Project

Carnegie Mellon University

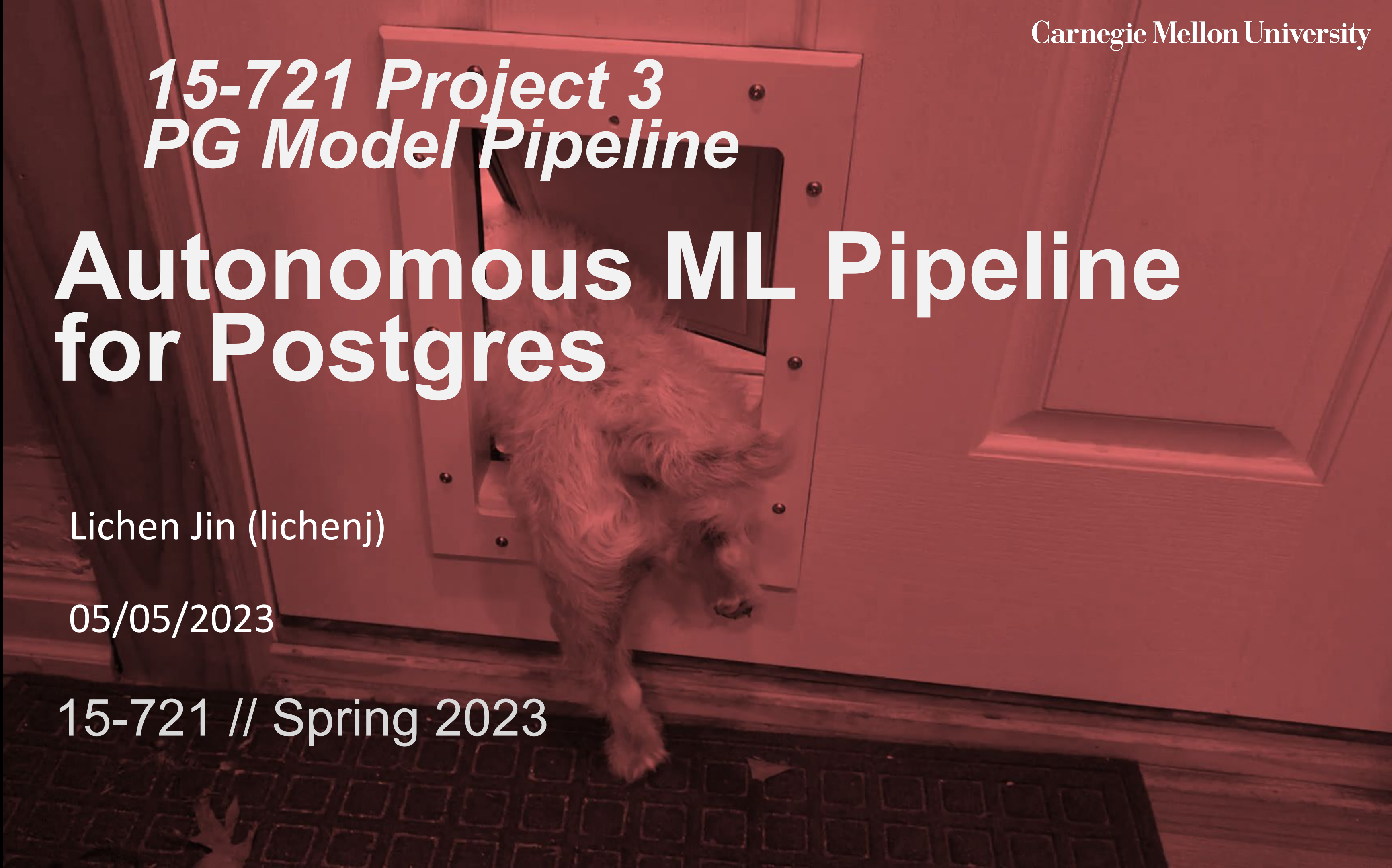
15-721 Project 3
PG Model Pipeline

Autonomous ML Pipeline for Postgres

Lichen Jin (lichenj)

