Batch Query Memory Prediction Using Deep Query Template Representations

GRADUATE PROGRAM IN ELECTRICAL AND COMPUTER ENGINEERING

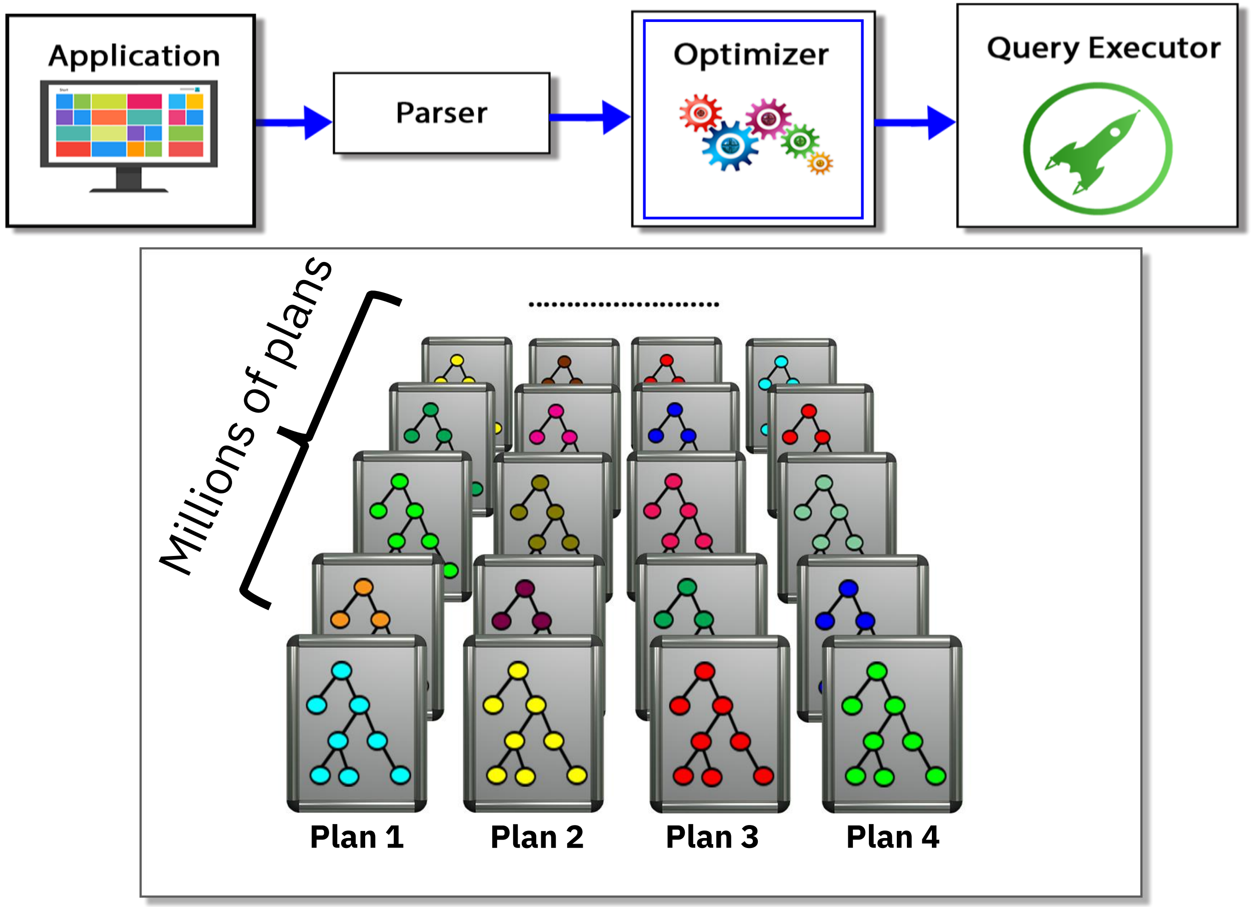
SUPERVISORS: MANOS PAPAGELIS, MARIN LITOIU

M.SC. THESIS OF NICOLAS ANDRES JARAMILLO DURAN

LASSONDE SCHOOL OF ENGINEERING | YORK U

Motivation

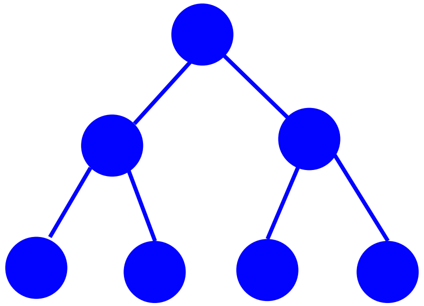
DBMS – Query Optimization



LASSONDE SCHOOL OF ENGINEERING | YORK U

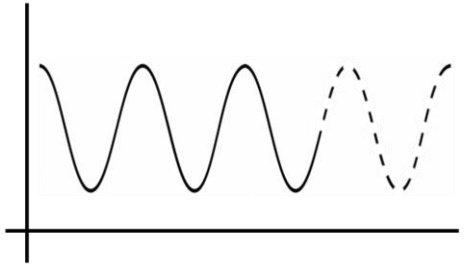
Towards an Autonomous DBMS

Physical Design



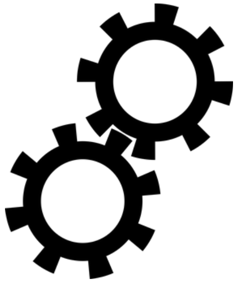
Which index should I create?

Query Performance Prediction



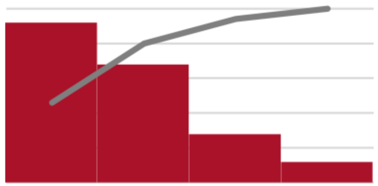
How long it will take to process the query at hand?

Configuration Tuning



What are the optimum configurations for my database?

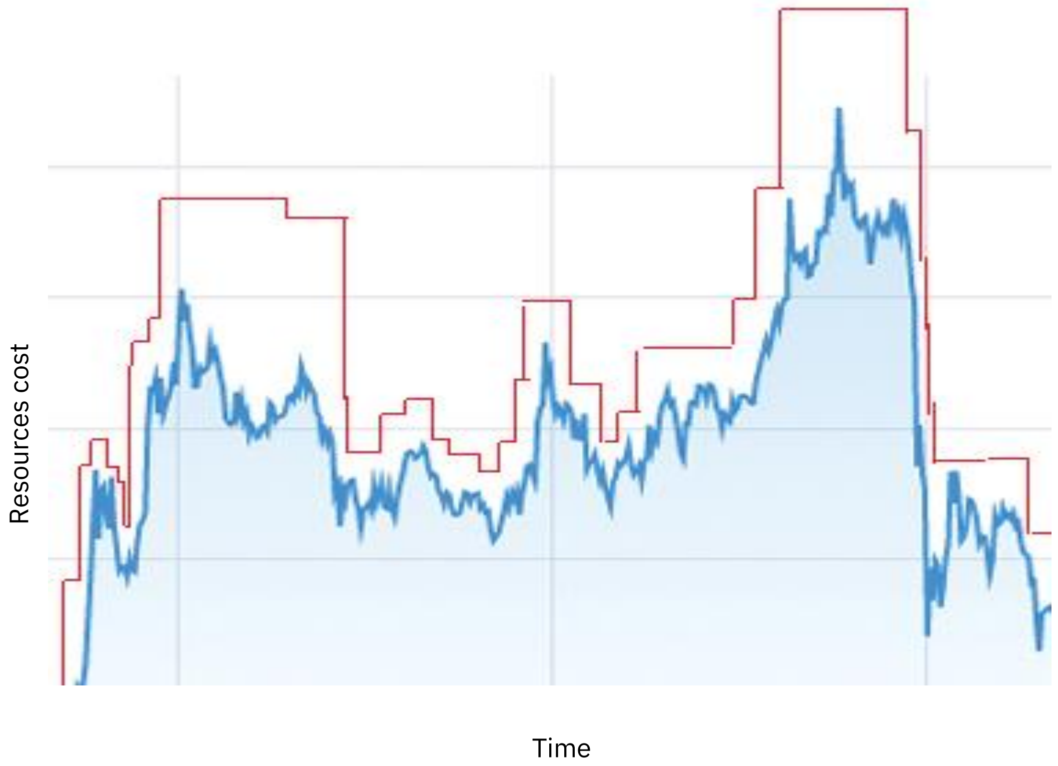
Resource Forecasting



What resources are needed to process this query?

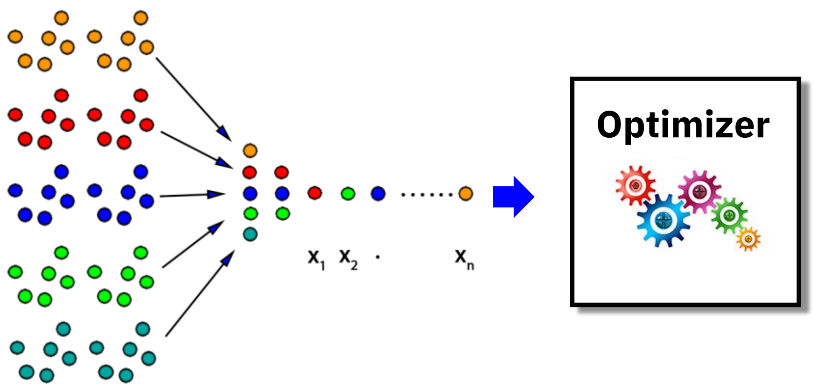
Resource Forecasting

- Resources
  - Memory
  - Disk I/O
  - Throughput
- Challenging problem
  - Excessive overhead
  - Inefficient scheduling
  - Inappropriate allocation of resources

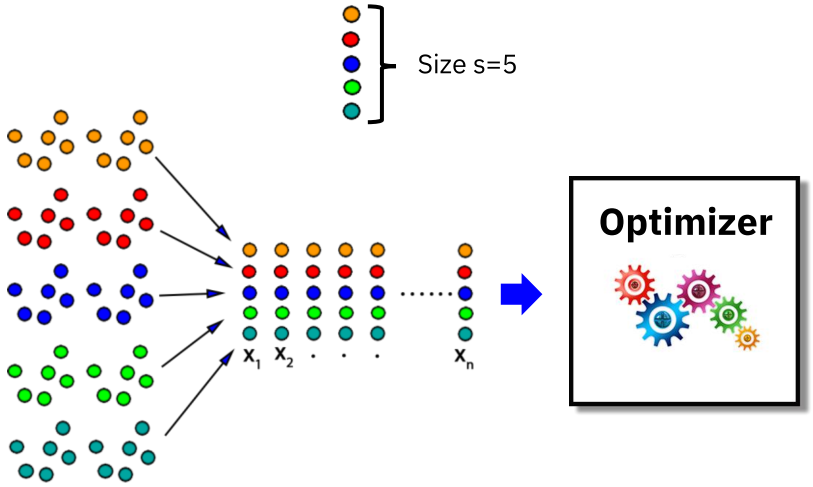


Single Query vs Workload (Batch Queries)

- Single Query



- Workload (Batch Queries)



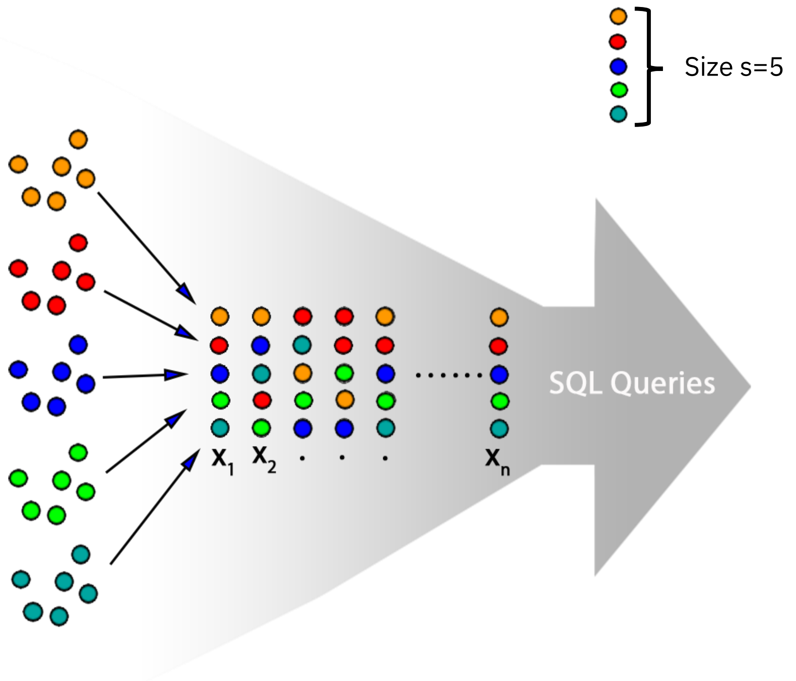
can we design **machine learning** models for predicting **memory** resource forecasting for a **batch of queries**?

