# CATWILD: Compiler Autotuning for TPU Workloads in the Wild

Ignacio Cano · Yu Emma Wang · Mike Burrows · Ziqiang Feng · Matheus Camargo · Chao Wang · David H. Liu · Tengyu Sun · Alexander Wertheim · Arissa Wongpanich · Christof Angermueller · Hyojun Kim · Wenqi Cao · Aleksey Orekhov · Amit Sabne · Emma Sevastian · Mehrdad Khani · Karthik Srinivasa Murthy · Berkin Ilbeyi · Subhankar Shah · Ryan Lefever · Arjun Khare · Ankit Sinha · Peter Ma · Matthew Bierbaum · Jeremiah Wilke · Emily Donahue · Sami Abu-El-Haija · Nikhil Sarda · Vineetha Govindaraj · Shobha Vasudevan · Kirill Gugaev · Idan Nachman · Jie Sun · Jose Baiocchi Paredes · Samrat Ghosh · Domagoj Babic · Zongwei Zhou · Naveen Kumar · Phitchaya Mangpo Phothilimthana

MLSys 2026
Bellevue, WA

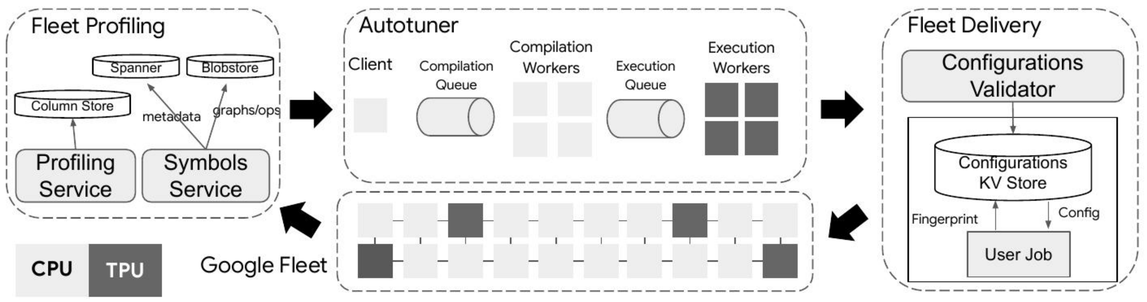
## Abstract

Compilers play a fundamental role in achieving peak performance for machine learning workloads. Compiler autotuning helps compilers find better optimization decisions. CATWILD is a system that automatically optimizes ML jobs in Google's TPU fleet using compiler autotuning techniques.

**DEPLOYMENT CHALLENGES:**

- Heterogeneity (workloads / hardware)
- Fast compilation for rapid experimentation
- Graphs use multi-chip topologies
- Compiler keeps on changing

## System Design




**(1) Fleet Profiling**: Symbol Service collects computation graphs. Analysis pipeline identifies highest-impact graphs/ops.

**(2) Autotuner**: Extends XTAT with a disaggregated architecture: a CPU pool for compilation and a TPU pool for execution, improving TPU duty cycle by 2−5× vs. a co-located design.

**(3) Fleet Delivery**: Tuned configs are stored in a Configurations KV Store keyed by graph fingerprint and embedded in user binaries for reproducibility. A Configurations Validator runs continuously as an eventually-consistent background process, invalidating stale or regressed configs.

### Single-Chip Performance Estimator

**Approach A:** Profile entire graph as-is. ✗ Requires large resources → infeasible

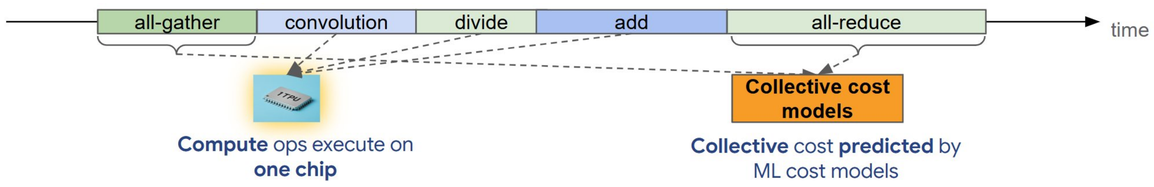
**Approach B:** Profile only compute ops, sum the total ✗ Inaccurate

**Our Approach:** Profile modified graph & estimate collectives

✓ Replace collectives with no-op stubs → runnable on 1 TPU

✓ Estimate communication with ML cost models

✓ Trace synthesis → backfill communication into profile

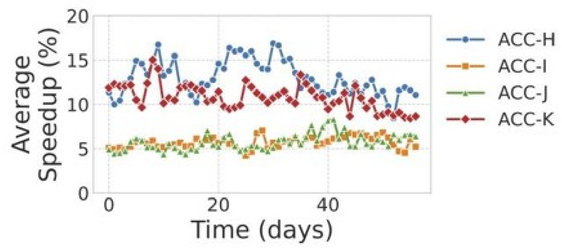



Compute ops execute on one chip; collective costs predicted by ML models.

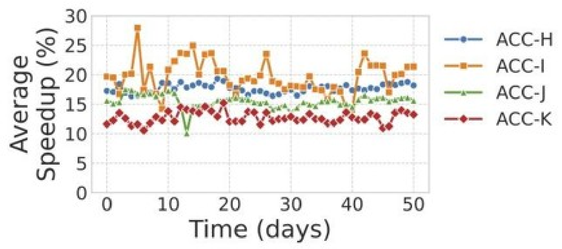
## Evaluation

**Key Results:**

- **Tuning ∼70% of top training workloads in the Google fleet**
- **100k+ autotuned configs used daily**
- **Savings / Cost = 8× − 30×**



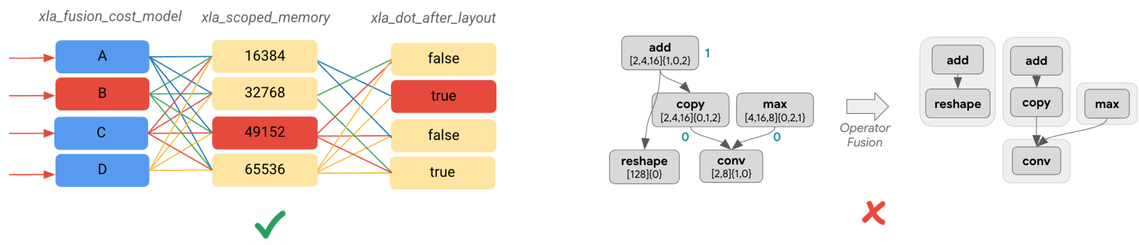

**Graph-level Flag Tuning: ∼5–15% speedups**




**Op Tile Size Tuning: ∼10–25% speedups**

## Insights

- **Coarse vs. Fine-grained Decisions**: high-level decisions more effective and less complex that op-level decisions.




- **Fleetwide View to Improve XLA**: improve out-of-the-box compiler performance (and correctness).


## Conclusion

**CATWILD continuously identifies, evaluates, and deploys autotuned XLA configurations for key TPU workloads in the Google fleet:**

- Continuous profiling and collection of TPU graphs/ops
- Disaggregated autotuner architecture off the user critical path
- Safe delivery of autotuned configurations back to the fleet
- Improve out-of-the-box XLA (performance & correctness)



nachocano@google.com