05/05/2023

15-721 // Spring 2023



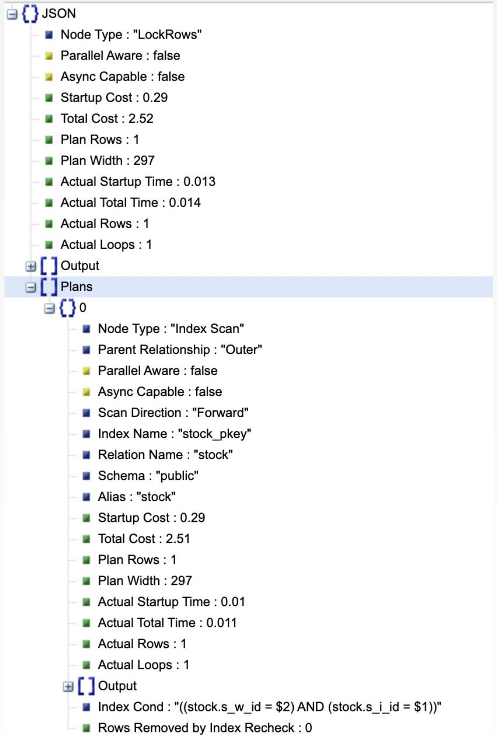
Recap

Project Overview

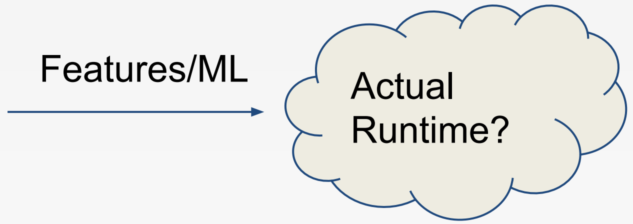
- Self-driving DBMS
  - Heuristic -> Machine Learning
  - Use history workloads
- QPP Task on PgSQL
- Unified pipeline as a daemon service
  - Python

