RCHUB: JOURNAL OF COMPUTATIONAL INTELLIGENCE SCIENCE AND ENGINEERING (JCISE)
VOL: 2, ISSUE 4, OCTOBER - DECEMBER 2026
RCHUB PUBLISHER
ISSN: 3107-8168
DOI: https://doi.org/10.65725/JCISE/2/4/008

EXPLAINABLE CNN-BASED CLASSIFICATION OF HETEROGENEOUS DATABASE WORKLOADS USING RUNTIME PERFORMANCE FEATURES

K. Jaya Bharathi,
Research Scholar, M.Tech., Ph.D.,
Rayalaseema University, Kurnool – 518007, Andhra Pradesh
Dr. S. Krishna Mohan Rao,(Retd.)
Professor and Dean, Department of CSE,
Siddhartha Institute of Engineering and Technology,
Ibrahimpatnam, Rangareddy District, Telangana.

Abstract

The current database management systems handle heterogeneous workloads, such as transactional, analytical, reporting, and mixed workloads that have varied patterns of execution and resource demands. Proper classification of workloads can be used to monitor performance of the database and optimize it based on workload. In the paper, a explainable Convolutional Neural Network (CNN)-based model is proposed to classify heterogeneous database workloads based on runtime performance characteristics, such as query execution time, CPU use, memory use, disk input/output activity, query complexity and parallelist. The features gathered are pre-processed and they are fed into a CNN model to determine four workload categories. SHapley Additive exPlanations (SHAP) is used to enhance interpretability, by determining the features that contributed to individual predictions and overall classification behavior. This framework is assessed in the form of accuracy, precision, recall, F1-score, classification latency, and computational overhead. It is compared to conventional classifiers, such as Decision Tree, Random Forest, Support Vector Machine, and Multilayer Perceptron. TPC-H and TPC-DS benchmarks are assumed to be used with analytical workloads, which are complemented by other workloads that model the other categories. The proposed framework will help to offer the correct and understandable workload classification and establish the basis of future adaptive query processing and optimization of the database.

Keywords: Database workload classification; Convolutional Neural Network; Explainable AI; SHAP; TPC-H; TPC-DS.

1. INTRODUCTION

The contemporary database management systems (DBMSs) handle various workloads, which involve transactional processing, analytical queries, reporting processes, and mixed workloads. These workloads vary in the complexity of queries, time taken to execute, concurrency, and

ISSN: 3107-8168
DOI: https://doi.org/10.65725/JCISE/2/4/008

resource usage. A database system therefore needs to be aware of its workload properties to facilitate efficient monitoring of its performance and workload-sensitive optimization. The traditional database systems are based on rules of thumb, statistical approximation, and cost-based query optimization schemes. Despite the widespread adoption of these approaches, they may not be effective due to an incorrect estimation of cardinality, defective cost models, and workload characteristics variations [1].

The recent development of deep learning has offered a chance to enhance the prediction of database performance and query optimization. Marcus et al. introduced Neo, an educated query optimizer, to create query execution plans and learn based on the previous execution results using deep neural networks [2]. In the same way, Bao integrates tree convolutional neural networks with a bandit-based learning algorithm to direct an already existing query optimizer with optimization hints [3]. These experiments exhibit the possibility of learning-based techniques in the field of database optimization. The choice of query plan and workload typology are however different research problems. There is no guarantee that a system that learns to choose an execution plan will categorize the incoming workloads as transactional, analytical, reporting, and mixed workloads.

Properly classified workloads can be a good source of information to comprehend database behaviour and to choose appropriate optimization strategies. Database workloads can be characterized in terms of runtime performance characteristics, such as query execution time, CPU use, memory usage, and disk input/output (I/O), query complexity, concurrency, and characteristics of the execution-plan. Convolutional Neural Network (CNN) could be explored to learn patterns based on a representation of these features in an appropriate structure. Its usefulness however, should be experimentally determined in comparison with appropriate baseline classifiers.

