Google AI

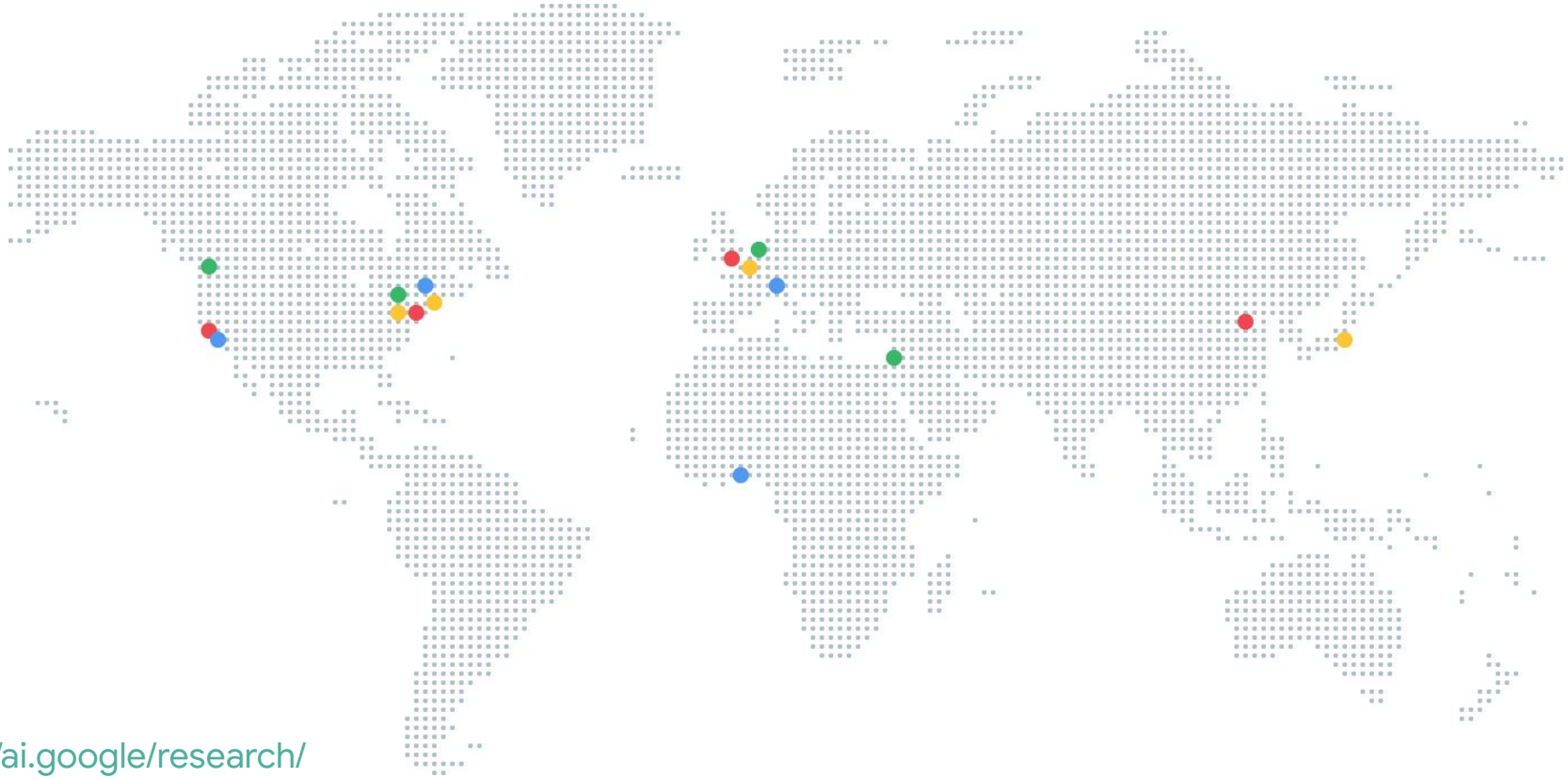
ML for ML Compilers at Google

Mangpo Phothilimthana
Research Scientist
mangpo@google.com

Presenting the work of **many** people at Google

Google Research

Google Research Overview



http://ai.google/research/
http://ai.google/research/teams

Our Approach

Cutting-edge Research

Help others Innovate

Smarter Products

Solving for Humanity's
Big Challenges

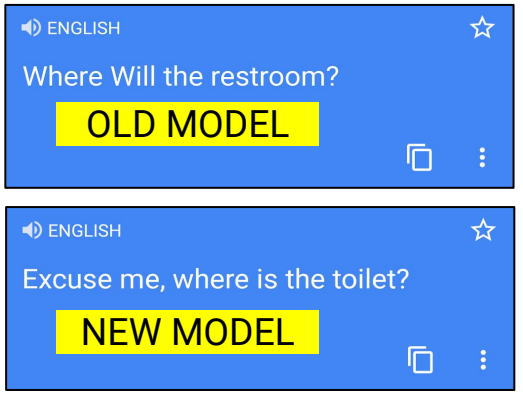
2014

Sequence to Sequence Learning with Neural Networks

Ilya Sutskever
Google
ilyasu@google.com

Oriol Vinyals
Google
vinyals@google.com

Quoc V. Le
Google
qvl@google.com



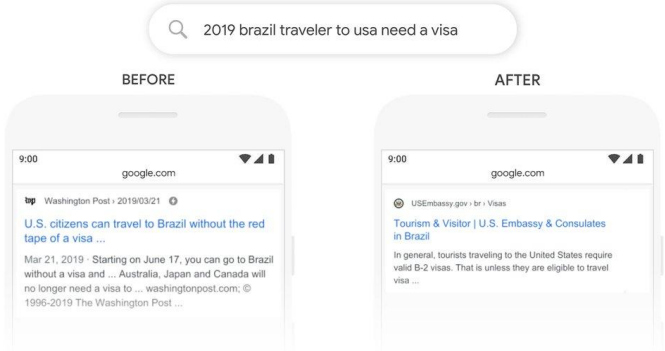
2015

2016

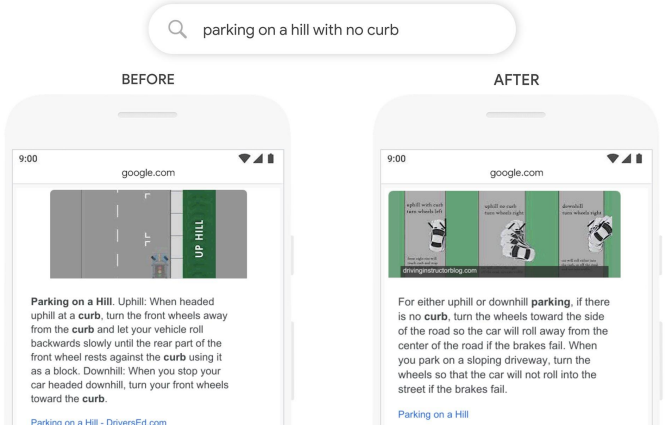
BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

Jacob Devlin **Ming-Wei Chang** **Kenton Lee** **Kristina Toutanova**
Google AI Language
{jacobdevlin,mingweichang,kentonl,kristout}@google.com

2018 - Open Sourced : Paper, Code, Trained Model
State-of-the-art in 11 different NLP tasks!!



2019



https://www.blog.google/products/search/search-language-understanding-bert/

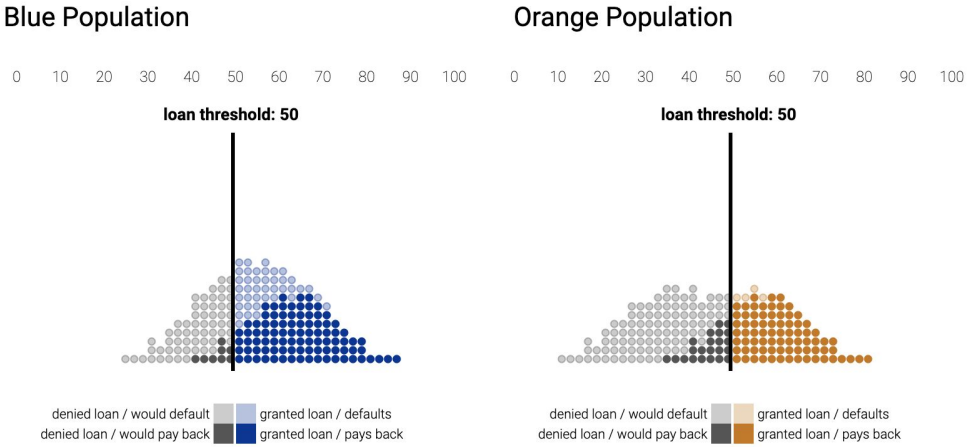
Fairness & Ethical AI

Fairness

Equality of Opportunity in ML

Hardt *et al.*

October 07, 2016



Google AI Blog ai.googleblog.com/2016/10/equality-of-opportunity-in-machine.html

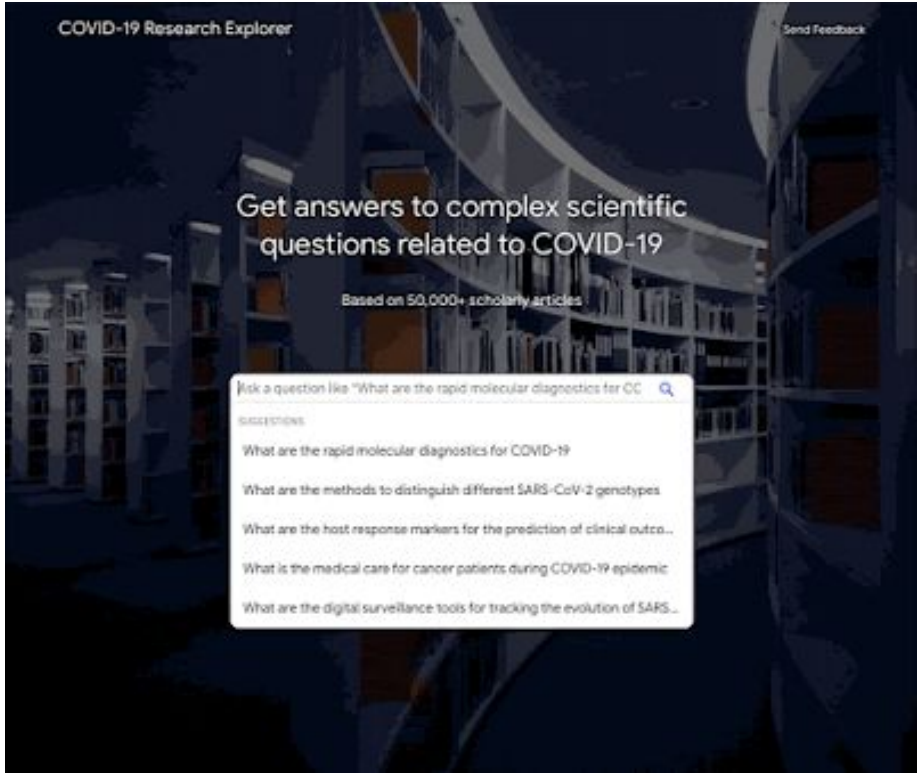
Fairness & Ethical AI Papers

- Diversity and Inclusion Metrics in Subset Selection. (Mitchell et al. 2020)
- Towards an Ethically-Informed Approach to Facial Recognition Auditing. (Raji et al. 2020)
- Towards a critical race methodology in algorithmic fairness. (Hanna et al. 2020)
- Lessons from archives: strategies for collecting sociocultural data in machine learning. (Jo and Gebru, 2020)
- Closing the AI Accountability Gap: Defining an End-to-End Framework for Internal Algorithmic Auditing. (Raji et al. 2020)
- Advances and open problems in federated learning. (Kairouz et al. 2020)
- Perturbation Sensitivity Analysis to Detect Unintended Model Biases. (Prabhakaran et al. 2019)
- Model Cards for Model Reporting. (Mitchell et al. 2019)
- InclusiveFaceNet: Improving Face Attribute Detection with Race and Gender Diversity. (Ryu et al. 2018)
- Mitigating Unwanted Biases with Adversarial Learning. (Zhang et al. 2018)
- Text Embedding Models Contain Bias. Here's Why That Matters. (Packer et al. 2018)
- Do algorithms reveal sexual orientation or just expose our stereotypes? (Blog post, Arcas et al. 2018)

