BOLT: BRIDGING THE GAP BETWEEN AUTO-TUNERS AND HARDWARE-NATIVE PERFORMANCE

Xing et. al. (2021)

Gabriel Mahler

Background

Auto-tuners vs hardware-native libraries

Background

Auto-tuners

- Structural tensor program optimization
- Generate training sets of sample programs - use performance to navigate the search space

Background

Auto-tuners

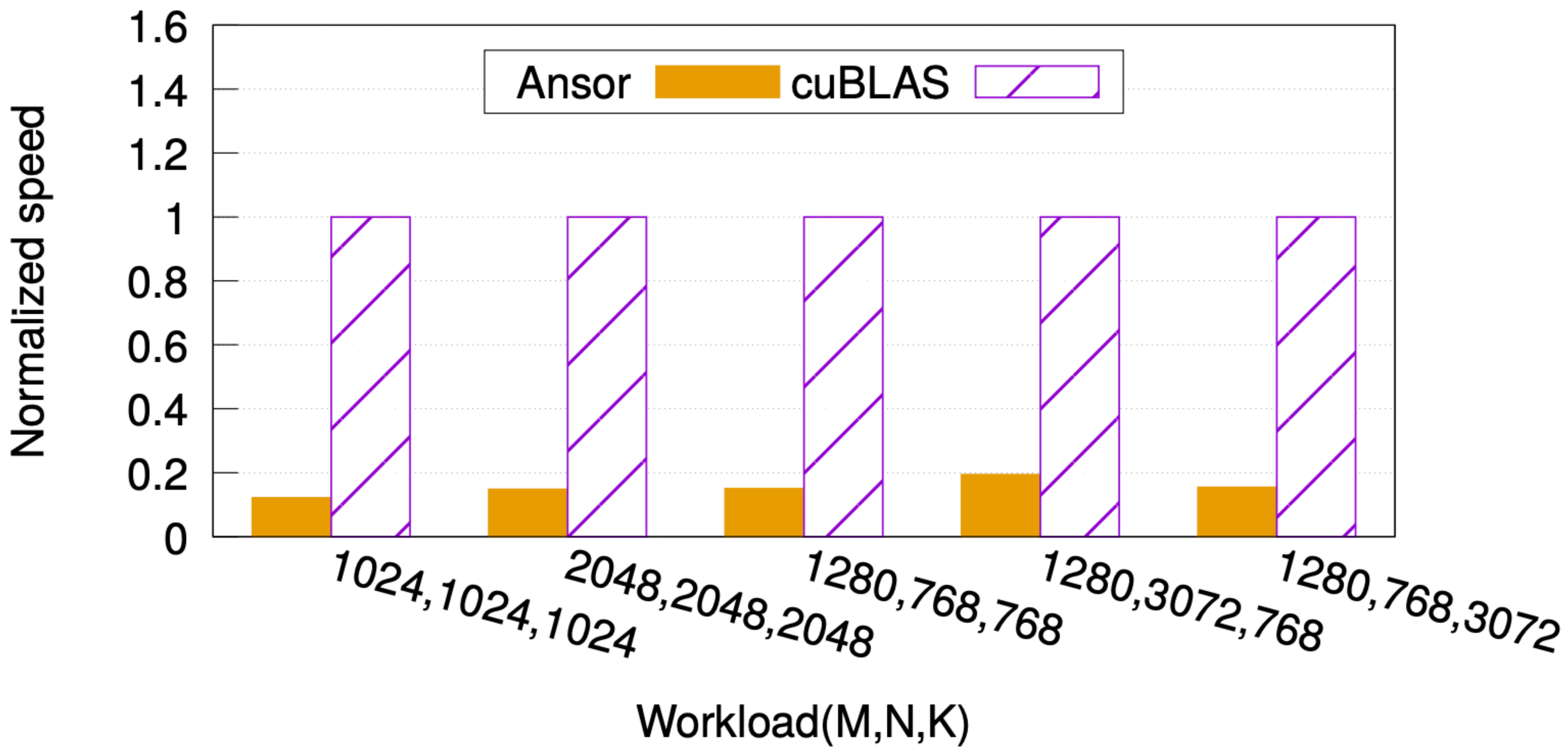
- Strenghts:
  - platform generality

Background

Auto-tuners

- Strenghts:
  - platform generality
- Weaknesses:
  - performance (vs. vendor libraries)

