EINNET: Optimizing Tensor Programs with Derivation-Based Transformations

Liyan Zheng, Haojie Wang, Jidong Zhai, Muyan Hu, Zixuan Ma, Tuowei Wang, Shuhong Huang, Xupeng Miao, Shizhi Tang, Kezhao Huang, Zhihao Jia

MOTIVATION

Tensor program transformations

- Optimize program performance
- Preserve program outputs

Automatic program optimizers

Superoptimization-based approaches

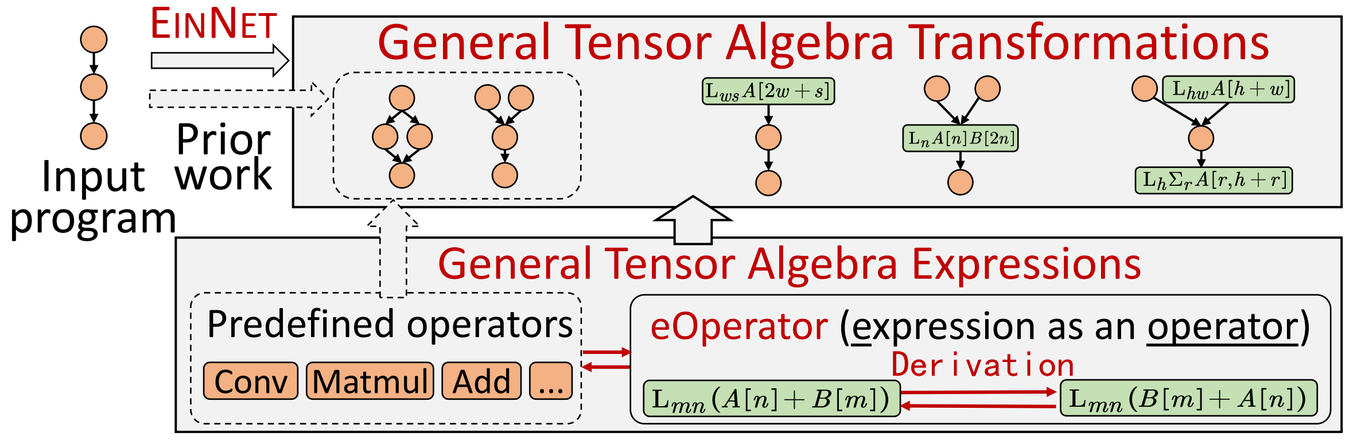
- Step I: enumerate candidate programs by predefined operators → limited space
- Step II: verify candidate programs with the original program → time-consuming

Derivation-based optimizer (our work)

🚀 Proposed technique: tensor expression derivation

🌀 Larger search space: tensor algebra transformations

😊 Better performance: up to 2.7x speedup



APPROACH

Tensor algebra expressions

Specify computation semantics mathematically

Nested expressions for multiple operators

$$B[n,f,h,w]=\text{Conv}(A,K)=\underset{nhwf}{L}\underset{crs}{\Sigma}A[n,c,h+r,w+s]K[r,s,f,c]$$

- Traversal notation notates the output shape
- Traversal and summation iterators with ranges
- Accessing elements with symbolic algebra expressions

