Bao: Learning to Steer Query Optimizers

Ryan Marcus, Parimarjan Negi, Hongzi Mao,Nesime Tatbul, Mohammad Alizadeh, and Tim Kraska

Presented by Armins Bagrats Stepanjans

Contributions

DBMS agnostic integration with the query optimizer

Competitive performance

Realistic evaluation setting

Structure of this presentation

Problem overview

Model's components

Evaluation

My review

Previous work

Fully learned systems (e.g. Neo, Kipf et al. Learned Cardinalities)

Heuristic query search (e.g Genetic Query Optimizer)

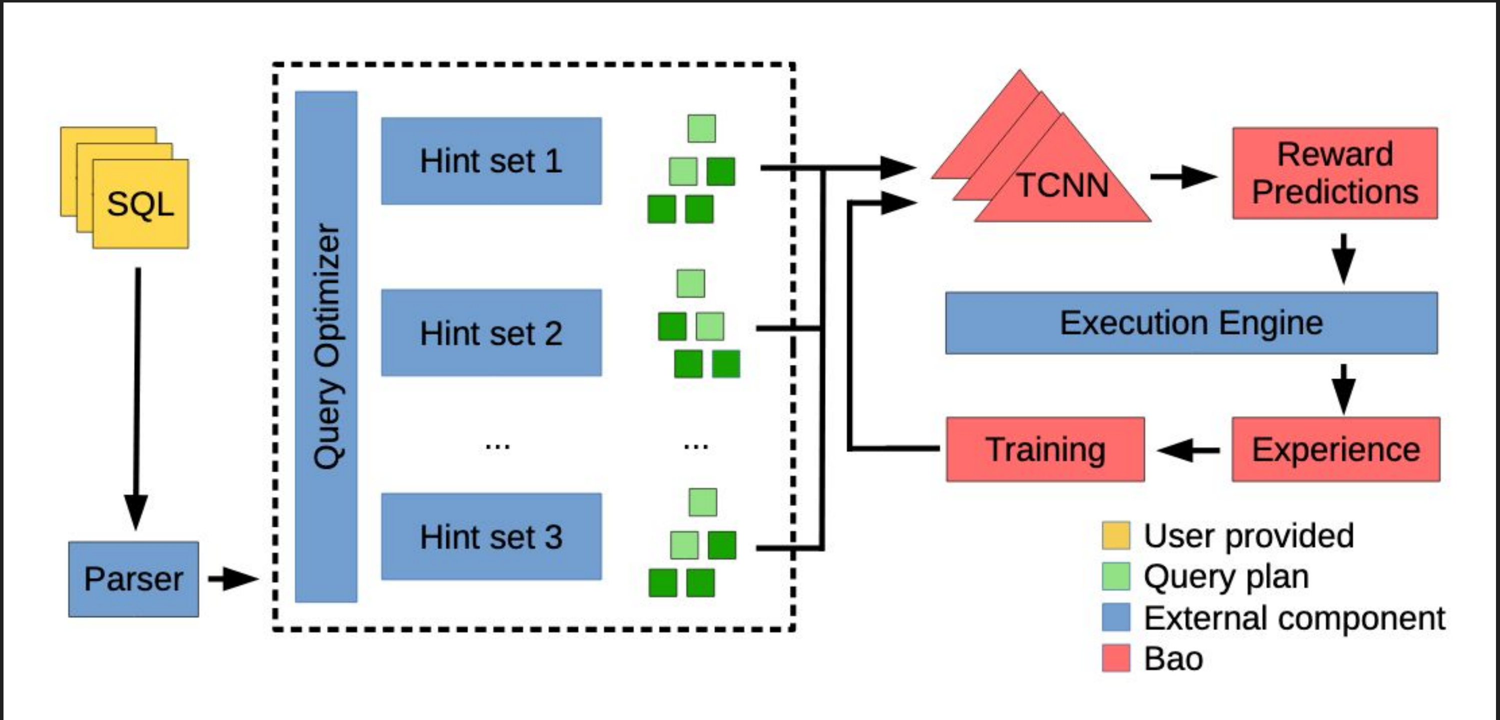
Issues Bao Addresses

Training efficiency

Robustness to changes in data

Tail performance

Query Steering



Thompson Sampling

Balances between exploration and exploitation when training.