Problem Definition

Problem Definition

Input

A query workload of size k



Output

Query Workload memory consumption estimate

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

Problem Definition

- We formulate the estimation of memory usage for an unseen workload as a **distribution regression problem**
- Problem:

  Given

  - $w_i \ = (Q_i, y_i)$
  - A training corpus $\{(Q_1, y_1), \ldots, (Q_n, y_n)\}$

  We wish to learn a function, $\hat{f}\ (\cdot)$, an approximation of $f(\cdot)$

  $\hat{f}\ (\cdot)$, can compute $\hat{y}$, an accurate estimate of the actual memory usage y

$$\hat{f}(w) = \hat{y}$$

- Assumptions:
  - An underlying function, $f(\cdot)$

$$f\ (w)\ =\ y$$

Methodology

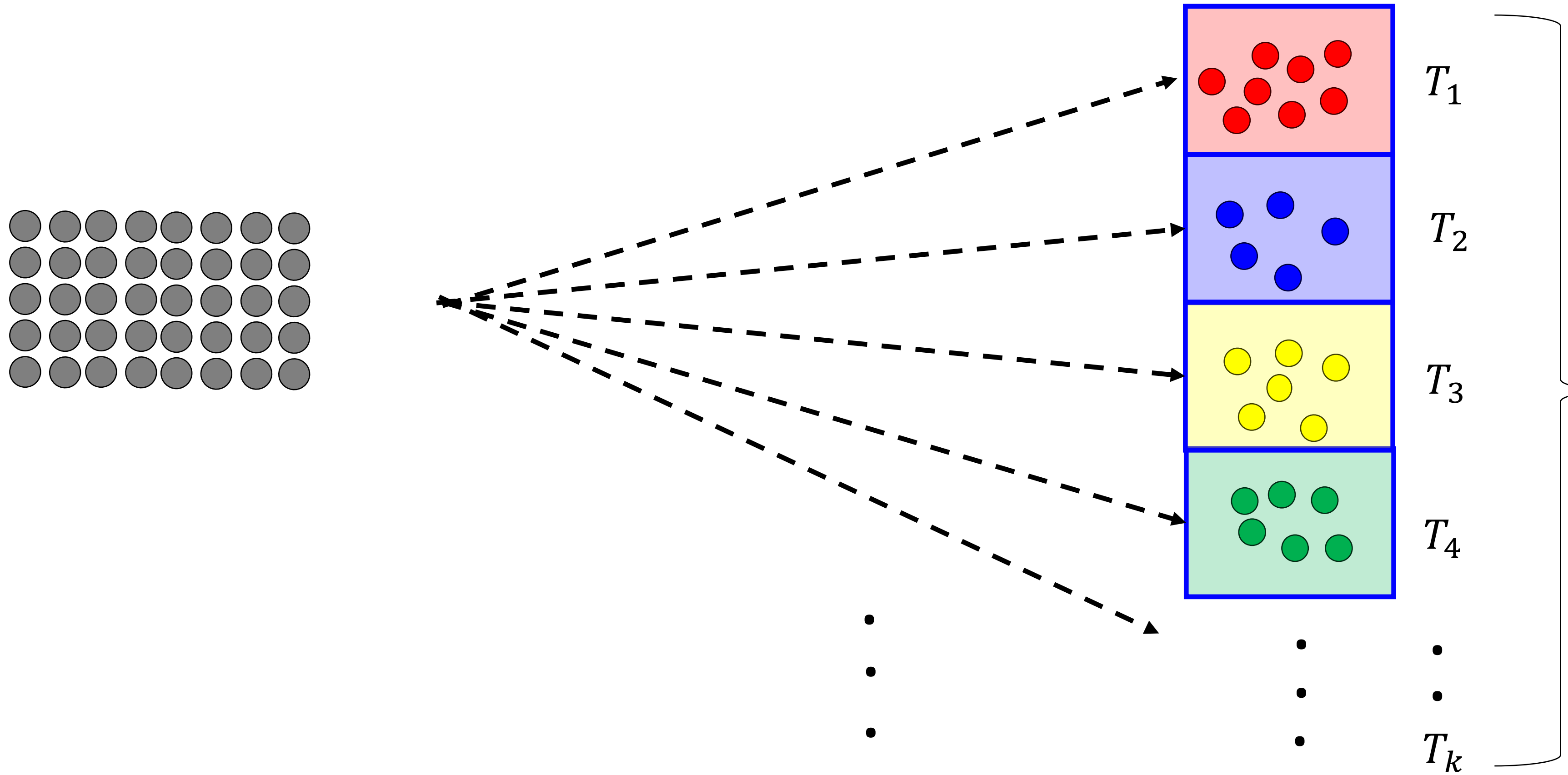
Three-Step Approach for Batch Query Memory Prediction

Phase 1
Learning Query Templates

Phase 2
Constructing Histograms from Workloads

Phase 3
Training a Distribution Regression Machine Learning Model

Phase 1: Learning Query Templates



Phase 1: Learning Query Templates

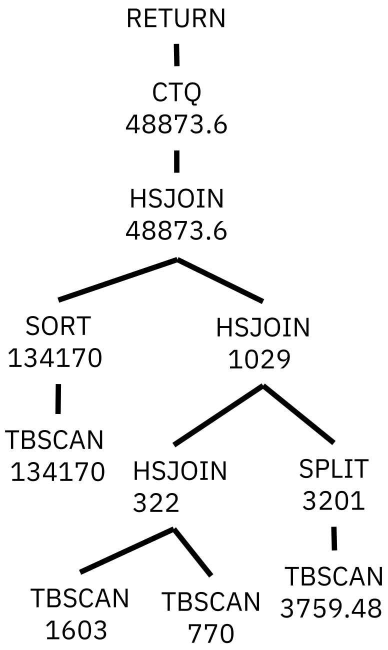
Rule Base			Map the generated queries back to their corresponding templates
			Group queries based on their estimated cardinality similarity
Clustering Base	K-Means	Query text feature encoding	Bag of words (BoW)
			Word embedding encoding
			Text mining approach
		Query plan feature encoding	Cardinality cost aggregation for each operator (CCAEO)
			Query plan encoding while maintaining tree structure (QPEWMTS)
	K-meadois	Query text feature encoding	Bag of words (BoW)
			Word embedding encoding
			Text mining approach
		Query plan feature encoding	Cardinality cost aggregation for each operator (CCAEO)
			Query plan encoding while maintaining tree structure (QPEWMTS)
	DBSCAN	Query text feature encoding	Bag of words (BoW)
			Word embedding encoding
			Text mining approach
		Query plan feature encoding	Cardinality cost aggregation for each operator (CCAEO)
			Query plan encoding while maintaining tree structure (QPEWMTS)

Phase 1: Learning Query Templates through CCAEO and K-Means

Query

```
SELECT
  Q9.NAME AS "COMPANY_NAME",
FROM
  AHNAF.TITLE AS Q1,
  AHNAF.MOVIE_LINK AS Q2,
  AHNAF.MOVIE_KEYWORD AS Q3,
WHERE
  (Q1.MOVIE_ID = Q2.MOVIE_ID) AND
  (Q3.MOVIE_ID = Q2.MOVIE_ID) AND
  (Q1.MOVIE_ID = Q3.MOVIE_ID) AND
```

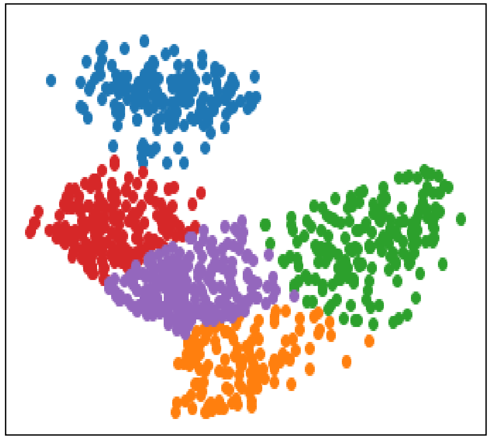
Query Plan



Vector

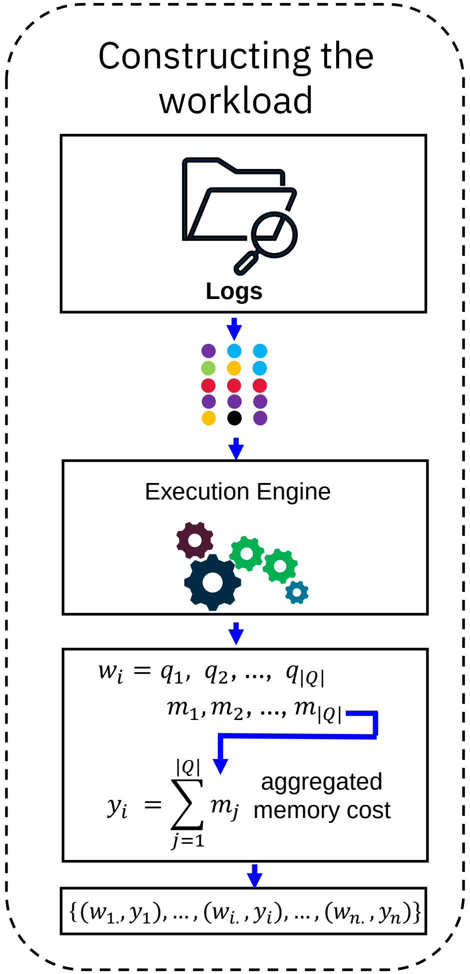
Operator	Count	Total Cardinality
TBSCAN	4	139532.48
HSJOIN	3	50224.6
SPLIT	1	3201
SORT	1	134170
CTQ	1	48873.6