Expression derivation

Mathematically equivalent rewrite 👉

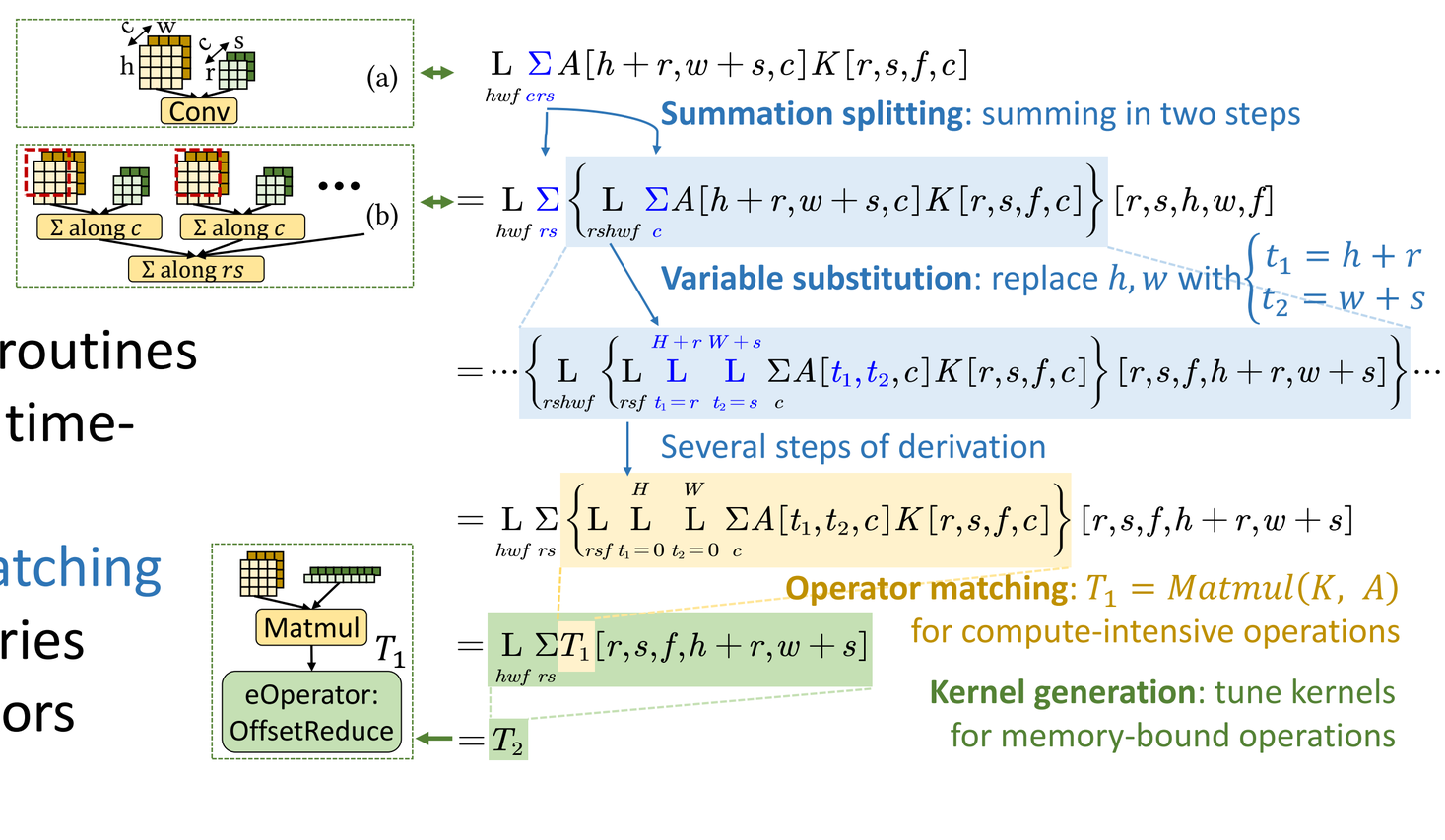
Expression execution

Different execution strategies

- Math libraries: efficient but only fixed routines
- Kernel generators: flexible but require time-consuming tuning

Get an ideal combination via operator matching

- Compute-intensive operations → libraries
- Memory-bound operations → generators



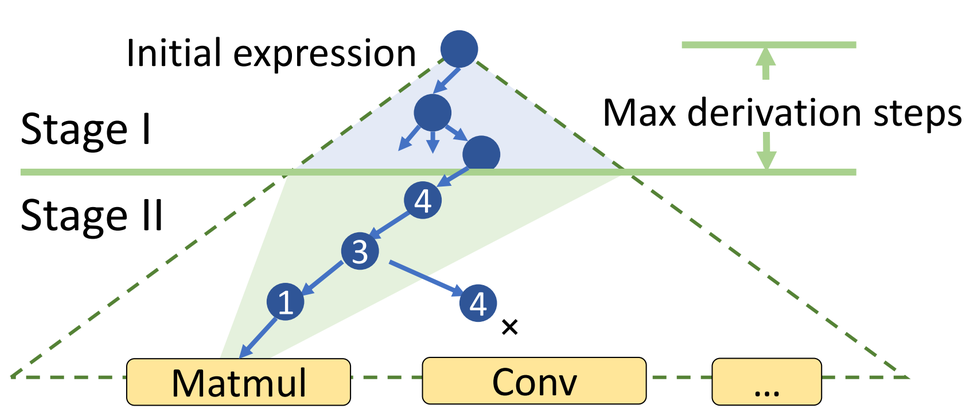
Search expression transformation space

Stage I: enlarge search space

- Apply all derivation rules under a search depth limit

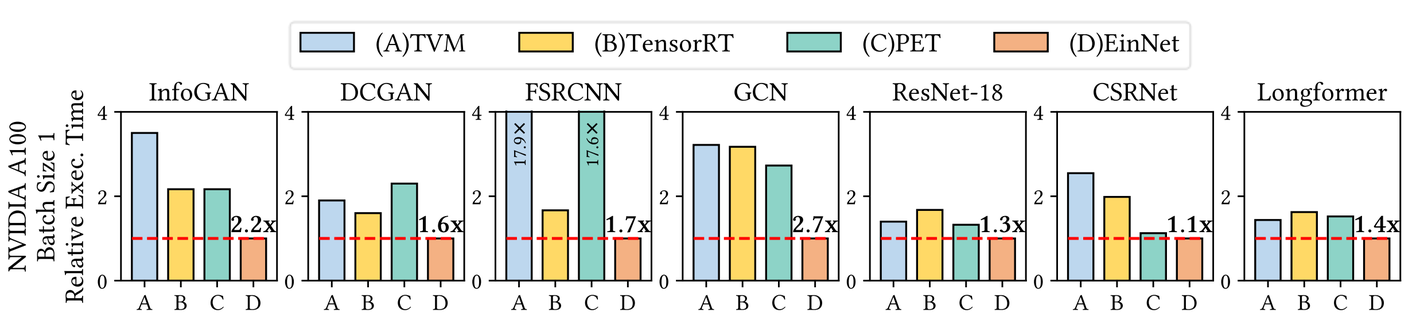
Stage II: opportunistically leverage math libraries

- Reduce expression distance to target



EVALUATION

End-to-end inference (up to 2.7x speedup)



Effective on different backends