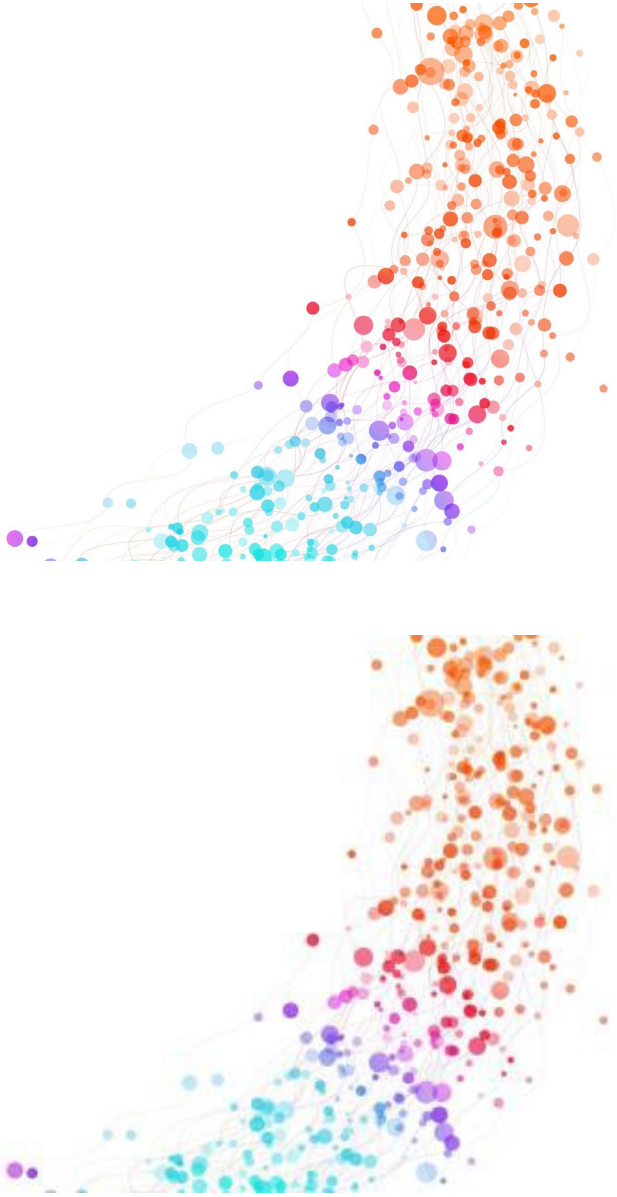
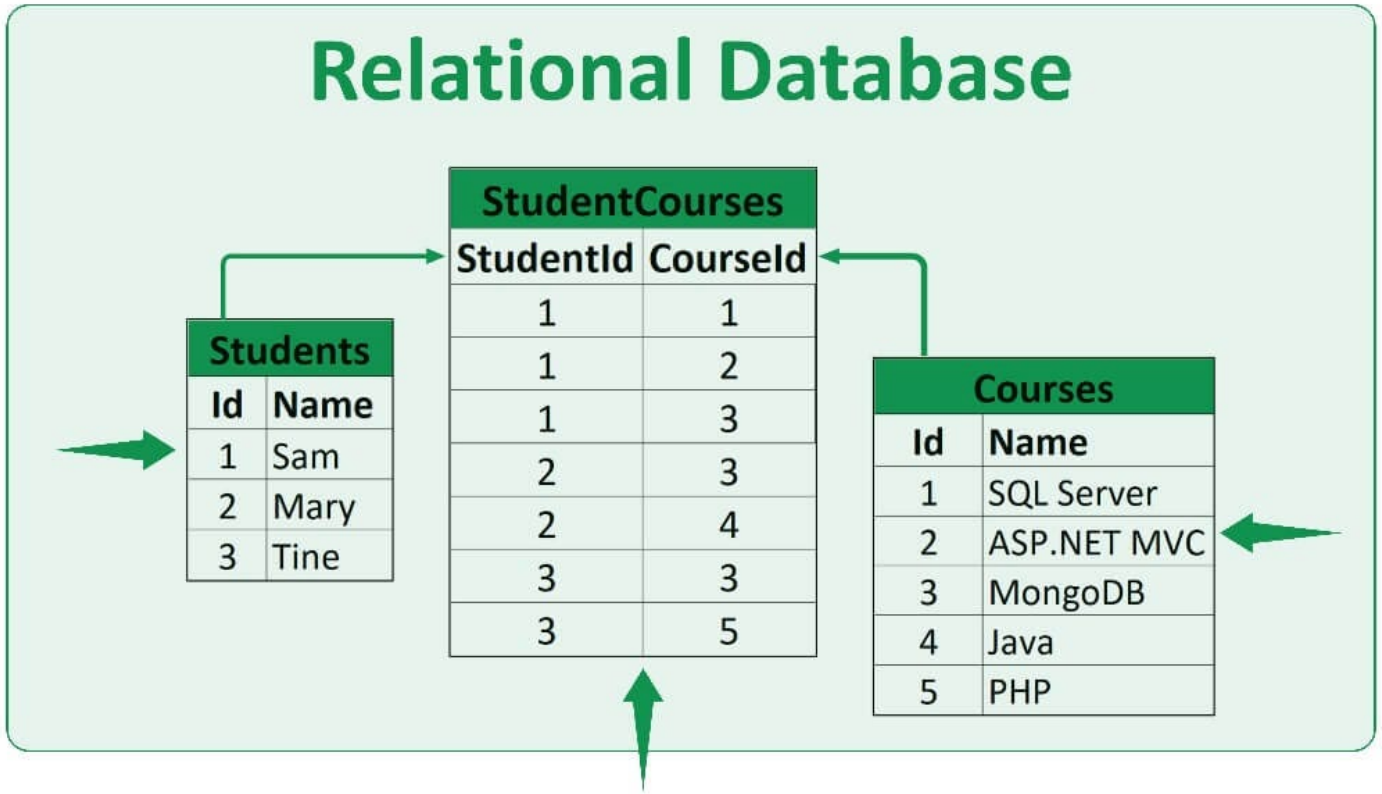
# Bao: Learning to Steer Query Optimizers

