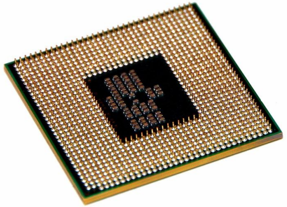
Google DeepMind

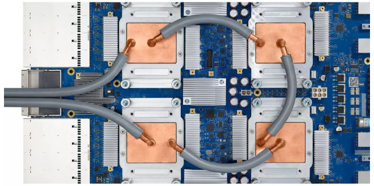
# ML for ML Compilers at Google

Phitchaya “Mangpo” Phothilimthana
mangpo@google.com

*Presenting the work of many people at Google*

# Production ML Compilation Stack at Google





# **Goal**:

**automatically** select optimal compiler configurations, **at scale** for all **ML workloads** in **Google's fleet**

# Compiler & Autotuner

## Compiler

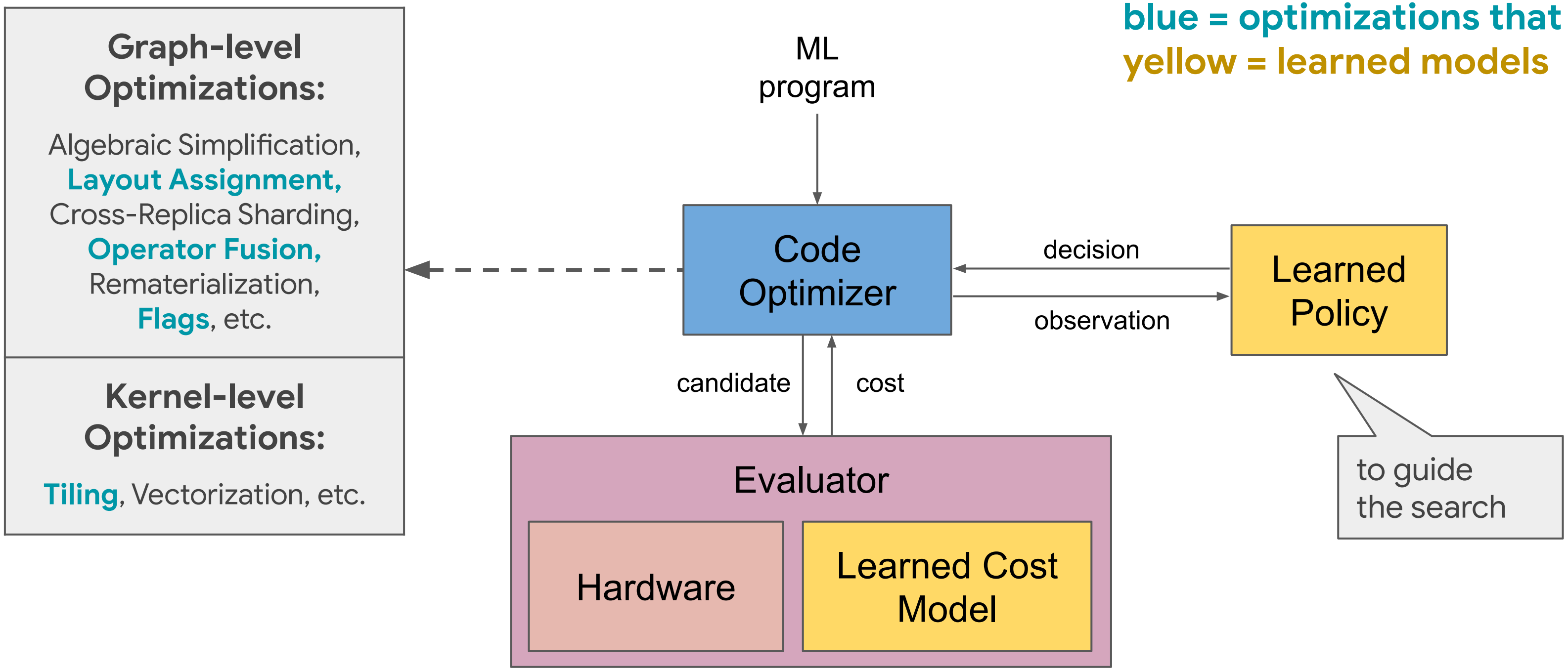
- **Transforms program** written in high-level language to low-level representation
- **Optimizes** program for performance **through heuristics** (often in polynomial time)

## Autotuner

- **Aids compiler** to find better optimization decisions
- **Searches** a space of configurations of a program
- **Selects the best configuration** according to a performance metric

