ML^{2}Tuner: Efficient Code Tuning via Multi-Level Machine Learning Models

JooHyoung Cha[1], Munyoung Lee[2], Jinse Kwon[2], Jubin Lee[3], Jemin Lee[2], Yongin Kwon[2]

[1]University of Science of Technology

[2]Electronics and Telecommunications Research Institute

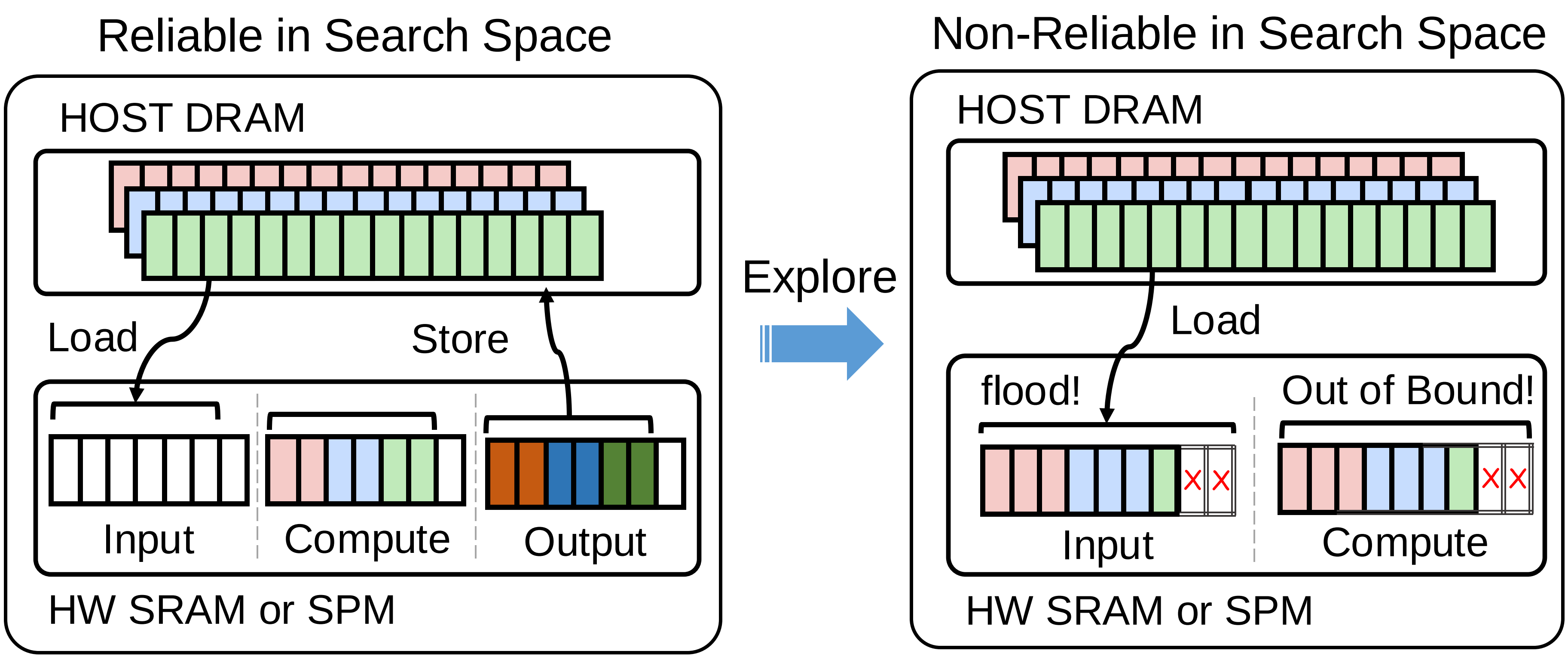
[3]Neubility

Workshop on ML for Systems at NeurIPS 2024

Motivation

Challenges in ML Code Auto-tuning

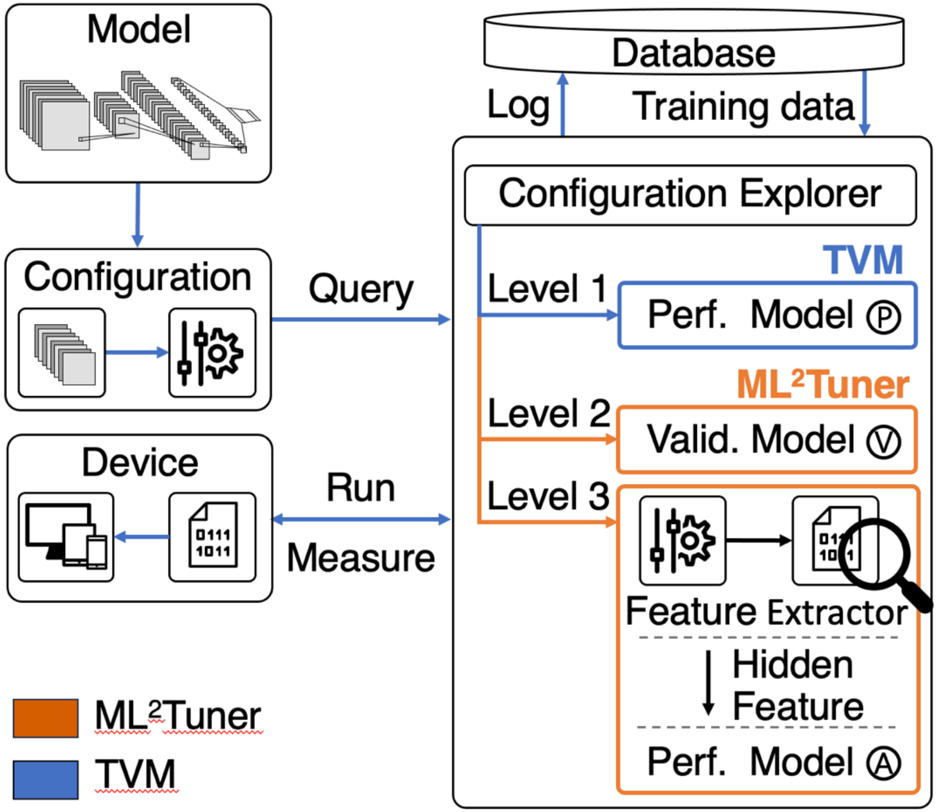
- Need to explore a vast search space
- Runtime errors in HW requires manual reset or system reboots.
- Aggressive optimization increases the chance of the runtime errors.



Example of Runtime Error in HW

Proposal