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






# Background: Query Optimization

- Query optimization in relational database systems

# Key Problem

**Query plan estimation**

Optimizing complicated joins

Handling dynamic workloads

Improve tail performance

# Solution: BAO

Bandit Optimizer using ML to give “hints” to query plan optimizer

Improves average-case performance

Improves tail performance

# Previous Works/Novelty

- Improved cardinality estimation using neural networks [4]
- Neo: Neural Optimizer for entire query plan optimizer based on deep RL [5]
  - Performs poorly in tail performance
- **Bao:** “first learned query optimisation system that outperforms both open source and commercial systems in cost and latency, all while adapting to changes in workload, data,and schema” [2]

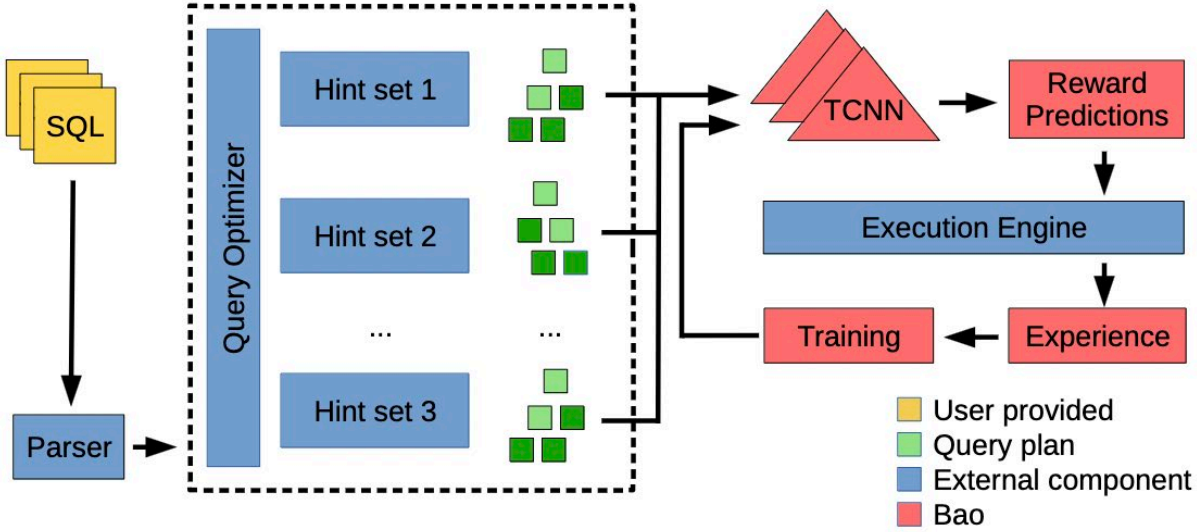



Figure 2: Bao system model

# Approach

- Selects a set of hints within existing parameter space
- Predict execution time using tree convolutional neural networks
- Thompson sampling

# Approach

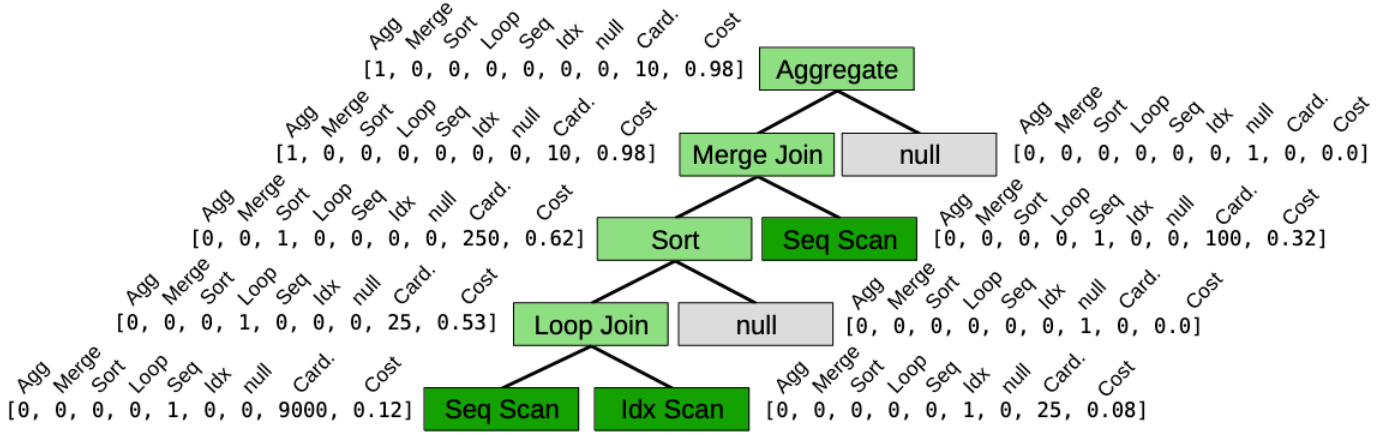



Figure 4: Vectorized query plan tree (vector tree)