Goal: Compare Modeling approaches fairly

QPP - Query Performance Prediction



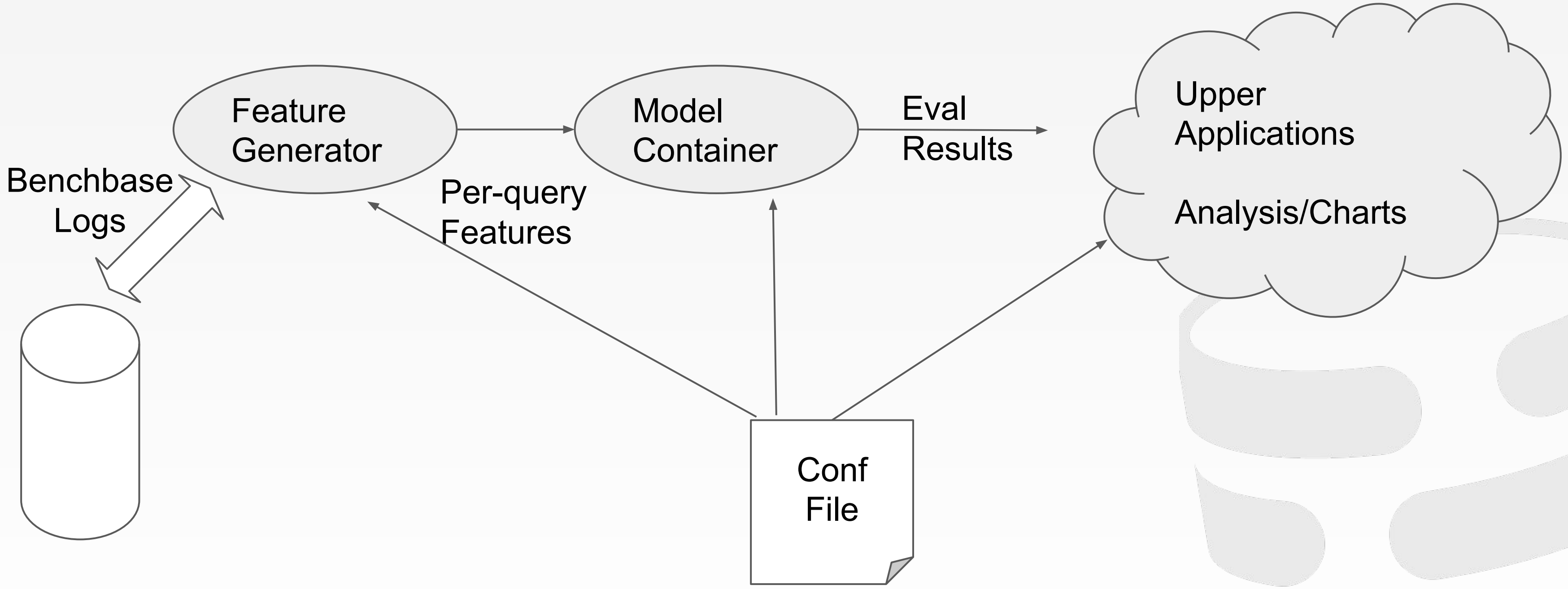
```
JSON
    Node Type : "LockRows"
    Parallel Aware : false
    Async Capable : false
    Startup Cost : 0.29
    Total Cost : 2.52
    Plan Rows : 1
    Plan Width : 297
    Actual Startup Time : 0.013
    Actual Total Time : 0.014
    Actual Rows : 1
    Actual Loops : 1
  Output
  Plans
    0
        Node Type : "Index Scan"
        Parent Relationship : "Outer"
        Parallel Aware : false
        Async Capable : false
        Scan Direction : "Forward"
        Index Name : "stock_pkey"
        Relation Name : "stock"
        Schema : "public"
        Alias : "stock"
        Startup Cost : 0.29
        Total Cost : 2.51
        Plan Rows : 1
        Plan Width : 297
        Actual Startup Time : 0.01
        Actual Total Time : 0.011
        Actual Rows : 1
        Actual Loops : 1
      Output
        Index Cond : "((stock.s_w_id = $2) AND (stock.s_i_id = $1))"
        Rows Removed by Index Recheck : 0
```



- Well studied in SOTA
- Features matters
- Model matters (trivial does not work well)
- Simple X/Y data set

- Messy SOTAs

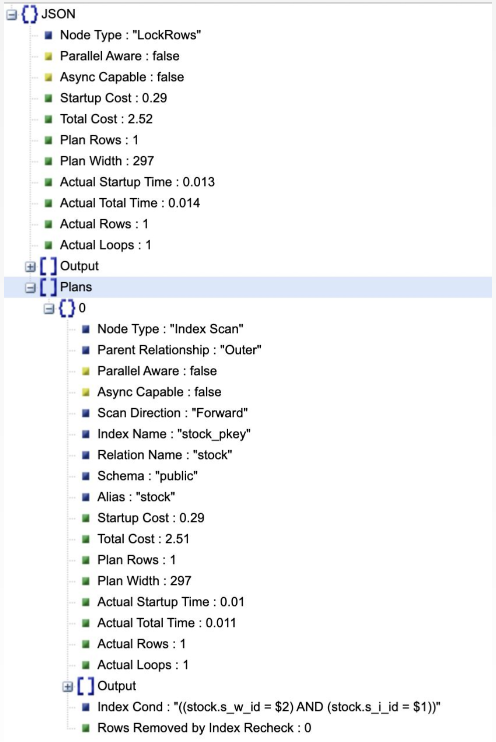
Recap - Design Rationale



Subgoals

- Explore and Validate Various QPP Methods
- Decouple and Integrate Python code into the DbGym (75%)
- Try to get findings in extensive experiments (100%)
- Various benchmark and SF coverage (125%)