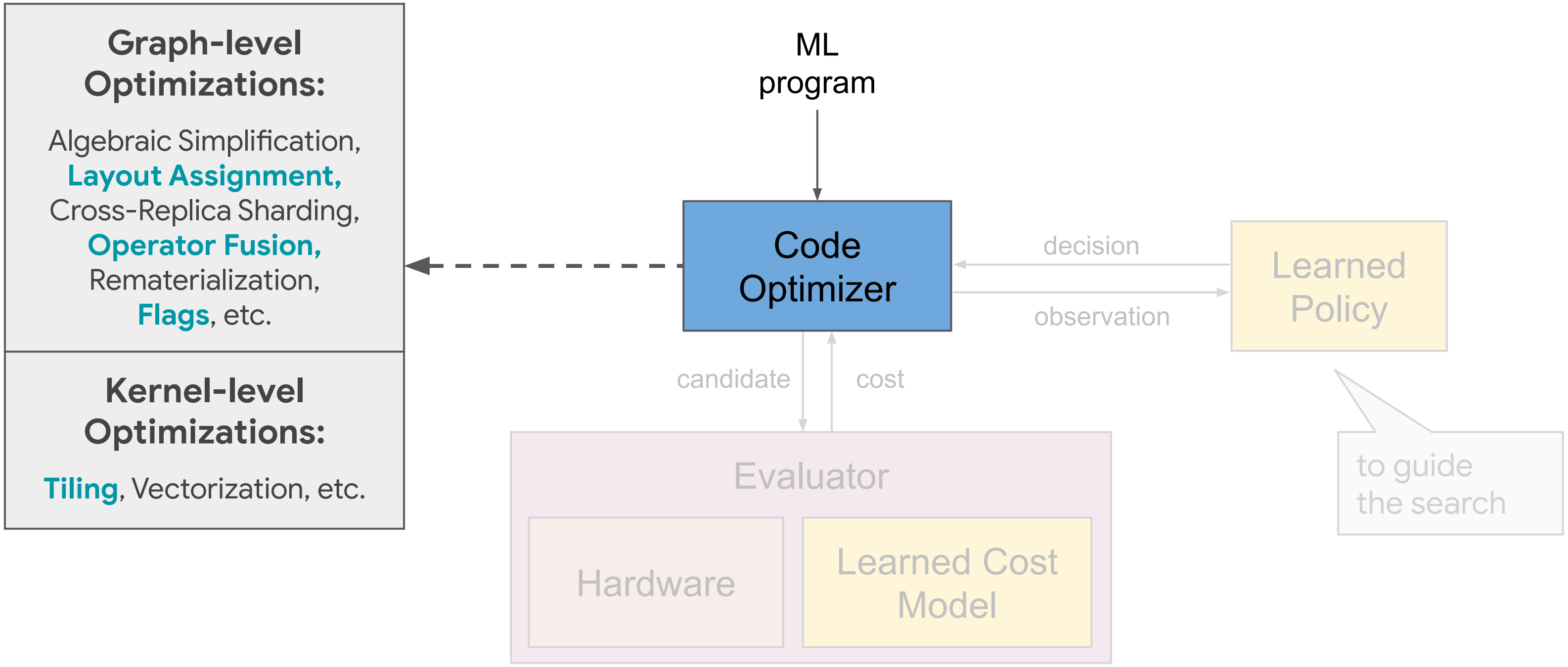
# XTAT: XLA TPU Autotuner

blue = optimizations that we tune
yellow = learned models



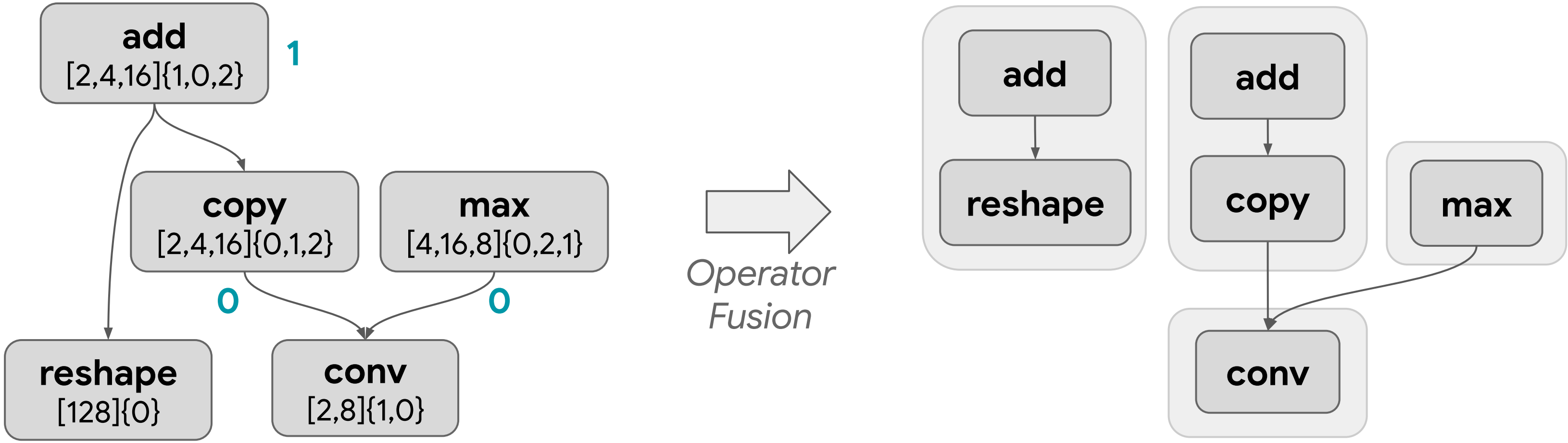

*Ref: Phothilimthana et al., A Flexible Approach to Autotuning Multi-Pass Machine Learning Compilers, PACT 2021.*