Background



Ansor
cuBLAS
Normalized speed
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
1024,1024,1024
2048,2048,2048
1280,768,768
1280,3072,768
1280,768,3072
Workload(M,N,K)

Background

Auto-tuners

- Strenghts:
  - Platform generality
- Weaknesses:
  - Performance (vs. vendor libraries)
  - Long search

Background

Templated Libraries

- Modularized and composable
  - Templates initiated for different hardware and workloads

Background

Templated Libraries

- Strenghts:
  - Performance superior to hardware-opaque auto-tuning

Background

Templated Libraries

- Strenghts:
  - Performance superior to hardware-opaque auto-tuning
- Weaknesses:
  - Parameters too low-level

Background

Templated Libraries

- Strengths:
  - Performance superior to hardware-opaque auto-tuning
- Weaknesses:
  - Parameters too low-level
  - Usually only for a part of a model

Bolt

- *“bridge the gap between auto-tuners and hardware-native performance”*

Bolt

- *“bridge the gap between auto-tuners and hardware-native performance”*
  - Deeper operator fusion (graph level)

Bolt

- *“bridge the gap between auto-tuners and hardware-native performance”*
    - Deeper operator fusion - graph level
    - Automated template code generation - operator level

Bolt

- *“bridge the gap between auto-tuners and hardware-native performance”*
    - Deeper operator fusion - graph level
    - Automated template code generation - operator level
    - (System-friendly models - model level)

Deeper Operator Fusion

Pre-requisite: **Epilogue Fusion** (provided by CUTLASS)

- General Matrix Multiply (GEMM)/Convolution Kernels
- Epilogue Kernels (element-wise operators, data type conversion, data type conversion, broadcast vector over columns, partial reduction over columns)

Deeper Operator Fusion

Persistent Kernel

- Fuses GEMMs & Convolutions
- Eliminates activation storing & loading in global memory
- Eliminates kernel initializations