Exploration only: choose actions at random

Exploitation only: choose the action that maximizes expected likelihood of the learned distribution.

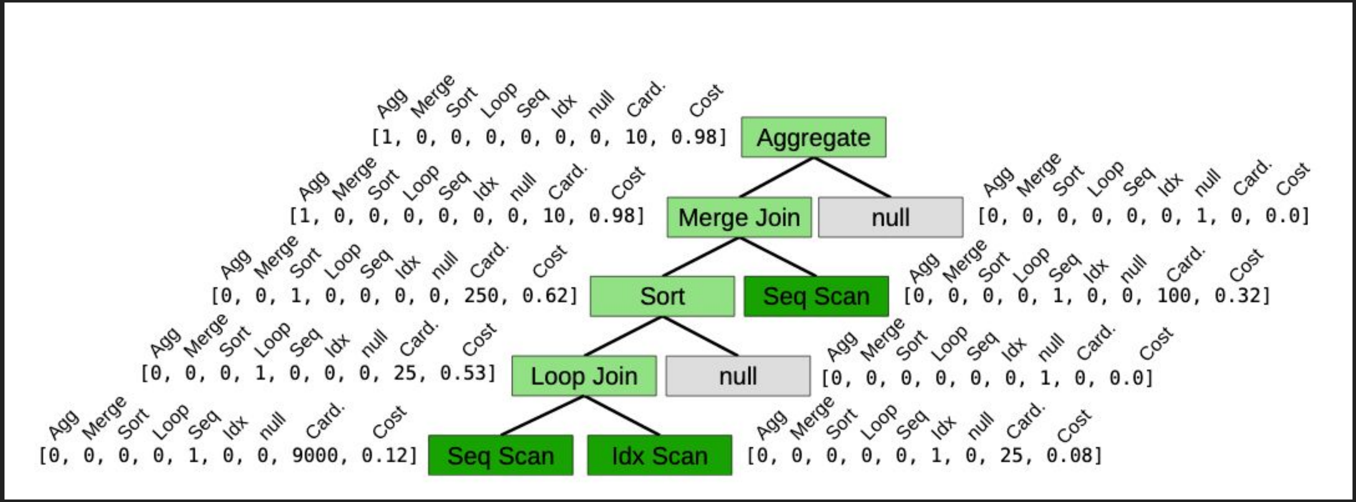
Thompson sampling: sample the action from the learned distribution.