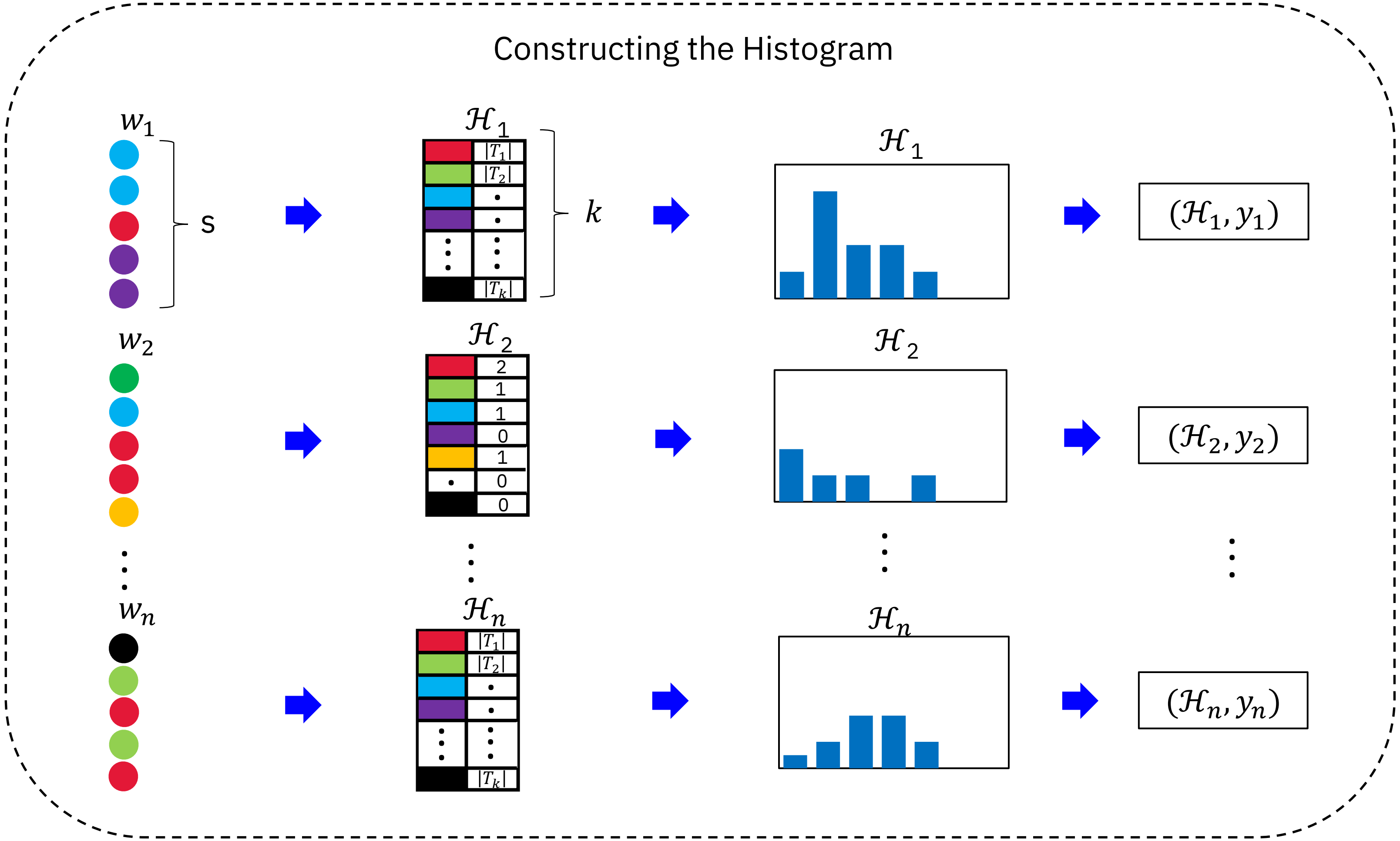
K-Means Clustering Algorithm



Motivation | Problem Definition | **Methodology** | Experimental Evaluation | Contribution & Future Work

Phase 2: Constructing Histograms from Workloads





Motivation | Problem Definition | **Methodology** | Experimental Evaluation | Contribution & Future Work

Phase 3: Training the Model

Ridge

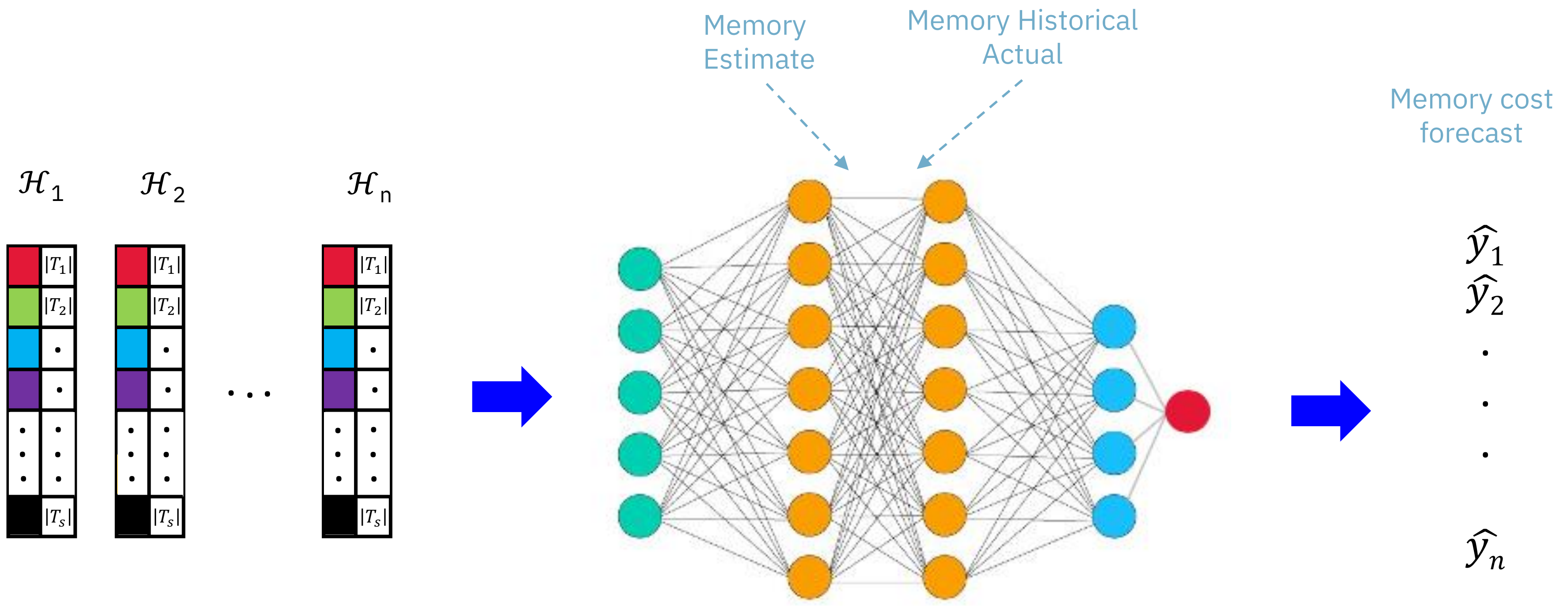
Decision Tree

Random Forest

XGBoost

Deep Neural Network

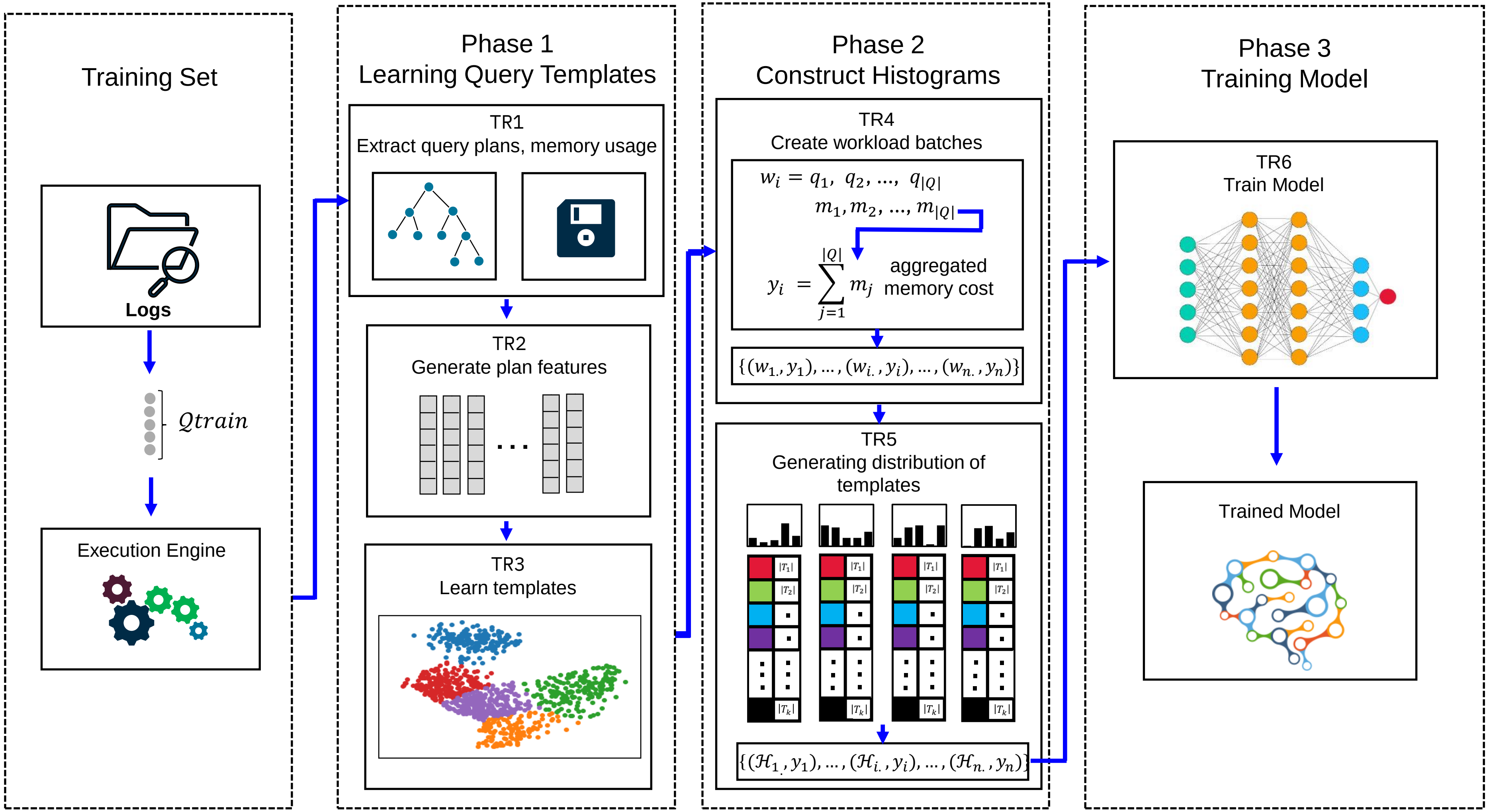
Phase 3: Training a Distribution Regression Deep Learning Model



Motivation | Problem Definition | Methodology | Experimental Evaluation | Contribution & Future Work