https://developers.google.com/machine-learning/fairness-overview/

COVID-19 Responses

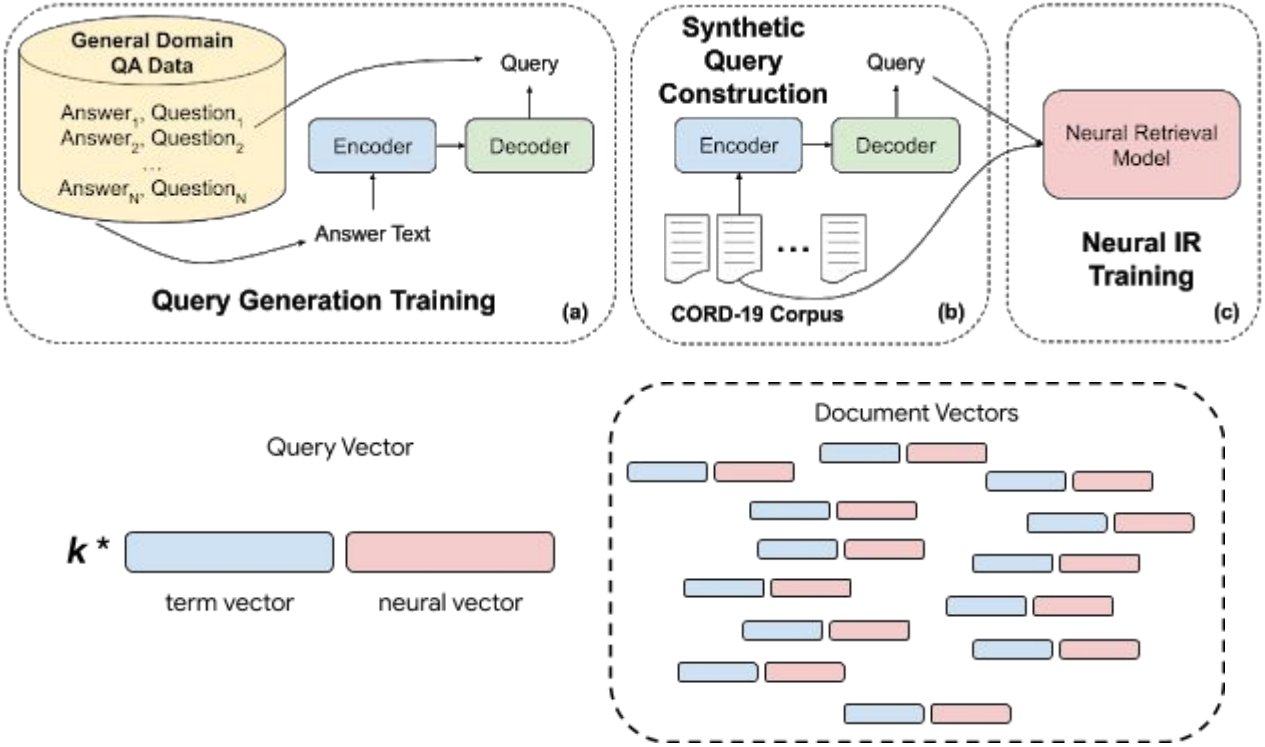
COVID-19 Research Explorer



Enables researchers to search papers related to COVID-19 using complex natural language queries.

https://covid19-research-explorer.appspot.com

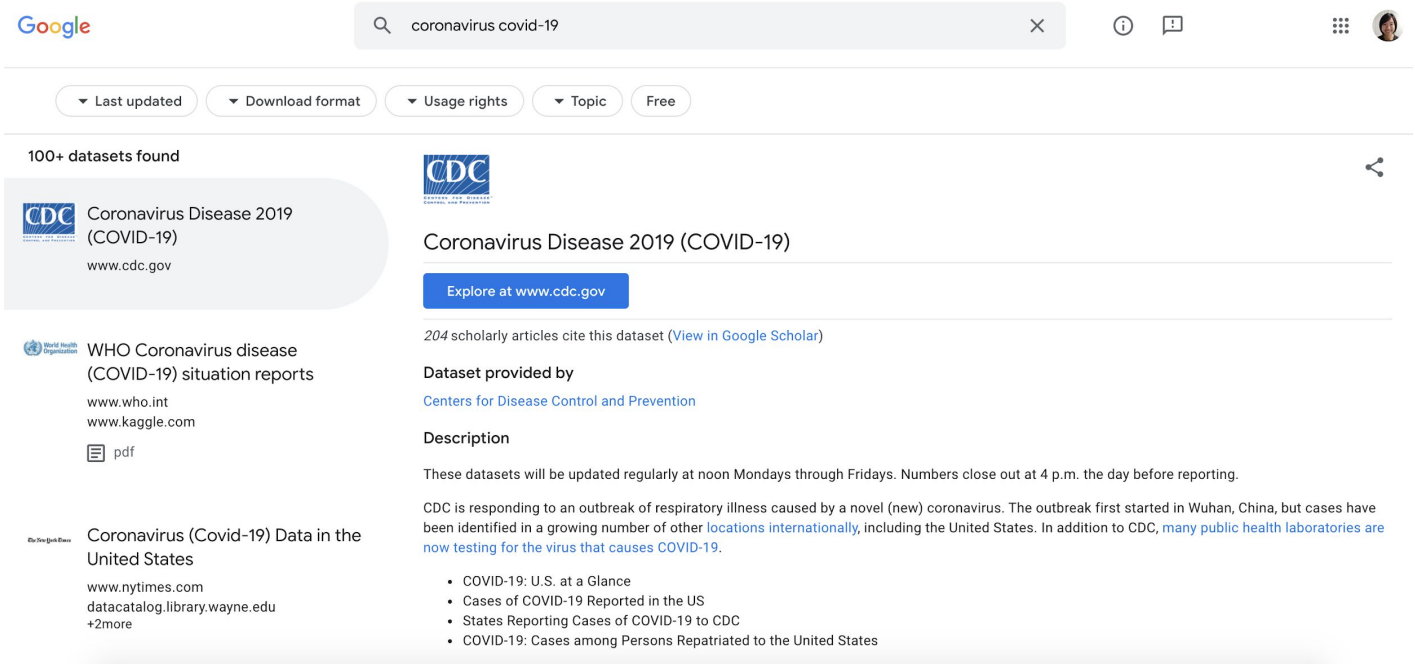
COVID-19 Research Explorer



Google AI Blog ai.googleblog.com/2020/05/an-nlu-powered-tool-to-explore-covid-19.html

Dataset Search

Dataset Search now includes selected coronavirus-related datasets including WHO, ECDE, and the COVID-19 Open Research Dataset.



COVID-19 Data

google.com/covid19/mobility

- Community mobility data reports **movement trends over time** by geography, across different places such as groceries, pharmacies, parks, transit stations, workplaces, and residential.

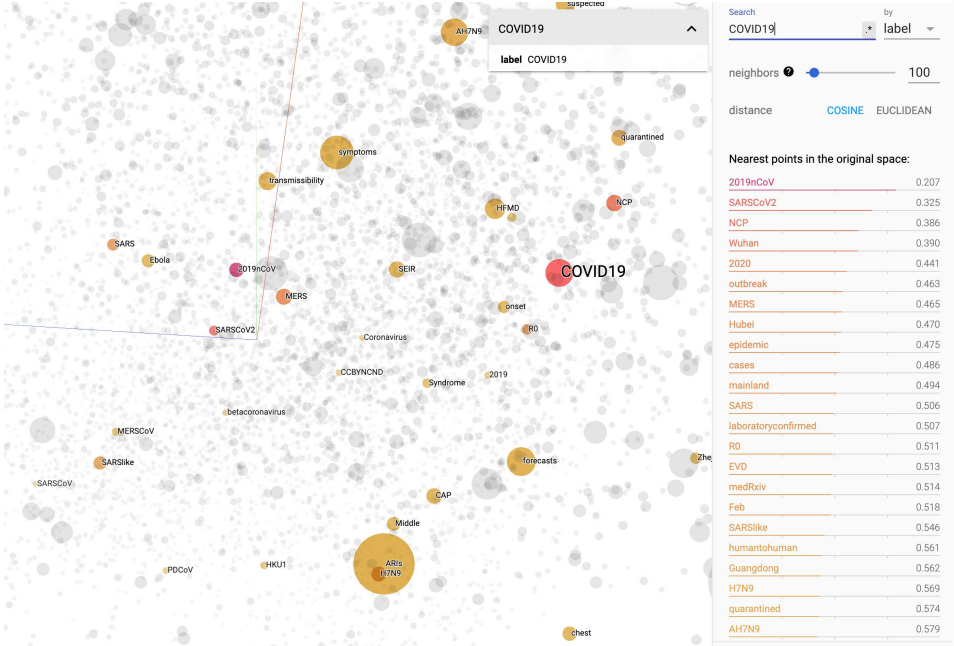
github.com/google-research/open-covid-19-data

- **Aggregates public COVID-19 data** sources into a single dataset.
- Includes cases, deaths, tests, hospitalizations, intensive care unit (ICU) cases, ventilator cases, and government interventions.
- Designed for researchers to build models quickly, and for engineers to add new data sources quickly.

Text Embedding for COVID-19

tfhub.dev/tensorflow/cord-19/swivel-128d/2

- Text embeddings trained with Swivel on COVID-19 Open Research Dataset (CORD-19)
- Released as a tool for researchers to use in COVID-19 research related natural language processing tasks



Engineer the Tools for Discovery

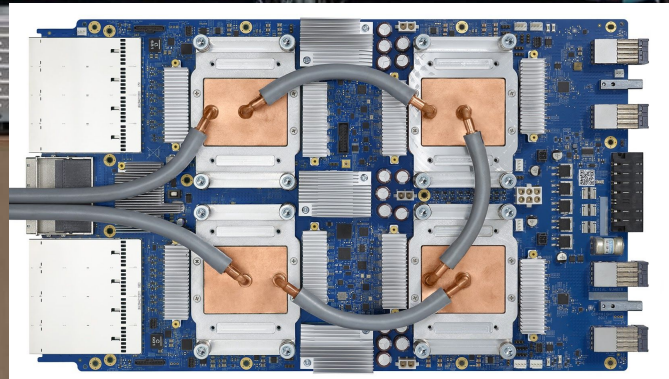
Programming Framework

TensorFlow

Open-source machine learning library

TPUs — Hardware for Deep Learning

TPU v3 Pod — >100 petaflops (10^{15} flops)



ML for ML Compilers

Contributors

Core

Mangpo Phothilimthana
Ulysse Beaugnon
Nikhil Sarda
Yanqi Zhou
Emma Wang
Sam Kaufman
Martin Maas
Mike Burrows
Sudip Roy
Albert Cohen

Others

Amit Sabne
Arun Chauhan
Berkin Ilbeyi
Dominik Grewe
Dong Li
Jinliang Wei
Karthik Srinivasa Murthy
Rezsa Farahani
Shen Wang
Amirali Abdolrashidi
Daniel Wong
Peter Ma
Qiumin Xu
Ming Zhong
Hanxiao Liu
Anna Goldie
Azalia Mirhoseini
James Laudon

Goal

Build an ML-based compiler for ML workloads that is as

fast (compilation time) and

better (efficient generated code) than

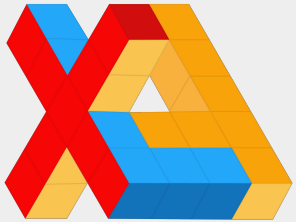
a mature heuristics-based production compiler.