Vectorized query tree

Tree of feature vectors

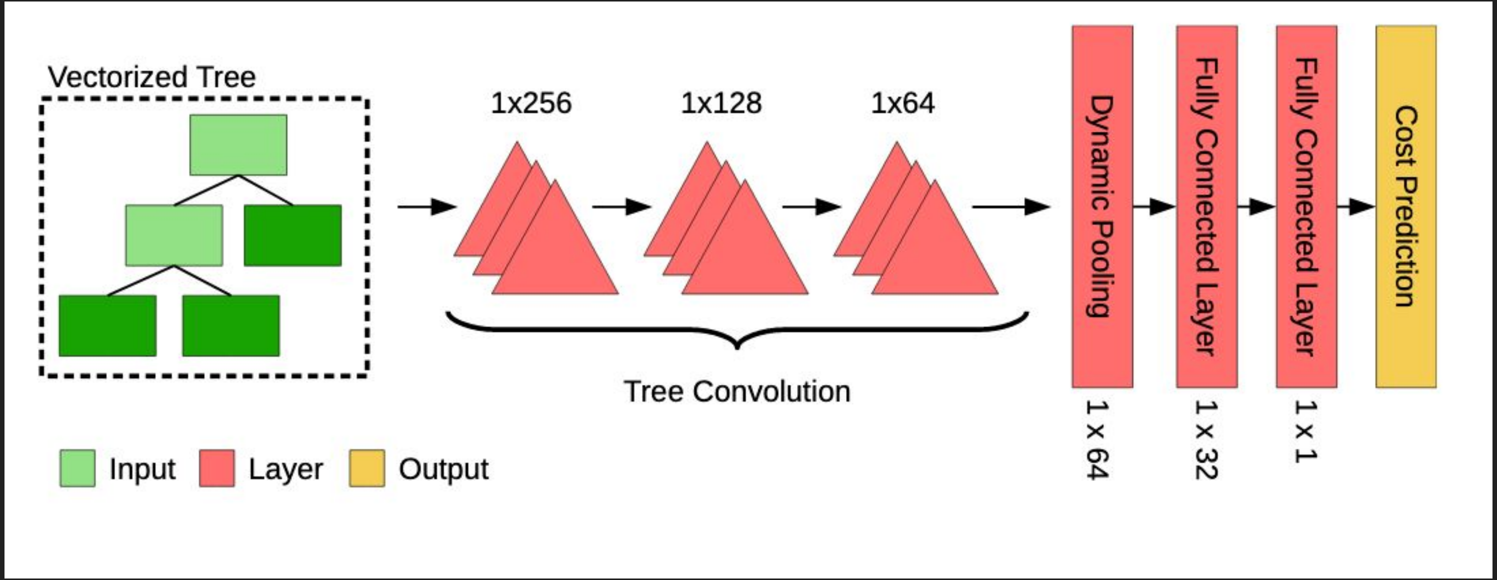
Schema independent

Extensible (new features can be added by concatenation)



Tree Convolutional Neural Networks

Identify patterns in query tree corresponding to (in)efficient queries.

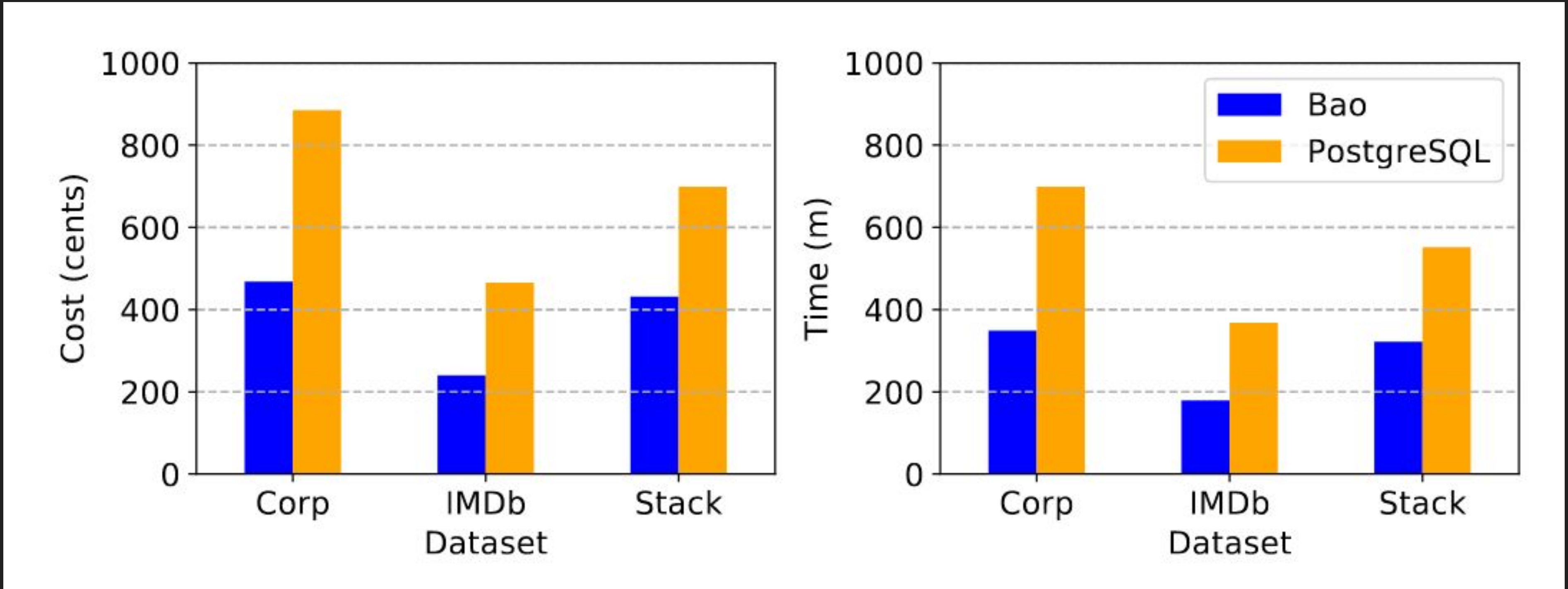


Cost function (regret)