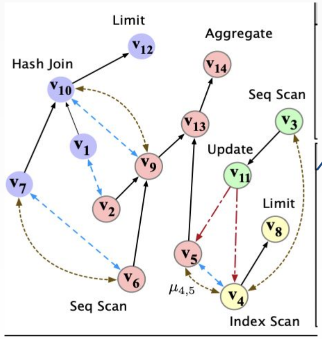
Integrated Method #1: GPredictor



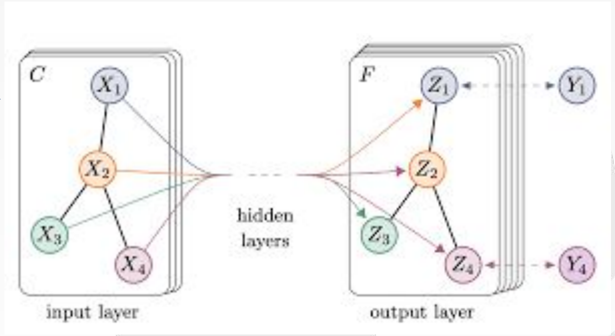
```
JSON
    Node Type : "LockRows"
    Parallel Aware : false
    Async Capable : false
    Startup Cost : 0.29
    Total Cost : 2.52
    Plan Rows : 1
    Plan Width : 297
    Actual Startup Time : 0.013
    Actual Total Time : 0.014
    Actual Rows : 1
    Actual Loops : 1
  [ ] Output
  [ ] Plans
    { } 0
        Node Type : "Index Scan"
        Parent Relationship : "Outer"
        Parallel Aware : false
        Async Capable : false
        Scan Direction : "Forward"
        Index Name : "stock_pkey"
        Relation Name : "stock"
        Schema : "public"
        Alias : "stock"
        Startup Cost : 0.29
        Total Cost : 2.51
        Plan Rows : 1
        Plan Width : 297
        Actual Startup Time : 0.01
        Actual Total Time : 0.011
        Actual Rows : 1
        Actual Loops : 1
      [ ] Output
        Index Cond : "((stock.s_w_id = $2) AND (stock.s_i_id = $1))"
        Rows Removed by Index Recheck : 0
```

Startup Cost: 0.xx
Total Cost: 1.xx
Actual???

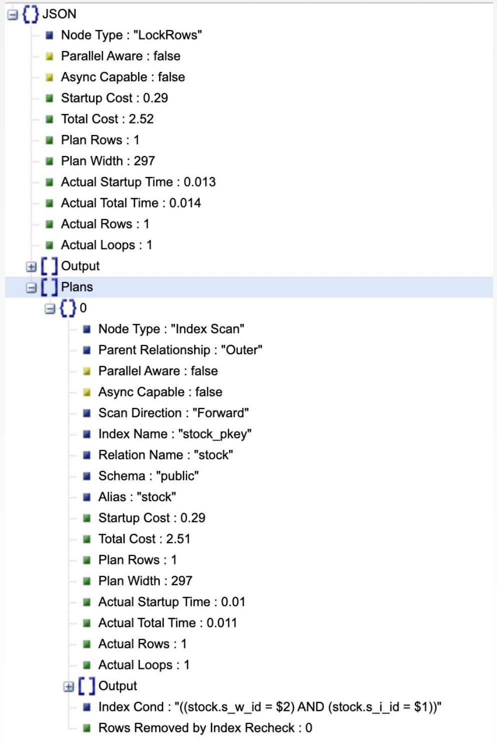
Per - Operator Feature Vector



- Original Impl
- My tweaks



Integrated Method #2: QPPNet



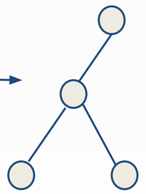
```
JSON
  Node Type : "LockRows"
  Parallel Aware : false
  Async Capable : false
  Startup Cost : 0.29
  Total Cost : 2.52
  Plan Rows : 1
  Plan Width : 297
  Actual Startup Time : 0.013
  Actual Total Time : 0.014
  Actual Rows : 1
  Actual Loops : 1
  [ ] Output
  [ ] Plans
    { } 0
      Node Type : "Index Scan"
      Parent Relationship : "Outer"
      Parallel Aware : false
      Async Capable : false
      Scan Direction : "Forward"
      Index Name : "stock_pkey"
      Relation Name : "stock"
      Schema : "public"
      Alias : "stock"
      Startup Cost : 0.29
      Total Cost : 2.51
      Plan Rows : 1
      Plan Width : 297
      Actual Startup Time : 0.01
      Actual Total Time : 0.011
      Actual Rows : 1
      Actual Loops : 1
      [ ] Output
      Index Cond : "((stock.s_w_id = $2) AND (stock.s_i_id = $1))"
      Rows Removed by Index Recheck : 0
```