Methology for Efficient Code Tuning via Multi Level



Level 1 (Model P) **:**
- Finding the Highest Performance

Level 2 (Model V) **:**
- Validate "Level 1"

Level 3 (Model A)**:**
- Finding the Highest Performance w/ Hidden Feature from Compiler

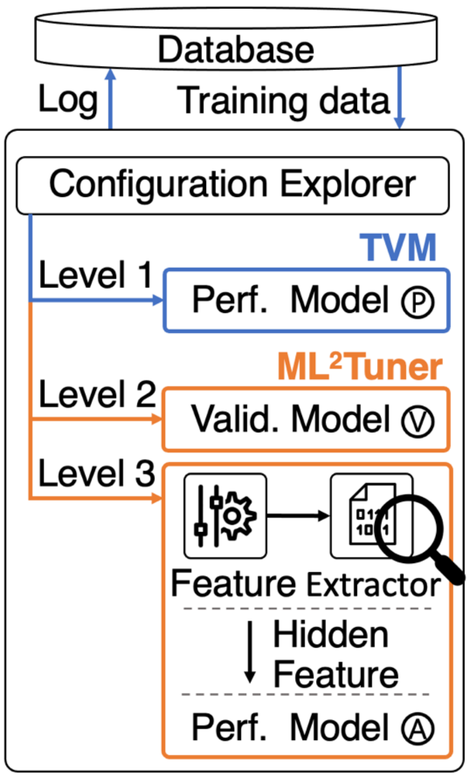
- **Feature Extractor:**
  - Compile Code from Configuration
  - Collecting Hidden Feature

*Configuration :
Tiling, Kernel, Layer Information

*Hidden Feature :
Flow Decision, Tiling Strategy, Weight Stationery, etc

Experimental Setting

- Hardware Platform : VTA on XILINX ZCU102
- Model : ResNet 18 trained with ImageNet 2012
- Compiler : NEST-C (based on GLOW)
- Optimizer : DMLC XGBoost (v2.1.1)