ML Compilation Stack at Google

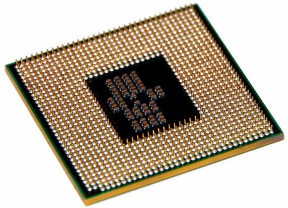
PyTorch

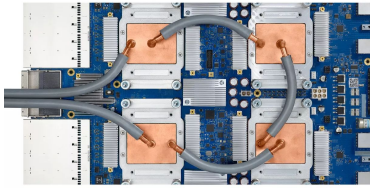
julia

...

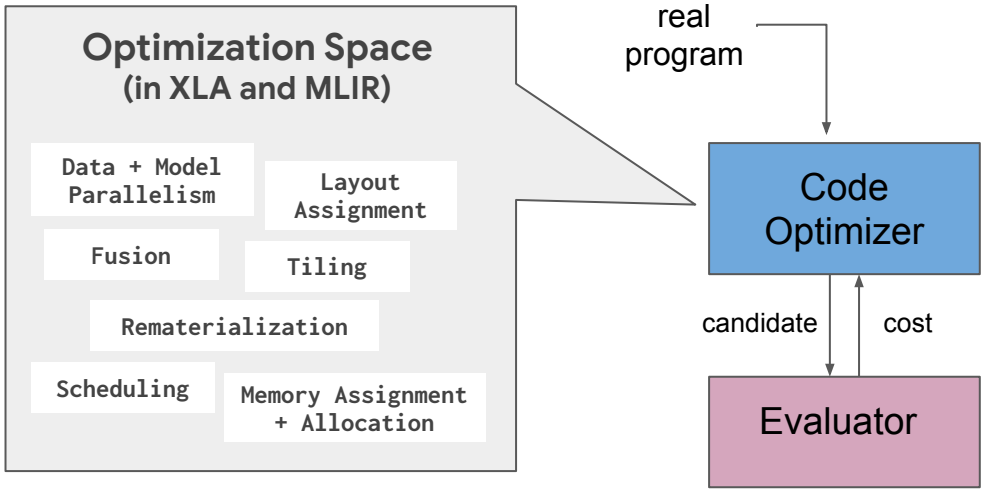


MLIR





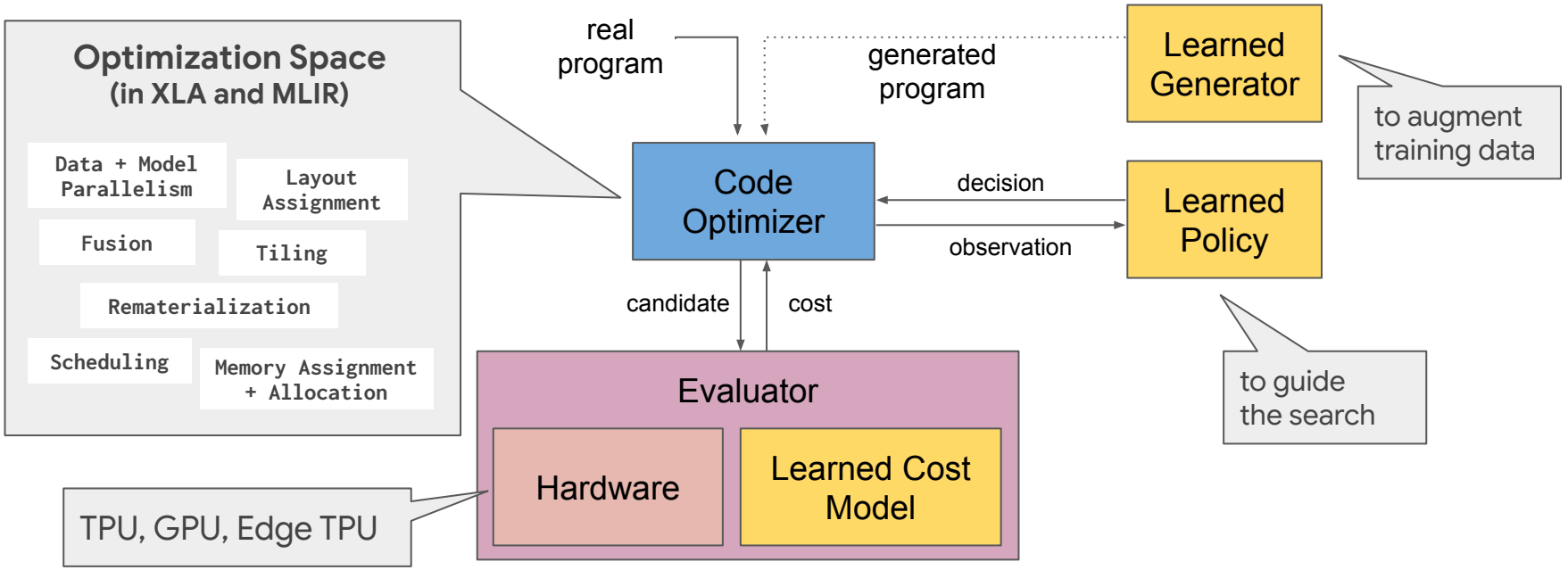
Overview



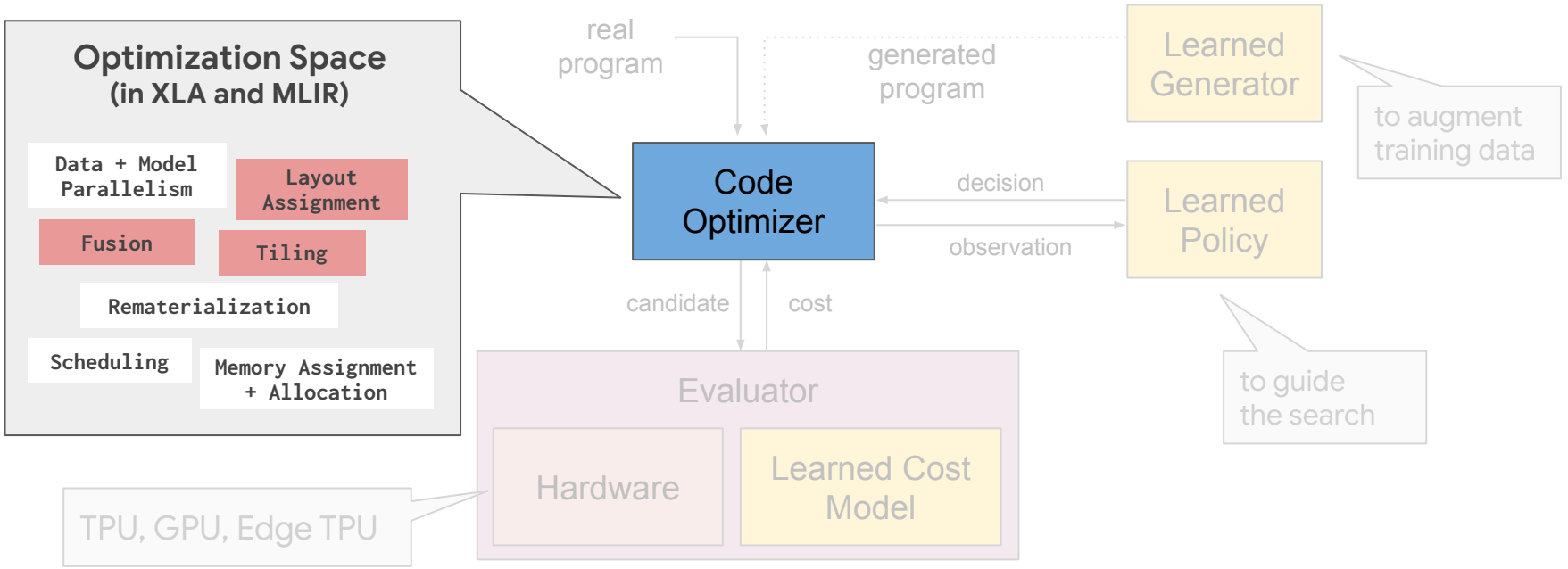
Autotuning

- **Searches** a space of configurations of a program
- **Selects the best configuration** according to a performance metric
- Used in ATLAS, FFTW, SPIRAL, PetaBricks, Halide, and OpenTuner

Overview

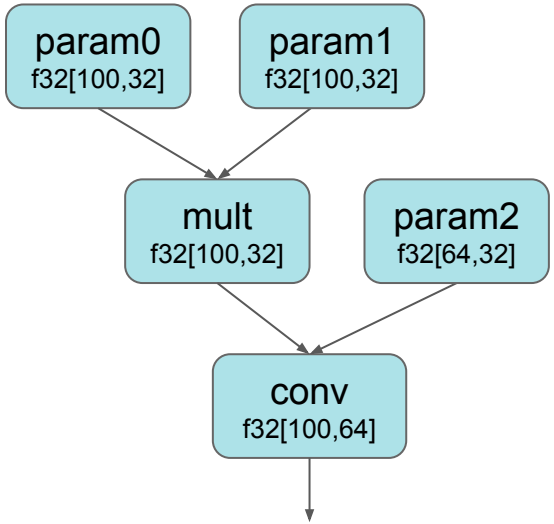

Optimization Space
(in XLA and MLIR)
Data + Model Parallelism
Layout Assignment
Fusion
Tiling
Rematerialization
Scheduling
Memory Assignment + Allocation
real program
generated program
Learned Generator
to augment training data
Code Optimizer
decision
observation
Learned Policy
candidate
cost
to guide the search
Evaluator
Hardware
Learned Cost Model
TPU, GPU, Edge TPU

Overview


Optimization Space
(in XLA and MLIR)
Data + Model Parallelism
Layout Assignment
Fusion
Tiling
Rematerialization
Scheduling
Memory Assignment + Allocation
real program
generated program
Learned Generator
to augment training data
Code Optimizer
decision
observation
Learned Policy
candidate
cost
to guide the search
Evaluator
Hardware
Learned Cost Model
TPU, GPU, Edge TPU

Layout Assignment

Example:

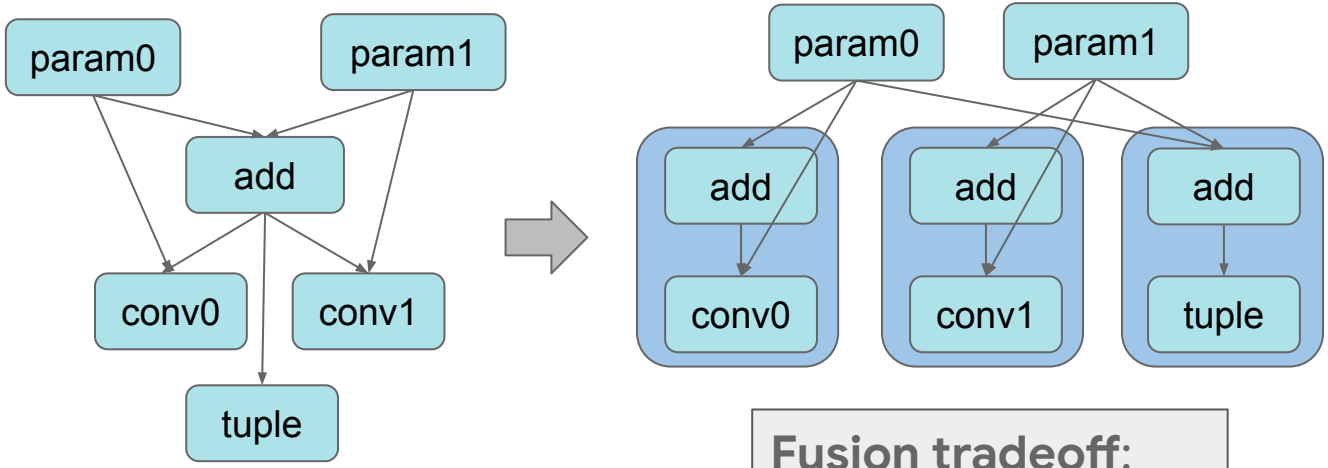


Layout configuration space:

Permutation of each tensor's dimensions

Operator Fusion

Example:

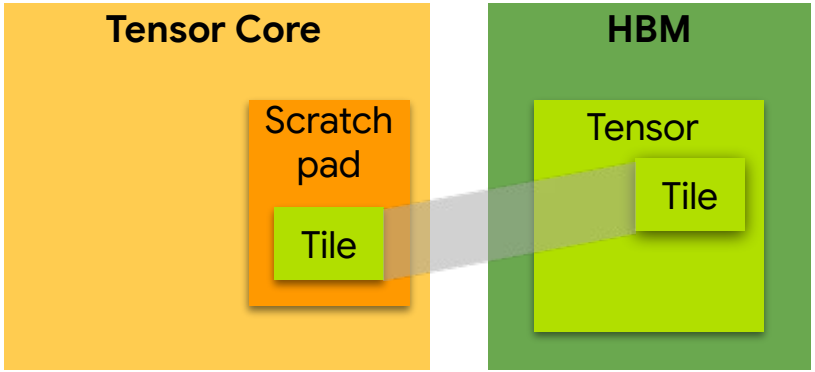


Fusion tradeoff:
less memory traffic
vs. recomputation

Fusion configuration:

- fuse or do-not-fuse per **node**
- If a node is marked fuse, it is fused into all its consumers.