# XTAT: XLA TPU Autotuner



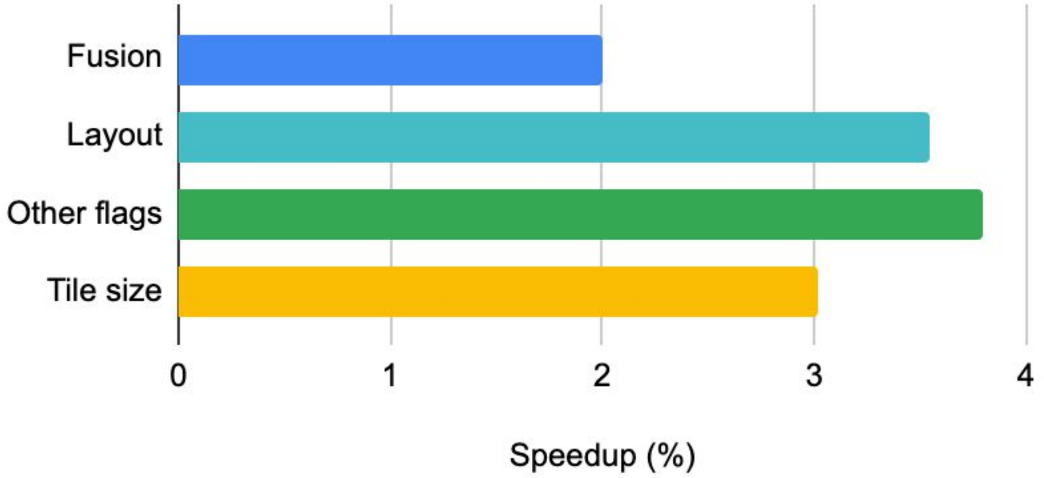
# Operator Fusion

## Example:



# Runtime Speedup

Average **speedup on top workloads** at Google




*on top 300 workloads on TPUs v4*

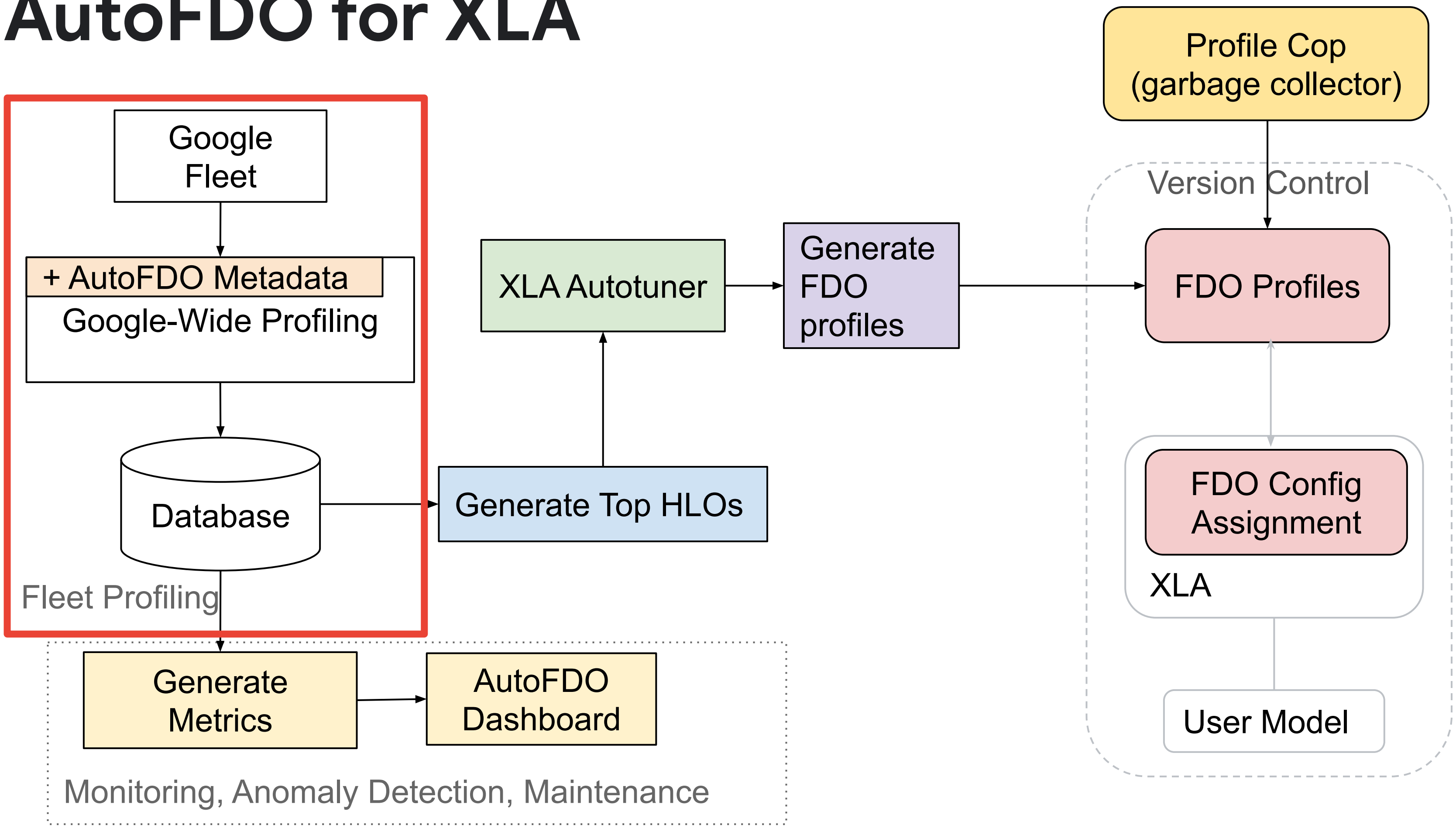
*on top 75% of workloads on TPUs v3**

Delivered **5-25% speedup** on **important production models** by tuning flags

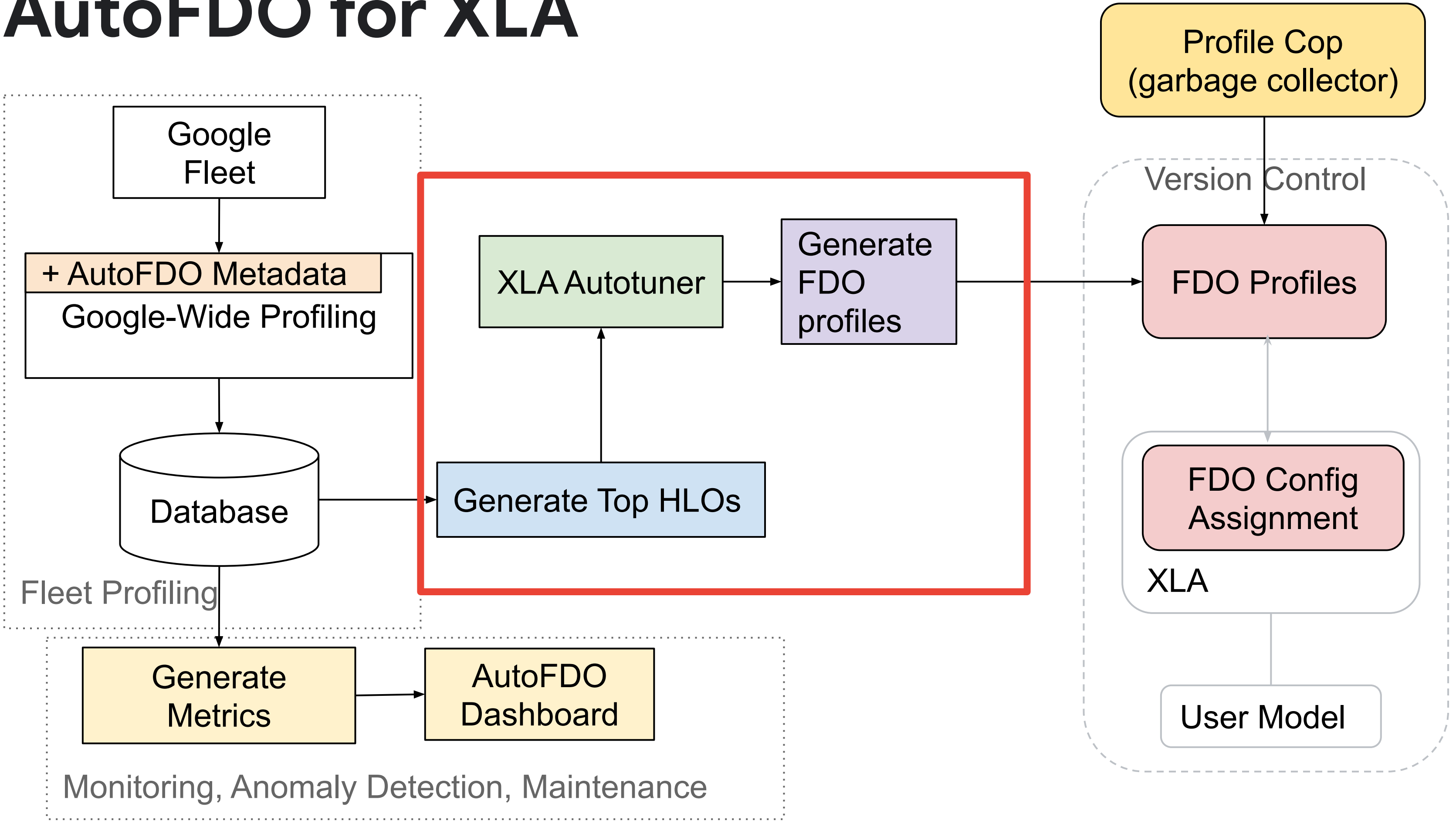
Google DeepMind