Startup Cost: 0.xx
Total Cost: 1.xx
Actual???

OpType-Specified Feature

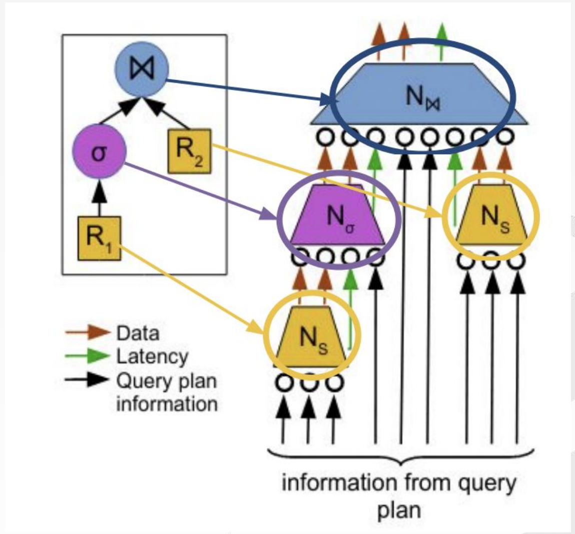
Max-min stat

Per - Operator Feature Vector

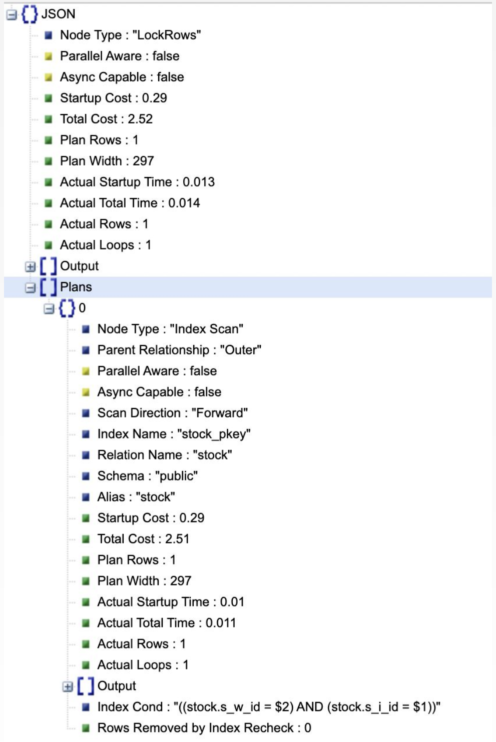


Tree Structure

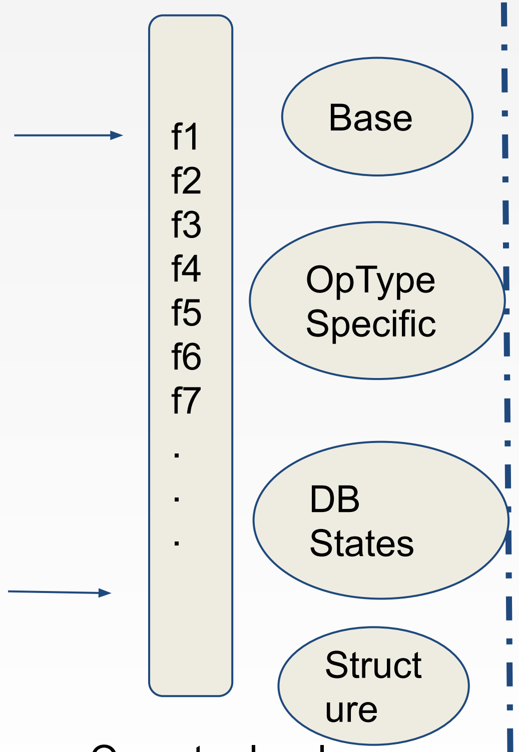
Both models claim to perform super good in their papers??