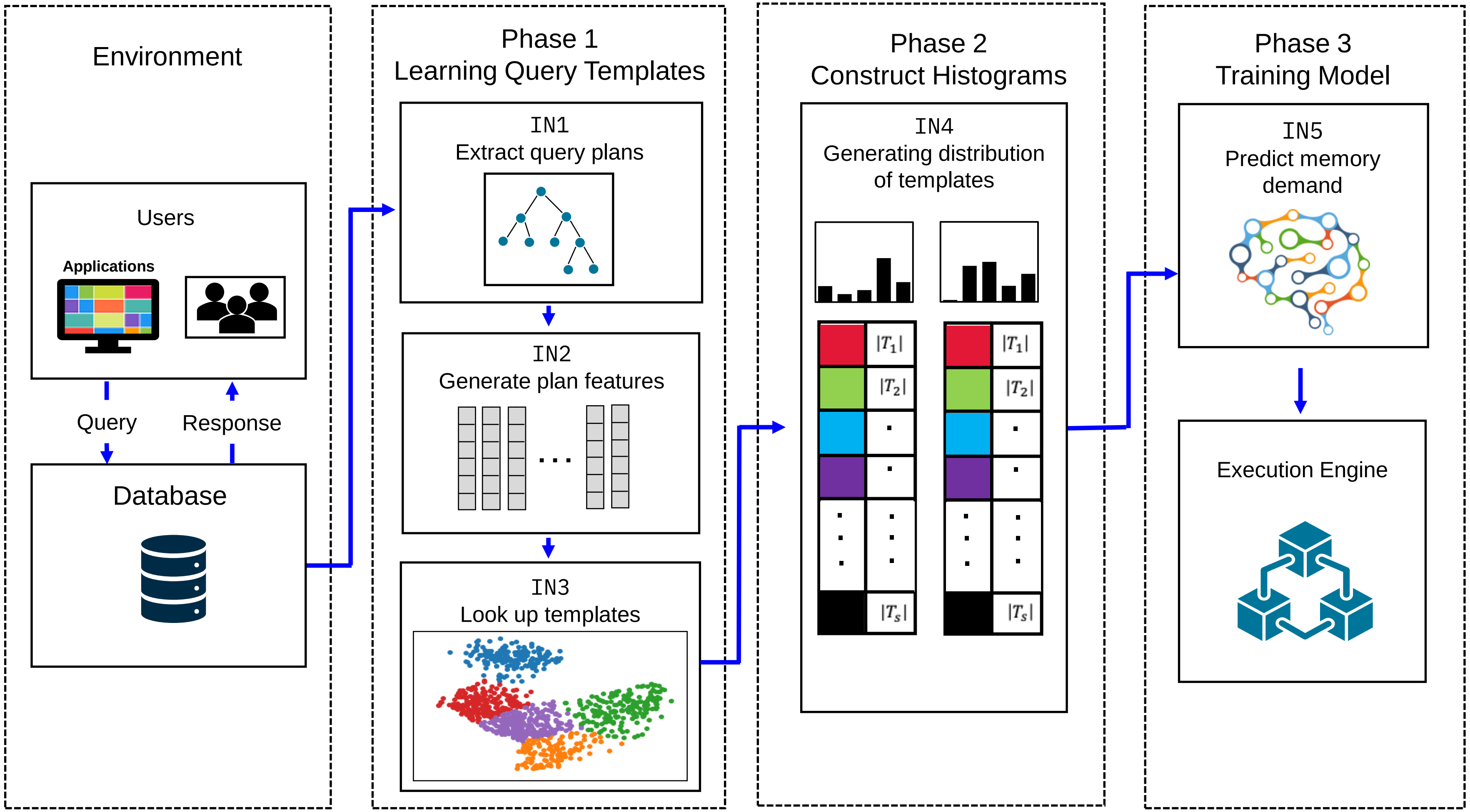
LASSONDE SCHOOL OF ENGINEERING YORK U

Training Pipeline Overview

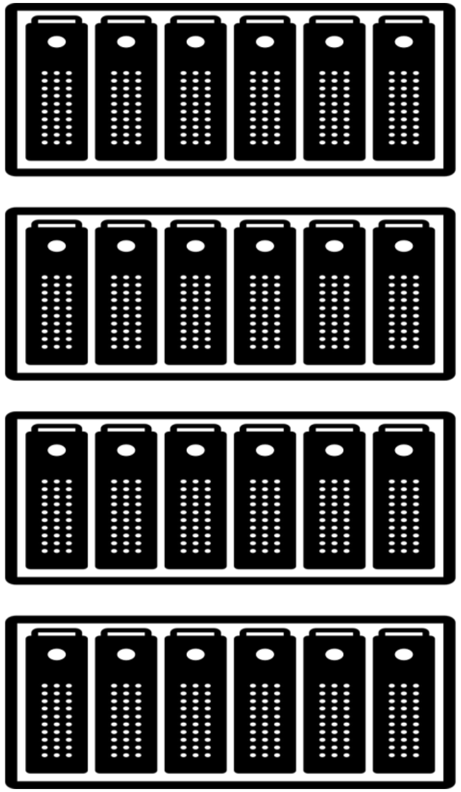


Motivation | Problem Definition | Methodology | Experimental Evaluation | Contribution & Future Work

Inferencing Pipeline Overview



Experimental Evaluation



Enviroment

- DB2 instance
- Linux system
- 8 CPU cores
- 32 GB of memory
- 500 GB of disk space

Datasets

TPCDS

- 93,000 queries
- OLAP – Transactional Workload

TPCC

- 3958 queries
- OLTP – Analytical Workload

JOB

- 2300 queries
- Join benchmark
- OLAP Transactional Workload

Experimental Evaluation Goals

Q1) Templates

- Learning query **templates performance**

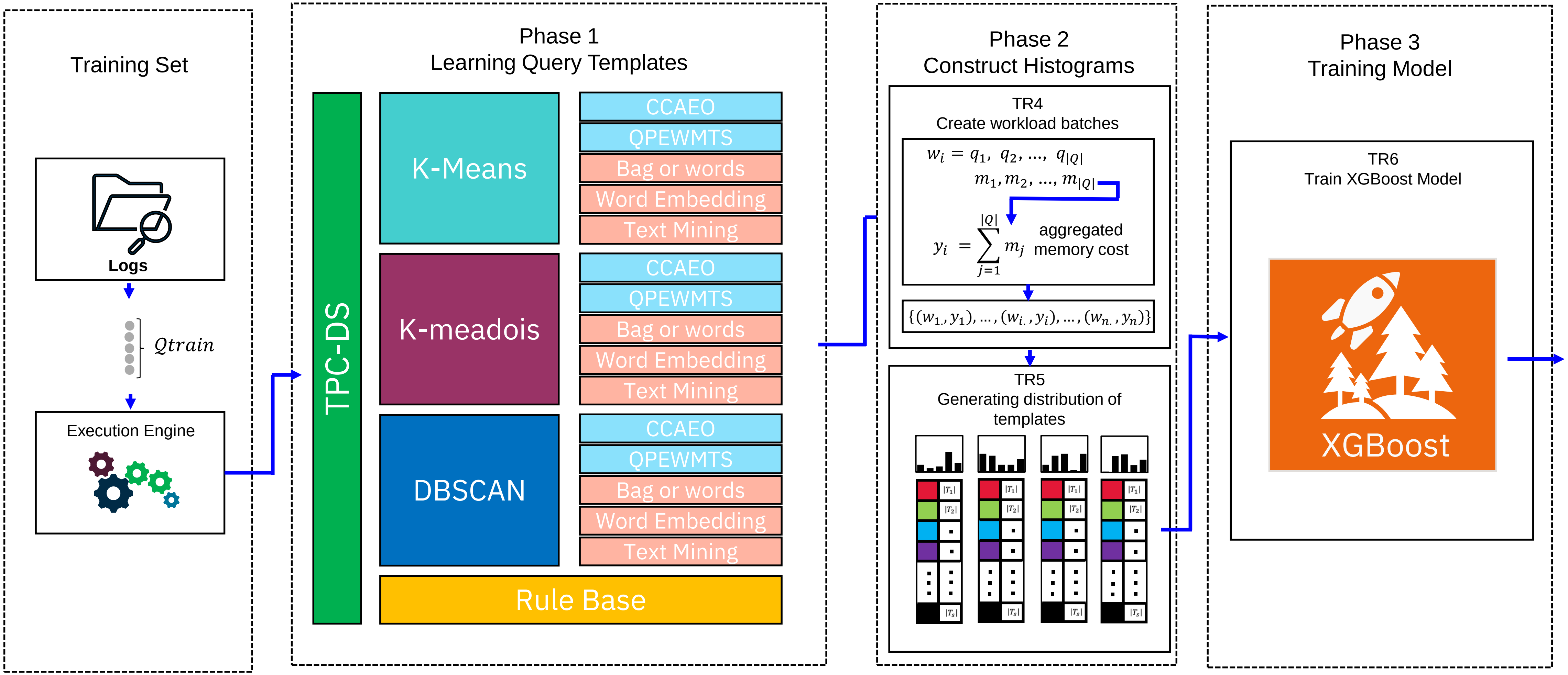
Q2) LearnedWMP

- LearnedWMP **accuracy performance**
- LearnedWMP **training** and **inference runtime cost**
- LearnedWMP **model size**

Q3) Parameter

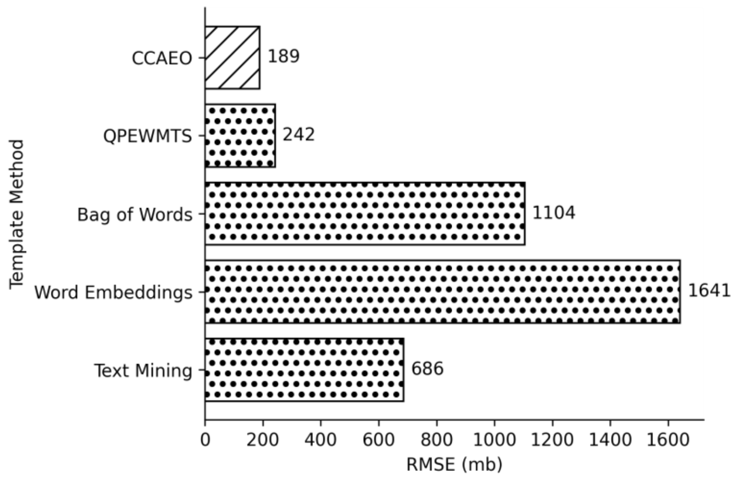
- Effect of the batch size parameter **s**