# Approach

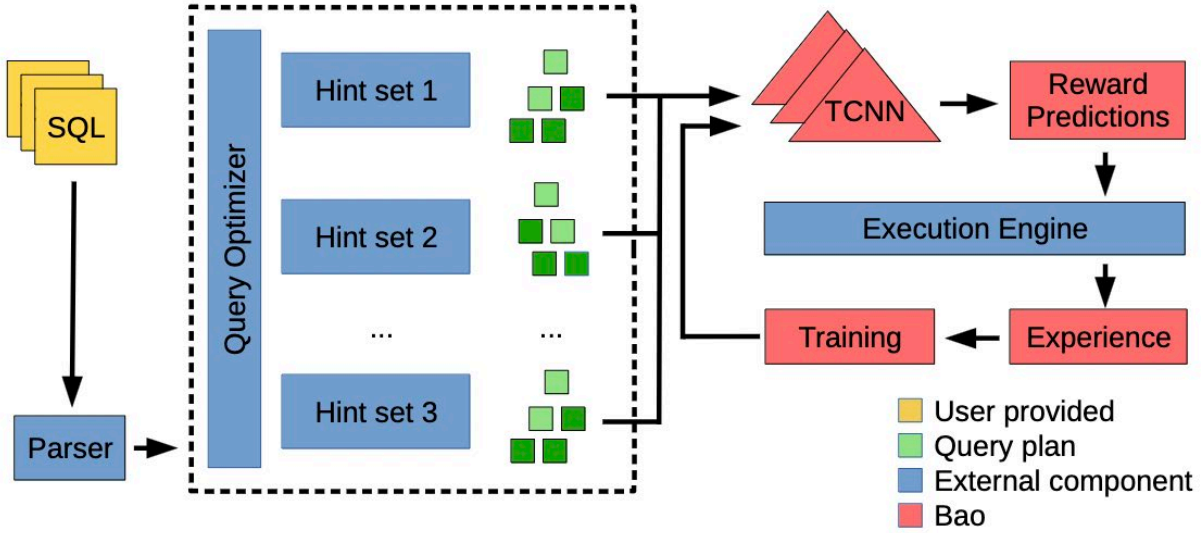
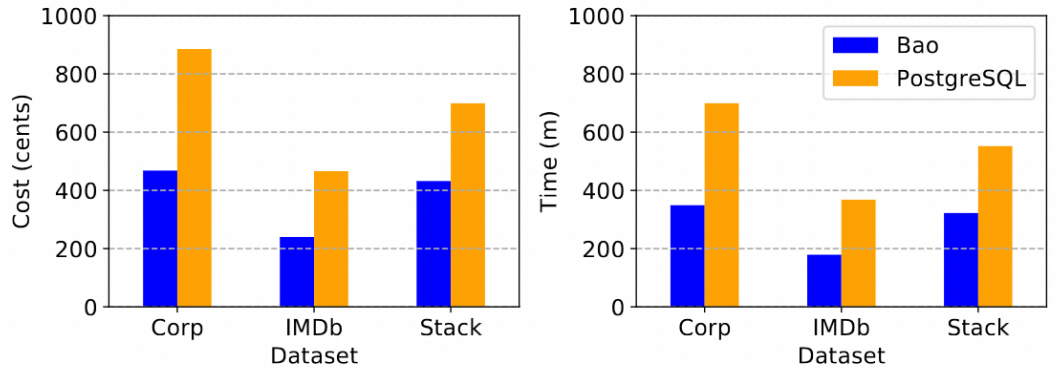
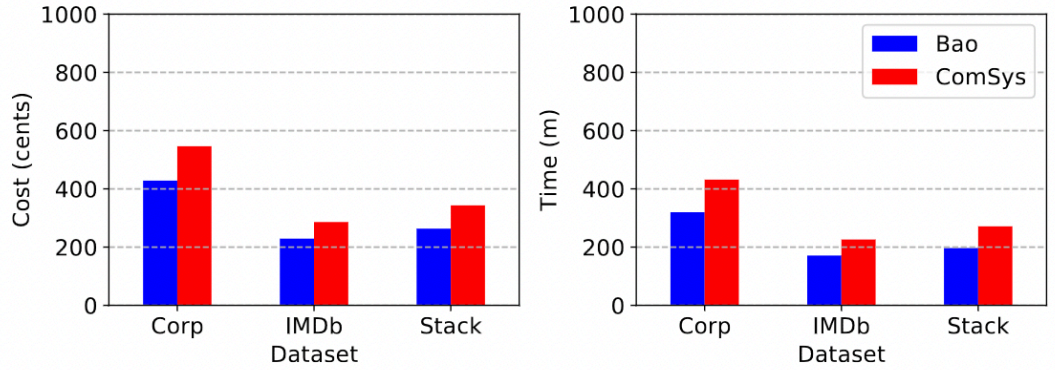



