodology for Neural Network Accelerator Designs

Xinfeng Xie, *Xing Hu, Peng Gu, Shuangchen Li, Yu Ji, and Yuan Xie*

University of California, Santa Barbara

02/17/2019

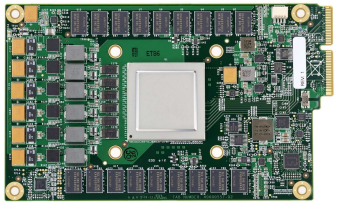
Outline

- Background & Motivation
  - NN Benchmark for Accelerator: Why, What?
- Benchmark Method
- NN Workload Characterization
  - Case Study: TensorFlow Model Zoo
- SW-HW Co-design Evaluation
  - Case Study: Neurocube, DianNao, and Cambricon-X
- Conclusion & Future Work

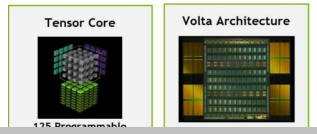
NN Benchmark: Why?

- NN accelerator has attracted a lot of attention
  - How good are existing accelerators?
  - How to design a better one?

TPU-v1
Systolic Array



GPU-Volta
Sea of Small Cores

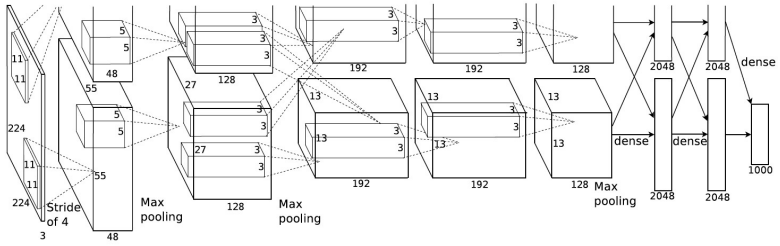


DeePhi
Sparse MXU

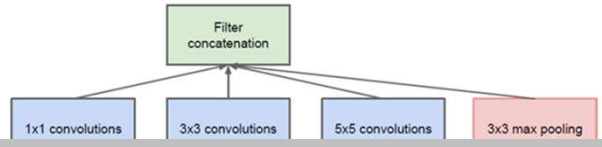
A benchmark-suite for **evaluating** and **providing guidelines** to accelerators with diverse and representative workloads.

NN Benchmark: What?

- 3Vs in NN models
  - ***V**olume*: a large amount of NN models
  - ***V**elocity*: a fast speed of volume growth
  - ***V**ariety*: various NN architectures



AlexNet

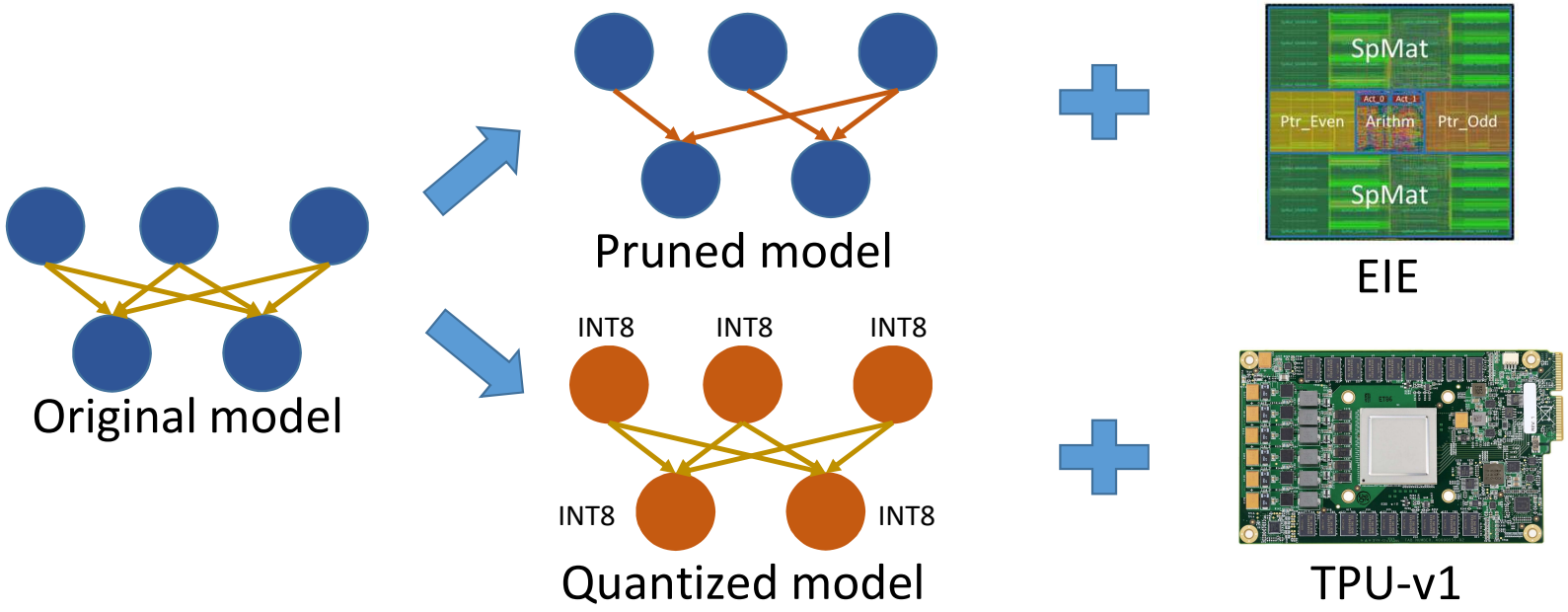


the building block of GoogleNet

A benchmark-suite needs to **select** representative NN models and **update** the suite.

