TASO: Optimizing Deep Learning with Automatic Generation of Graph Substitutions

Zhihao Jia, Oded Padon, James Thomas, Todd Warszawski, Matei Zaharia, and Alex Aiken

Stanford University
Current Rule-based DNN Optimizations

Computation Graph

Input
- conv3x3
  - relu
  - conv3x3
    - add
      - relu
- conv1x1
  - relu

Rule-based Optimizer

Fuse conv + relu

Optimized Graph

Input
- conv3x3 + relu
  - add
    - relu
- conv1x1 + relu
  - relu
Current Rule-based DNN Optimizations

TensorFlow currently includes ~200 rules (~53,000 LOC)
When I turned on XLA (TensorFlow’s graph optimizer), the training speed is about 20% slower.

With XLA, my program is almost 2x slower than without XLA.
Limitations of Rule-based Optimizations

**Robustness**
Experts’ heuristics do not apply to all DNNs/hardware

**Scalability**
New operators and graph structures require more rules

TensorFlow currently uses ~4K LOC to optimize convolution
Limitations of Rule-based Optimizations

**Robustness**
Experts’ heuristics do not apply to all DNNs/hardware

**Scalability**
New operators and graph structures require more rules

**Performance**
Miss subtle optimizations for specific DNNs/hardware
Motivating Example

The final graph is 30% faster on V100 but 10% slower on K80.
How should we address the complexity of designing DNN graph optimizations?
TASO: Tensor Algebra SuperOptimizer

- Key idea: replace manually-designed graph optimizations with *automated generation and verification* of graph substitutions for deep learning

- **Less engineering effort**: 53,000 LOC for manual graph optimizations in TensorFlow → **1,400** LOC in TASO

- **Better performance**: outperform existing optimizers by up to **2.8x**
Graph Substitution
TASO Workflow

Operator Specifications

Graph Subst. Generator

... Candidate Substitutions

Graph Subst. Verifier

... Verified Substitutions

Graph Optimizer
TASO Workflow

Input Comp. Graph → Search-Based Graph Optimizer → Optimized Comp. Graph

Verified Substitutions
Key Challenges

1. How to generate potential substitutions?
   - Graph fingerprints

2. How to verify their correctness?
   - Operator specifications + theorem prover
Graph Substitution Generator

Enumerate all possible subgraphs up to a fixed size using available operators

~66M graphs with up to 4 operators
Graph Substitution Generator

Compute output fingerprints with random input tensors
Graph Substitution Generator

Pairs of graphs with identical fingerprint are candidate substitutions
Graph Substitution Generator

TASO generates \(~\textbf{29,000}\) substitutions by enumerating graphs w/ up to 4 operators

\textbf{743} substitutions remain after applying pruning techniques to eliminate redundancy
Graph Substitution Verifier

- **Candidate Substitutions**
- **Verified Substitutions**

**P1. conv is distributive over concatenation**

**P2. conv is bilinear**

... 

**Pn.**

∀x, w₁, w₂.

Conv(x, Concat(w₁, w₂)) = Concat(Conv(x, w₁), Conv(x, w₂))

**Operator Specifications**
Verification Workflow

∃x, w₁, w₂ .
(Conv(x, w₁), Conv(x, w₂))
≠ Split(Conv(x, Concat(w₁, w₂)))

P1. ∀x, w₁, w₂ .
Conv(x, Concat(w₁, w₂)) = 
Concat(Conv(x, w₁), Conv(x, w₂))

P2. ...

Operator Specifications

Theorem Prover

UNSAT
Verification Effort

TASO generates all 743 substitutions in 5 minutes, and verifies them against 43 operator properties in 10 minutes.

Supporting a new operator requires a few hours of human effort to discover its properties.

Operator specifications in TASO \(\approx 1,400\) LOC
Manual graph optimizations in TensorFlow \(\approx 53,000\) LOC
Search-Based Graph Optimizer (MetaFlow [SysML19])

- **Goal**: applying verified substitutions to obtain an optimized graph

- **Cost model**
  - Based on the sum of individual operators’ cost
  - Measure the cost of each operator on hardware

- **Cost-based backtracking search**
  - Backtrack local optimal solutions
  - Optimizing a DNN model takes less than 10 minutes
End-to-end Inference Performance (V100 GPU w/ cuDNN)

- ResNet-50
- NasNet-A
- ResNeXt-50
- NasRNN
- BERT-Large

TensorFlow 1.0x, TensorRT 1.3x, MetaFlow 2.8x, TASO 1.4x

Competitive on standard models
Larger speedups on emerging models
Similar speedups on the TVM backend
Different DNN models require **different** substitutions.
Conclusion

TASO is the first DNN optimizer that automatically generates substitutions
• Less engineering effort
• Better performance
• Formal verification

https://github.com/jiazhihao/taso
• Support DNN models in ONNX, TensorFlow, and PyTorch