Tile Size Selection



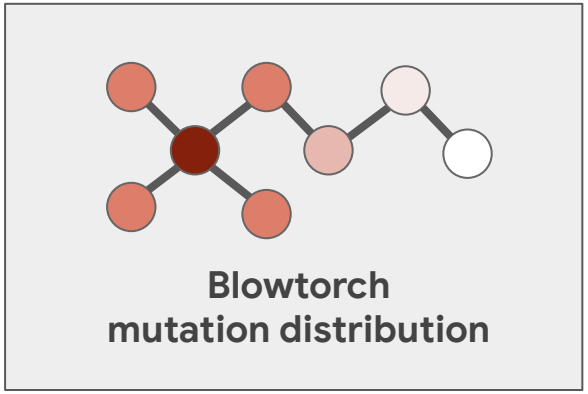
TPUs process one (fused) tensor op at a time.

- For each operation, tile output tensor, copy input tiles into scratchpad.
- Store intermediates in scratchpad.

Autotuning Strategies

Mature techniques

- Exhaustive search (for tile size)
- **Blowtorch** simulated annealing (for layout and fusion)



Experimental

- Reinforcement learning
- Co-evolution of different search strategies

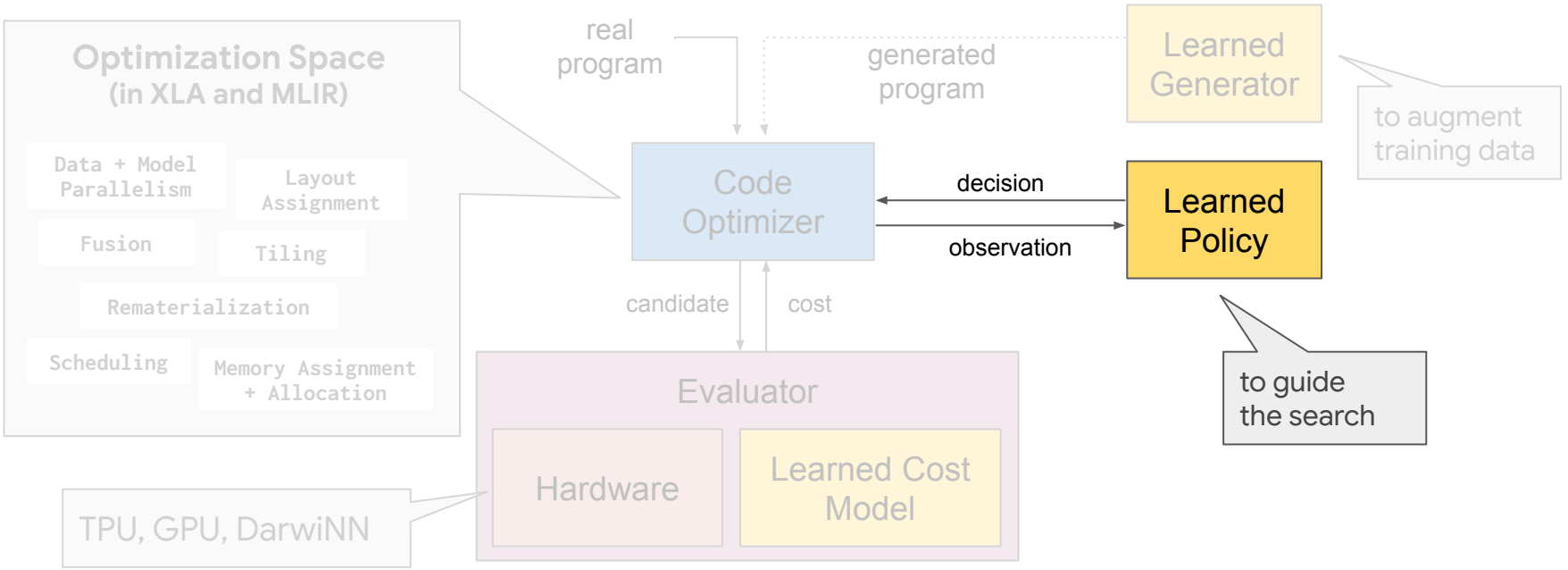
Highlighted Results

Model	Runtime Speedup	From
Translate Inference Transformer	26%	Tile-size
Translate Inference (Production)	6%	Tile-size
EfficientNet Training	12%	Tile-size
Search Ranking (Production)	10%	Fusion
GraphNet	15%	Fusion
OpenAI RNN	45%	Layout
Magenta	37%	Layout

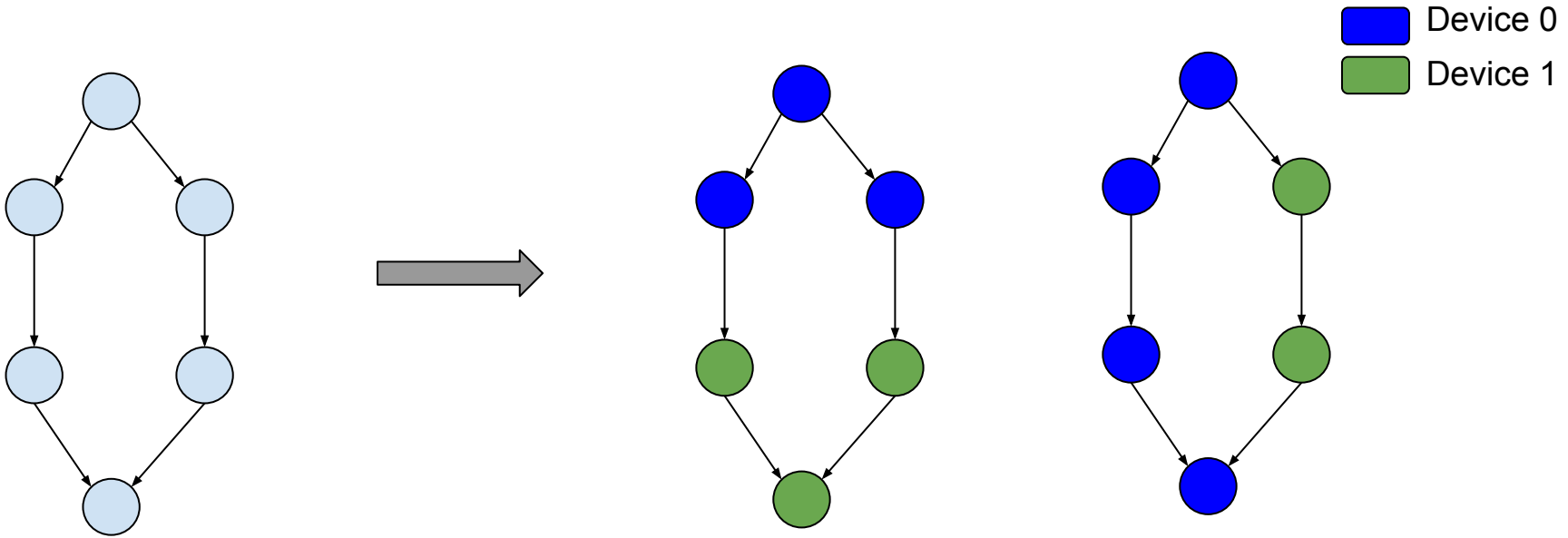
Manual cost model fix in XLA (realizing partial speedup)

Led to discovery of manual layout fix: 18% speedup.

Learned Policy



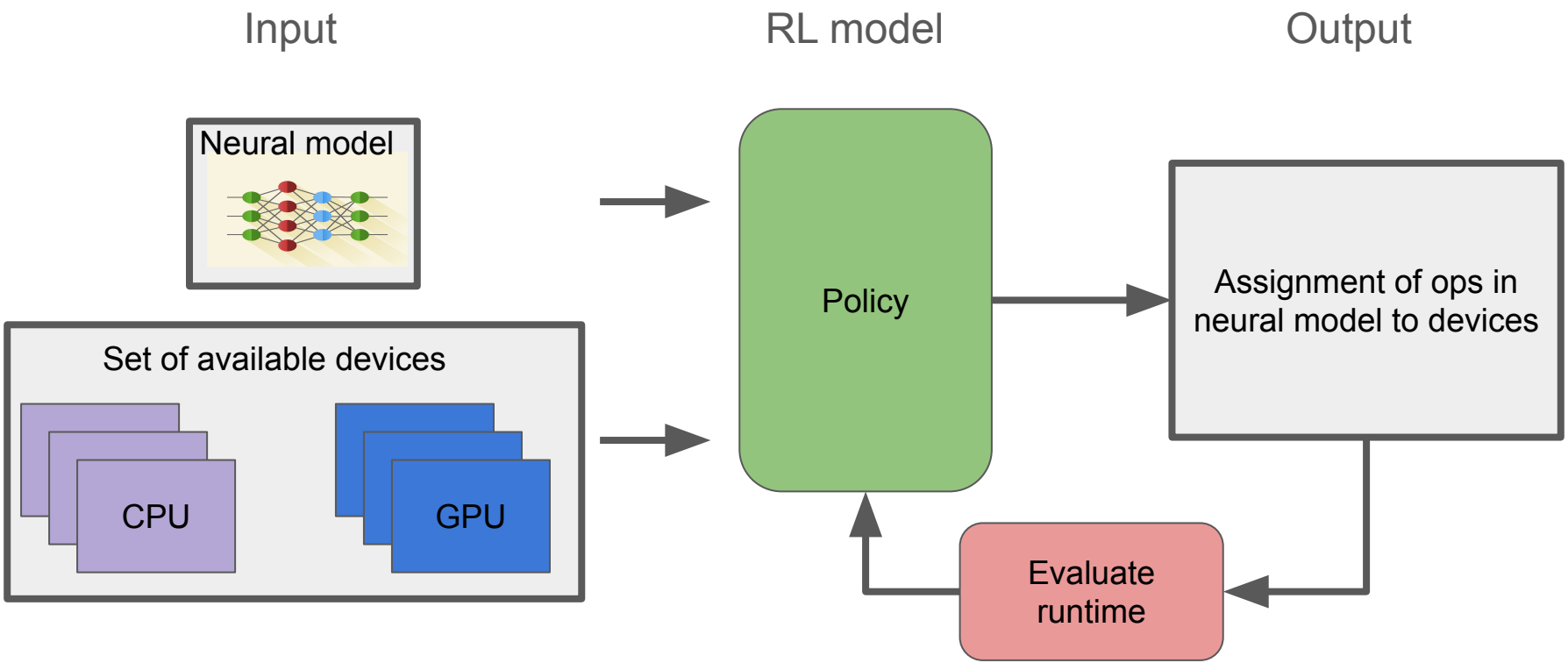
Model Parallelism via Op Placement



Good placement

- Peak memory < device memory
- Minimize step time

Posing Device Placement as an RL Problem



Mirhoseini et al, “Device placement optimization with reinforcement learning.” 2017