Interpretability is another factor that should be taken into account. Deep learning models can give predictions without necessarily stating which input features affected their decisions. Models SHapley Additive exPlanations (SHAP) are explainable Artificial Intelligence (XAI) methods that can be used to determine the impact of different features on model predictions [4]. These types of explanations can assist database administrators to learn about the workload typologies and explore the performance behaviour.

Thus, this paper will suggest a explainable CNN-based architecture to classify heterogeneous database workloads based on runtime performance characteristics. The framework is centered

ISSN: 3107-8168
DOI: https://doi.org/10.65725/JCISE/2/4/008

around four types of workloads, which include transactional, analytical, reporting, and mixed workloads. Accuracy, precision, recall, F1-score, classification latency, and computational overhead can be used to evaluate its performance. The analytical workload scenarios can be delivered by TPC-H and TPC-DS, and other transactional, reporting and mixed workload traces should be created or monitored so as to have sufficient coverage of all four categories. The primary aim is to explore the possibility of accurate and interpretable workload identification through CNN-based classification along with feature-level explanations.

2. LITERATURE SURVEY

The authors provided an in-depth survey of database Cardinality estimation, cost modeling, and query plan enumeration are all components of query optimization. [1]. Their research described how errors and constraints in cost models can cause sub-optimal query execution plans. It also compared conventional and learning-based methods of enhancing optimization in the optimizer's performance. As much as this piece of work sets the value of proper database estimation and optimization, it is more concerned with query optimization as opposed to explainable classification of heterogeneous workloads based on runtime features.

Researchers came up with Neo, an educated query optimizer which engages deep reinforcement learning to produce query execution plans [2]. The model is informed via prior decisions made by an optimizer and the result of query execution. The paper has shown that optimization by learning can be competitive to other well known methods of optimization of databases. Neo however does not specifically deal with the classification of workloads into transactional, analytical, reporting, and mixed workloads, but merely deals with the selection of query execution plans.

Author later suggested Bao, a realistic learned query optimization model that uses tree convolutional neural networks in conjunction with Thompson sampling [3]. Bao is an optimizer assistant that gives optimization tips to an existing query optimizer and adjusts its decision-making according to measured performance. The fact that it is responsive to workload and changes in data is indicative of the potential of learning-based database methods. However, Bao is not designed to do a explicit multi-class workload classification, but to optimize queries. Its method also fails to directly respond to the need of SHAP-based explanations of workload-classification predictions.

Researchers proposed SHAP, a unified system of interpretation of machine learning predictions based on the creation of contribution values on input features [4]. SHAP may be used to justify

ISSN: 3107-8168
DOI: https://doi.org/10.65725/JCISE/2/4/008

how a classifier categorizes a workload to an observation. An example use is to determine whether query complexity, number of joins or available runtime statistics were a strong predictor. But SHAP is an explanatory method but not a workload classifier; its utility is based on the predictive model and the quality of the input data.

Balsa is a query optimizer proposed by the authors, which is trained using deep reinforcement learning without having to demonstrate expert optimizers [5]. Balsa is also learnt in a simulator, and then refined by being run in a real database environment. Its test on the Join Order Benchmark showed the reinforcement learning could be used to learn query optimization policies. However, Balsa concentrates on learning query execution strategies and does not concentrate on categorizing explicitly the heterogeneous workloads and elaborating on the characteristics of the features that lead to the category.

arXiv+1

Researchers introduced BitE, an ensemble-learning system that aims at speeding up optimization of learned queries in mixed-workload context [6]. The framework leverages database statistics and metadata to steer optimization, expands the search space with hintsets, and mitigates bias towards default execution plans. The authors cited that the number of queries improved and the number of query regressions reduced with the new methods as compared to the traditional methods tried in their research. BitE is especially pertinent as a foundational paper since it takes into account the issues of optimal mixed workloads. However, its primary goal is the learned query optimization rather than direct classification into transactional, analytical, reporting, and mixed categories with SHAP-based explanations.

arXiv

Commended the difficulties of query-plan enumeration, incorrect cardinality estimates, and error of cost-model [1]. The issues are especially significant when there is a variation in query complexity, data features, and execution environments. They survey indicates that workload changes and effective model updating remain to be challenges in the optimization of learning-based database. This encourages the exploration of the possibility of explicit workload classification as a source of valuable information that can be used to select a workload-conscious plan in the future.