Integrated Method #3: AutoML



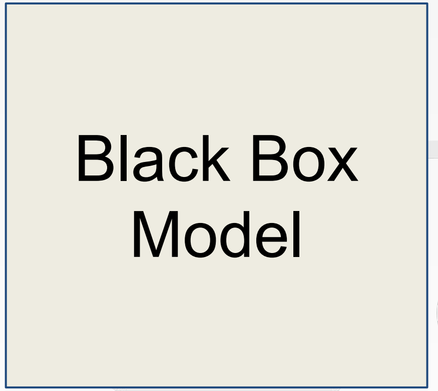
```
{} JSON
    Node Type : "LockRows"
    Parallel Aware : false
    Async Capable : false
    Startup Cost : 0.29
    Total Cost : 2.52
    Plan Rows : 1
    Plan Width : 297
    Actual Startup Time : 0.013
    Actual Total Time : 0.014
    Actual Rows : 1
    Actual Loops : 1
  [ ] Output
  [ ] Plans
    {} 0
        Node Type : "Index Scan"
        Parent Relationship : "Outer"
        Parallel Aware : false
        Async Capable : false
        Scan Direction : "Forward"
        Index Name : "stock_pkey"
        Relation Name : "stock"
        Schema : "public"
        Alias : "stock"
        Startup Cost : 0.29
        Total Cost : 2.51
        Plan Rows : 1
        Plan Width : 297
        Actual Startup Time : 0.01
        Actual Total Time : 0.011
        Actual Rows : 1
        Actual Loops : 1
      [ ] Output
        Index Cond : "((stock.s_w_id = $2) AND (stock.s_i_id = $1))"
        Rows Removed by Index Recheck : 0
```



Operator-level Feature Vector

Makes life easier

Research: How to parse the feature set from the DB?