Prior Approach

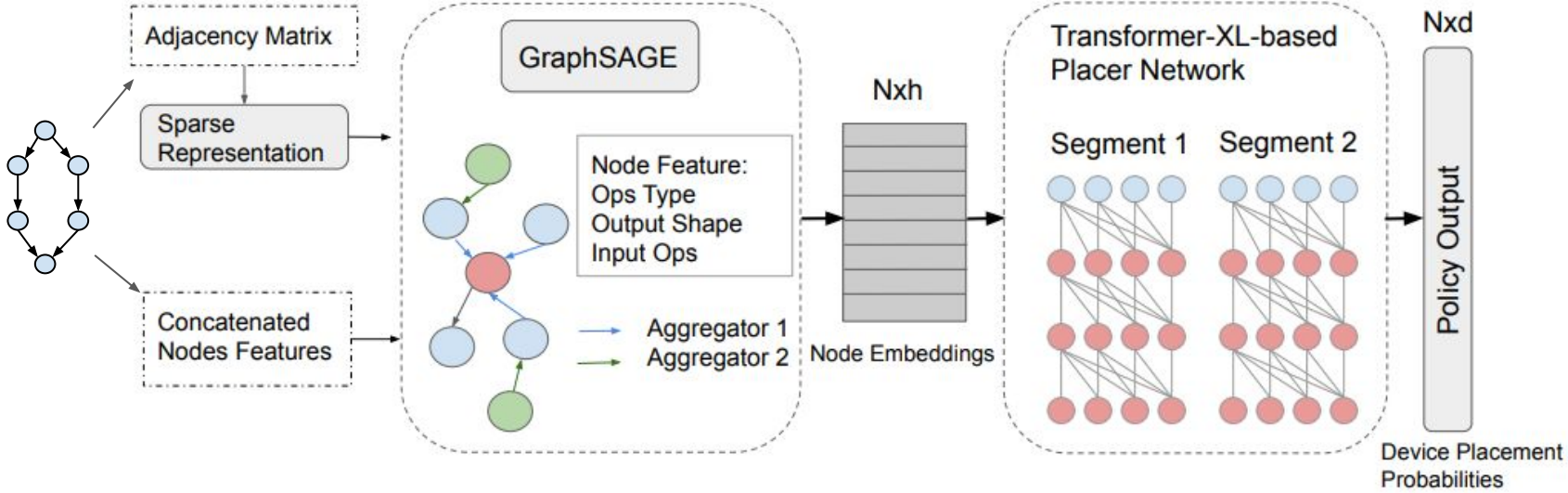
- LSTM does not scale to large graphs > 50k nodes (8-layer GNMT).
- Progressive placement is slow.
- Impractical for production.
  - Training a policy using RL takes a long time.
  - The solution does not generalize to new graphs.

Mirhoseini et al, “Device placement optimization with reinforcement learning.” 2017

GDP: Generalized Device Placement

Inductive graph embedding for generalization

Single-shot placement generation for speed



Representation learning

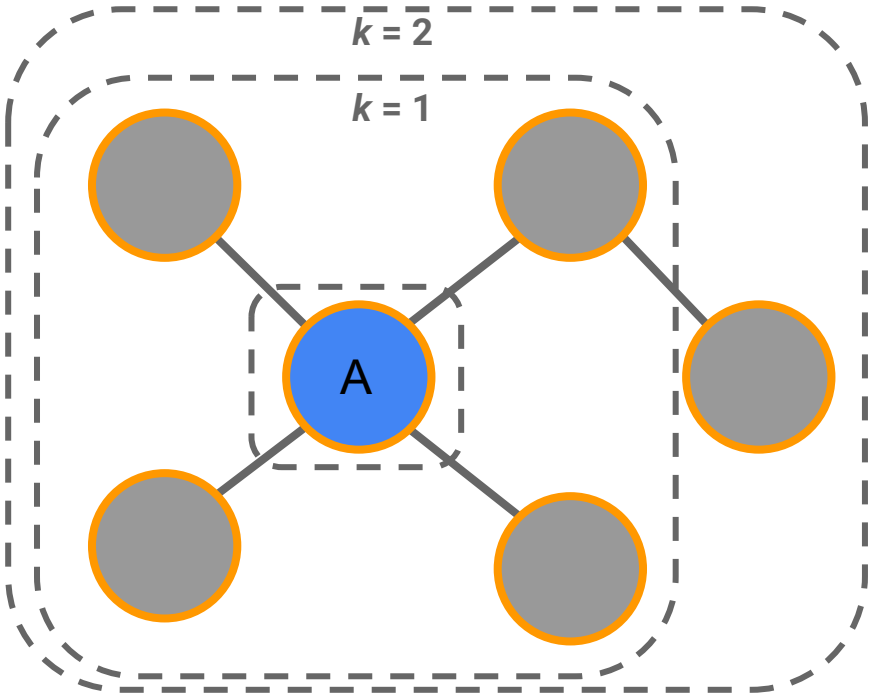
Policy Network

Zhou et al, “GDP: Generalized Device Placement for Dataflow Graphs.” 2019

GraphSAGE

A neural network model to learn node representations in graph-structured data.

- Conditions on neighbors (local structure).
- Transfer between graphs within a domain.



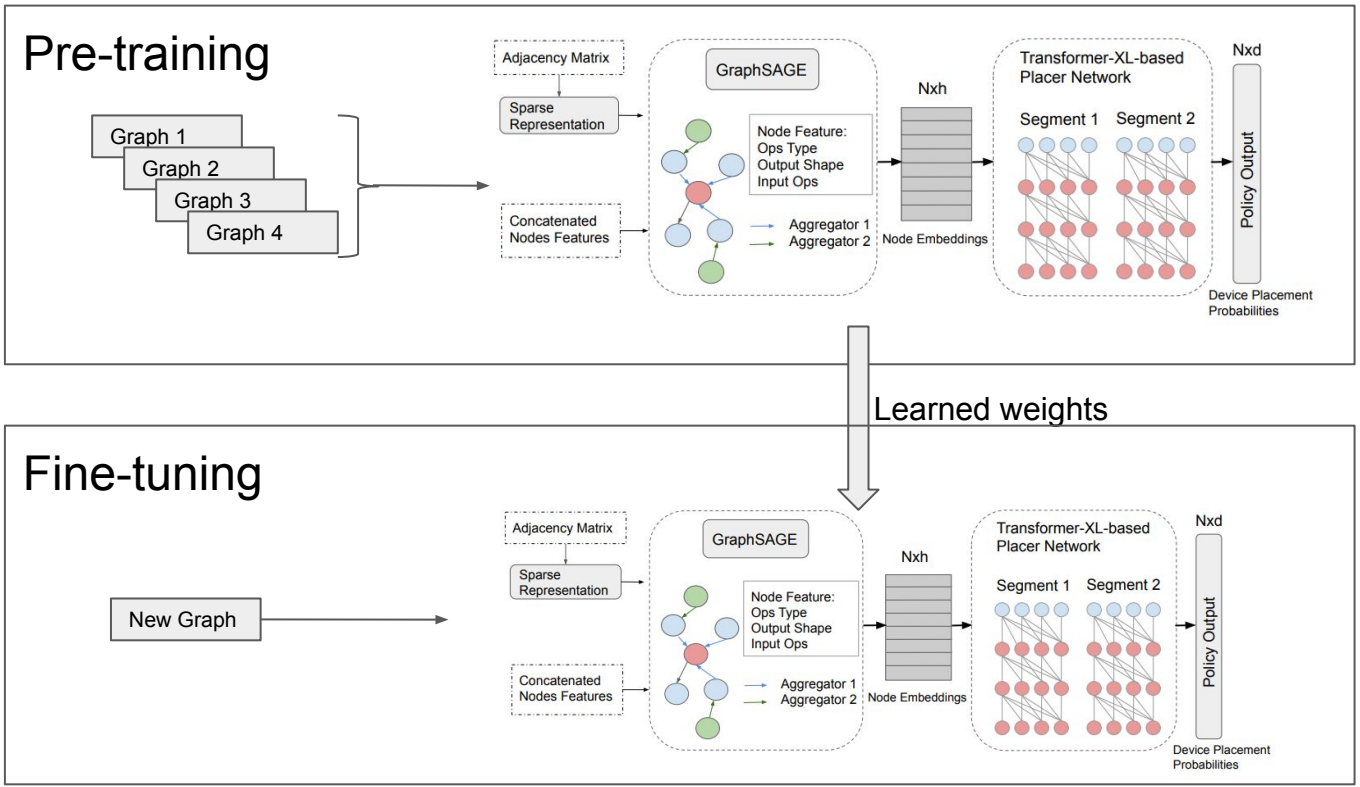
Hamilton et al, “Inductive Representation Learning on Large Graphs.” 2017

Placement Results

Model (#devices)	**GDP-one** (s)	HP (s)	METIS (s)	HDP (s)	Run time speed up over HP / HDP	Search speed up
2-layer RNNLM (2)	0.234	0.257	0.355	0.243	9.8% / 4%	2.95x
4-layer RNNLM (4)	0.409	0.48	OOM	0.490	17.4% / 19.8%	1.76x
2-layer GNMT (2)	0.301	0.384	OOM	0.376	27.6% / 24.9%	30x
4-layer GNMT (4)	0.409	0.469	OOM	0.520	14.7% / 27.1%	58.8x
8-layer GNMT (8)	0.649	0.610	OOM	0.693	-6% / 6.8%	7.35x
2-layer Transformer-XL (2)	0.386	0.473	OOM	0.435	22.5% / 12.7%	40x
4-layer Transformer-XL (4)	0.580	0.641	OOM	0.621	11.4% / 7.1%	26.7x
8-layer Transformer-XL (8)	0.748	0.813	OOM	0.789	8.9% / 5.5%	16.7x
Inception (2)	0.405	0.418	0.423	0.417	3.2% / 3%	13.5x
AmoebaNet (4)	0.394	0.44	0.426	0.418	26.1% / 6.1%	58.8x
2-stack 18-layer WaveNet (2)	0.317	0.376	OOM	0.354	18.6% / 11.7%	6.67x
4-stack 36-layer WaveNet (4)	0.659	0.988	OOM	0.721	50% / 9.4%	20x
GEOMEAN	-	-	-	-	**16% / 9.2%**	**15x**