Q1 - Learning Query Templates Experimental Setup

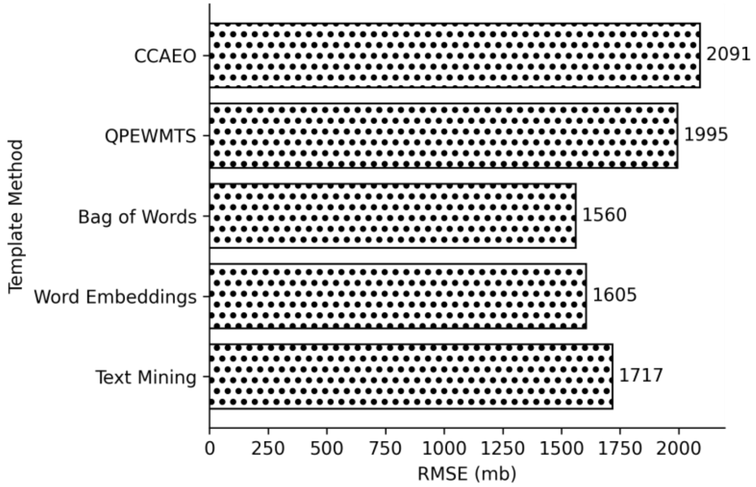


Q1 - Learning Query Templates Performance on TPC-DS Dataset

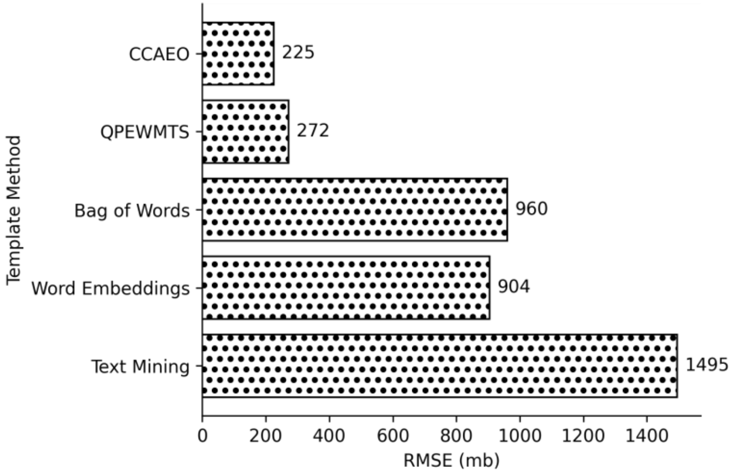
K-means



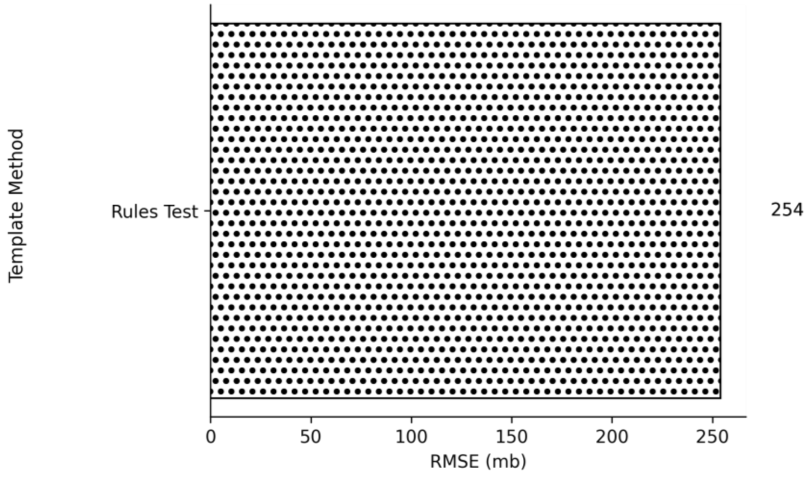
DBSCAN



K-medoids

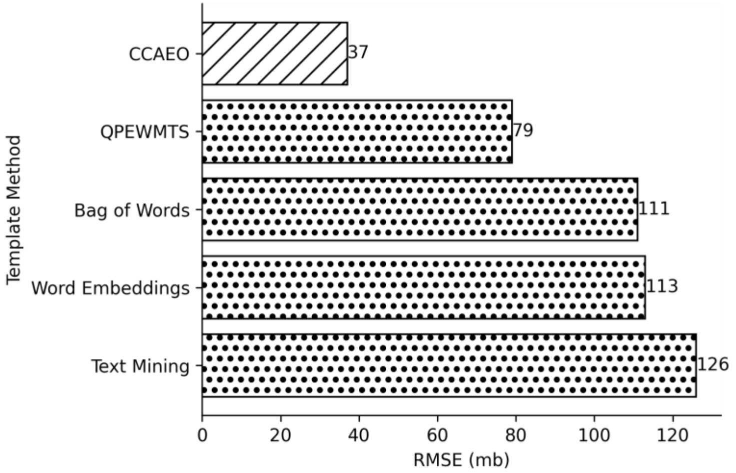


Rule Base

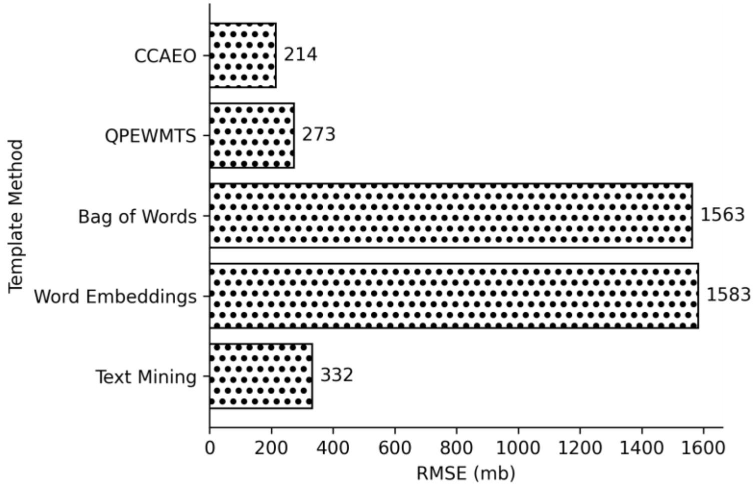


Q1 - Learning Query Templates Performance on JOB Dataset

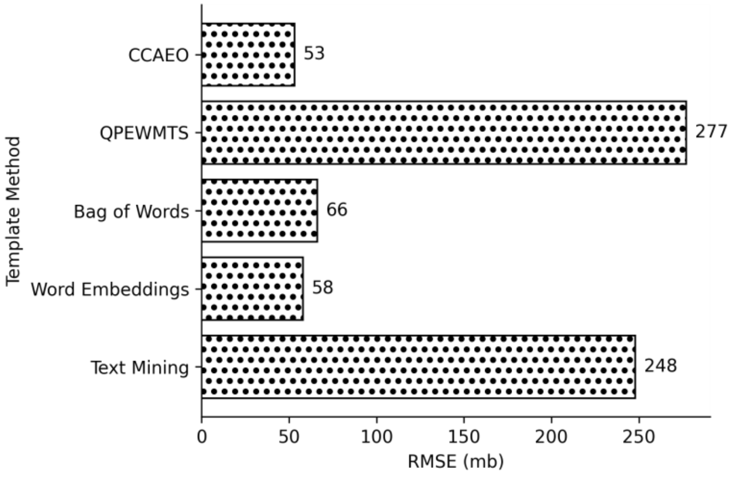
K-means



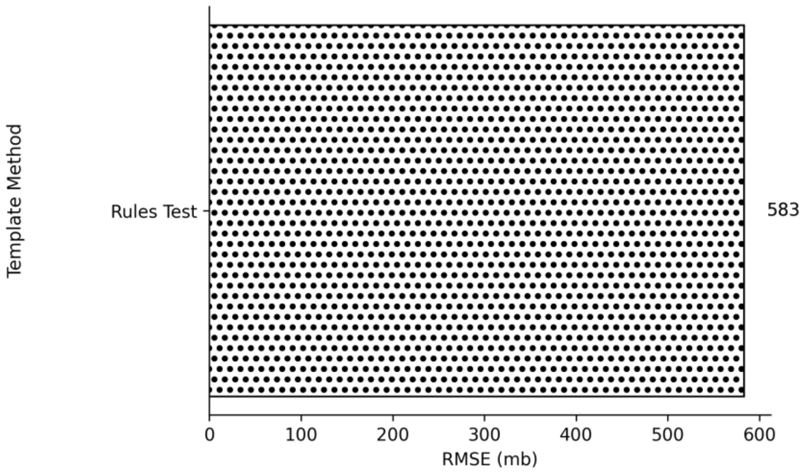
DBSCAN



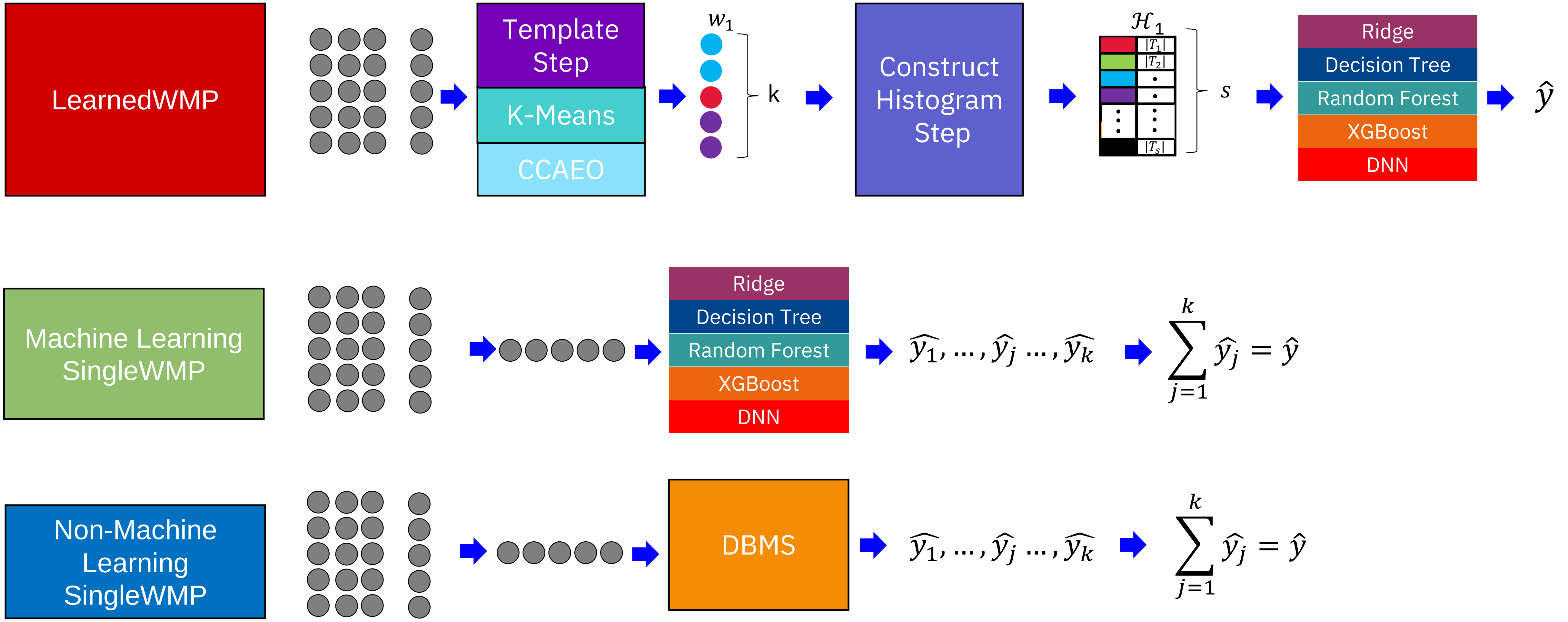
K-medoids



Rule Base



Q2 - LearnedWMP Experimental Setup



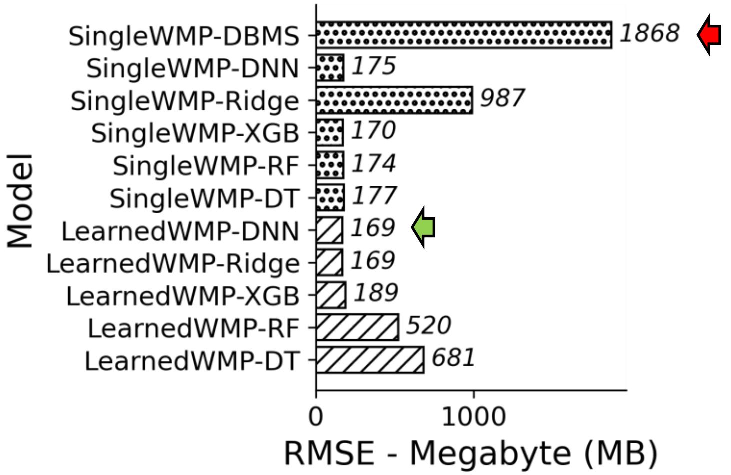
Q2 - LearnedWMP Experimental Setup

- LearnedWMP
  - LearnedWMP-DNN
  - LearnedWMP-Ridge
  - LearnedWMP-DT
  - LearnedWMP-RF
  - LearnedWMP-XGB