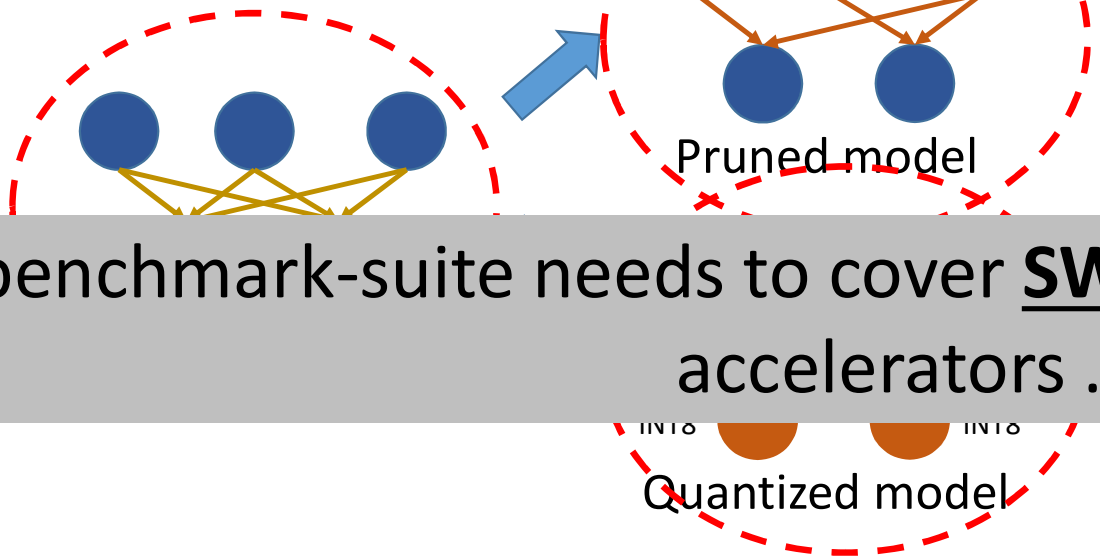
NN Benchmark: What?

- SW-HW co-design: model compression + hardware design
  - Pruning: prune out insignificant weight
  - Quantization: use lower number of bits for data representation



NN Benchmark: What?

- SW-HW co-design: model compression + hardware design
  - Pruning: prune out insignificant weight
  - Quantization: use lower number of bits for data rep[illegible]



How can I include one of them to evaluate SW-HW co-designs?

A benchmark-suite needs to cover **SW-HW co-designs** for NN accelerators .

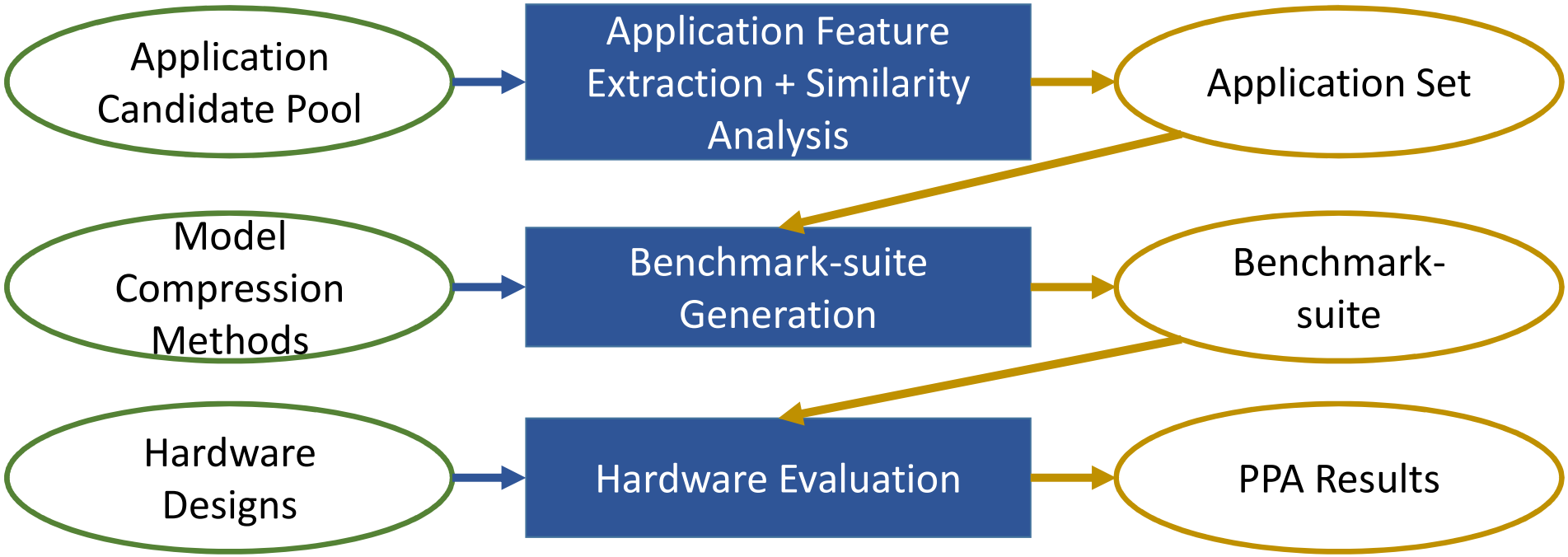
NN Benchmark: Related Work

- We need a new NN benchmark for accelerators!