Deeper Operator Fusion

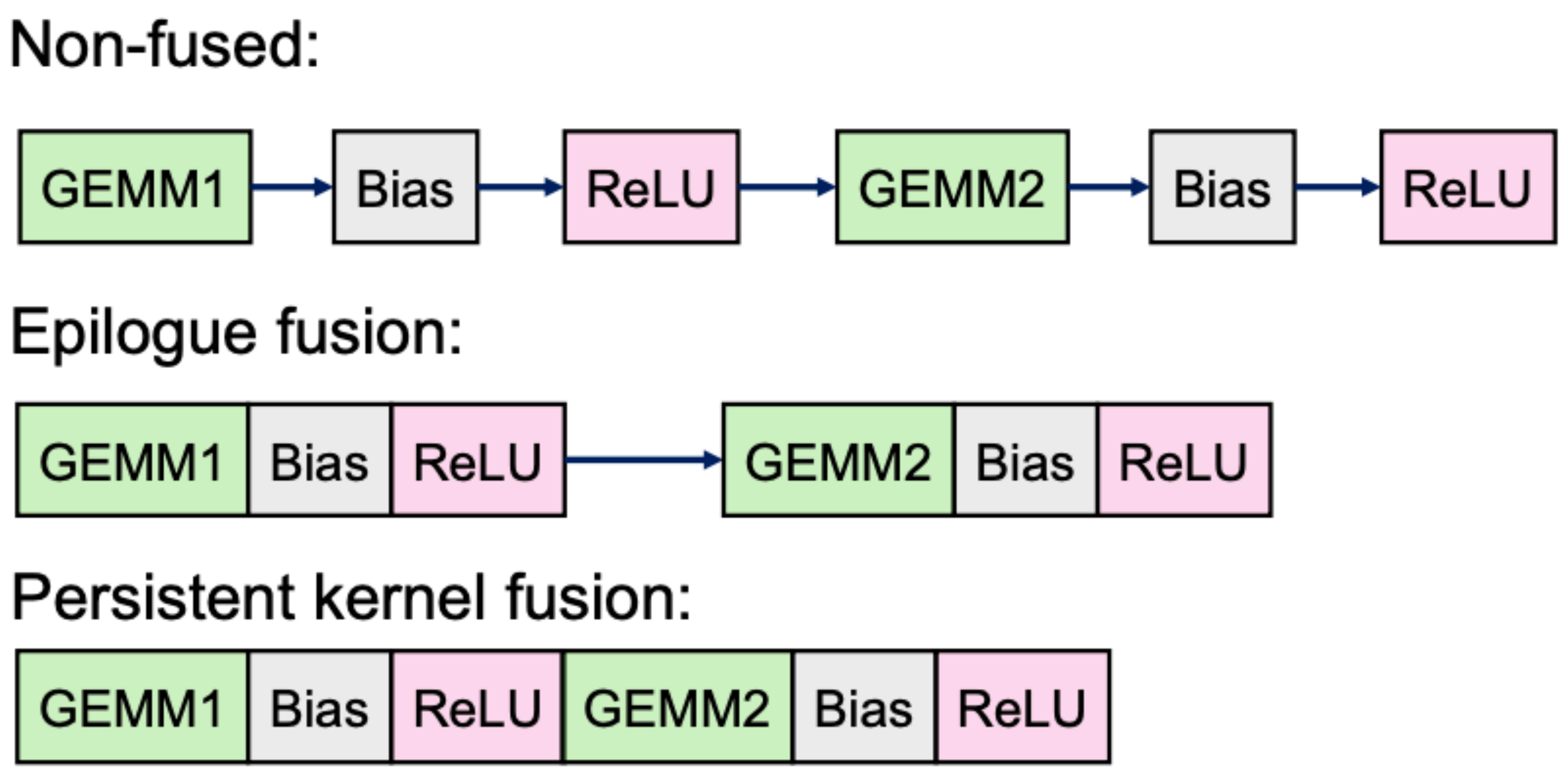
Persistent Kernel

- Requires:
  - GEMM: dimensional compatibility
  - Convolution: filter restrictions

Deeper Operator Fusion

Persistent Kernel

- Bolt:
  - Identifies opportunities to use persistent kernels
  - Generates new code using templates (using CUDA code)


Non-fused:
GEMM1
Bias
ReLU
GEMM2
Bias
ReLU
Epilogue fusion:
GEMM1
Bias
ReLU
GEMM2
Bias
ReLU
Persistent kernel fusion:
GEMM1
Bias
ReLU
GEMM2
Bias
ReLU

Automated Template Code Generation

BYOC (bring your own codegen)

Automated Template Code Generation

BYOC (bring your own codegen)

- Offload parts of code from the compiler (TVM) to templated libraries (CUTLASS)

Automated Template Code Generation

“Light-weight performance profiler”

Automated Template Code Generation

“Light-weight performance profiler”

- Searches for best template parameters

Automated Template Code Generation

“Light-weight performance profiler”

- Searches for best template parameters
  - Target hardware-informed

Automated Template Code Generation

“Light-weight performance profiler”

- Searches for best template parameters
  - Target hardware-informed
  - Templated libraries - whiteboxes
    - Further optimization possibilities