In general, the analyzed articles prove the relevance of the optimization of databases, deep learning, and workload adaptation, as well as the interpretability of models. Nonetheless, the studies are concerned with various sides of the issue: Neo, Bao, and Balsa discuss the learned

ISSN: 3107-8168
DOI: https://doi.org/10.65725/JCISE/2/4/008

query optimization, BitE is devoted to the optimization under the mixed load, and SHAP assists the interpretation of the predictions. The proposed paper explores the combination of CNN-based SQL workload classification, runtime performance characteristics, and SHAP-based explanations in one classification. It will compare the CNN with Decision Tree, Random Forest, Support Vector Machine and Multilayer Perceptron models to understand whether CNN has any quantifiable advantages. The wider context will then be based on the classification output to facilitate the workload-sensitive plan choice and dynamics at runtime by the suggested algorithms EHWC, WAAPS and IEADO. The effectiveness of these contributions will have to be determined by experimental results.

3. PROPOSED METHODOLOGY

The suggested model adopts a Convolutional neural network (CNN) to categorize heterogeneous database workloads in terms of runtime performance characteristics and SHapley Additive explanations (SHAP) to explain the ensuing predictions. The data collection methodology involves workload data collection, preprocessing, CNN training, workload classification, model explanation and performance evaluation.

3.1 Workload Data Collection and Preprocessing

The workloads are performed by means of PostgreSQL and TPC-H and TPC-DS are utilized in analytical cases. Further, transactional, reporting and mixed workloads are created or accumulated to reflect all the four target classes. Runtime characteristics are the execution time of queries, CPU usage, memory usage, disk I/O usage, query complexity, the number of joins, concurrency, and the execution plan.

Data collected are cleaned, normalized and coded where needed. It consists of a training, validation, and testing subset and is split in a representative way (e.g. 70, 15, 15). Splitting into groups is used where required to avoid the representation of observations with the same query template or execution session on training and testing sets.

3.2 CNN-Based Workload Classification

The preprocessed features are converted to a unified input representation to the CNN. Convolutional layers are used to learn patterns on the input features and fully connected layers are used to map the learned representations to four categories of workloads: Transactional, Analytical, Reporting and Mixed. The likelihood of each category is approximated by a softmax output layer. Validation data is used to tune model parameters.

3.3 Explainability and Evaluation

ISSN: 3107-8168
DOI: https://doi.org/10.65725/JCISE/2/4/008

SHAP is used on the trained classifier to show the global feature importance and to explain a single prediction. A confusion matrix, precision, accuracy, F1-score, and classification performance are used to evaluate the performance of classification. Also considered are prediction latency, computational overhead and quality of explanations. CNN is compared with Decision tree, random forest, support vector machine, and multilayer perceptron classifiers with uniform data partitions and evaluation environment.

4. PROPOSED ALGORITHM

Algorithm 1: Heterogeneous Database Workload Classification through explainable CNNs.

Input:

- ((D): Named workload dataset on the database.
- (P): Set of runtime performance features.
- (C): Workload - classes - Transactional, Analytical, Reporting, Mixed.
- (E): Number of training epochs - maximum.

Output:

- Workload predicted class (hat y).
- P = Probability vector (P) of a particular class (i).
- SHAP-based explanations (S) of features.
- Measurements of classification (M)

Steps:

1. Gather workload execution logs of PostgreSQL and benchmark or generated workloads.
2. Retrieve runtime characteristics (F), such as execution time, CPU, memory, I/O, query complexity, joins, and concurrency.
3. Label each of the class set (C) observations with a workload label.
4. Clean the data set through the processing of missing values, duplicates and invalid records.
5. Standardize numerical variables and code nominal variables and values.
6. Split the data into training, validation, and testing data without leakage of data.
7. Produce a uniform systematized depiction of the input features in CNN process.
8. Pre-train the CNN by using convolutional layers, activation functions, optional Softmax output layer, fully connected layer, and pooling layer.