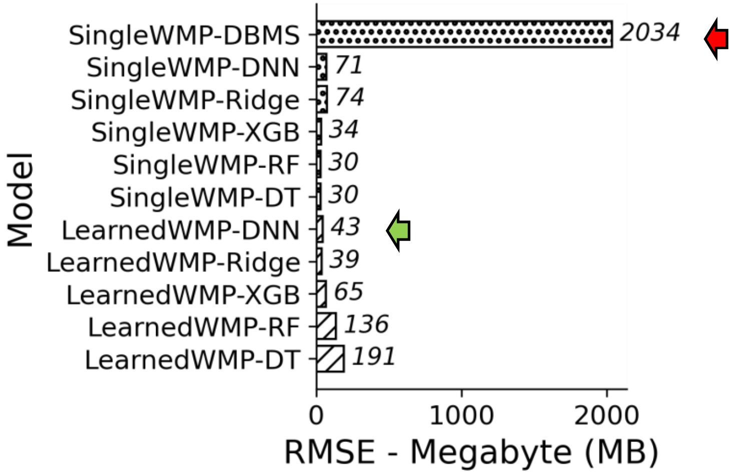
- SingleWMP
  - SingleWMP-DNN
  - SingleWMP-Ridge
  - SingleWMP-DT
  - SingleWMP-RF
  - SingleWMP-XGB
  - SingleWMP-DBMS