# Data-Center Scale Deployment

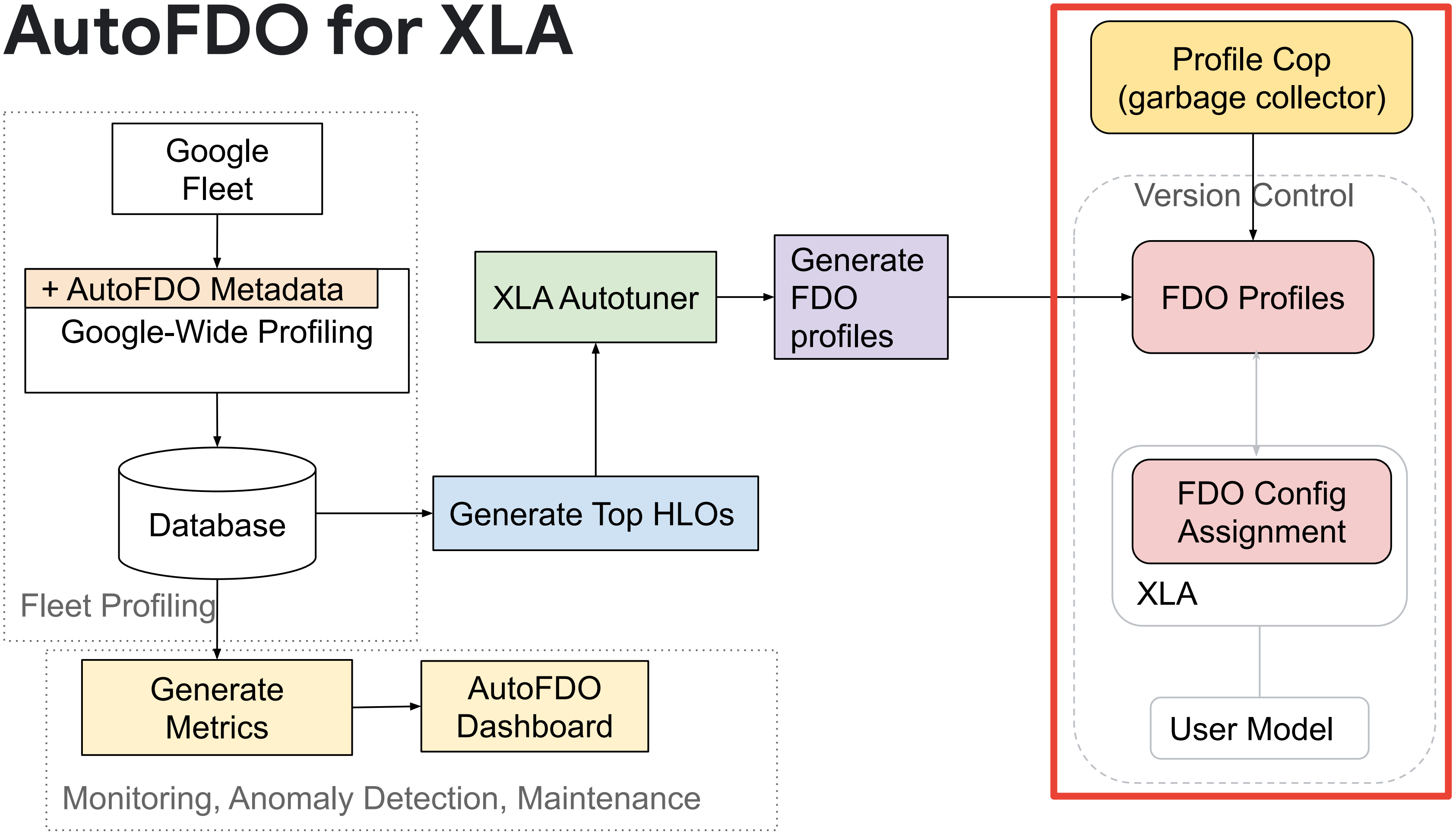
# AutoFDO for XLA



# AutoFDO for XLA



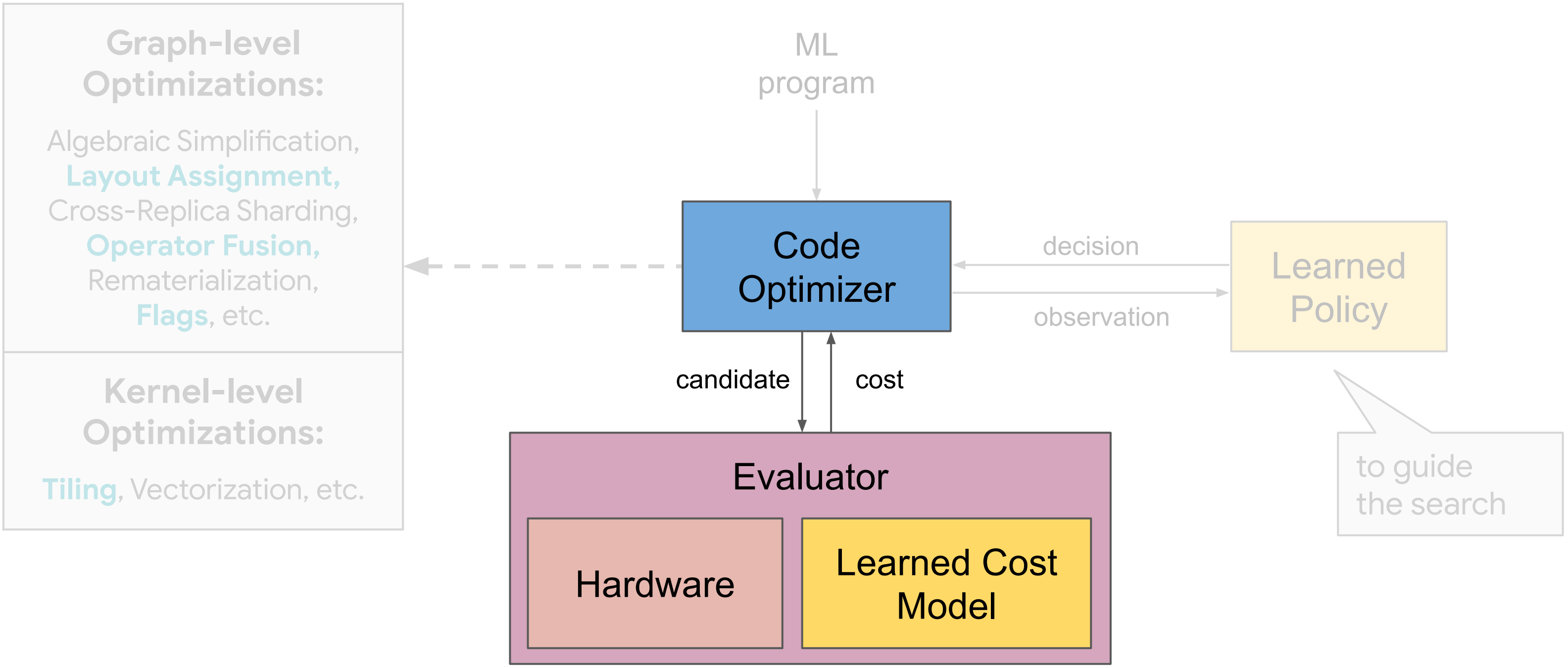
# AutoFDO for XLA



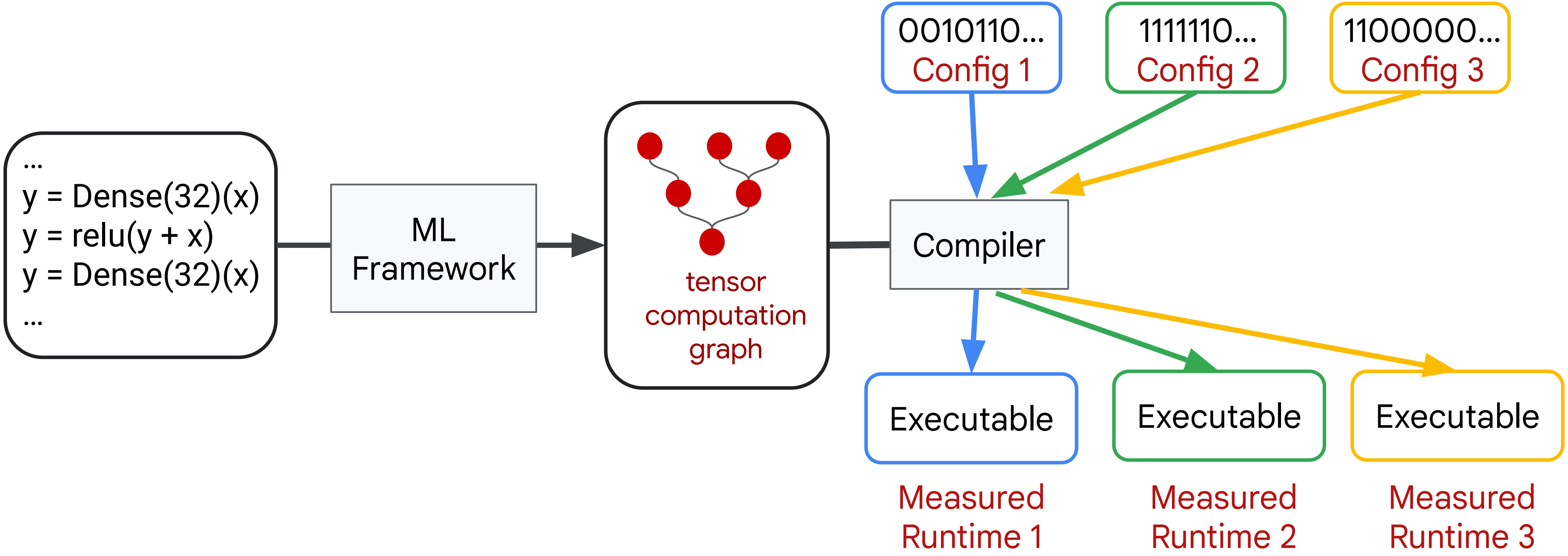
# AutoFDO for XLA

- Have deployed the **tile size autotuning** to optimize top workloads in the TPU fleet daily
- **Save ~2%** of total TPU consumption
- Savings / tuning cost: ~15x
- **Learned cost model** enabled tuning 20x more kernels per day