System-Friendly Models

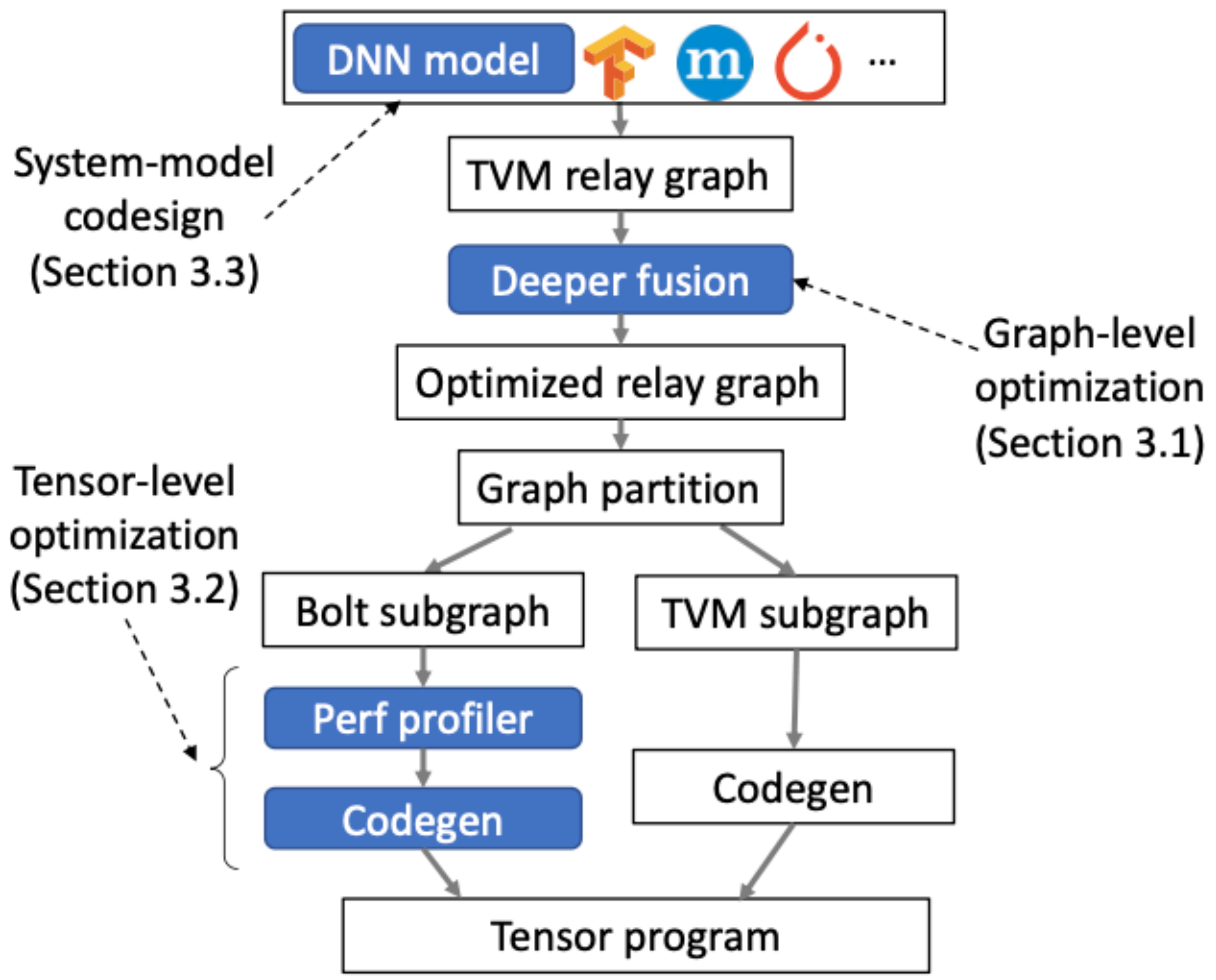
- Epilogue fusion: explore activation functions