Q2 - LearnedWMP Accuracy Performance

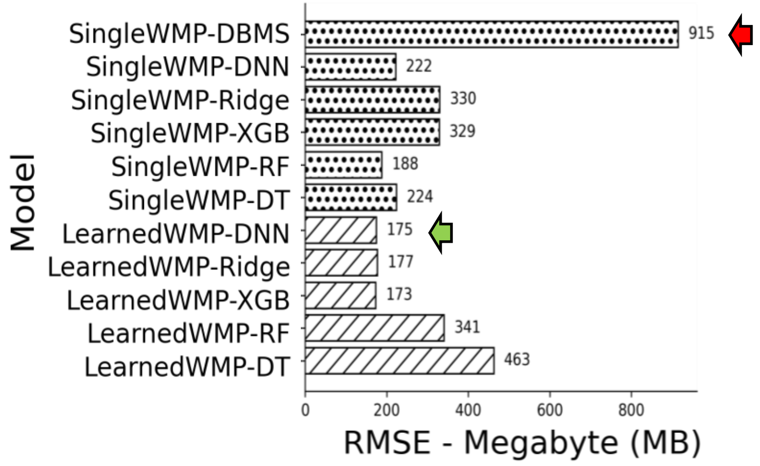
TPC-DS



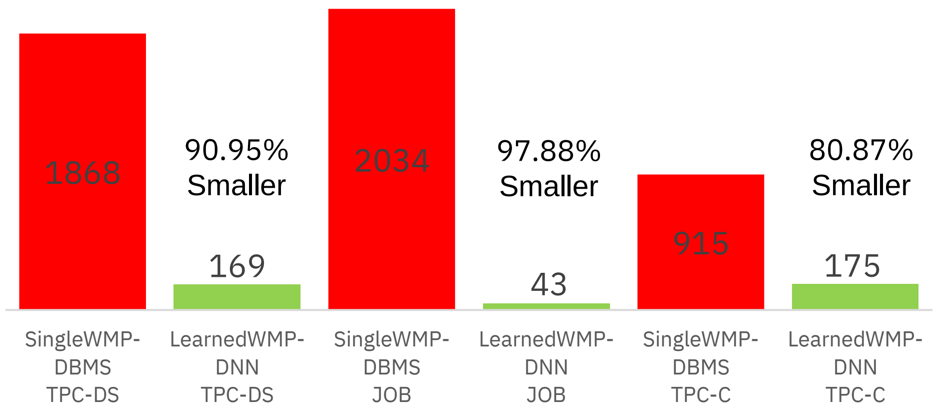
JOB



TPC-C

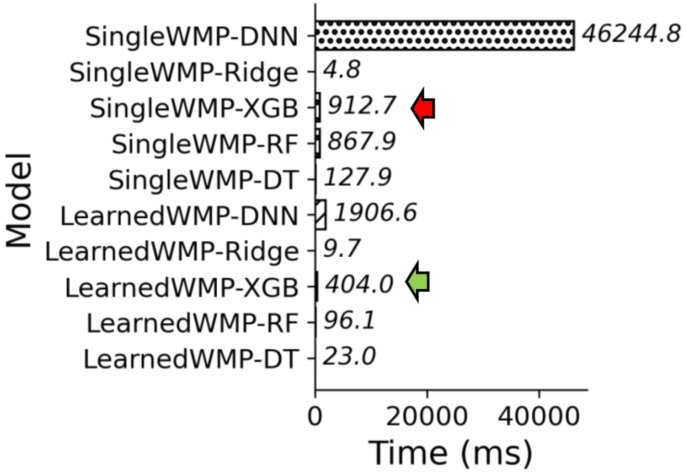


Observation

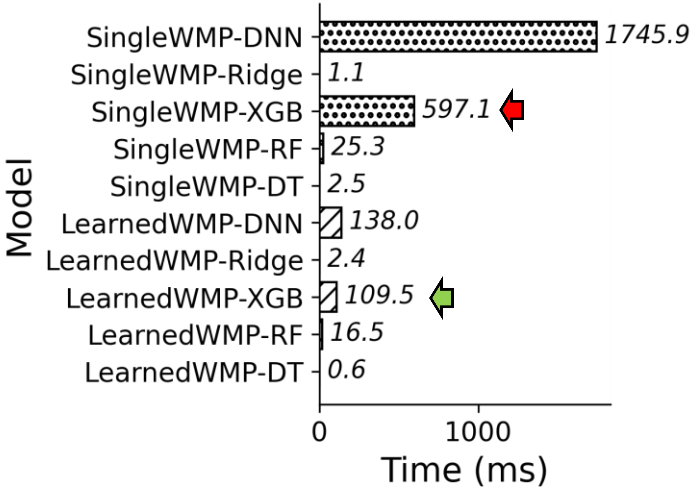


Q2 - LearnedWMP Training Runtime Performance

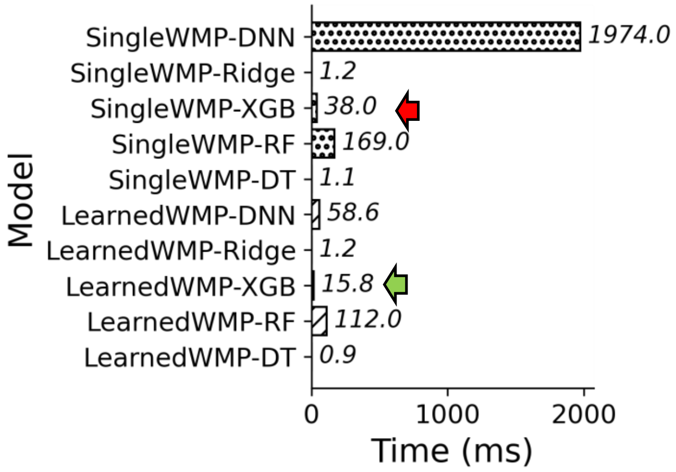
TPC-DS



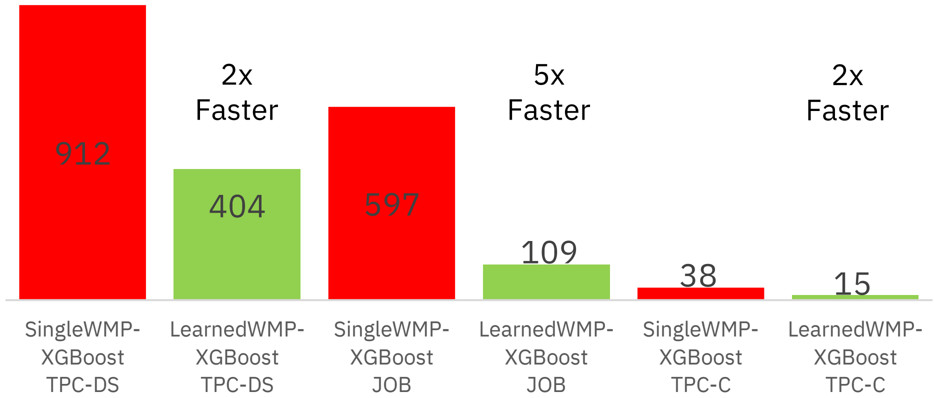
JOB



TPC-C

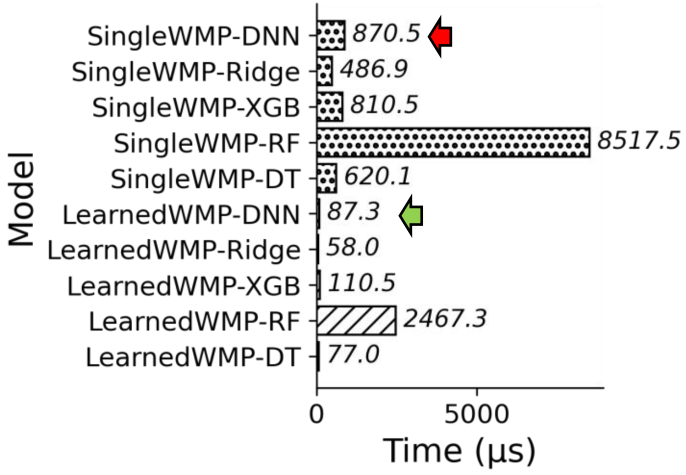


Observation

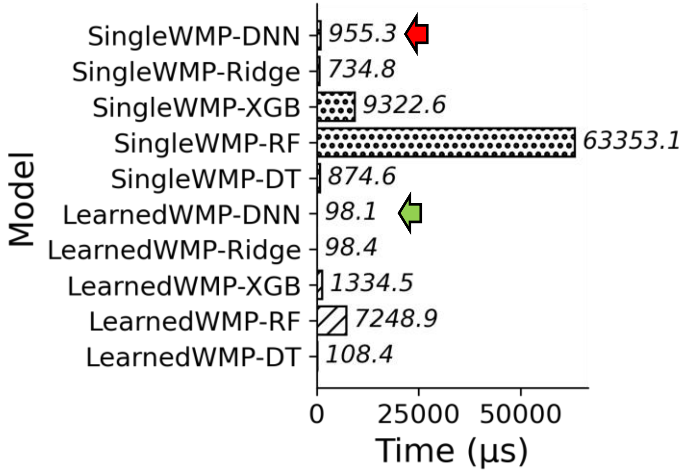


Q2 - LearnedWMP Inference Runtime Performance

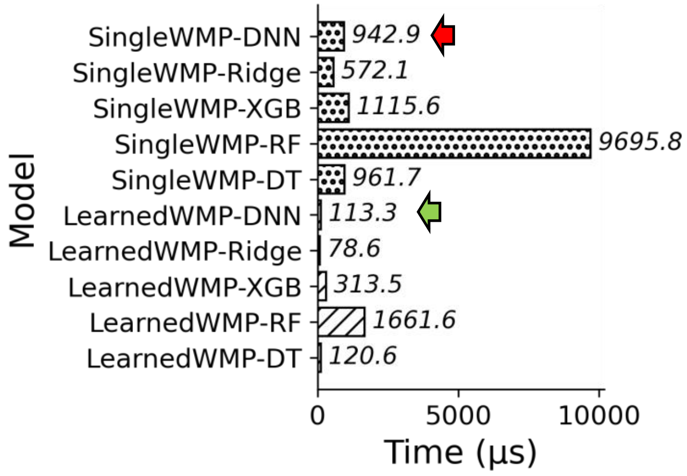
TPC-DS



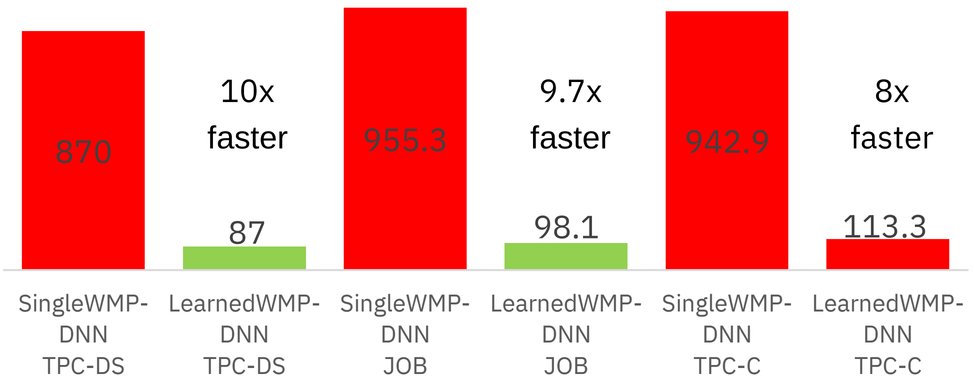
JOB



TPC-C

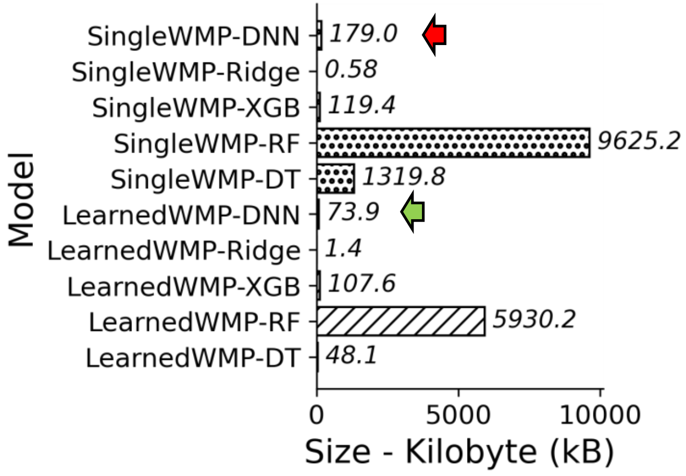


Observation

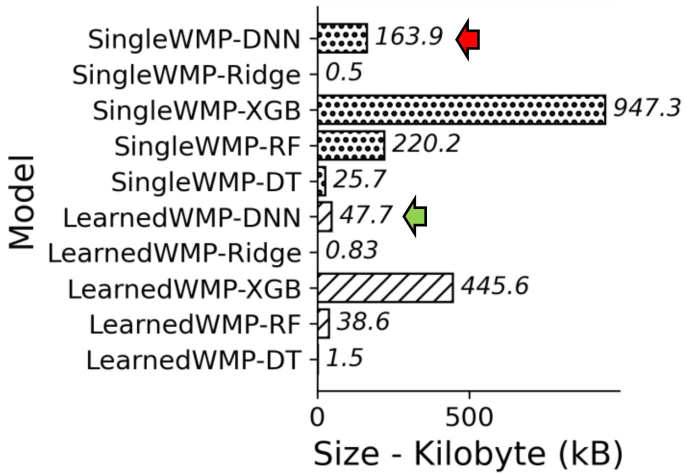


Q2 - LearnedWMP Model Size

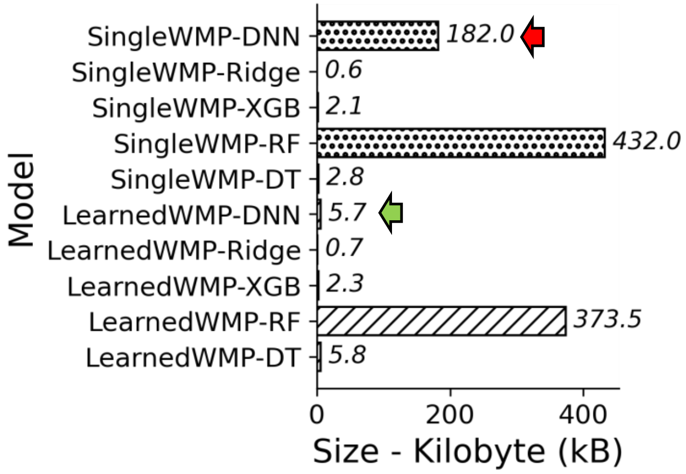
TPC-DS



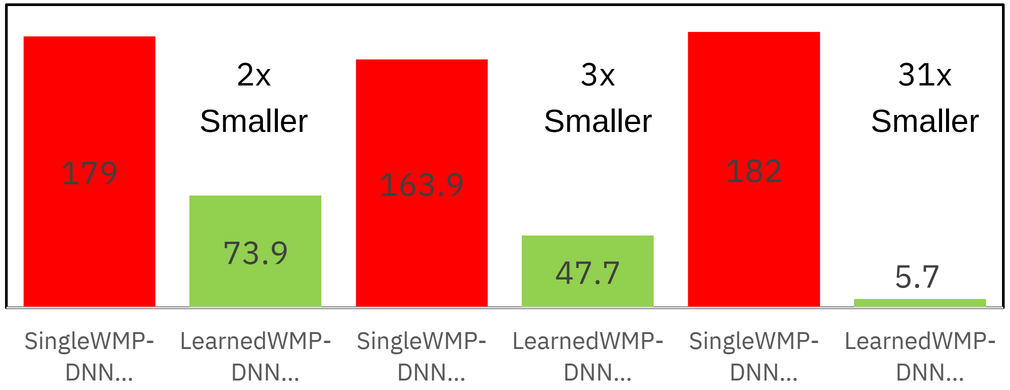
JOB



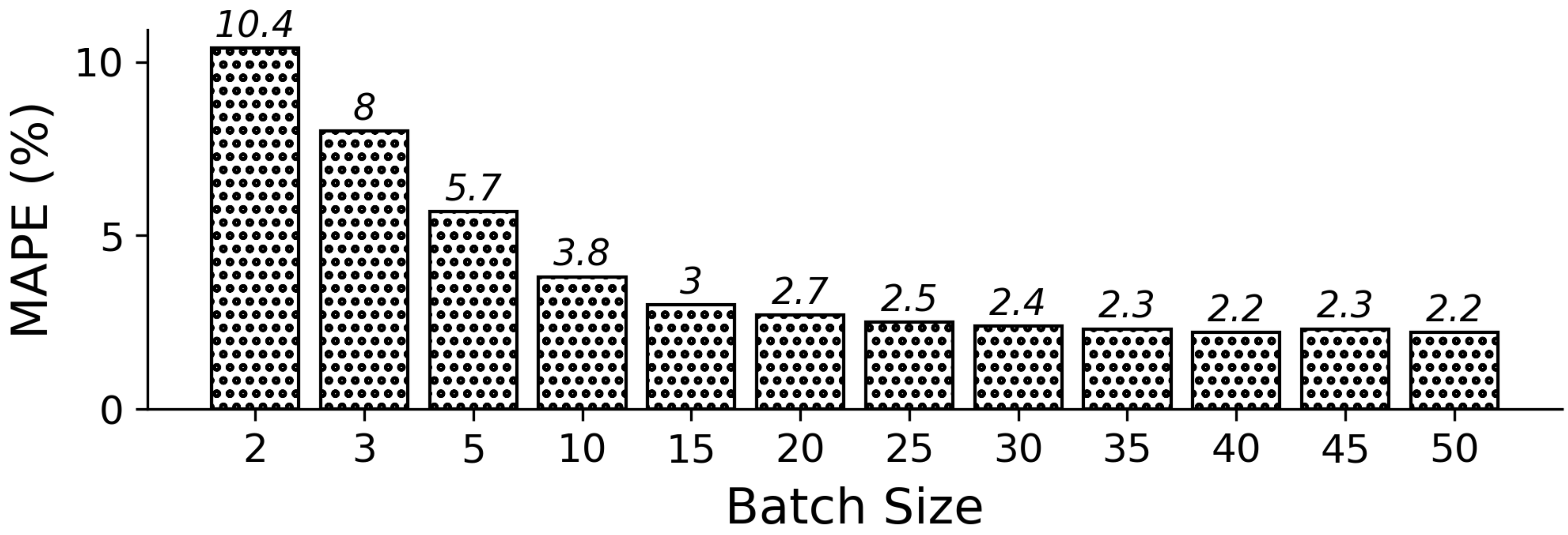
TPC-C



Observation



Q3 - LearnedWMP Batch Size Parameter s



Motivation | Problem Definition | Methodology | Experimental Evaluation | Contribution & Future Work

Contribution and Future Work

Contributions

- Introduce a novel problem of **workload memory prediction**
- Workload memory prediction as a **distribution regression problem**
- Propose **LearnedWMP**, a novel prediction **model** that can estimate the **memory** demand of a **batch of SQL queries**
  - Better Performance
  - Faster training and inference time
  - Smaller model size
- Our model reduced the memory estimation errors of **DBMS** by at least **47.16%**

Future Work - Datasets

- Current Datasets
  - TPC-DS
  - TPC-C
  - JOB

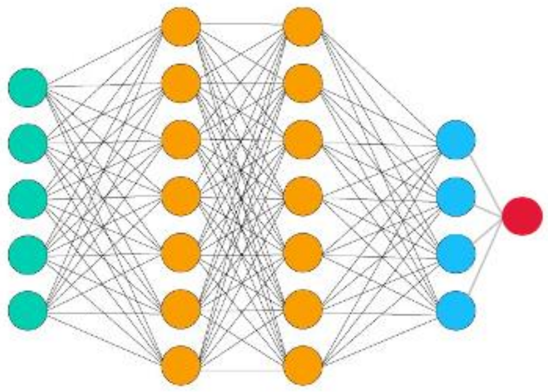
- Future Datasets
  - Production DB and dataset

Future Work – Resource Prediction

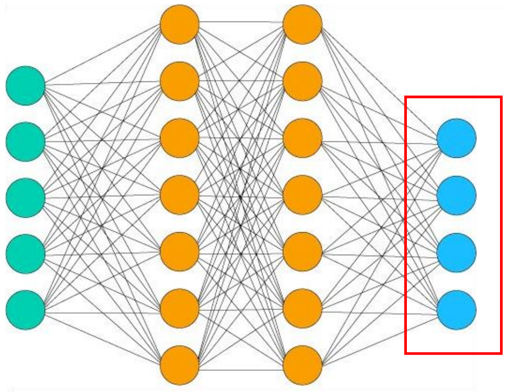
- Current Predicted Resource
  - Memory
- Future Resources to predict
  - Disk I/O
  - Throughput
  - CPU