Hyperparameters for DMLC XGBoost Models

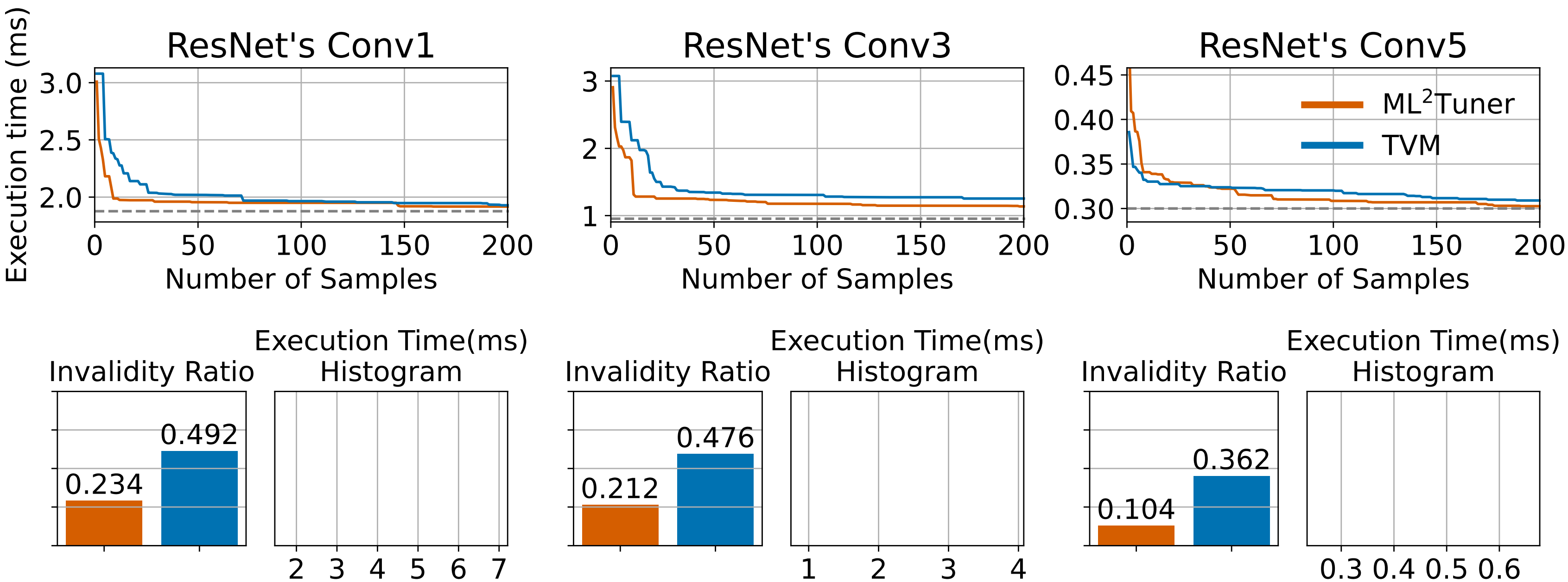
Parameter	Model P	Model V	Model A
objective	reg:squarederror	binary:hinge	reg:squarederror
boost round	300	300	300
max depth	14	5	14
min child weight	3	3	3
gamma	0.0	0.0	0.0
subsample	1.0	0.6	1.0
colsample bytree	1.0	0.6	1.0
learning rate	0.01	0.1	0.01
reg alpha	1×10^{-5}	1×10^{-2}	1×10^{-5}

Level 1 Level 2 Level 3

Experimental Result

Impact of Model V : Validity Model

- Equivalent performance with 12.3% fewer Samples
- Reducing the invalidity ratio to 60.8%
- Provide increased opportunities for exploration

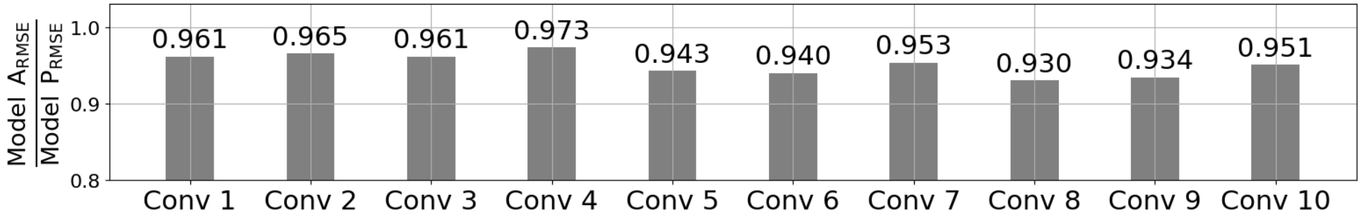


Result for Conv1, Conv3 and Conv5 of ResNet18 on VTA

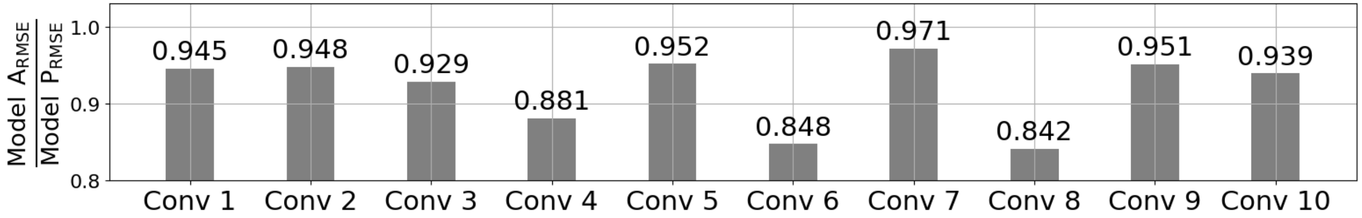
Experimental Result

Impact of Model A : Advanced Performance Prediction

- Each RMSE value was calculated as the average over 10 repeated iterations.
- Model A's average RMSE achives 91.6% of Model P’s from XGBoost.
- Leverage the compiler's information to optimize tuning faster than Model P.



Set Boost Round of XGBoost to 100



Set Boost Round of XGBoost to 300

Evaluate Robustness Model P and Model A

Summary

Our Contribution:

- Error-Aware Search Refinement using **Multi Level**
- Reducing the invalidity ratio and Equivalent performance with fewer Samples
- Hidden feature in Compiler improves tuning outcomes about 9.17%.

What did we skip?

1. Validate Evaluation on Diverse Environment

- Testing Diverse hardware to assess generalizability.
- Testing Algorithm to further refine the tuning process.

2. Auto-identifying features from binaries

- Study on improved accuracy and probabilistic error detection systems.

Thank you for your attention.