Project Name	Platform	Phase	App Selection	SW-HW Co-design
Fathom	CPU/GPU	Training + Inference	Empirical	✗
BenchIP	Accelerator	Inference	Empirical	✗
MLPerf	Cloud + Mobile	Training + ~~Inference~~	Empirical	✗
NNBench-X	**Accelerator**	**Inference**	**Quantitative**	☑

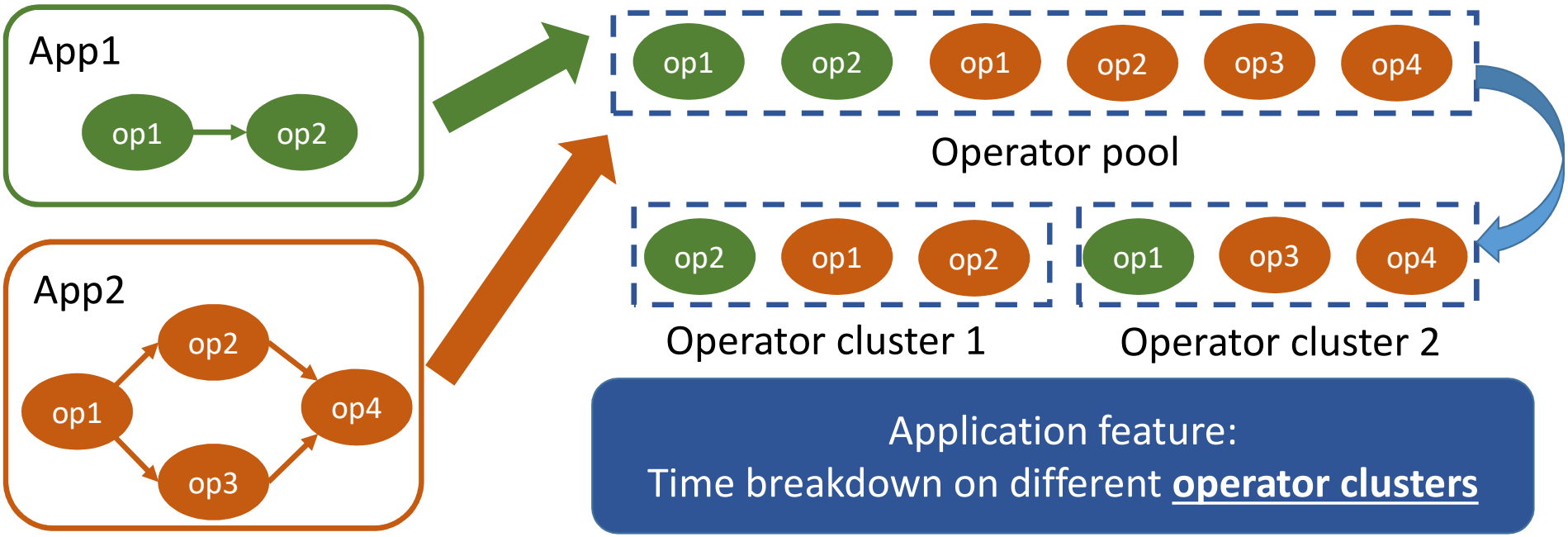
Benchmark Method

- Overall idea: both SW and HW designs are input



NN Workload Characterization

- Application feature for NN applications
  - Two-level analysis: operator-level and application-level

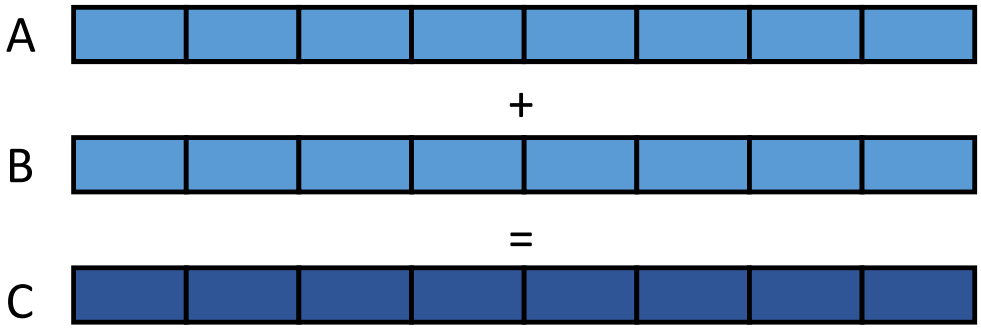


Operator Feature

- Operator features
  - Locality: #data / #comps
  - Parallelism: the ratio of #comps can be parallelized