Figure 2: Bao system model




(a) Across our three evaluation datasets, Bao on the PostgreSQL engine vs. PostgreSQL optimizer on the PostgreSQL engine.




(b) Across our three evaluation datasets, Bao on the ComSys engine vs. ComSys optimizer on the ComSys engine.

# Results

- Improves average performance on open source and commercial systems

# Results

- Avoids tail catastrophe problem

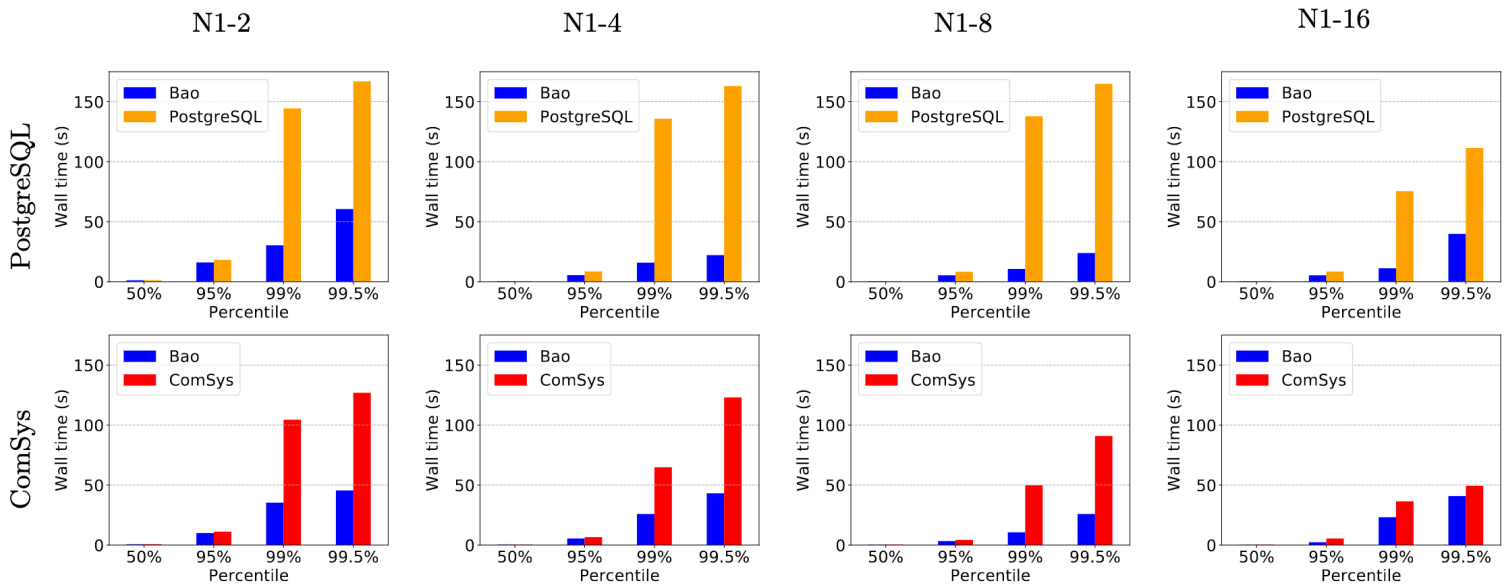
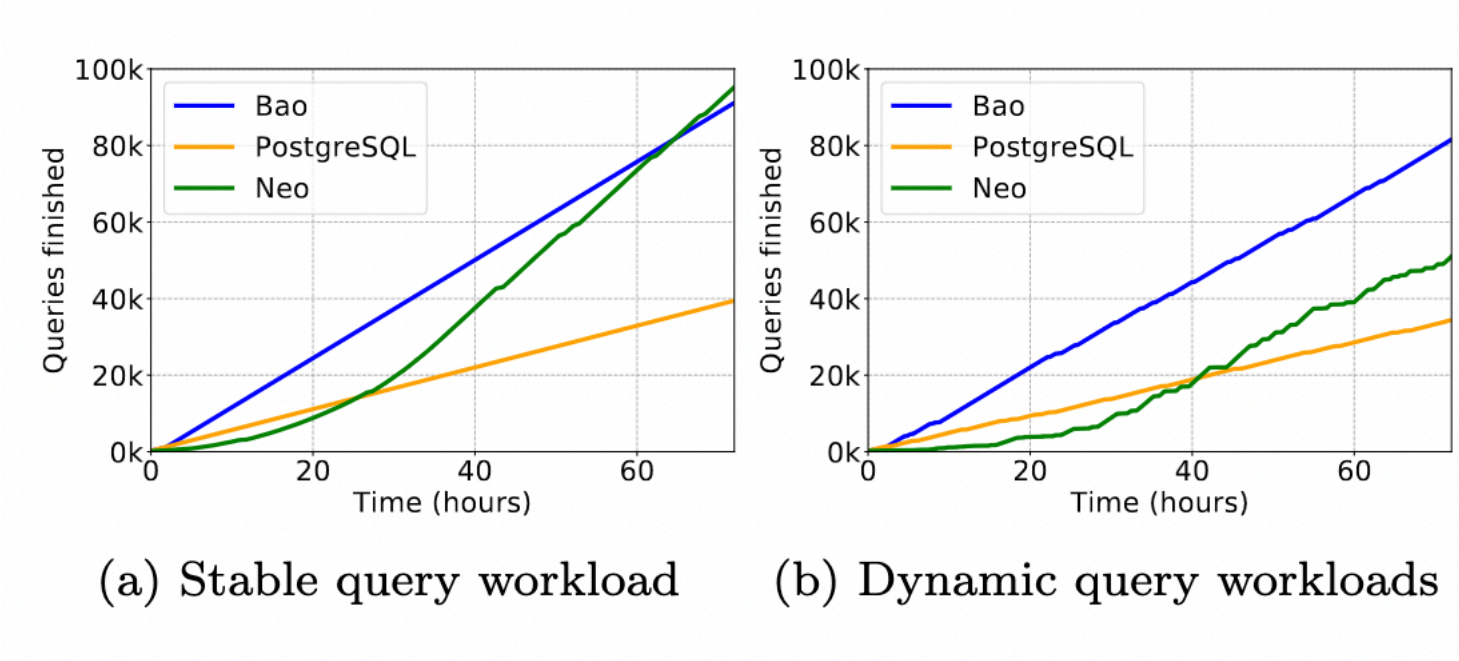