Zhou et al, “GDP: Generalized Device Placement for Dataflow Graphs.” 2019

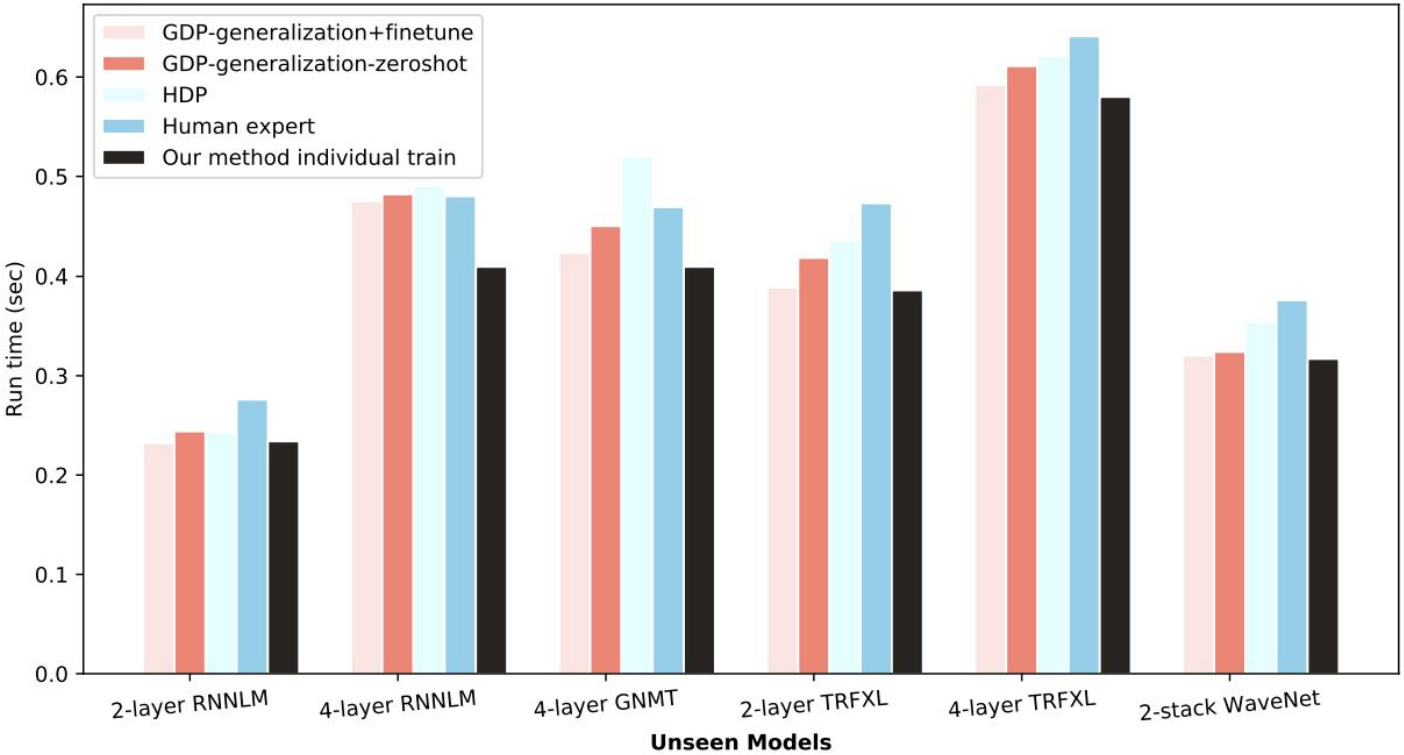
Fine-tuning vs. Zero-shot for Unseen Graphs



Fine-tune for
< 50 steps
(takes < 1 minute)

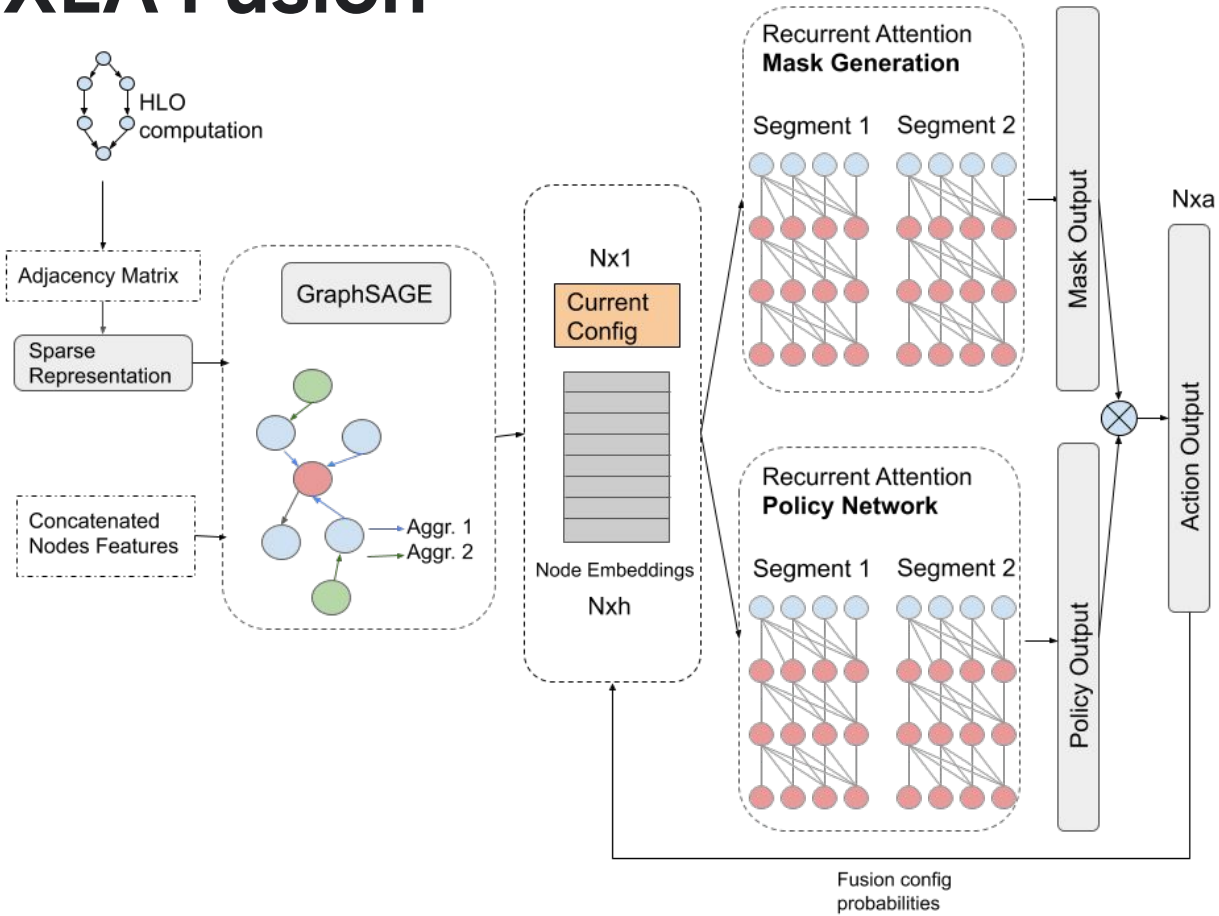
Zhou et al, “GDP: Generalized Device Placement for Dataflow Graphs.” 2019

Generalization



Zhou et al, “GDP: Generalized Device Placement for Dataflow Graphs.” 2019

RL for XLA Fusion



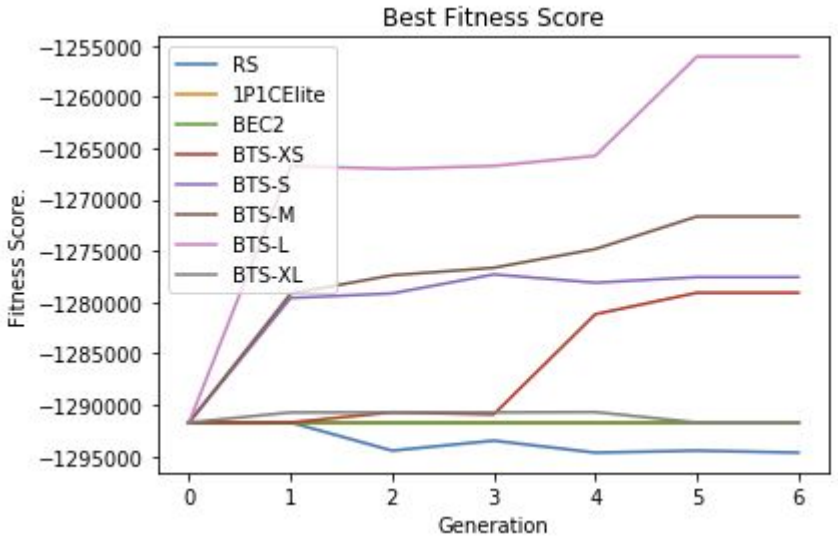
Preliminary Results: RL vs SA

Compare RL vs. Simulated Annealing (SA) starting from default config.

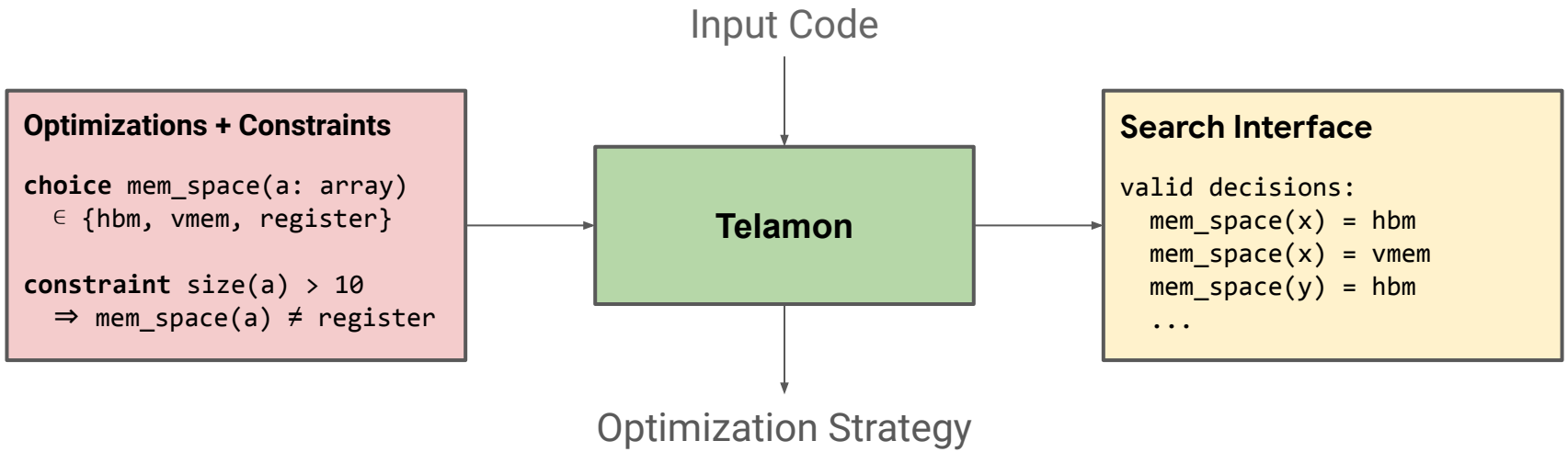
Workload	Speedup (RL)	Samples (RL)	Speedup (SA)	Samples (SA)
Ranking	4%	100	5%	900
GraphNet	**20%**	500	15%	1,700
Unet	3%	-	5%	1,500
Xception	2%	200	1%	1,300
ResNet	1%	-	2%	1,000

Combine Different Search Techniques

- Evolving Learning Communities
- Platform for co-learning evolution of different search strategies
- Migrate best solutions between communities
- Initially designed for NAS