System-Friendly Models

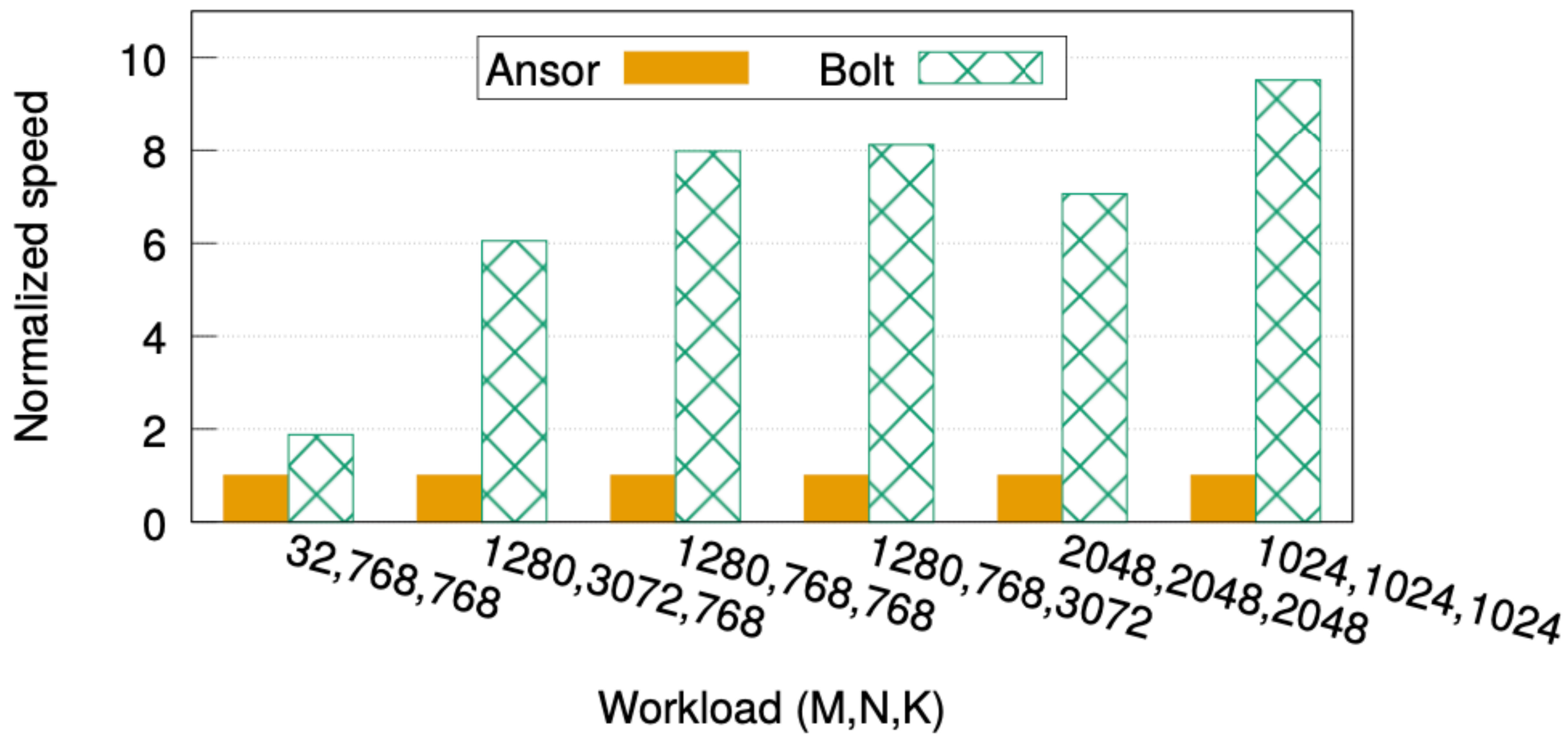
- Epilogue fusion: explore activation functions
- Persistent kernel: model deepening with 1x1 convolutions

System-Friendly Models

- Epilogue fusion: explore activation functions
- Persistent kernel: model deepening with 1x1 convolutions
- Efficient tensor shapes: padding of unaligned tensors