#data:
sizeof(A) + sizeof(B) + sizeof(C)
#comps:
length(A) scalar add oprs

Locality: #data / #comps
Parallelism: 100%

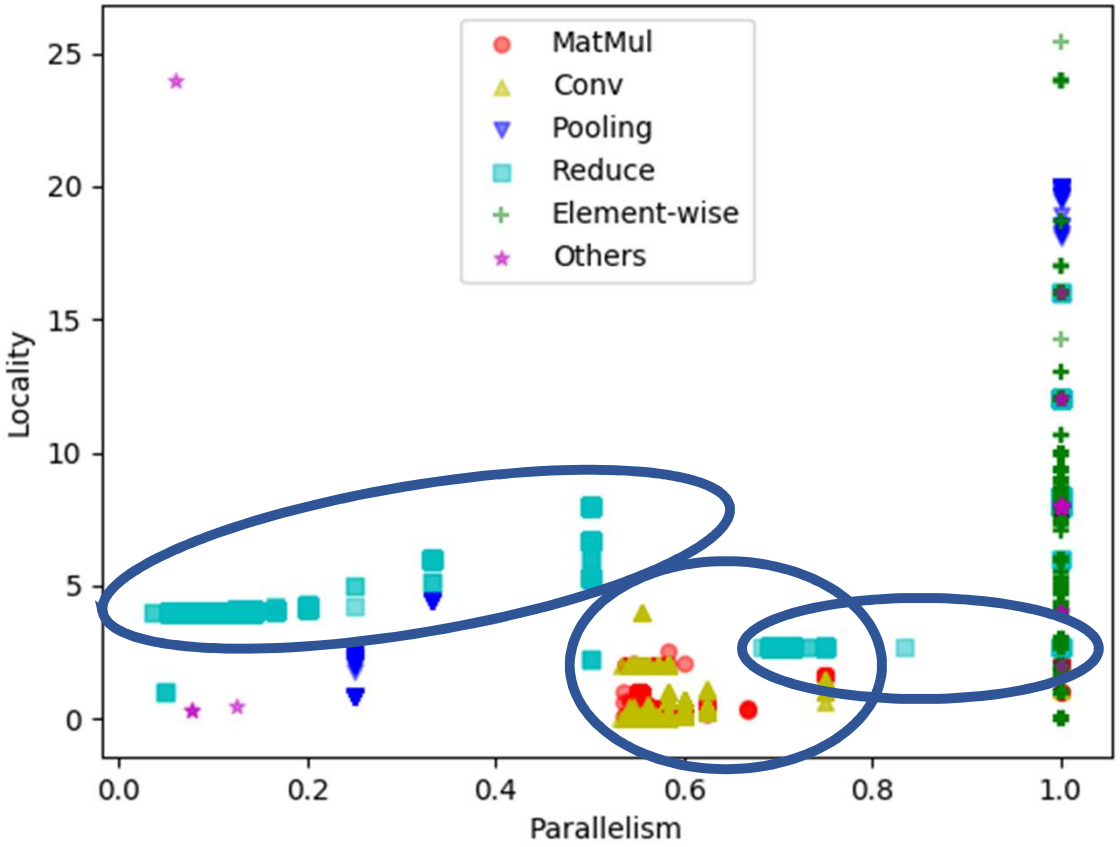


An example of element-wise add

Case Study: TensorFlow Model Zoo

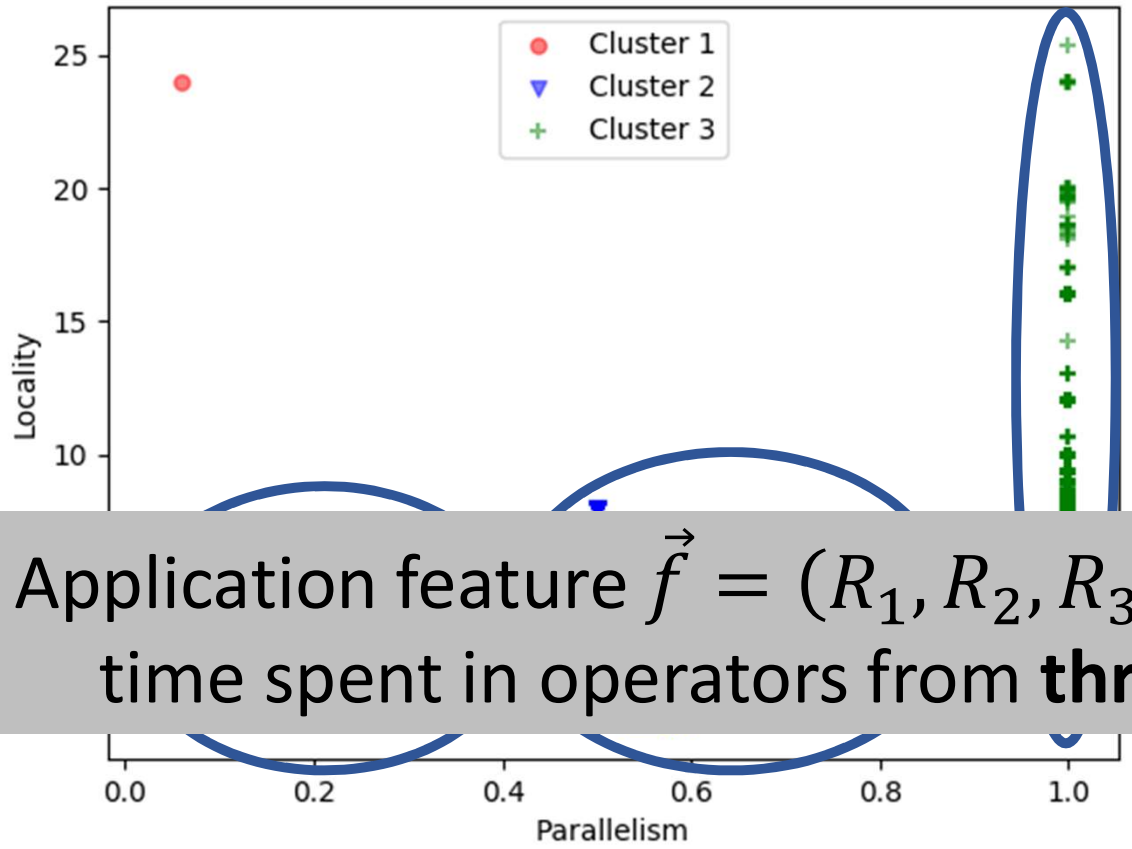
- **Up-to-date models from the machine learning community**
  - Source code: https://github.com/tensorflow/models
- **A wide range of application domains:**
  - Computer vision (CV), natural language processing (NLP), informatics etc.
  - 24 NN applications with 57 models.
- **Diverse neural network architectures and learning methods:**
  - Convolutional neural network (CNN), recurrent neural network (RNN) etc.
  - Supervised learning, unsupervised learning, reinforcement learning etc.

Workload Characterization (1/5)



- **Observation #1:** Convolution and matrix multiplication operators are similar to each other in terms of locality and parallelism features.
- **Observation #2:** Operators with the same functionality can exhibit very different locality and parallelism features.