Feature Comb
Feature Norm
Model Tuning

Data Set and Exp Settings

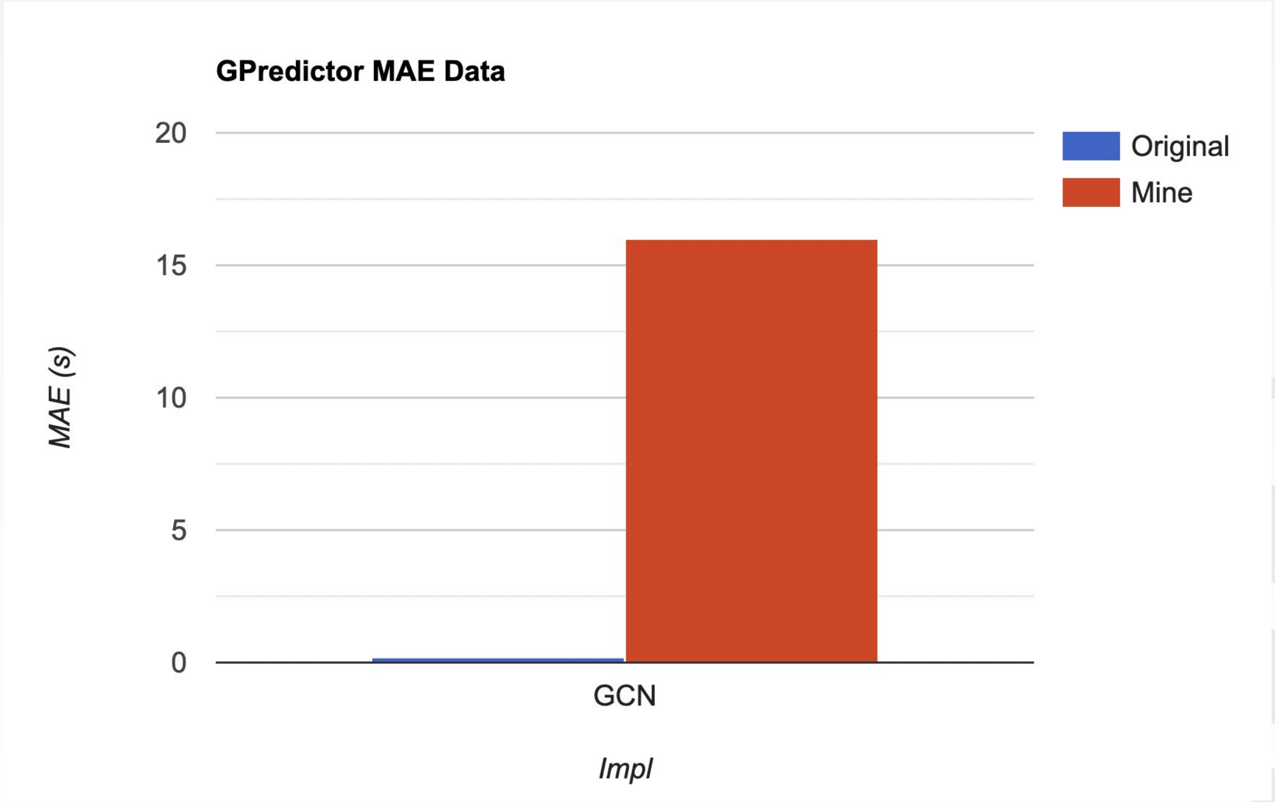
- TPC-H, SF=10, Terminal=1
- EXPLAIN ANALYZE json dict for 3000 queries.
- ~30000 operators. Train-test split.

- Running on PDL-dev7, assuming in-memory execution.

- Average elapsed time per query: 20 seconds.

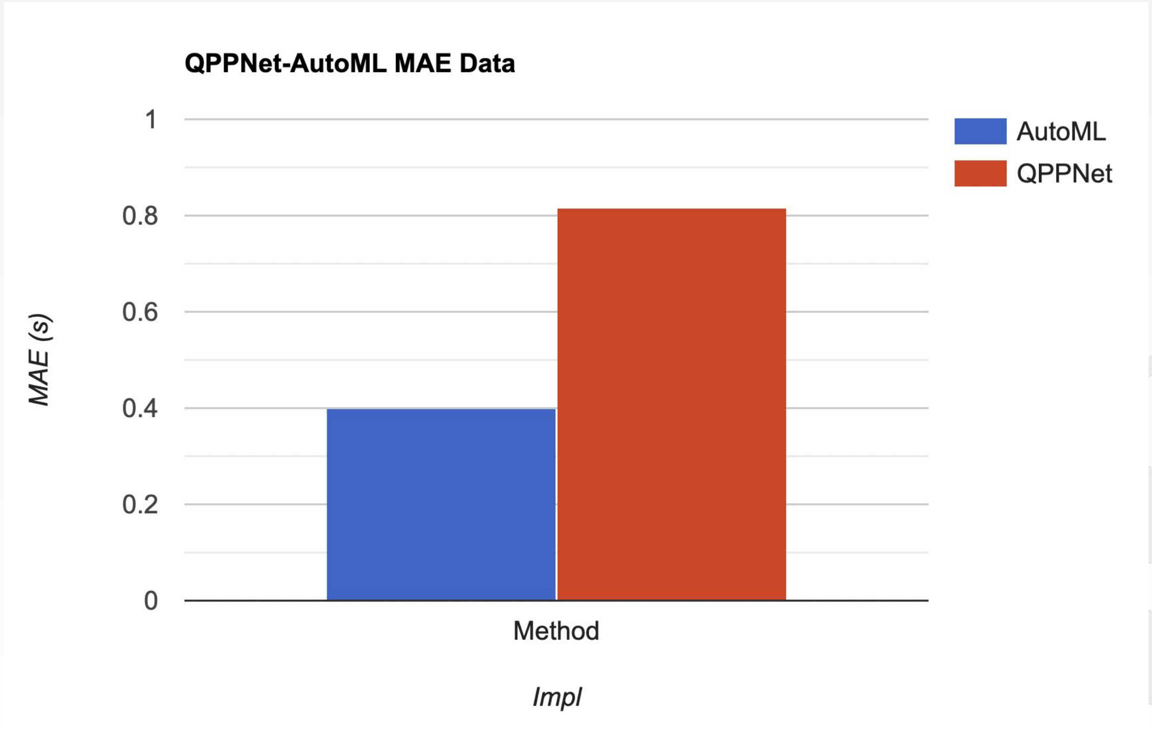
- Metric: Mean Absolute Error (MAE) for the model predictions