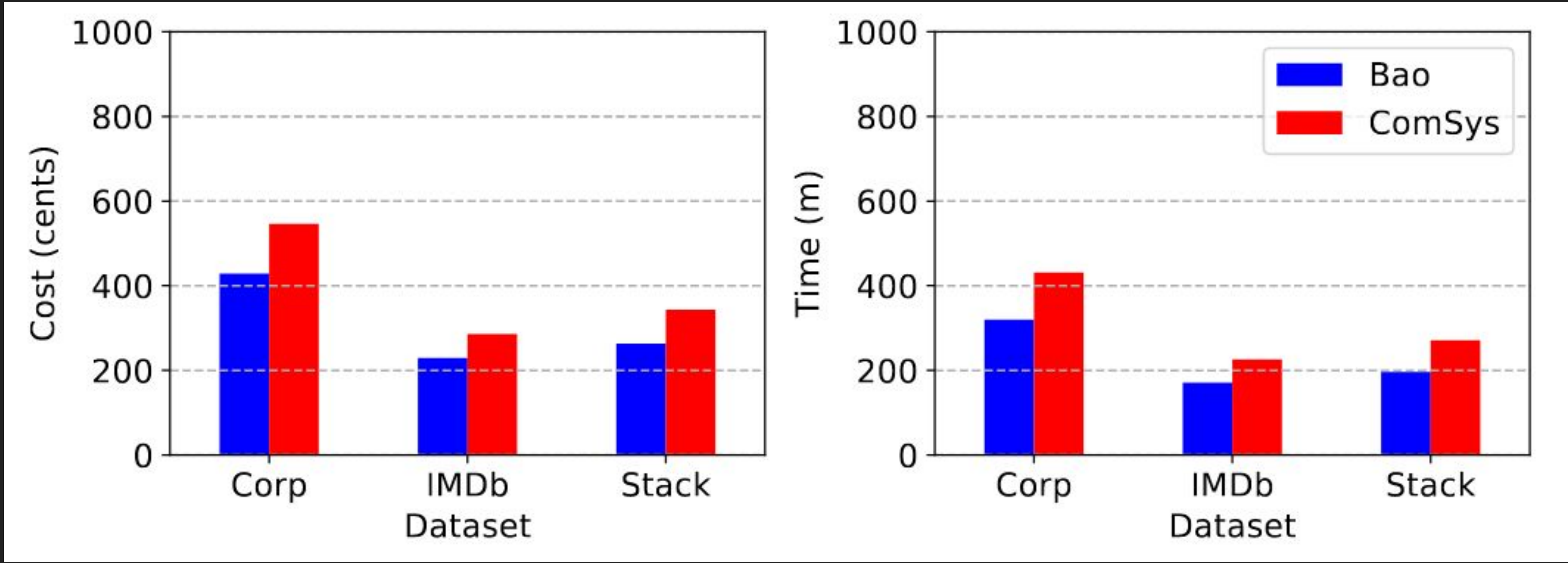
Mean squared distance from *optimal performance*

It is up to the user to define the *performance metric*

Bao v. PostgreSQL



Bao v. ComSys(?)


1000
800
600
400
200
0
Cost (cents)
Corp
IMDb
Stack
Dataset
1000
800
600
400
200
0
Time (m)
Bao
ComSys
Corp
IMDb
Stack
Dataset

Criticism

Parts of the evaluation were performed on proprietary systems and datasets.

Query optimiser is run 48 times for each Bao prediction.

Fails to incorporate the ideas expressed by the authors in SageDB

Summary

Bao integrates query optimizers into multi-handed bandit optimization problem.

The paper provides a DBMS and schema agnostic query tree representation.

The model is evaluated with representative datasets and cost metrics.

Please ask questions!

Thank you for your time