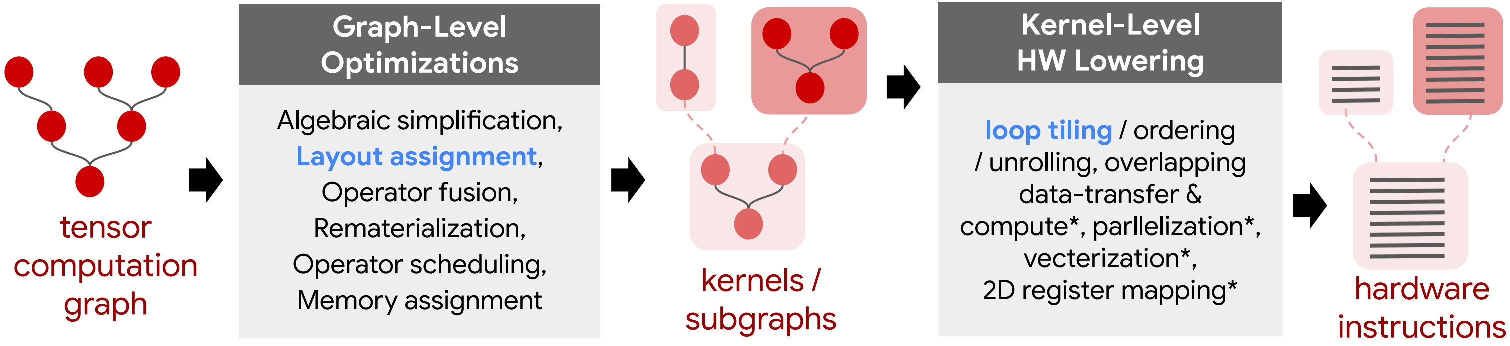
# Learned Cost Model



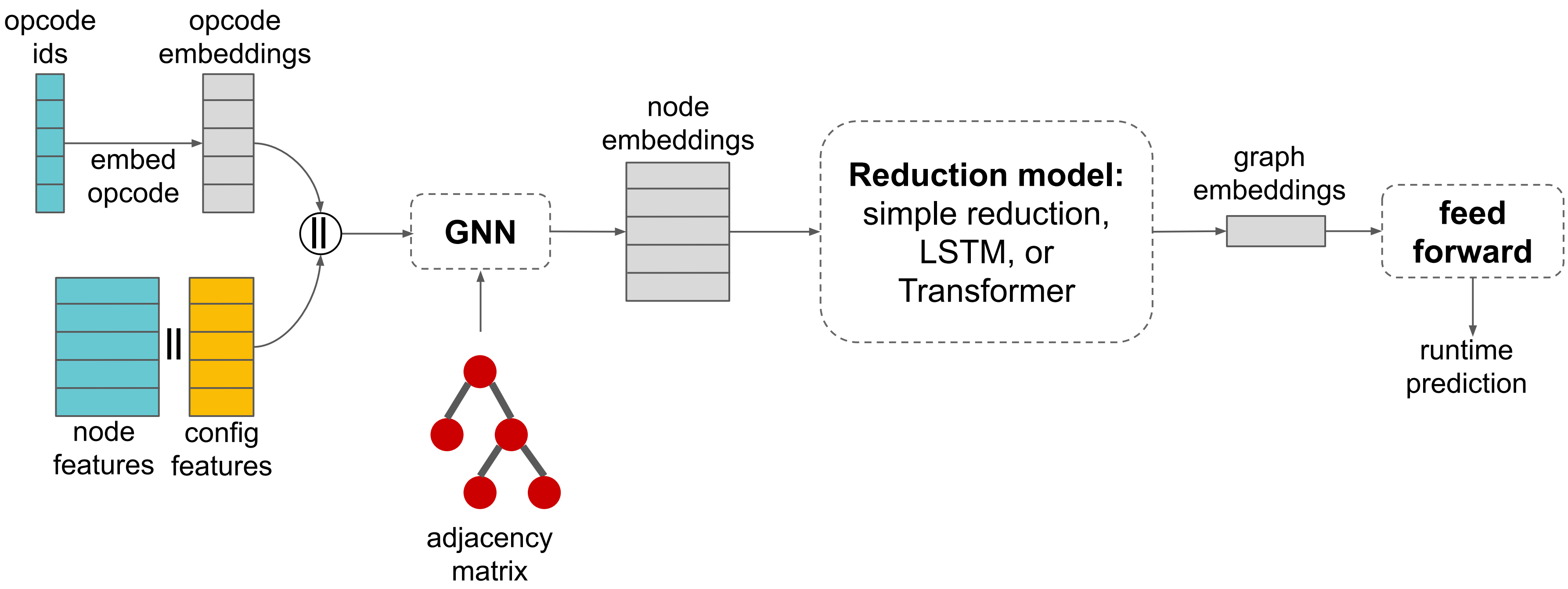
# Task: Config Ranking



# Target Optimizations

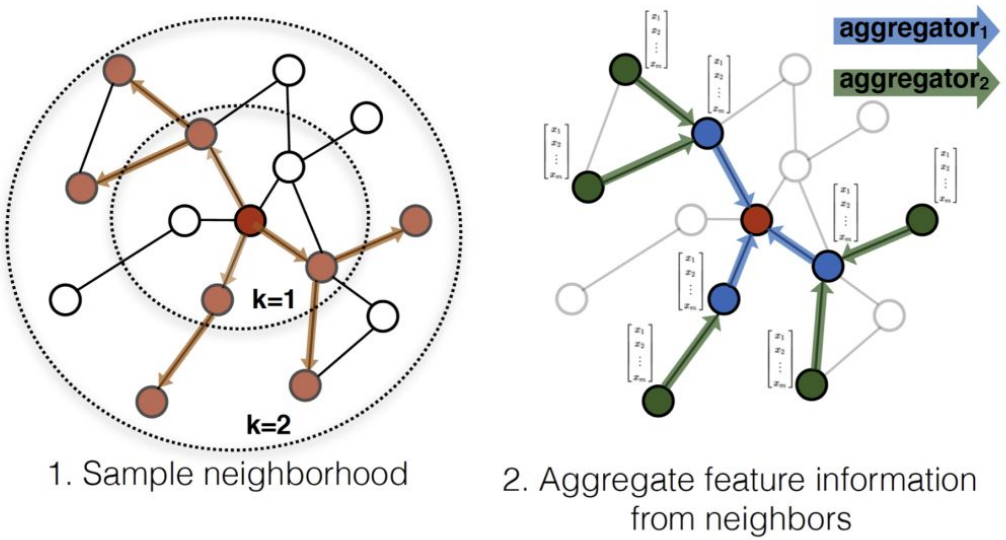


# Model Architecture




*Ref: Kaufman and Phothilimthana et al., A Learned Performance Model for Tensor Processing Units, MLSys 2021.*

# GraphSage




*Ref: Hamilton and Ying et al., Inductive Representation Learning on Large Graphs, NeurIPS 2017.*

## Node embedding:

$$\varepsilon_i^k = l_2\left(f_3^k\left(concat\left(\varepsilon_i^{k-1}, \sum_{j\in neighbors(i)} f_2^k(\varepsilon_j^{k-1})\right)\right)\right)$$

$f$: feedforward
$l_2$: L2 norm

# Losses

**Mean Squared Error**
for absolute runtime prediction.
Targets are log-transformed.

$$L = \sum_{i=1}^{n} (y_i' - y_i)^2$$

**Pairwise Rank Loss**

for relative runtime prediction.

$$L = \sum_{i=1}^{n} \sum_{j=1}^{n} \frac{\phi(y_i' - y_j') \cdot pos(y_i - y_j)}{n \cdot (n-1)/2}$$

$$\phi(z) = \begin{cases} (1-z)_+ & \text{hinge function } \textbf{or} \\ log(1+e^{-z}) & \text{logistic function} \end{cases}$$

$$pos(z) = \begin{cases} 1 & \text{if } z > 0 \\ 0 & \text{otherwise} \end{cases}$$

# Evaluation Metrics

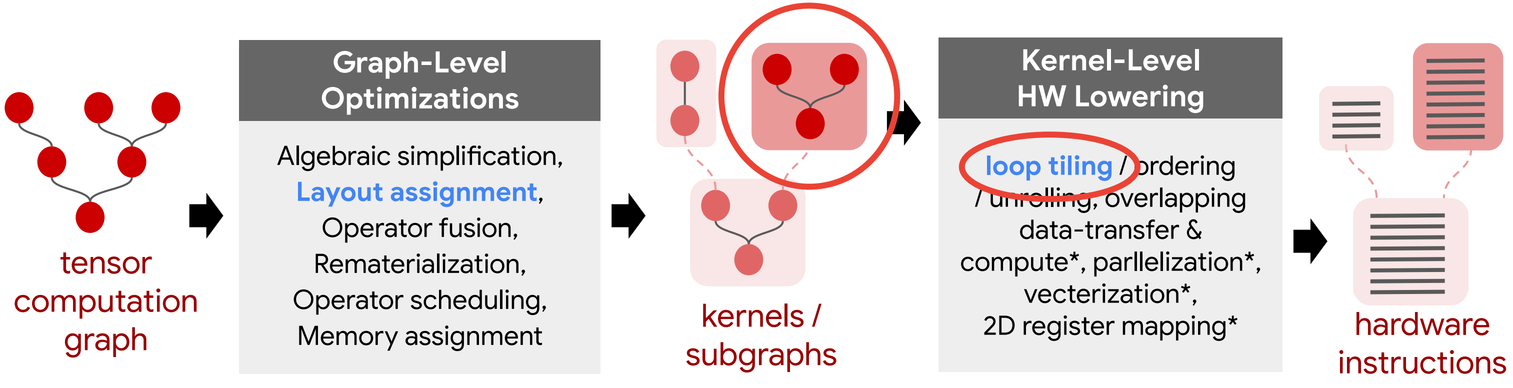
**Top-K Error**: slow down compared to optimal

$$\frac{\text{The best runtime of the top-k predictions}}{\text{The best runtime of all configurations}} - 1 = \frac{\min_{i \in K} y_i}{\min_{i \in A} y_i} - 1$$

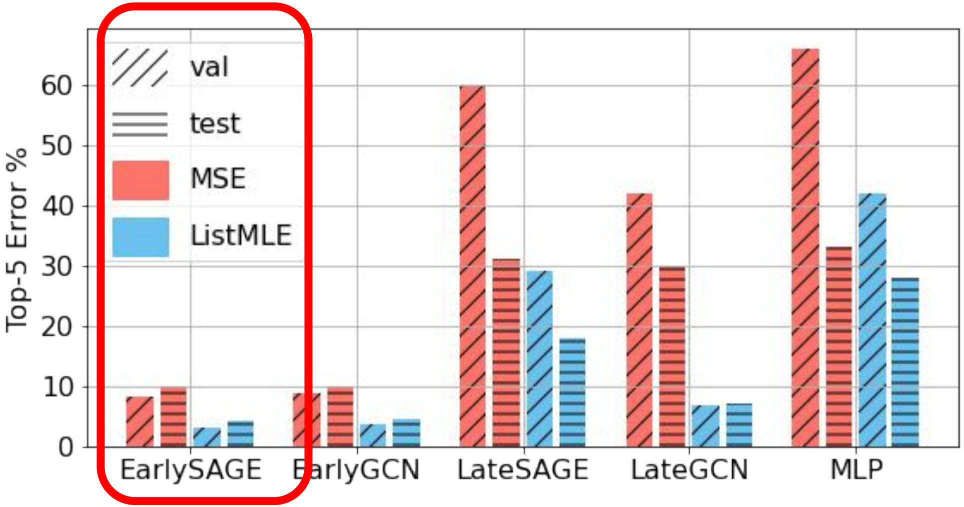
**Ranking Correlation**: ability to guide the search

```
Kendall-Tau(model rank, gound-truth rank)
```

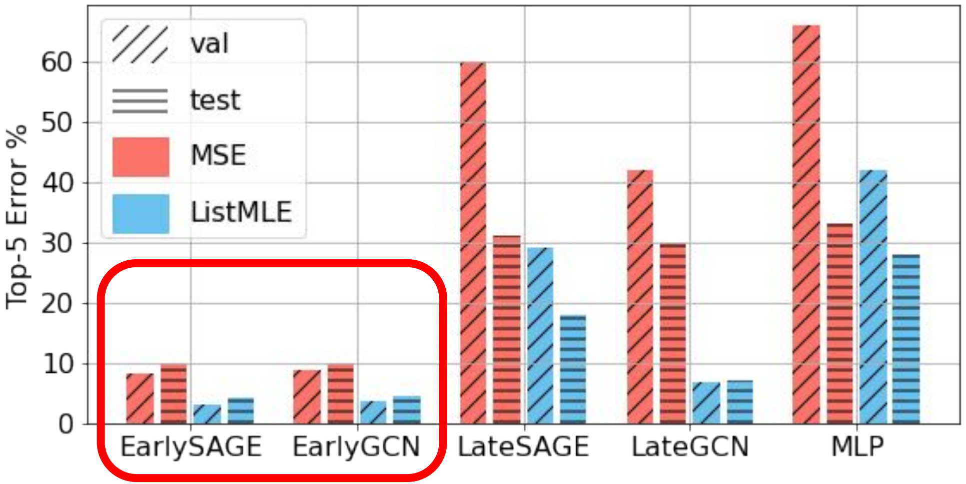
# Tile Size Runtime Prediction (Kernel Level)