Telamon: Search with Constraints

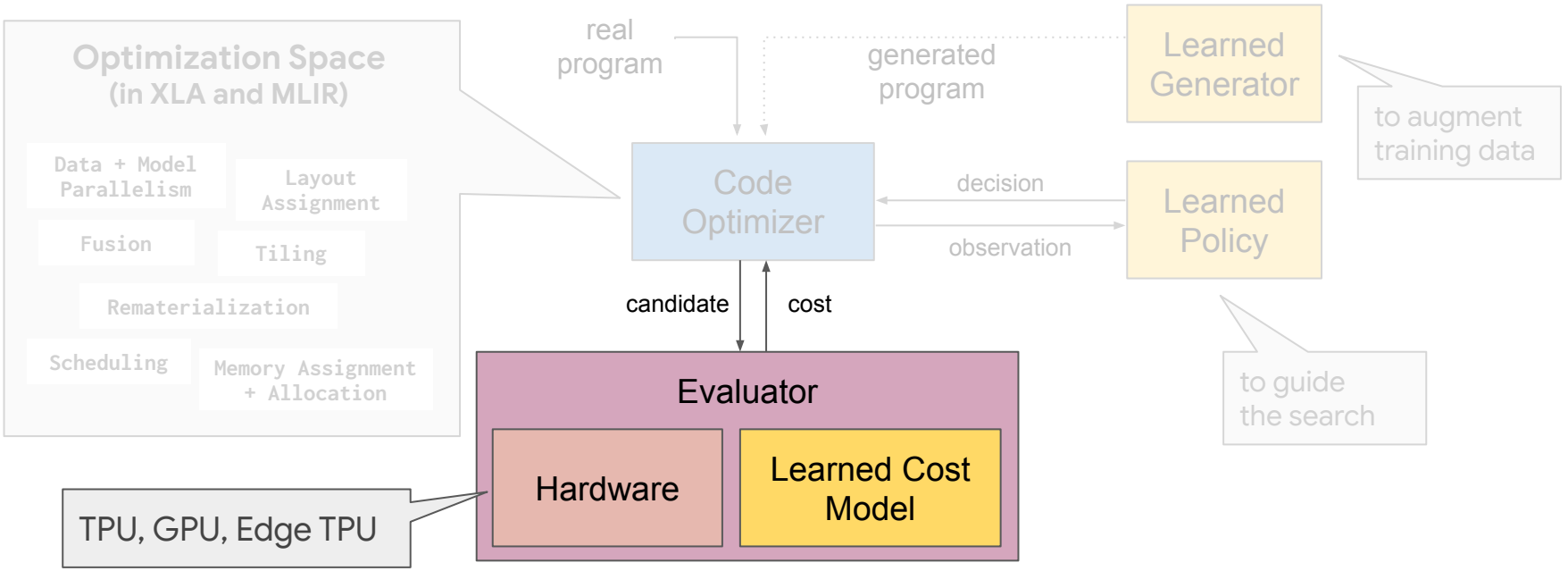


- Enforce correctness constraints
- Declarative search space description

Next Steps: Application to Memory Allocation + Tensor Distribution on Edge TPU

Extension of Beaugnon et al, “Optimization space pruning without regrets”

Learned Cost Model



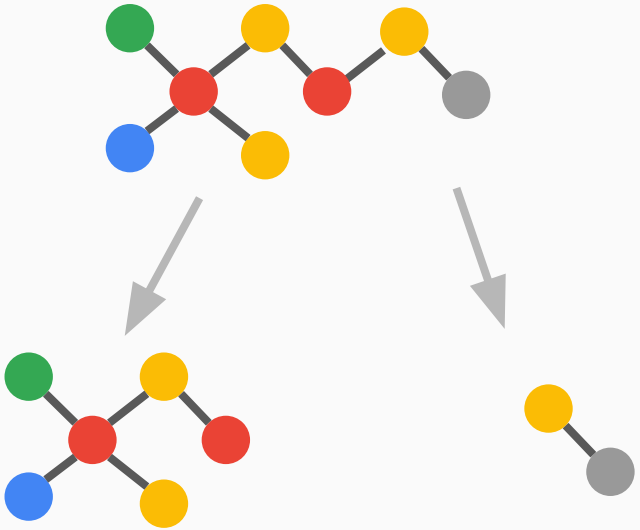
Extension of [Kaufman et al, "Learned TPU Cost Model for XLA Tensor Programs"]

Analytical Cost Model: Challenges

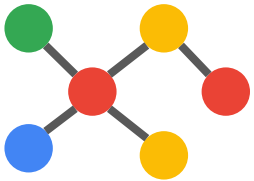
- **Hardware**: complex processor architecture
- **Compiler**: performance-affecting decisions that are made during compilation
- **IR**: lack of in-depth view of the processor architecture and low-level generated code
- **Program characteristics**: target programs contain complex, multi-level nested loops
- **Demand**: proliferation of accelerators demands rapid development of performance models

Overview of Cost Model

1. Decompose Into Kernels



2. Regression Per Kernel

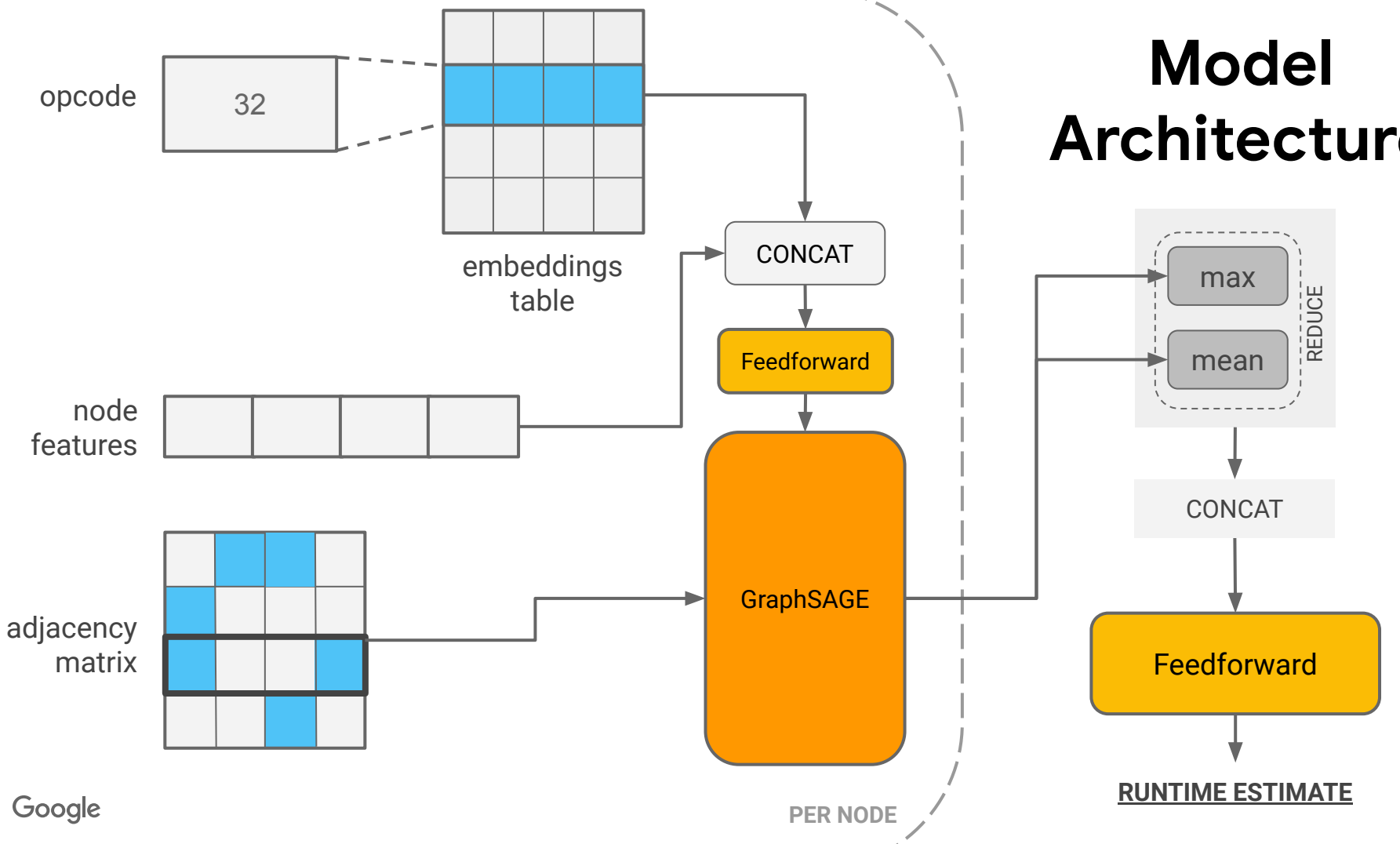
$f($  $)$

KERNEL

$\cong 5.2s$

RUNTIME

Model Architecture



Loss Functions

Absolute loss (when we need absolute runtime prediction)

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

Rank loss (when we need relative runtime prediction)

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

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

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

Tasks & Datasets

Fusion

- Run fusion autotuner with a random search to generate 50,000 fusion configs or until timeout (4 hours on 50 machines) on each input graph.
- Optimized graphs are decomposed, yielding 207 million fused kernels.

Tile Size

- Compile each XLA program using the compiler's default fusion heuristics, obtaining an optimized graph that we decompose into kernels.
- Run tile-size autotuner on each kernel with exhaustive search until timeout (30 mins on 50 machines per kernel).

Dataset Stats

Split	Manual Split Programs		Kernels	
	Fusion	Tile-Size	Fusion	Tile-Size
Train	78	93	157.5M	21.8M
Val.	8	8	30.1M	1.6M
Test	8	8	20.3M	1.4M

- Split by programs not kernels to test model's ability to generalize.
- Validation and test programs were chosen randomly.

Fusion Dataset: Prediction Accuracy

Analytical: manual cost model used for tile-size selection in XLA

Table reports accuracy on test kernels whose runtimes are >= 5 us.

	MAPE			Kendall's τ		
	Our Model	LSTM	Analytical	Our Model	LSTM	Analytical
ConvDRAW	38.2	62.3	21.6	0.63	0.52	0.77
WaveRNN	17.8	29.7	322.9	0.81	0.73	0.70
NMT Model	79.2	68.9	26.3	0.83	0.82	0.91
SSD	24.0	49.3	55.9	0.76	0.70	0.76
RNN	8.7	23.5	20.5	0.92	0.85	0.86
ResNet 1	6.8	14.4	11.5	0.92	0.84	0.88
ResNet 2	5.7	14.3	13.3	0.92	0.83	0.86
Translate	17.9	22.4	27.2	0.85	0.81	0.74
Median	17.9	26.6	23.9	0.84	0.81	0.81

Tile-Size Dataset: Prediction Accuracy

Analytical: manual cost model built specifically for this task

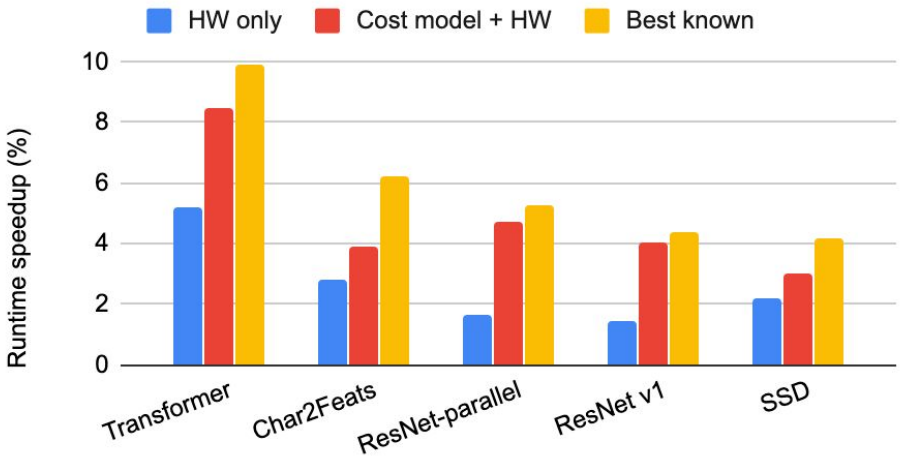
	Analytical	Our Model (Rank Loss)	Our Model (MSE Loss)
ConvDRAW	0.79	0.64	0.66
WaveRNN	0.65	0.46	0.56
NMT Model	0.81	0.74	0.66
SSD	0.77	0.64	0.60
RNN	0.55	0.42	0.37
ResNet v1	0.73	0.72	0.67
ResNet v2	0.73	0.74	0.69
Translate	0.92	0.76	0.62
Median	0.75	0.68	0.64

Fusion Autotuning with Learned Cost Model

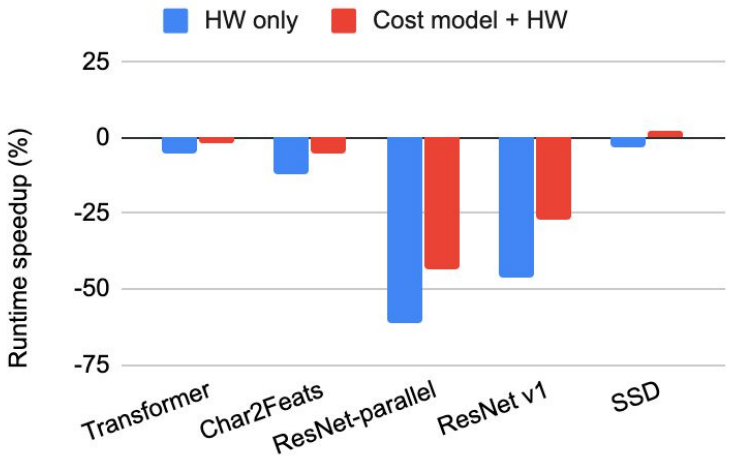
HW only: run autotuner on TPU for 5 min.