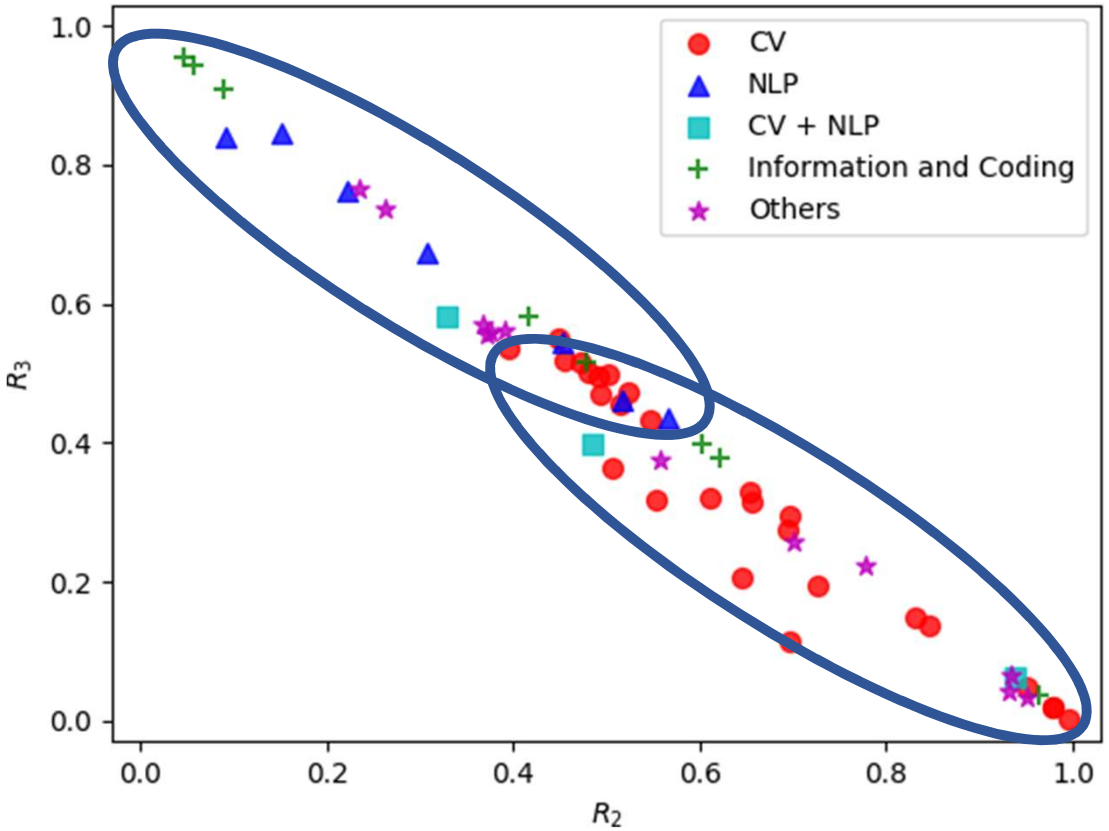
Workload Characterization (2/5)



- **Cluster 1:** Inferior parallelism
  - Hard to be parallelized.
  - Bad news from Amdahl's Law.
- **Cluster 2:** Moderate parallelism and locality
  - Benefit from parallelization and cache hierarchy.
- **Cluster 3:** Ample parallelism

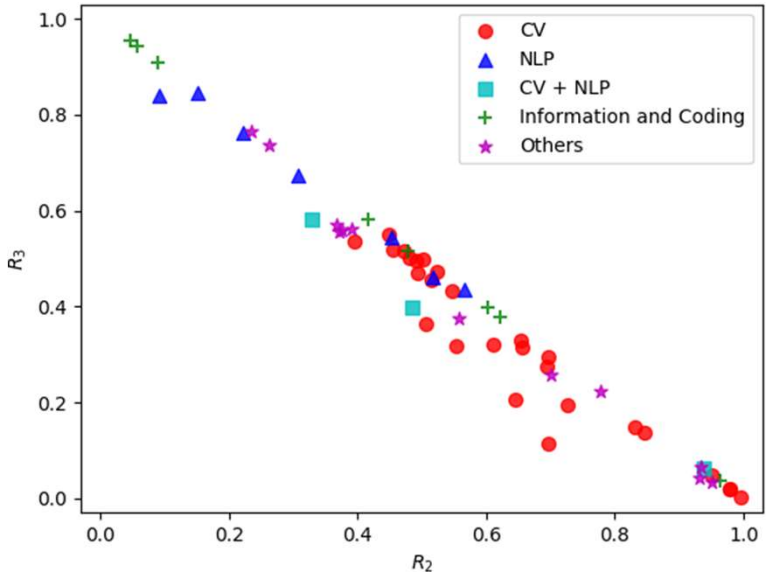
Application feature $\vec{f} = (R_1, R_2, R_3)$, where **$R_1$, R_2, and R_3** are time spent in operators from **three clusters** respectively.

Workload Characterization (3/5)

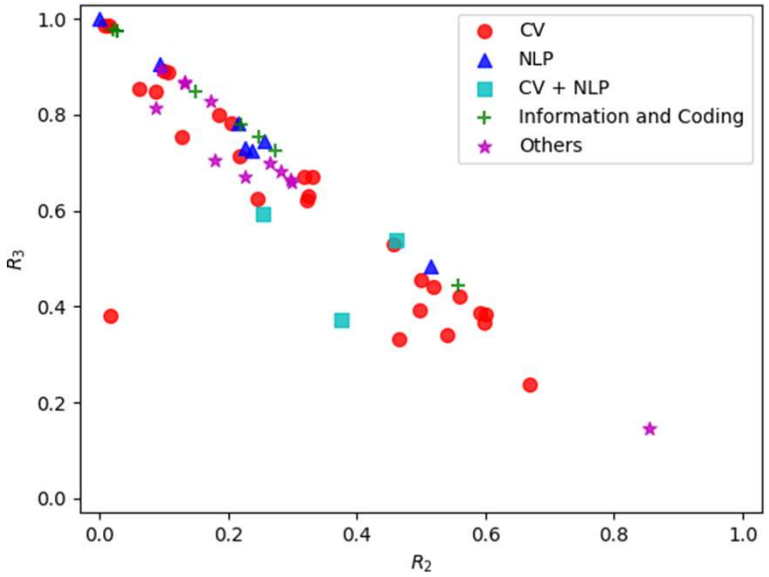


- **Observation #3:** The bottleneck of application is related to its application domain.
- CV applications are bounded by R_2 (mostly Conv and MatMul).
- NLP applications are bounded by R_3 (mostly Element-wise)