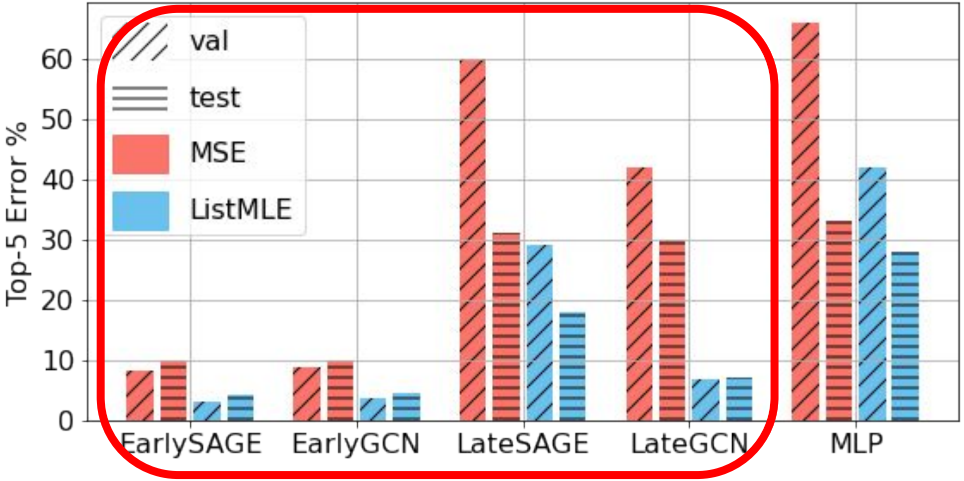
# Results: Top-K Error

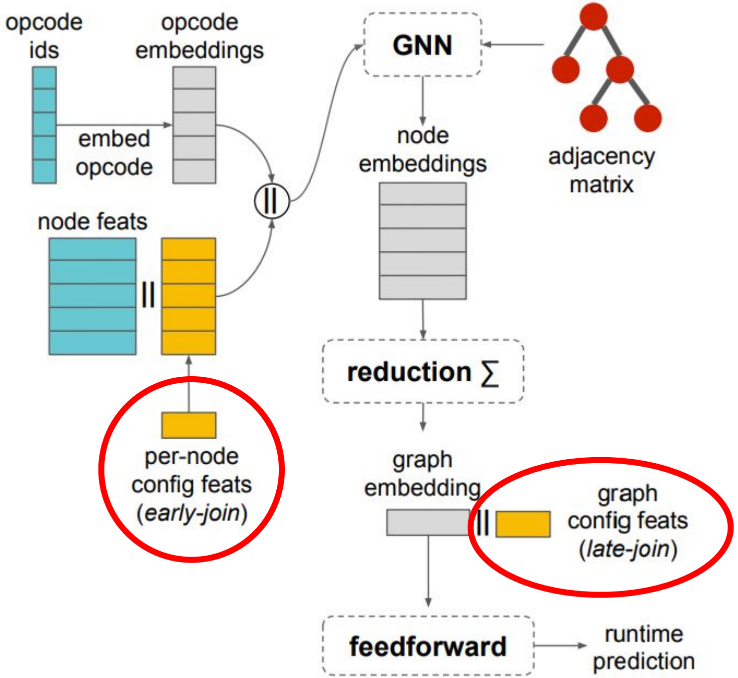


# Results: Top-K Error

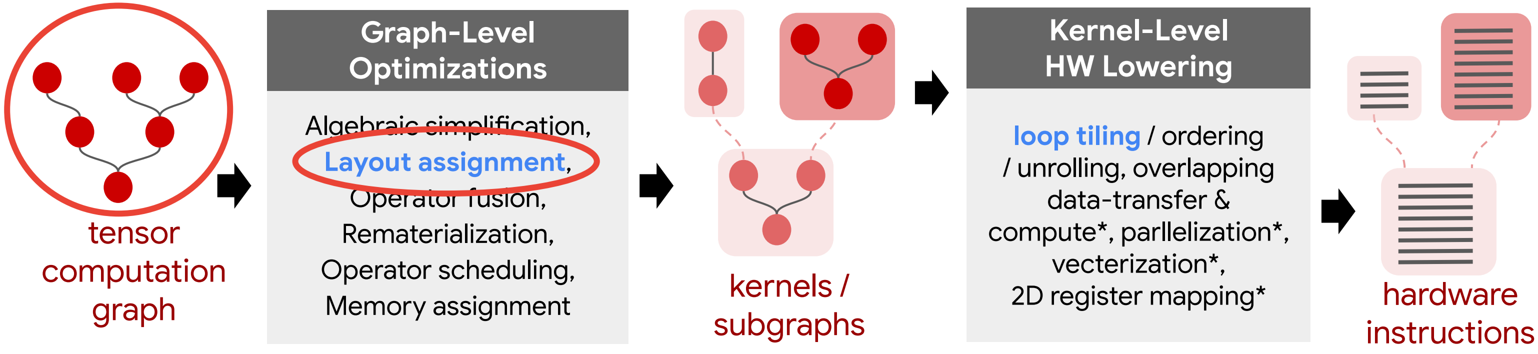


# Results: Top-K Error



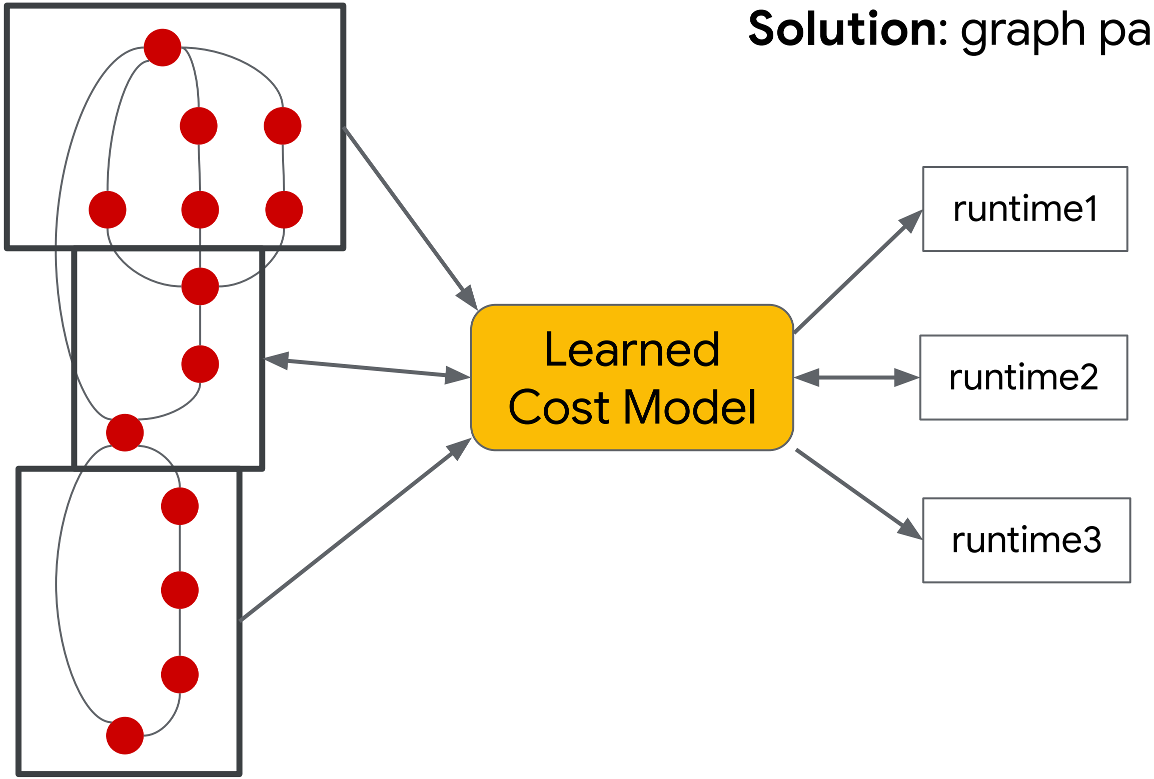



# Layout Runtime Prediction (Graph Level)



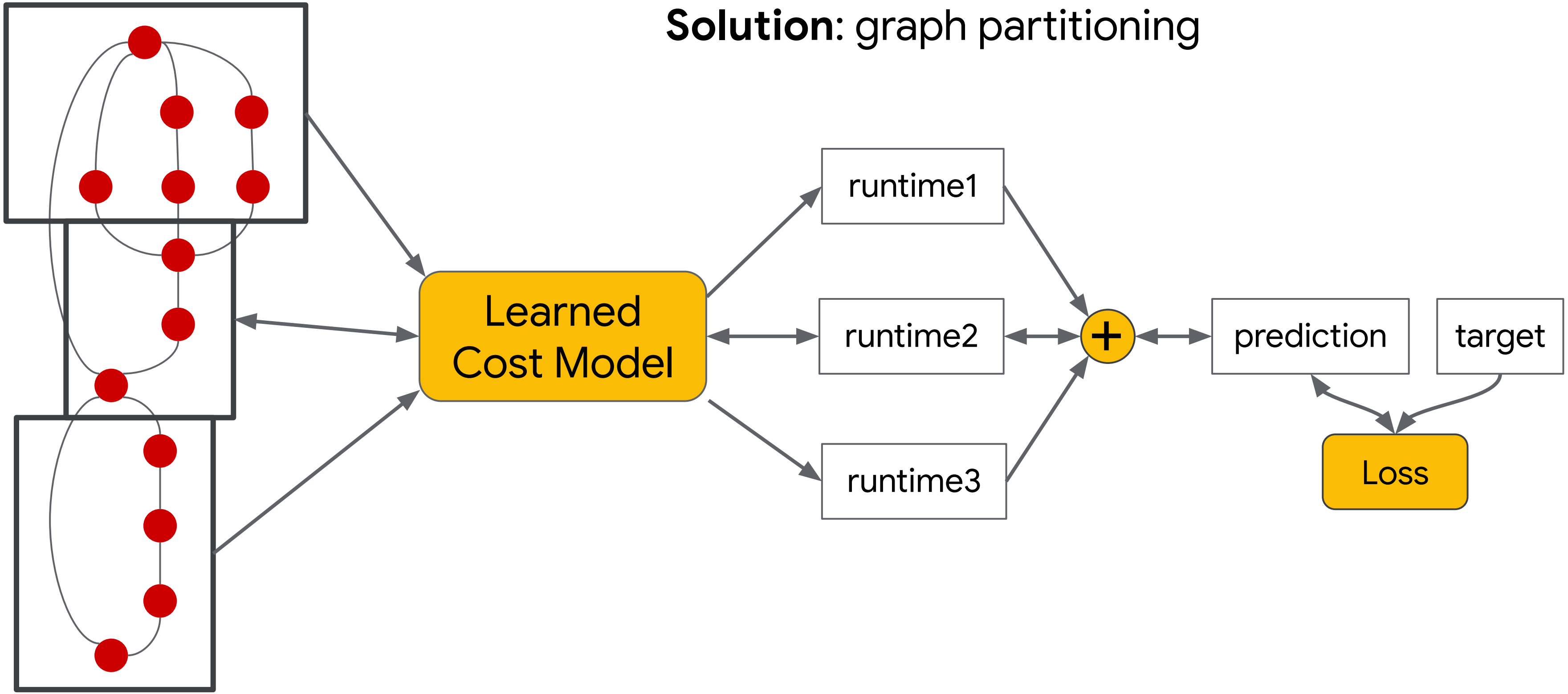