ISSN: 3107-8168
DOI: https://doi.org/10.65725/JCISE/2/4/008

9. Train CNN on the training subset with a multi-class classification loss.
10. Test the model on the validation set and optimize its hyperparameters.
11. Choose the model setting that has the highest validation performance.
12. Use the chosen model on the testing subset to get predicted class probabilities (P).
13. Label observations with the most probable class:

$$\hat{y} = \arg\max_{c \in C} P(c \mid F)$$

14. Use SHAP to interpret the model predictions and get feature contributions (S).
15. Accuracy, precision, recall, F1-score, confusion matrix, prediction latency, and computational overhead computed.
16. Train and test baseline classifiers with the same data partitions.
17. Compare CNN and the baseline classifiers and report the results obtained.
18. Return

    $\hat{y}$, (P), (S), and (M).

Table-1 Runtime feature evaluation

Runtime feature	**Sample value**
Query execution time	4.8 seconds
CPU utilization	82%
Memory consumption	1.6 GB
Disk read activity	740 MB
Number of joins	6
Concurrent queries	12
Query complexity	High

Table-2: Workload classification output

ISSN: 3107-8168
DOI: https://doi.org/10.65725/JCISE/2/4/008

Query ID	Execution time (s)	CPU (%)	Memory (GB)	I/O (MB)	Predicted class
Q001	0.12	28	0.4	12	Transactional
Q002	4.8	82	1.6	740	Analytical
Q003	1.9	46	0.8	95	Reporting
Q004	3.2	76	1.4	420	Mixed
Q005	0.18	32	0.5	18	Transactional
Q006	5.1	85	1.8	810	Analytical

Table-3: Sample Classification Performance

Classifier	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)
Decision Tree	86	85	84	84.5
Random Forest	91	90	90	90
SVM	88	87	86	86.5
MLP	90	89	89	89
Proposed CNN	94	93	93	93