Influence of “Ground Truth” Features



Actual Startup Time: The ground truth can be mostly indicated from this variable..

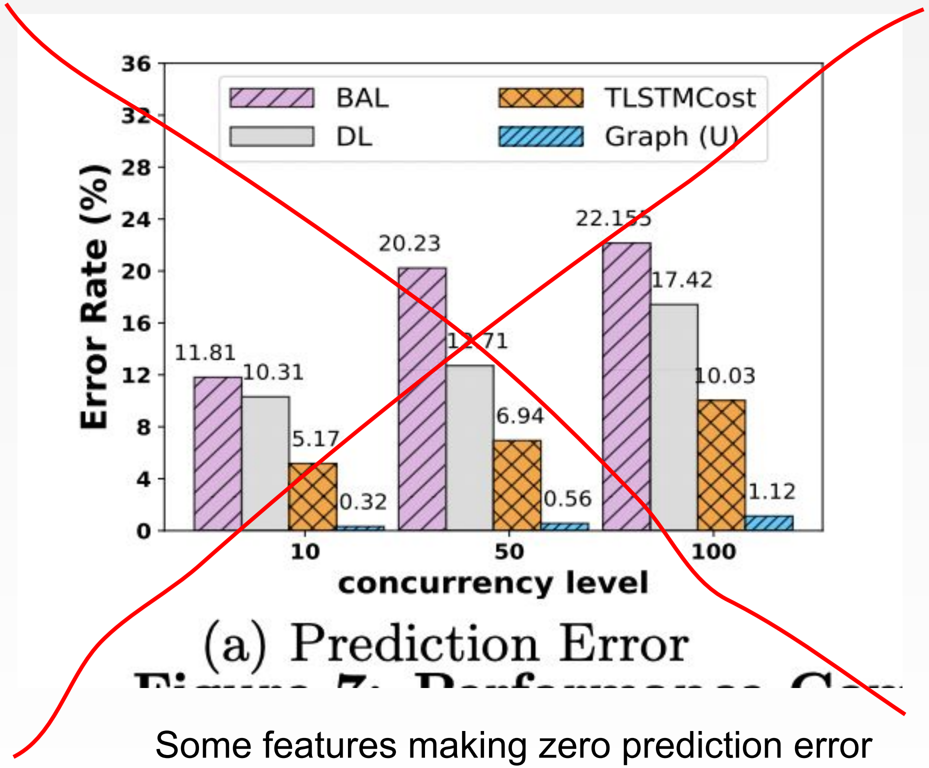
QPPNet and AutoML(QPPNet Feat)



More specific features matters

AutoML is on par and even better than well designed SOTA.

Findings / Takeaways



Some features making zero prediction error

- Information parsed into feature vectors are important (DB, not ML researches)
- Maybe QPPNet makes some sense compared to their results in their paper.
  - Distribution & Stat?
- AutoML shows potentials to reach SOTA modeling performance on QPP.

Future Works

- Run experiments on more benchmarks
- Auto feature set without manual rules for AutoML (under discussion)
- Other typical tasks in ML for DBMS

- Technical debts: Hacks of passing traces files in containerized service.

Acknowledgements

Idea: Andy, Wan, William

DbGym Framework: Wan

Resources: CMU-DB