# Challenge 1: HLO graphs are huge! (up to 500k nodes)

**Solution**: graph partitioning



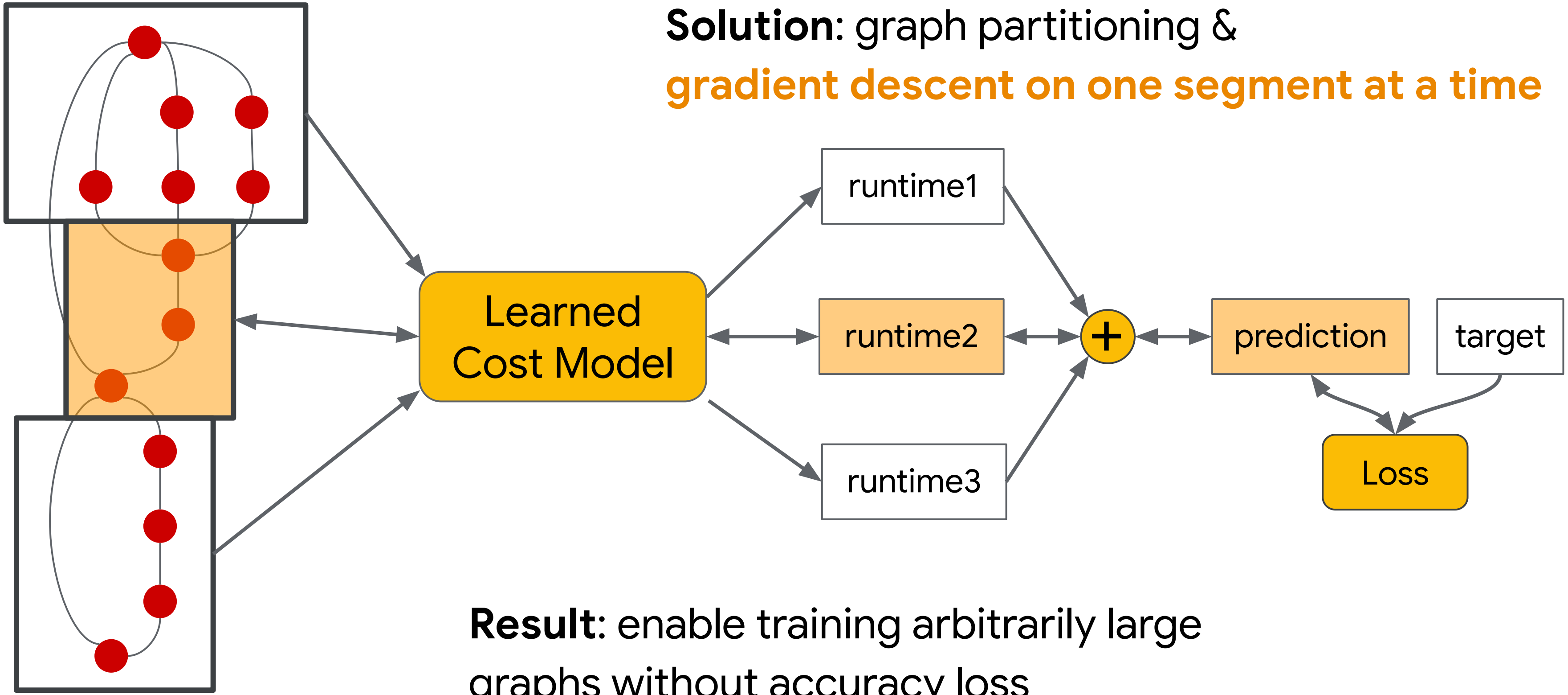
# Challenge 1: HLO graphs are huge! (up to 500k nodes)

**Solution**: graph partitioning



# Challenge 1: HLO graphs are huge! (up to 500k nodes)

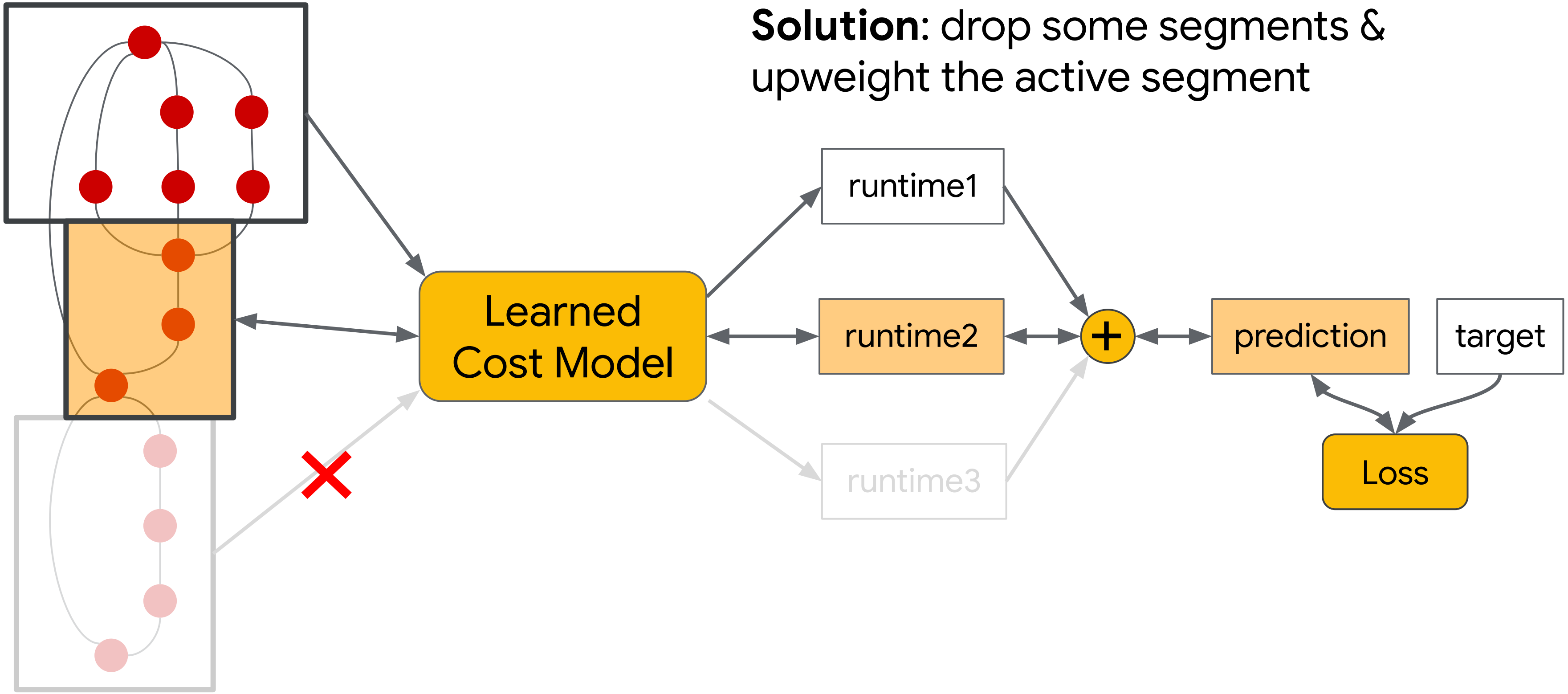
**Solution**: graph partitioning &
**gradient descent on one segment at a time**




