Learning to Optimize Tensor Programs

Tianqi Chen, Lianmin Zheng, Eddie Yan, Ziheng Jiang, Thierry Moreau, Luis Ceze, Carlos Guestrin, Arvind Krishnamurthy
Goal: Deploy Deep Learning Everywhere

Explosion of models and frameworks

Huge gap between model/frameworks and hardware backends

Explosion of hardware backends
Existing Approach

Frameworks

High-level data flow graph

Hardware
Existing Approach

Frameworks

High-level data flow graph

Primitive Tensor operators such as Conv2D

Hardware

NVIDIA
Existing Approach

Frameworks

High-level data flow graph

Primitive Tensor operators such as Conv2D

eg. cuDNN

Offload to heavily optimized DNN operator library

Hardware
Limitations of Existing Approach

cuDNN
Limitations of Existing Approach

cuDNN

Frameworks
Limitations of Existing Approach

Frameworks

cuDNN

NVIDIA
Limitations of Existing Approach

Frameworks

New operators

cuDNN

NVIDIA
Limitations of Existing Approach

Frameworks

New operators

cuDNN

NVIDIA
Limitations of Existing Approach

Frameworks

New operators

cuDNN

NVIDIA

Intel processor

Raspberry Pi

iPhone

Others
Limitations of Existing Approach

Frameworks: cuDNN

New operators

Engineering intensive

cuDNN

NVIDIA
Learning to Optimize Tensor Programs

Frameworks

High-level data flow graph and optimizations

Hardware
Learning to Optimize Tensor Programs

Frameworks

High-level data flow graph and optimizations

Hardware
Learning to Optimize Tensor Programs

Frameworks

High-level data flow graph and optimizations

Machine Learning based Program Optimizer

Hardware
Learning to Optimize Tensor Programs

Machine Learning based Program Optimizer

Learning to generate optimized program for new operator workloads and hardware
Search over Possible Program Transformations

Compute Description

\[
C = \text{tvm.compute}((m, n), \lambda y, x: \text{tvm.sum}(A[k, y] \times B[k, x], \text{axis}=k))
\]

- Loop Transformations
- Thread Bindings
- Cache Locality
- Thread Cooperation
- Tensorization
- Latency Hiding

Hardware
Search over Possible Program Transformations

Compute Description

\[
C = \text{tvm.compute}((m, n), \\
\quad \lambda y, x: \text{tvm.sum}(A[k, y] \times B[k, x], \text{axis}=k))
\]
Search over Possible Program Transformations

Billions of possible optimization choices

Compute Description

\[ C = \text{tvm.compute}((m, n), \lambda y, x: \text{tvm.sum}(A[k, y] \times B[k, x], \text{axis}=k)) \]
Learning-based Program Optimizer

Program Optimizer → Code Generator → Program
Learning-based Program Optimizer

- Program Optimizer
- Code Generator
- Program
- Training data $D$
Learning-based Program Optimizer

Program Optimizer → Code Generator → Program

Learning

Statistical Cost Model → Training data

Training data
Learning-based Program Optimizer

- Relatively low experiment cost
- Domain-specific problem structure
- Large quantity of similar tasks

Unique Problem Characteristics

- Learning
- Program Optimizer
- Code Generator
- Statistical Cost Model
- Training data
Program-aware Cost Modeling

High-Level Configuration
Program-aware Cost Modeling

High-Level Configuration

```
for y in range(8):
    for x in range(8):
        C[y][x]=0
    for k in range(8):
        C[y][x]+=A[k][y]*B[k][x]
```

Low-level Abstract Syntax Tree
(shared between tasks)
Program-aware Cost Modeling

High-Level Configuration

for y in range(8):
    for x in range(8):
        C[y][x]=0
    for k in range(8):
        C[y][x]+=A[k][y]*B[k][x]

Low-level Abstract Syntax Tree (shared between tasks)

Boosted Tree Ensembles

<table>
<thead>
<tr>
<th></th>
<th>C</th>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>y</td>
<td>64</td>
<td>64</td>
<td>64</td>
</tr>
<tr>
<td>x</td>
<td>8</td>
<td>8</td>
<td>64</td>
</tr>
<tr>
<td>k</td>
<td>1</td>
<td>8</td>
<td>8</td>
</tr>
</tbody>
</table>

- touched memory
- outer loop length
- statistical features
Program-aware Cost Modeling

High-Level Configuration

for y in range(8):
    for x in range(8):
        C[y][x] = 0
    for k in range(8):
        C[y][x] += A[k][y] * B[k][x]

Low-level Abstract Syntax Tree (shared between tasks)

Boosted Tree Ensembles

touched memory

<table>
<thead>
<tr>
<th>C</th>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>y</td>
<td>64</td>
<td>64</td>
</tr>
<tr>
<td>x</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>k</td>
<td>1</td>
<td>8</td>
</tr>
</tbody>
</table>

outer loop length

<table>
<thead>
<tr>
<th>Y</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>8</td>
</tr>
<tr>
<td>k</td>
<td>64</td>
</tr>
</tbody>
</table>

statistical features

TreeGRU

touched memory

for y in context vec of y

for x in context vec of x

for k in context vec of k

final embedding

soft scatter

outer loop length

for context vec of y

for context vec of x

for context vec of k

for context vec of k

for y in range(8):

for x in range(8):

for k in range(8):

for context vec of y

for context vec of x

for context vec of k

for context vec of y

for context vec of x

for context vec of k

for context vec of y

for context vec of x

for context vec of k

for context vec of y

for context vec of x

for context vec of k

for context vec of y

for context vec of x

for context vec of k

for context vec of y

for context vec of x

for context vec of k

for context vec of y

for context vec of x

for context vec of k

for context vec of y

for context vec of x

for context vec of k

for context vec of y

for context vec of x

for context vec of k

for context vec of y

for context vec of x

for context vec of k

for context vec of y

for context vec of x

for context vec of k

for context vec of y

for context vec of x

for context vec of k

for context vec of y

for context vec of x

for context vec of k

for context vec of y

for context vec of x

for context vec of k

for context vec of y

for context vec of x

for context vec of k

for context vec of y

for context vec of x

for context vec of k

for context vec of y

for context vec of x

for context vec of k

for context vec of y

for context vec of x

for context vec of k

for context vec of y

for context vec of x

for context vec of k

for context vec of y

for context vec of x

for context vec of k
Transfer Learning Among Different Workloads

Historical Optimization Tasks

Domain Invariant Program Representations

Transferable Models to speedup new tasks
State of Art Performance

Nvidia GPU

ARM CPU

ARM GPU
State of Art Performance

Nvidia GPU

ARM CPU

ARM GPU

In production use inside several major companies