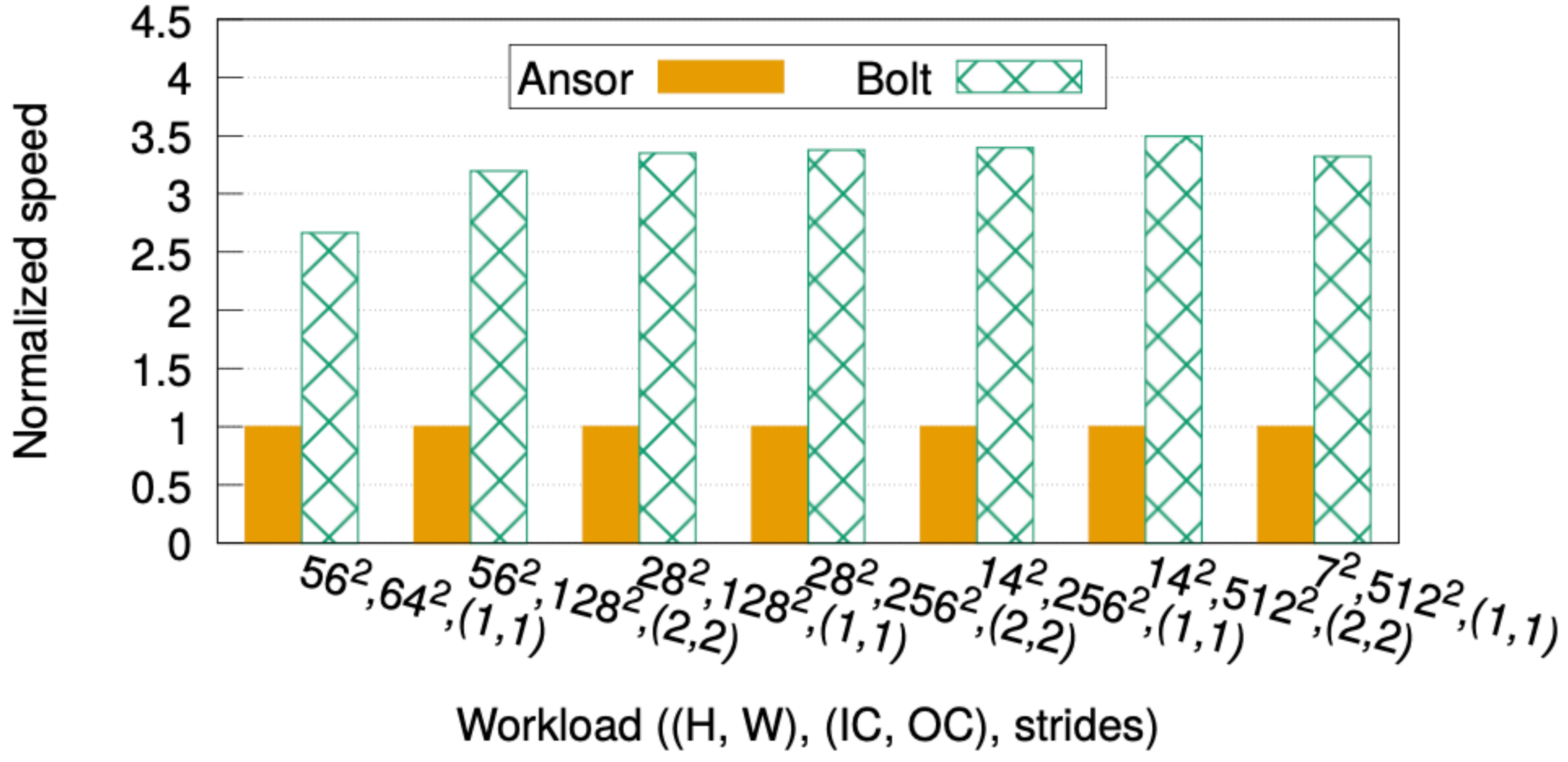

DNN model
...
TVM relay graph
System-model
codesign
(Section 3.3)
Deeper fusion
Graph-level
optimization
(Section 3.1)
Optimized relay graph
Graph partition
Tensor-level
optimization
(Section 3.2)
Bolt subgraph
TVM subgraph
Perf profiler
Codegen
Codegen
Tensor program