Future Work – Multiple Resource Prediction

Single Resource Prediction

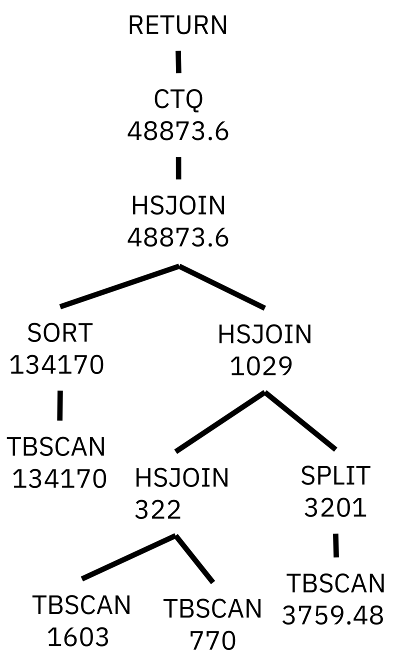


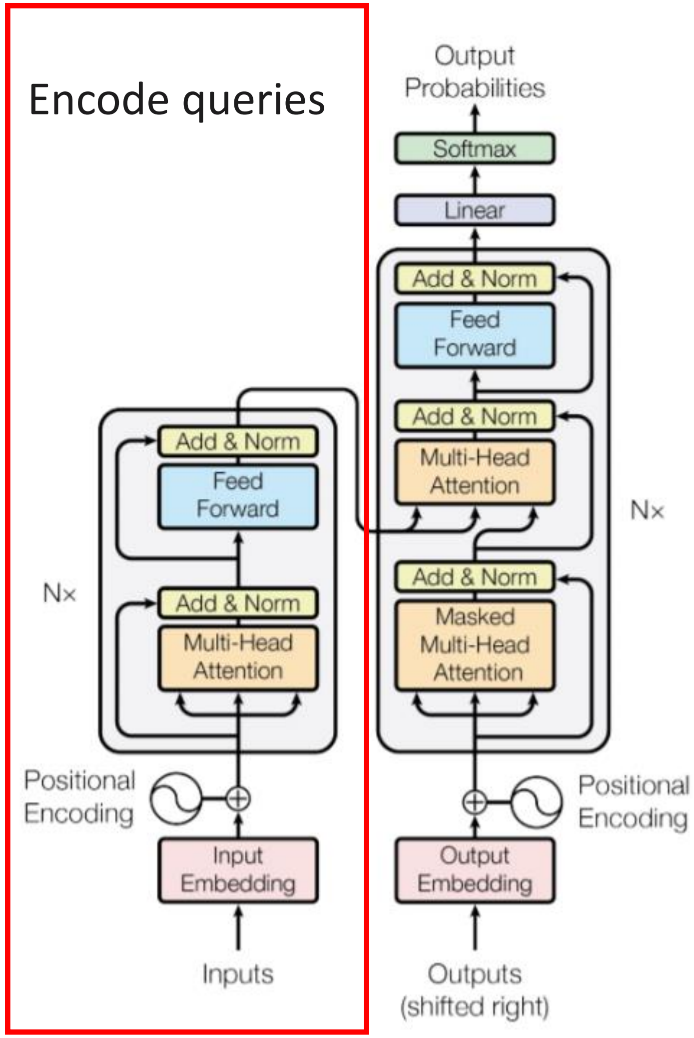
Multiple Resource Prediction



Future Work – Learn Templates

Keep the structure
of the query plan





LASSONDE SCHOOL OF ENGINEERING YORK U

Thank you

Nicolas Andres Jaramillo Duran

Data Driven Industries

Problem Definition

- Definitions:

Query $q = (e, p, m)$	e is the query expression, p is the query plan, and m is the memory usage.
$w = (\mathcal{Q}, y)$	$\mathcal{Q}$ is a set of queries where $q_i \in \mathcal{Q}$ is a tuple (e_i, p_i, m_i), and y is the actual memory utilization of all queries in $\mathcal{Q}$ such that $y = \sum_{i=1}^{\mathcal{Q}} m_i$
n	number of workloads in the training corpus of the form $\{(w_1, y_1), \ldots, (w_n, y_n)\}$
$\mathcal{T} = \{t_1, \ldots, t_k\}$	query template $t_i \in \mathcal{T}$ represents a class of queries with similar memory requirements
$\mathcal{H} = [c_1, \ldots, c_k]$	let $w = (\mathcal{Q}, y)$ be a workload, c_i is the number of queries in $\mathcal{Q}$ that can be mapped to query template $t_i \in \mathcal{T}$. the counts of queries in $\mathcal{Q}$ that map to different query templates in $\mathcal{T}$ are recorded in a $1 - d$ vector of length $k = \lvert T \rvert$ such that $\lvert \mathcal{Q} \rvert = \sum_{i=1}^{k} c_i$

- Assumptions:
  - The distribution of queries among the query templates (i.e., the workload histogram bins) is uniform.
  - The query templates are independently and identically distributed
  - An underlying function, $f(\cdot)$, exists that can accurately compute any workload's memory usage, y, from the workload histogram, $\mathcal{H}$.

$$f(\mathcal{H}) = y$$

We don't know $f(\cdot)$ nor have access to the set of all possible workload examples to derive $f(\cdot)$.

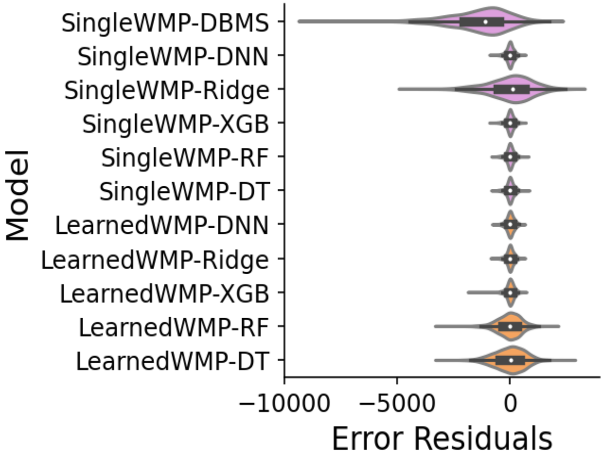
Problem Definition

- Problem:
  - We formulate estimating memory usage of an unseen workload as a distribution regression problem, where the estimate is computed from an input probability distribution - the distribution of queries $\mathcal{Q}$ among templates $\mathcal{T}$.
  - For w_i, $\mathcal{H}_i$ is the workload histogram and y_i is the collective historical memory utilization of all queries in the workload.
  - Let us assume we have a training corpus $\{(\mathcal{H}_1, y_1), \ldots, (\mathcal{H}_n, y_n)\}$ of n workload histograms, one for each workload.
  - Using distribution regression, we wish to learn a function, $\hat{f}\,(\cdot)$, an approximation of $f\,(\cdot)$.
  - $\hat{f}\,(\cdot)$, can compute $\hat{y}$, an accurate estimate of the actual memory usage y

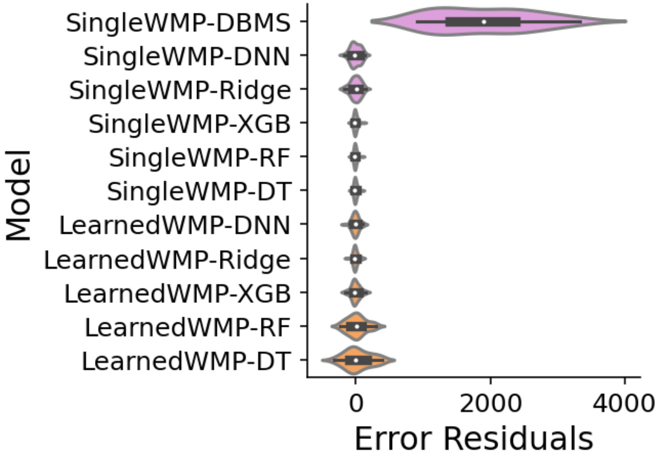
$$f\,(\widehat{\mathcal{H}}) = \hat{y}$$

LearnedWMP Accuracy Performance

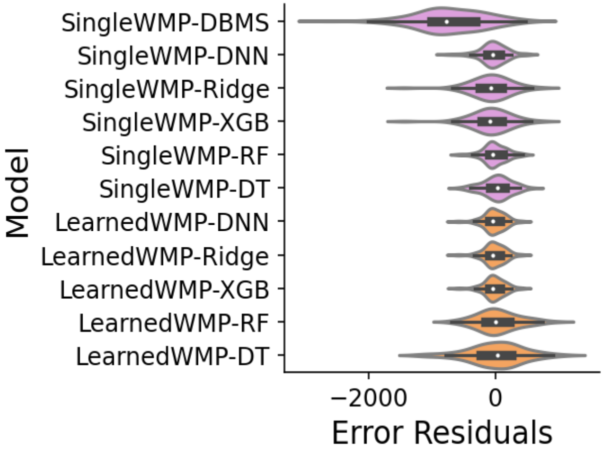
TPC-DS



JOB



TPC-C

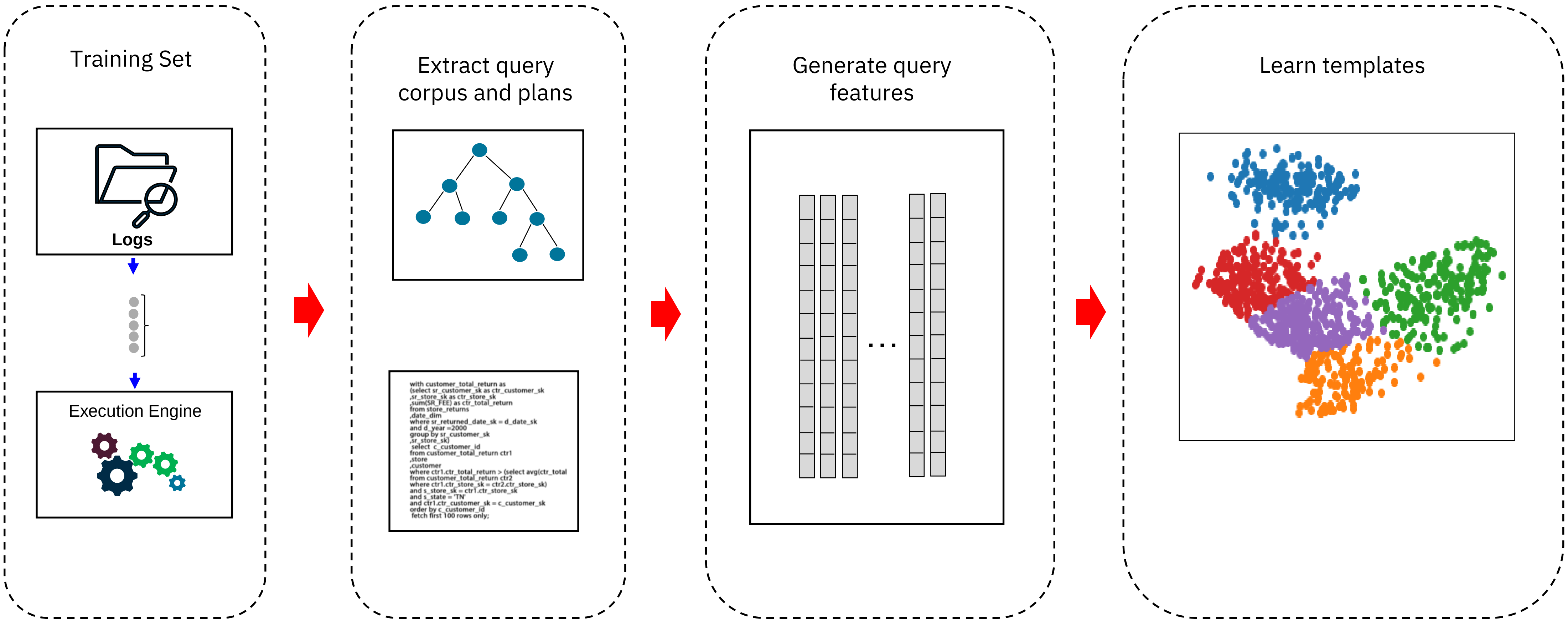


Phase 1: Learning Query Templates

- Learning rule-based query templates
  - Map the generated queries back to their corresponding templates.
  - Group queries based on their estimated cardinality similarity.

- Learning clustering-based templates
  - Query text feature encoding
    - Bag of words (BoW)
    - Word embedding encoding
    - Text mining approach
  - Query plan base feature encoding
    - Cardinality cost aggregation for each operator (CCAEO)
    - Query plan encoding while maintaining tree structure (QPEWMTS)

Phase 1: Learning Query Templates



Motivation | Problem Definition | **Methodology** | Experimental Evaluation | Contribution & Future Work