Using Data-Driven and Zero-Shot Learning to learn DBMS Components

Abstract

Workload-driven learning is a technique to replace a DBMS component with a machine learning model

- **Issue**: For each new database or component, a new model must be trained. This makes it very inflexible and \bullet expensive to train.
- Solution: Use data-driven and transfer learning approaches to reduce training effort and make the model generalizable to unseen databases

Data-Driven Learning

Idea: Model **learns data characteristics** like the data's distribution and correlation across complex relational databases

- No training workload needed as the model relies on data only
- Retraining the model only takes a few minutes
- Support for tasks that do not consider workload (cardinality estimation, AQP, indexing)

Goal: Construct Relational Sum-Product Network (RSPN) from

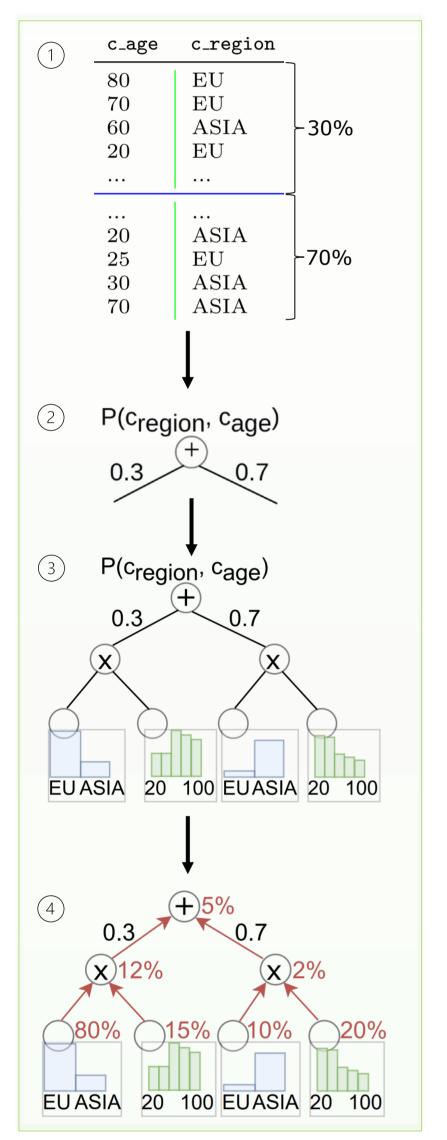
Zero-Shot Learning for Databases

Idea: Inline to other zero-shot approached (e.g. GPT-3), train a model that can generalize to unseen databases out-of-the-box.

- No queries on database are required for training
- Broader applicability to different tasks (physical cost estimation, knob tuning, physical design tuning)

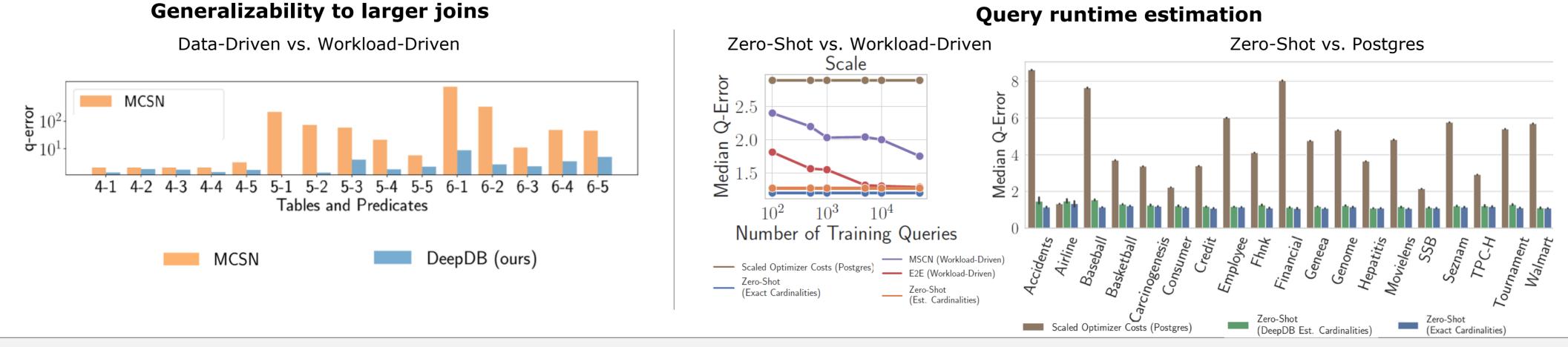
Concept

database



Training a Zero-Shot Cost Model Inference on Unseen Database (one-time-effort) (for every new database) + Workload₁ \longrightarrow Runtimes DB₁ Split independent rows into row DB₂ + Workload₂ Runtimes DB + Workload, Training Data clusters (e.g. using KMeans) DB_n + Workload_n Runtimes Zero-Shot Cost Model Features Features Labels Zero-Shot Cost Model Use sum node and add weights Pre-training across multiple DBs Reuse model on unseen DB corresponding to the row cluster sizes to the edges **Key Challenges** In each row cluster, split independent 1. Challenge: Query encoding that columns into column clusters (product generalizes across databases Zero-Shot Encoding node) (Transferable Representation) **Problem:** Can't use representation of • If not all columns are independent, **Graph Encoding** Transferable Featurization workload-driven models as they don't allow start again with the first step, Node Featurization generalizes Encode Physical Plan Operators, Predicates Tables and Columns as a Graph across Databases transfer across DBs otherwise continue Solution: Operator (One-Hot) Cardinality 1 0 0 1 Aggregate • Capture query plans as graph encoding Hash Join 230K 0 1 MIN(... ⁽⁴⁾ • Use RSPN to compute probabilities • Lean how expensive a certain Hash Seq Scan 0 1 0 550K 4 operation on a dataset is on arbitrary attributes of the table Seq Scan Predicate Operator (One-Hot) • Encode information (e.g. data type, • Example: SELECT COUNT(*) FROM 0 0 tuple width) of data where operations Data Type (One-Hot) Width Customer C WHERE c region='EU' are executed on 4 company_type AND c age<30 yields 5% No Tuples No Pages Annotate operators with the production_year 21 21 movie_companies intermediate sizes they are executed title on (\rightarrow) informed by data-driven models) Estimated value can be used to **select** optimal query plan Generalization 2. Challenge: Sufficient training performance stagnates 🖌 10M Median Q-Erro 1.75**Goal**: Find out when model is sufficiently trained Idea: Estimate using holdout databases. If .50 estimated performance is acceptable or channel='ONLINE region='EU annel='ONLINE stagnates, stop .25Orders Orderlines Customers Orders Orderlines Customers **Assumption**: Every DB and workload is 51015sampled i.i.d. Number of training datasets

Evaluation



Vincent Melisch

Diploma Computer Science Student at the TU Dresden

E-Mail: vincent.melisch@tu-dresden.de

Resources

Based on Prof. Dr. Carsten Binnig's lecture Learned DBMS Components as part of the Lecture Series on Database Research

Graphics are taken from the slides of Prof. Dr. Carsten Binnig



Runtimes

Predictions