Figure 8: Percentile latency for queries, IMDb workload. Each column represents a VM type, from smallest to largest. The top row compares Bao against the PostgreSQL optimizer on the PostgreSQL engine. The bottom row compares Bao against a commercial database system on the commercial system's engine. Measured across the entire (dynamic) IMDb workload.

# Results

- Improvement over Neo
- Resiliency testing




(a) Stable query workload (b) Dynamic query workloads

# Summary

- Bao uses a hint optimization system
- Works within any existing optimization system

# Pros

- Extensive testing
- Clear results
- Bao can easily adapt and generalize

# Cons

- Restricted to traditional query plan
- Fixed number of hints

# Impact and Future Work

- Impressive and novel system that makes significant optimization improvements
- 39 citations since 2020
- Learn to produce independent physical plans
  - Balsa [3]

# Works Cited


- 1. Ryan Marcus, Parimarjan Negi, Hongzi Mao, Nesime Tatbul, Mohammad Alizadeh, Tim Kraska.Bao:Learning to Steer Query Optimizers.2020. arXiv preprint https://arxiv.org/pdf/2004.03814.pdf.
- 2. Ryan Marcus et al. “Bao: Making learned query optimization practical”.In:ACM SIGMOD Record51.1 (2022), pp. 6–13.
- 3. *Yang, Z., Chiang, W., Luan, S., Mittal, G., Luo, M., & Stoica, I. (2022). Balsa: Learning a Query Optimizer Without Expert Demonstrations. Proceedings of the 2022 International Conference on Management of Data.*
- *4.* Henry Liu, Mingbin Xu, Ziting Yu, Vincent Corvinelli, Calisto Zuzarte.Cardinality estimation usingneural networks.ACM ‘15. 2015
- *5.* Ryan Marcus, Parimarjan Negi, Hongzi Mao, Chi Zhang, Mohammad Alizadeh, Tim Kraska, OlgaPapaemmanouil, Nesime Tatbul.Neo: A learned query optimizer.2019. Proceedings of the VLDBEndowment. https://doi.org/10.14778/3342263.3342644.

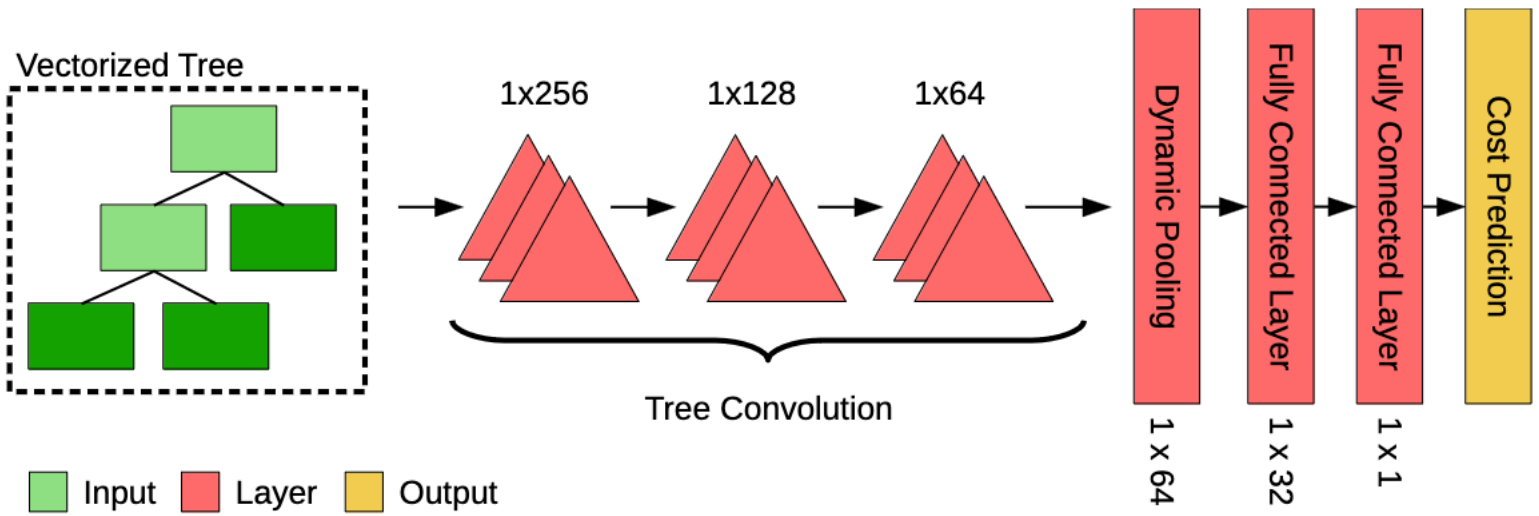



Figure 5: Bao prediction model architecture