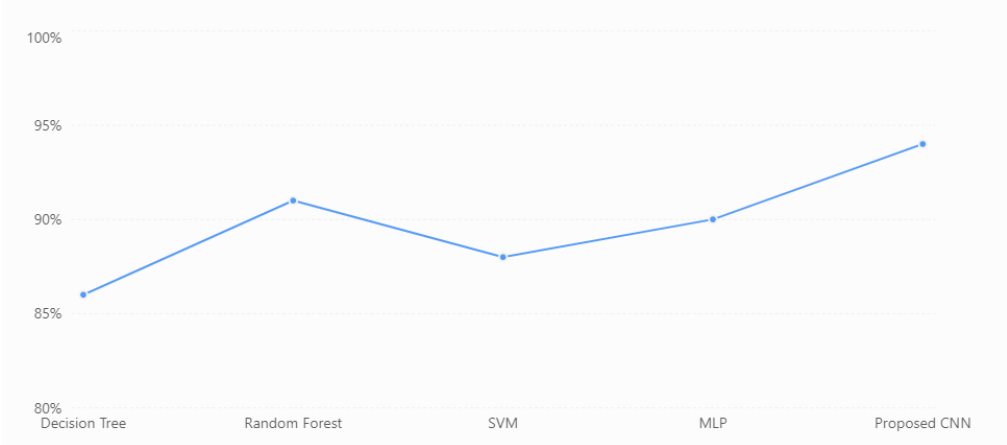


Figure-1 Accuracy Comparison

ISSN: 3107-8168
DOI: https://doi.org/10.65725/JCISE/2/4/008

The previous workflow, explained below, is proposed as the Algorithm 1, Explainable CNN-Based Heterogeneous Database Workload Classification, which groups database workloads into four categories: Transactional, Analytical, Reporting, and Mixed. First, PostgreSQL workload data are gathered with runtime performance metrics like query execution time, CPU usage, memory usage, disk I/O activity, query complexity, number of joins, and concurrency. The raw data are pre-processed to handle missing values, normalization, and label encoding and partitioned into training, validation, and testing datasets. A Convolutional Neural Network (CNN) is subsequently trained to acquire the patterns based on the structured representation of features and make predictions regarding the type of workload that is to be applied. The predicted class and its probability are contained in the output, and SHAP is utilized to reveal how individual features contributed to the classification decision. The illustrative output categorizes query Q025 as Analytical with an approximate probability of 91 and query execution time, disk I/O activity and join complexity as some of the features that affect it. Accuracy, precision, recall, F1-score, a confusion-based performance, prediction latency, and computational overhead are the measures of evaluation of the framework and the performance is compared with Decision Tree, Random Forest, Support Vector Machine and Multilayer Perceptron classifiers. As an example, illustrative test accuracy of 93% illustrates how the findings can be reported; nevertheless, it is necessary to conduct actual experiments in order to confirm these figures and prove whether the proposed CNN has superiority to the baseline methods.

Conclusion

In this paper, a framework that can be used to explainably classify heterogeneous database workloads based on runtime performance features is proposed, which involves an explainable Convolutional Neural Network (CNN)-based framework. The framework takes into account four types of workloads: Transactional, Analytical, Reporting and Mixed, based on the features of query execution time, CPU utilization, memory consumption, disk I/O activity, query complexity, and concurrency. CNN-based classification is used together with SHapley Additive explanations (SHAP) to enhance the interpretability of workload predictions through determining the features that affect classification decisions. It is measured by accuracy and precision, recall, F1-score, classification latency, and computational overhead and compared to traditional classifiers including Decision Tree, Random Forest, Support Vector Machine, and Multilayer Perceptron. TPC-H/TPC-DS offers an analytical workload scenario and more representative workloads are needed to represent all four categories. The suggested solution

ISSN: 3107-8168
DOI: https://doi.org/10.65725/JCISE/2/4/008

will facilitate meaningful and understandable database workload identification that can form a basis of future adaptive query processing and workload-conscious optimization. The effectiveness of the framework, computational efficiency and applicability to other database settings need to be further tested through experiments.

REFERENCES

[1.] H. Lan, Z. Bao, and Y. Peng, "A Survey on Advancing the DBMS Query Optimizer: Cardinality Estimation, Cost Model, and Plan Enumeration," Data Science and Engineering, vol. 6, pp. 86–101, 2021. https://doi.org/10.1007/s41019-020-00149-7. Springer Nature Link

[2.] R. Marcus et al., "Neo: A Learned Query Optimizer," Proceedings of the VLDB Endowment, vol. 12, no. 11, pp. 1705–1718, 2019. https://doi.org/10.14778/3342263.3342644.

[3.] R. Marcus et al., "Bao: Making Learned Query Optimization Practical," in Proceedings of the ACM SIGMOD International Conference on Management of Data, 2021.

[4.] S. M. Lundberg and S.-I. Lee, "A Unified Approach to Interpreting Model Predictions," in Advances in Neural Information Processing Systems, vol. 30, 2017. Read paper.

[5.] Z. Yang, W.-L. Chiang, S. Luan, G. Mittal, M. Luo, and I. Stoica, "Balsa: Learning a Query Optimizer Without Expert Demonstrations," in Proceedings of the 2022 ACM SIGMOD International Conference on Management of Data, 2022. https://doi.org/10.1145/3514221.3517885.
arXiv

[6.] Y. Kim, Y. Choi, Y. Gil, S. Lee, H. Shin, and J. Chong, "BitE: Accelerating Learned Query Optimization in a Mixed-Workload Environment," arXiv preprint arXiv:2306.00845, 2023. https://arxiv.org/abs/2306.00845.
arXiv