**Result**: enable training arbitrarily large graphs without accuracy loss

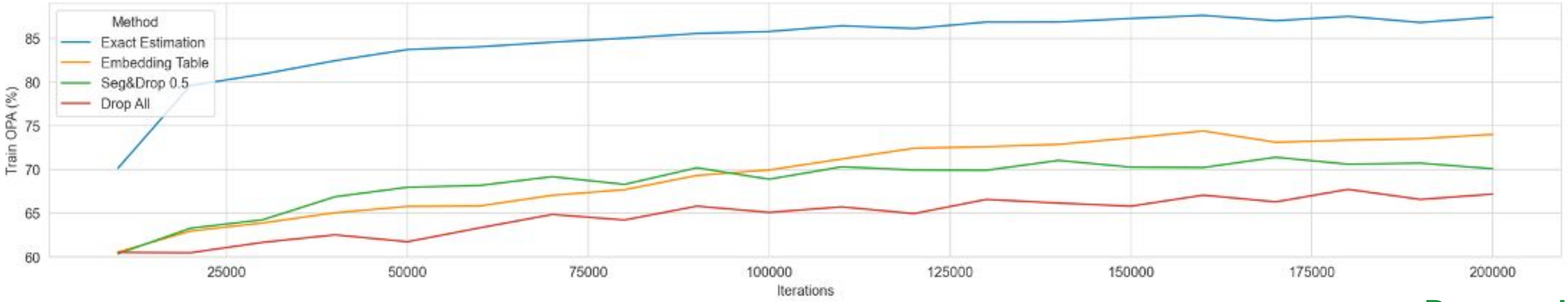
# Challenge 2: HLO graphs are very diverse

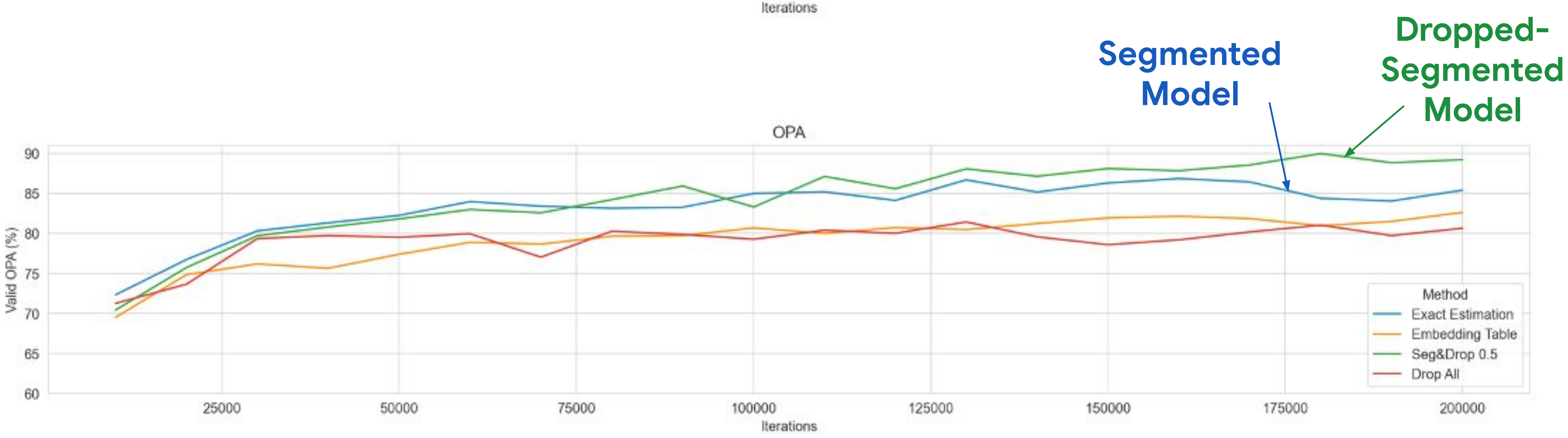
**Solution:** drop some segments & upweight the active segment



# Result: Better Generalization

OPA = Ordered Pair Accuracy



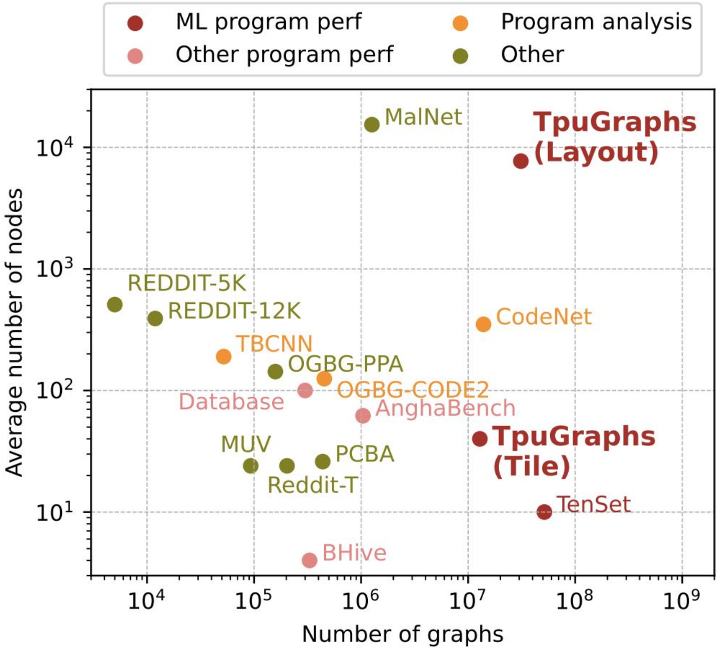




*Ref: Cao et al., Learning Large Graph Property Prediction via Graph Segment Training, NeurIPS 2023*

# Ablation Study: Top-K Error

| Model | Top-1 E | | Top-10 E | | Top-100 E | |
|---|---|---|---|---|---|---|
| | Val | Test | Val | Test | Val | Test |
| Best | 24.3 | 25.3 | 6.4 | 10.4 | 0.4 | 1.2 |
| Full Graph | 34.3 | 39.6 | 11.5 | 14.9 | 0.7 | 2.6 |
| Small Segment | 37.9 | 47.3 | 13.3 | 17.9 | 1.4 | 3.5 |
| Topo Partition | 27.5 | 27.1 | 6.5 | 10.1 | 0.6 | 1.5 |
| Fewer Layers | 26.9 | 28.2 | 7.9 | 12.5 | 0.7 | 1.7 |
| MSE loss | 42.7 | 53.1 | 12.6 | 18.8 | 1.6 | 3.8 |
| Random | 58.1 | 90.5 | 15.7 | 20.6 | 1.8 | 3.6 |

# TpuGraphs dataset




*Ref: Phothilimthana et al., TpuGraphs: A Performance Prediction Dataset on Large Tensor Computational Graphs, NeurIPS 2023*

# TpuGraphs dataset

| Collection {opt}:{src}:{space} | Avg # of Nodes | # of Graphs + Configs |
|---|---|---|
| Layout:XLA:Default | 14,105 (372 - 43,614) | 771,496 |
| Layout:XLA:Random | | 908,561 |
| Layout:NLP:Default | 5,659 (876-21,919) | 13,285,415 |
| Layout:NLP:Random | | 16,125,781 |
| Tile:XLA | 40 | 12,870,077 |

*Ref: Phothilimthana et al., TpuGraphs: A Performance Prediction Dataset on Large Tensor Computational Graphs, NeurIPS 2023*

# TpuGraphs dataset

**Dataset:** github.com/google-research-datasets/tpu_graphs

**Competition:** kaggle.com/competitions/predict-ai-model-runtime

- Final submission deadline: **November 17**
- Total prizes: **$50,000**
- Winners will be invited to present at ML for Systems Workshop @ NeurIPS

Google DeepMind

# References


Phothilimthana et al., **A Flexible Approach to Autotuning Multi-Pass Machine Learning Compilers**, PACT 2021.

Kaufman and Phothilimthana et al., **A Learned Performance Model for Tensor Processing Units**, MLSys 2021.

Cao et al., **Learning Large Graph Property Prediction via Graph Segment Training**, NeurIPS 2023.

Phothilimthana et al., **TpuGraphs: A Performance Prediction Dataset on Large Tensor Computational Graphs**, NeurIPS 2023.