Workload Characterization (4/5)



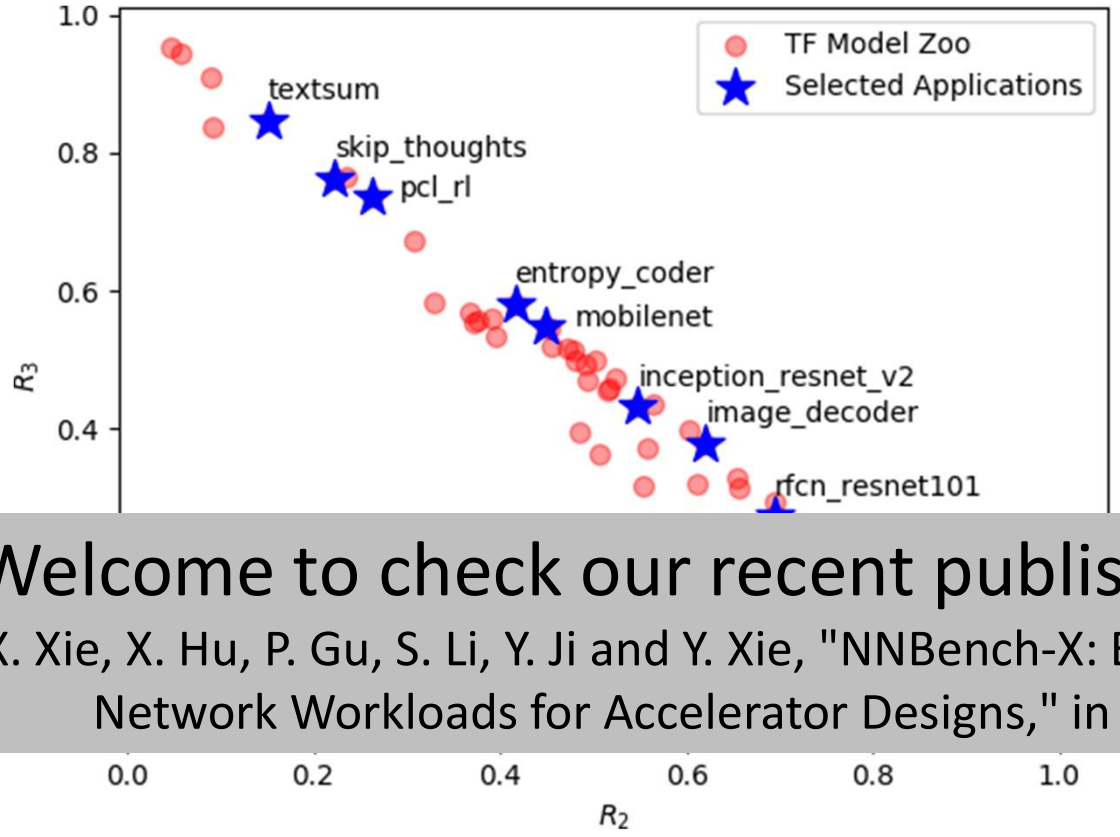
(a) CPU



(b) GPU

- **Observation #4:** Applications on GPU have a larger R_1 because parallelizable parts are well accelerated. (Amdahl's Law)

Workload Characterization (5/5)



- Select applications along the line $R_2 + R_3 = 1$

Table: Brief descriptions for ten applications in NNBench-X.

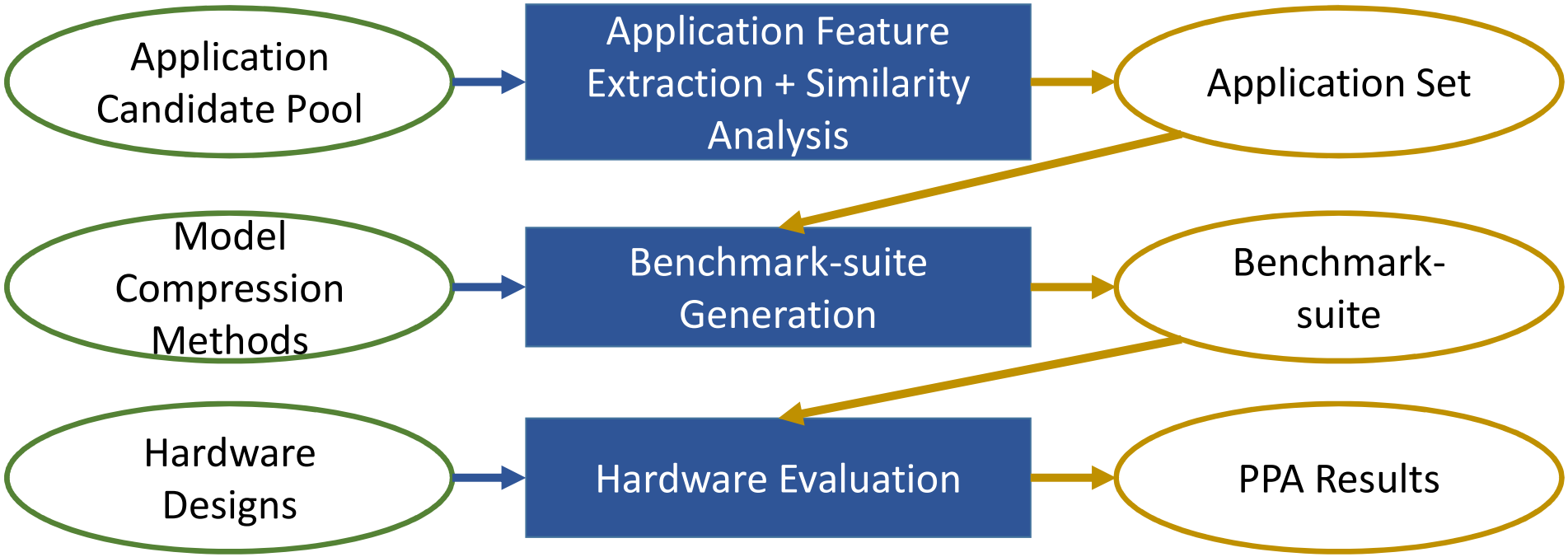
Application	Description
textsum	Text summarization
skip_thoughts	Sentence-to-vector encoder
pcl_rl	Reinforcement learning
entropy_coder	Image file compression