Cost model + HW: run autotuner on CPU for 1 hr and then TPU for 5 min.

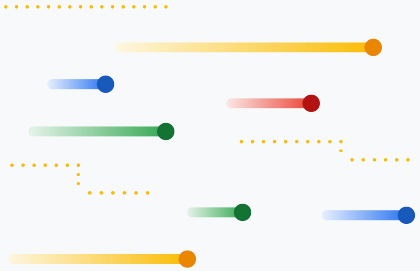
Best known: run autotuner on TPU for 4 hr.



Autotuning from default config



Autotuning from random config



Google Research

Deployment Strategies

Deployment Strategies

Offline autotuning	**Users run autotuner** offline on their workloads. Save best configs and use them in future compilation.
Online autotuning	**Compiler runs autotuner** automatically during compilation.
Profile-guided autotuning	**System runs autotuner** automatically **on top workloads** and saves best configs in shared database. Compiler uses configs from database if exist.

Deployment Strategies

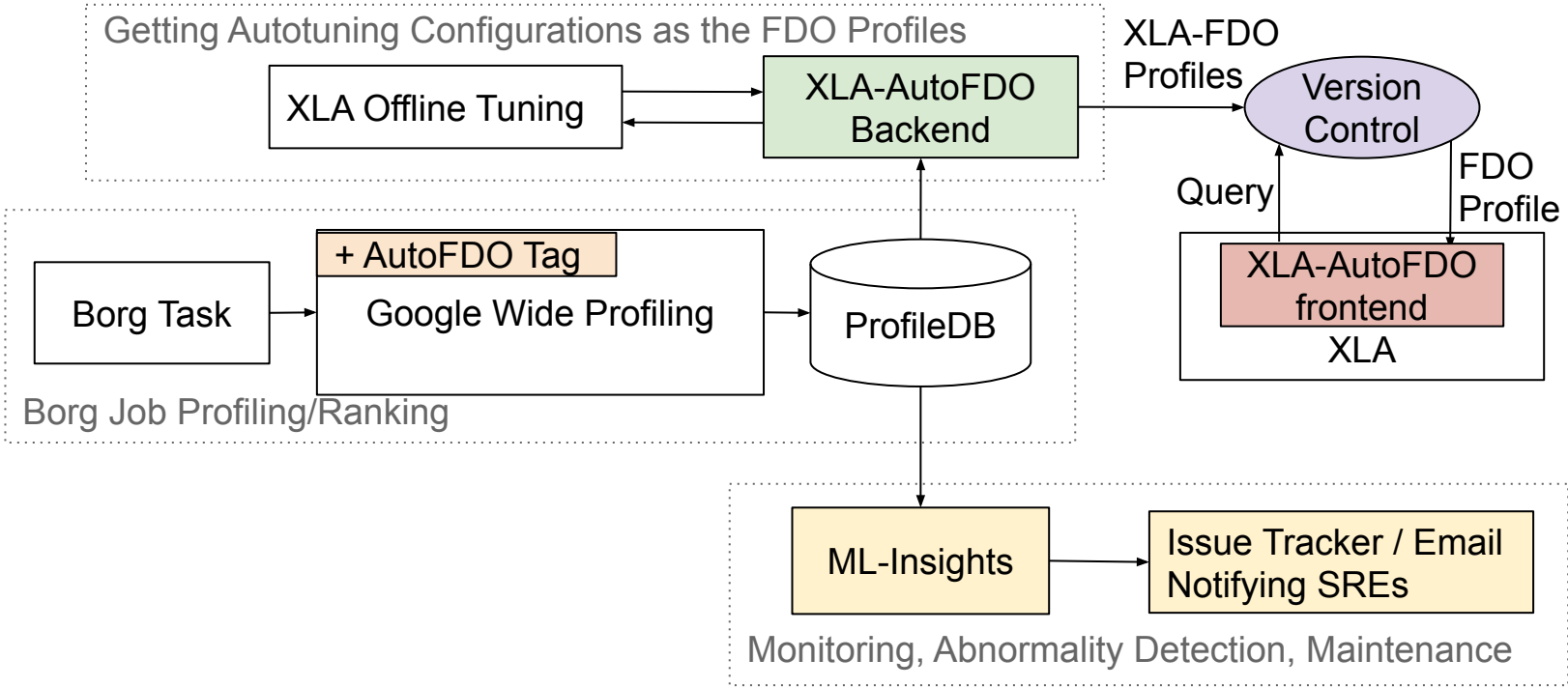
	Pros	Cons
Offline autotuning	Fast compilation. More time for autotuning.	Require more user's effort.
Online autotuning	Easy to use.	Small time budget for autotuning. Reproducibility issue.
Profile-guided autotuning	Easy to use. Fast compilation. More time for autotuning.	Some workloads won't get benefit.

Profile-Guided Autotuning: XLA-AutoFDO

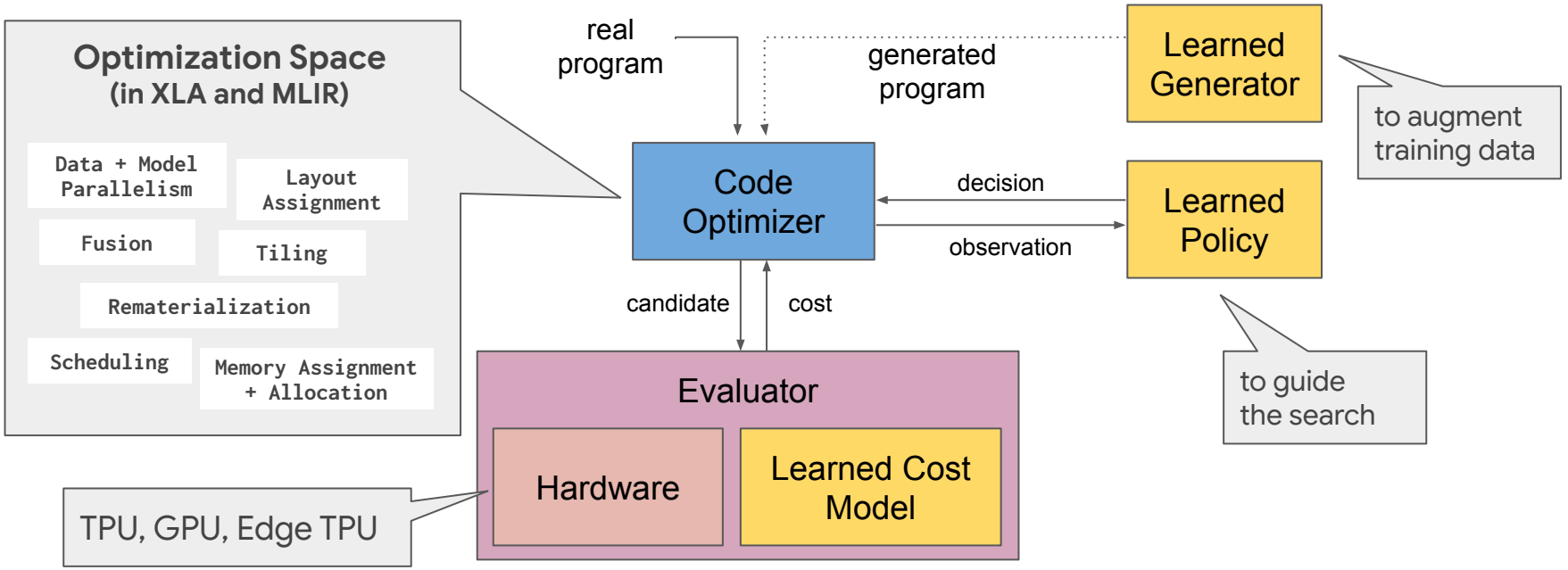
Guarantees:

- **No change to users' existing workflow**.
- **No overhead to fleet jobs in execution**: being able to generate the runtime profiles without affecting current jobs execution.
- **No overhead to compilation**: compiling new binaries with the runtime profiles pre-generated offline.
- **Reproducibility**: being able to reproduce performance results.
- **Rollback-ability**: being able to roll back to a stable version when things go wrong in production.

System Design



Summary


Optimization Space
(in XLA and MLIR)
Data + Model Parallelism
Layout Assignment
Fusion
Tiling
Rematerialization
Scheduling
Memory Assignment + Allocation
real program
generated program
Learned Generator
to augment training data
Code Optimizer
decision
observation
Learned Policy
candidate
cost
to guide the search
Evaluator
Hardware
Learned Cost Model
TPU, GPU, Edge TPU

Our Publication

Learned policy:
Zhou et al, “GDP: Generalized Device Placement for Dataflow Graphs.” 2019

Learned cost model:
Kaufman et al. “Learned TPU Cost Model for XLA Tensor Programs”, ML for Systems workshop, 2019

Related Work: ML for Compilers

Learned policy/optimizations:

Haj-Ali and Huang et al. “AutoPhase: Juggling HLS Phase Orderings in Random Forests with Deep Reinforcement Learning,” MLSys 2020.
Haj-Ali et al. “NeuroVectorizer: End-to-End Vectorization with Deep Reinforcement Learning,” CGO 2020.
Mendis et al. “Compiler Auto-Vectorization with Imitation Learning,” NeurIPS 2019.
Cummins et al. “End-to-end Deep Learning of Optimization Heuristics,” PACT 2017.

Learned cost model:

Mendis et al. “Ithemal: Accurate, Portable and Fast Basic Block Throughput Estimation using Deep Neural Networks,” ICML 2019.
Adams et al. “Learning to Optimize Halide with Tree Search and Random Programs,” SIGGRAPH 2019.
Chen et al. “Learning to Optimize Tensor Programs,” NeurIPS 2018.

ML compiler optimizations:

Jia et al. “TASO: Optimizing Deep Learning Computation with Automated Generation of Graph Substitutions,” SOSP 2019.
Jia et al. “Beyond Data and Model Parallelism for Deep Neural Networks,” *MLSys 2019*.
Narayanan et al. “PipeDream: Generalized Pipeline Parallelism for DNN Training,” *SOSP 2019*.

Faculty Programs

Faculty Research Awards
Focused Awards
Visiting Faculty
Consulting Researcher
Focused Research Workshops
Cloud Grants for Academia

Student Programs

PhD Fellowships
Internships
AI Residency
Student Research
CIFRE / Joint PhD Advising
Scholarships
Travel Grants
PhD Summits
Research Talk Series

http://ai.google/research/outreach