Evaluation



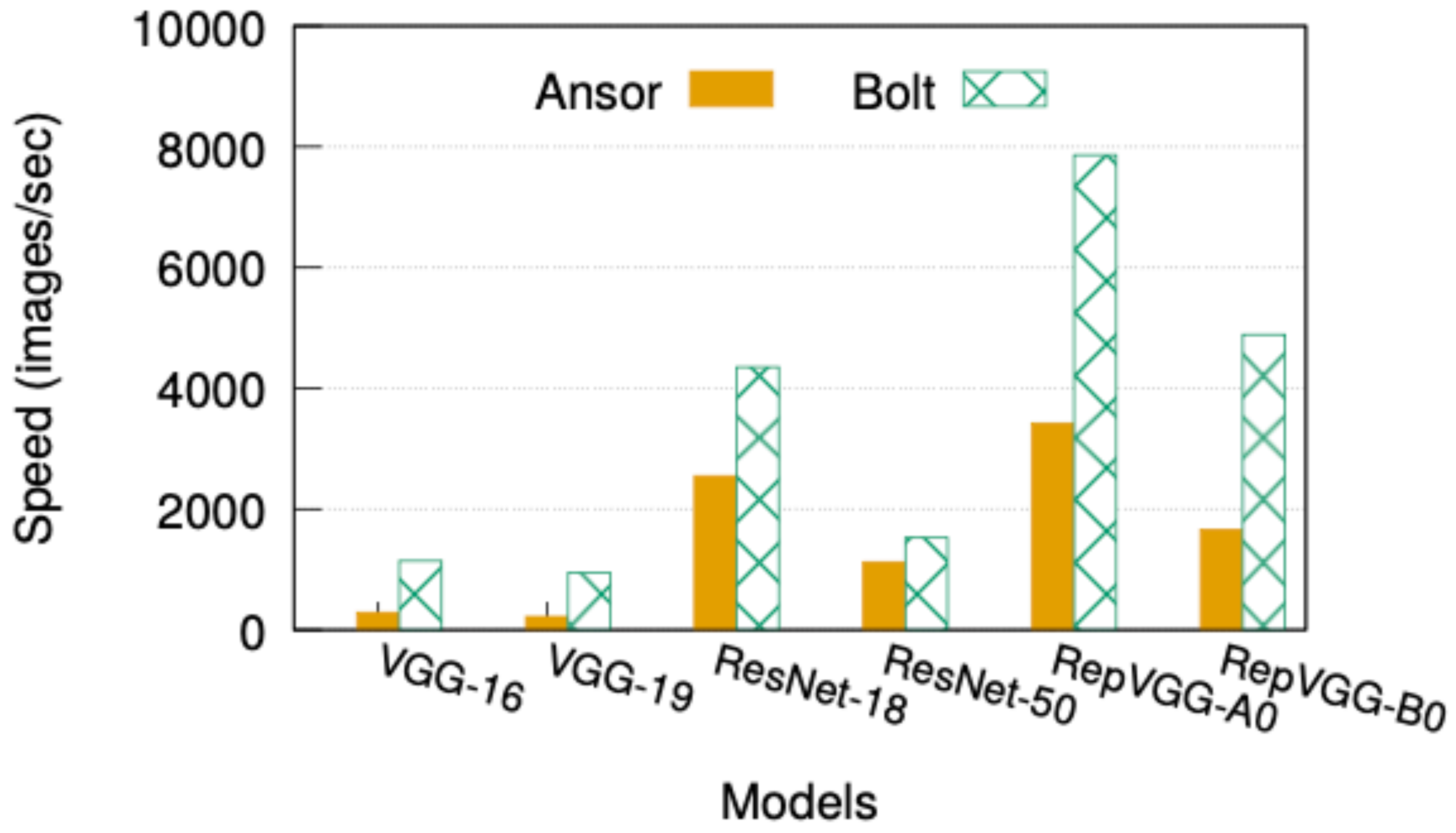
Comparison: GEMMs

Evaluation



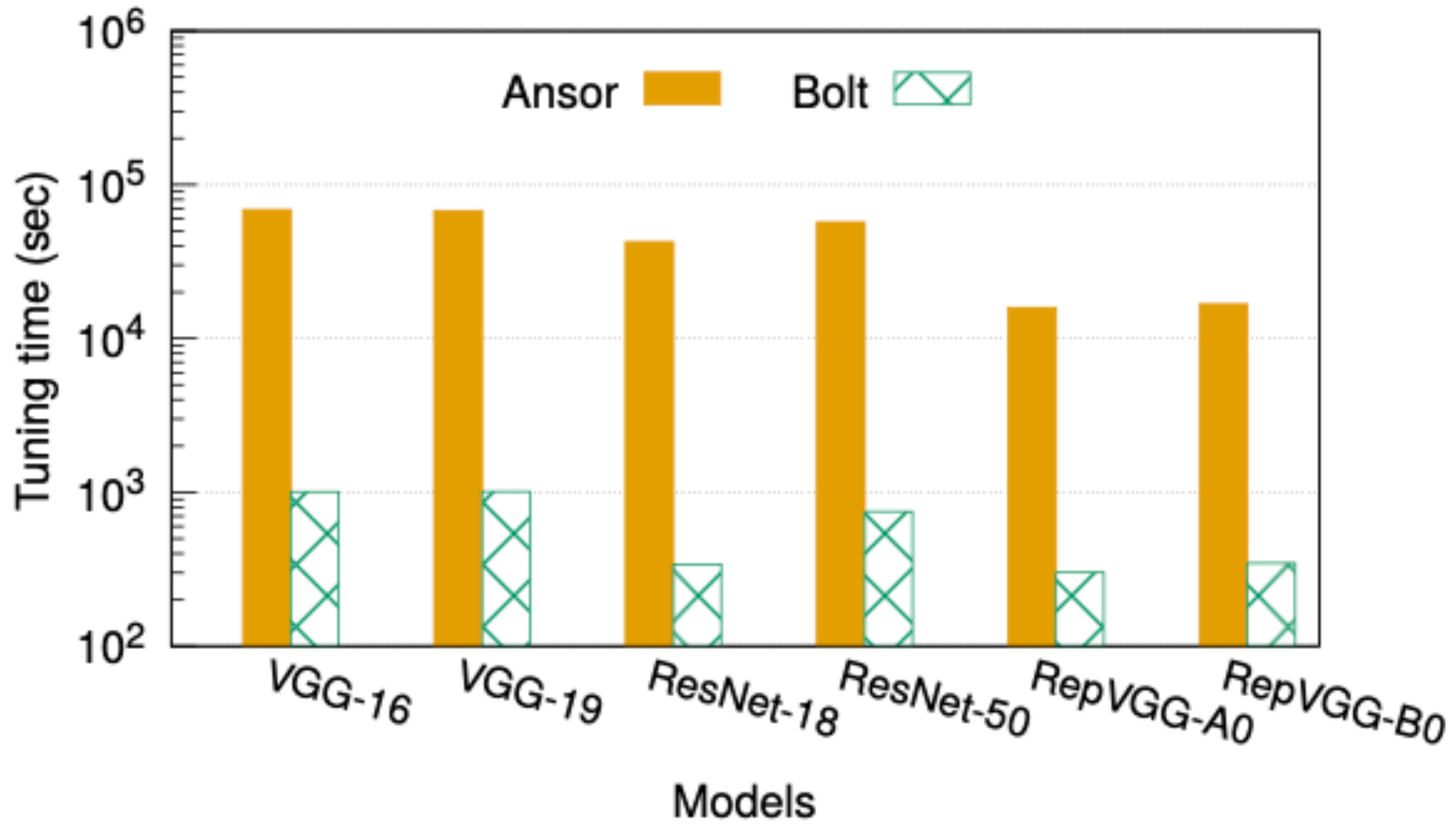
Comparison: Convolutions

Evaluation



Comparison: end-to-end inference speed

Evaluation



Comparison: end-to-end tuning speed

Bolt Today

- Integrated into TVM (CUTLASS)
    - https://github.com/apache/tvm/pull/9261
- Actively used by Bytedance

Critique

Critique

- End-to-end testing only on convolution based models

Critique

- End-to-end testing only on convolution based models
- Limited fusion exploration

Critique

- End-to-end testing only on convolution based models
- Limited fusion exploration
- Paper not too accessible (eg. many unexplained abbreviations and names)