Welcome to check our recent published paper for more details:

X. Xie, X. Hu, P. Gu, S. Li, Y. Ji and Y. Xie, "NNBench-X: Benchmarking and Understanding Neural Network Workloads for Accelerator Designs," in *IEEE Computer Architecture Letters*.

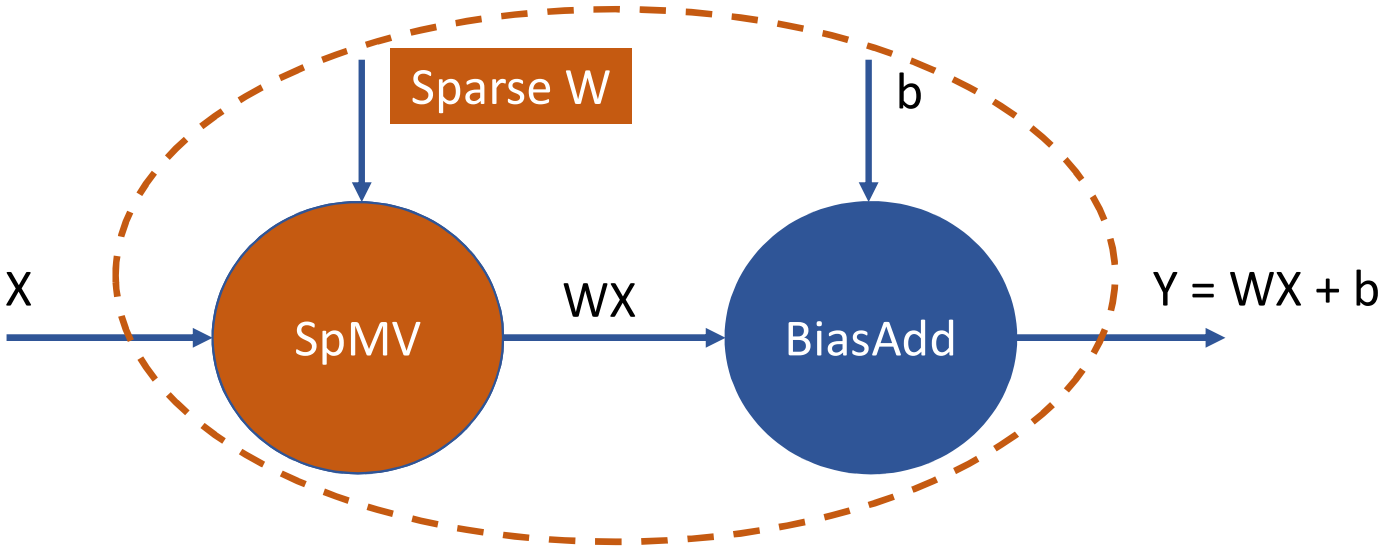
Benchmark Method

- After the first stage, we obtained the application set.



Benchmark-suite Generation

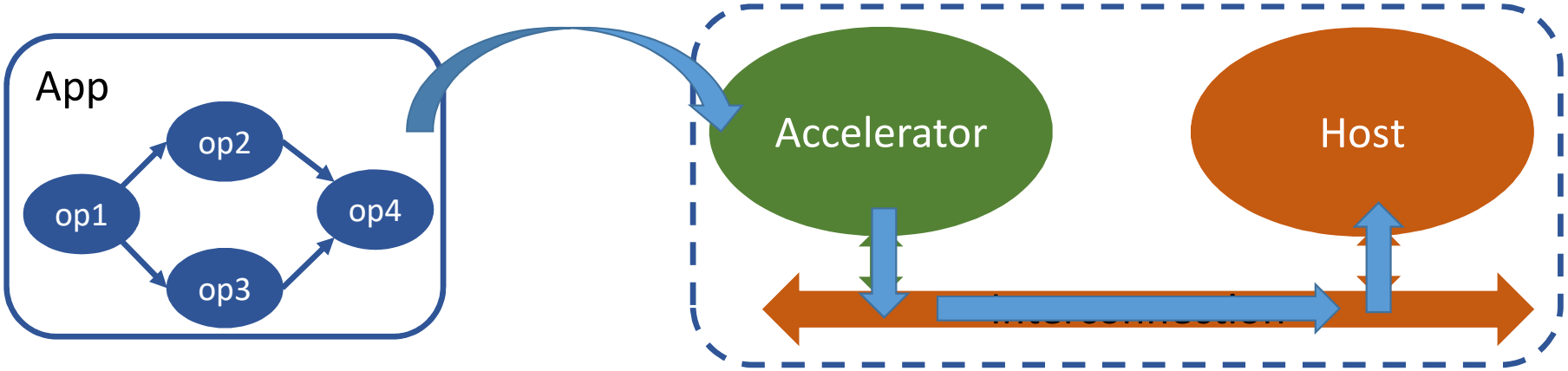
- Export a new computation graph according to the input model compression technique



An example: exporting a pruned model

Hardware Evaluation

- Operator-based simulation framework

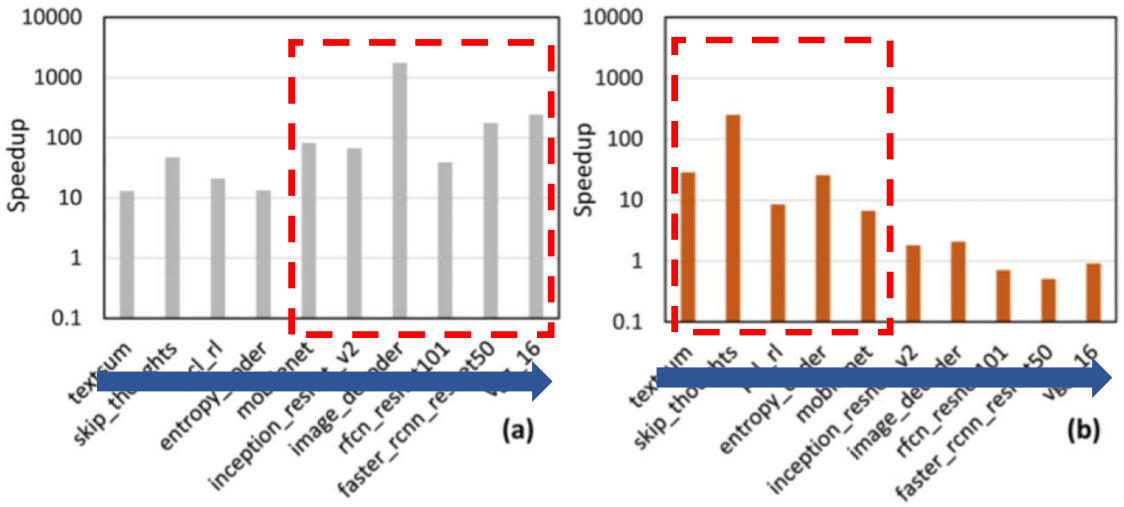


- Scheduling strategy:
  - Schedule operators to accelerator
  - **Fallback:** (unsupported by the accelerator) schedule into the host

SW-HW Co-design Evaluation

- Evaluated Hardware:
  - GPU, Neurocube, DianNao, and Cambricon-X
- Case Study I: Memory-centric vs. Compute-centric Designs
  - Evaluated hardware: GPU and Neurocube
- Case Study II: Benefits of Model Compression
  - Solution I: DianNao + Dense models
  - Solution II: Cambricon-X + Sparse models (90% sparsity)
  - Solution III: Cambricon-X + Sparse models (95% sparsity)

Compute-centric vs. Memory-centric

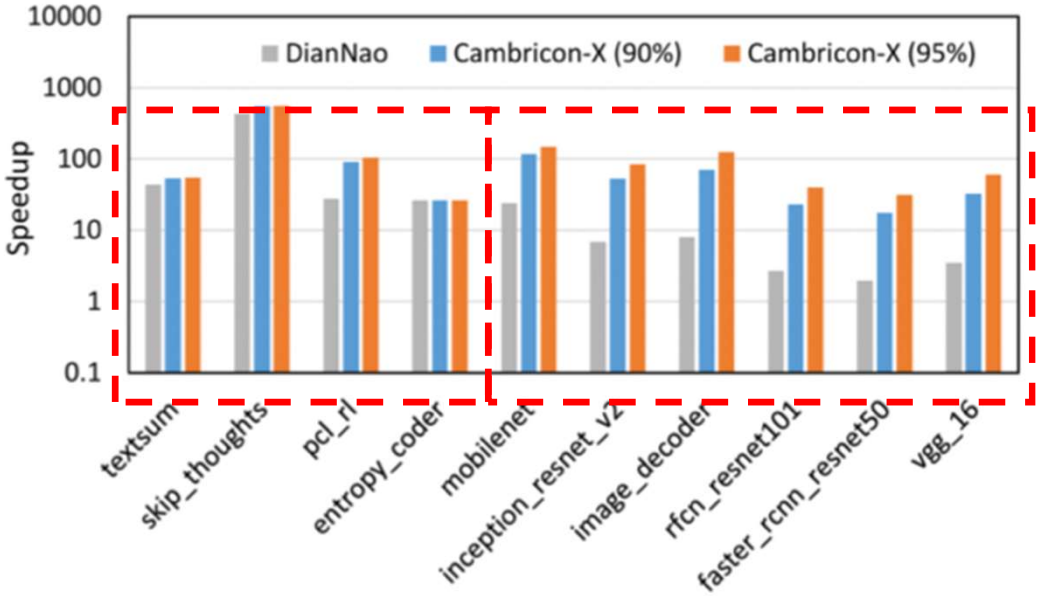


(a) GPU (b) Neurocube

Applications are listed in an increasing R_2 order along the x-axis. (decreasing R_3 order)

- **Observation #5:** GPU benefits applications bounded by R_2 because of rich on-chip computation resources and scratchpad memory.
- **Observation #6:** Neurocube benefits applications bounded by R_3 by providing large effective memory bandwidth.

Benefits of Model Compression



DianNao: 0% weight sparsity

Cambricon-X (90%): 90% weight sparsity

Cambricon-X (95%): 95% weight sparsity

- **Observation #7:** Pruning weights helps CV and NLP applications differently.
- Pruning weights help CV applications significantly.
- NLP applications are not so sensitive to weight sparsity as CV applications.

Conclusion & Future Work

- Two Main Takeaways:
  - CV and NLP applications are very different from the perspective of NN accelerator designs.
  - Conv and MatMul are not always the bottleneck of NN applications.

- Future Work:
  - Hardware modeling in the early design stage of accelerators.
  - Other model compression techniques in addition to quantization and pruning.
  - Value-dependent behaviors in NN applications, such as graphical convolution network (GCN).

Thank You!

Q & A

Please contact the authors for further discussion.

E-mail:
xinfeng@ucsb.edu
yuanxie@ucsb.edu