Methodology: Automatic Database Index Tuning Using Machine Learning

Mounicasri Valavala
University of the Cumberlands
Williamsburg, Kentucky, USA
mvalavala1456@ucumberlands.edu

Wasim Alhamdani
University of the Cumberlands
Williamsburg, Kentucky, USA
wasim.alhamdani@ucumberlands.edu

***Abstract*—Index plays a crucial role in determining the query response time and enables fast data access. The Automatic index-tuning model using Machine Learning (ML) increases the index's adaptability to variable workloads by including the column's future usage, column properties, and query operations. The paper presents the methodology to create an automated model for index selection and is part of the Automatic Database Index Tuning series. Selecting the index using ML is the first of its kind. As a result, there is no available dataset. Hence, the research finalizes the features influencing index selection by interviewing the database professionals and data collection by analyzing the existing indexes. The paper will cover the data collection methods, data analysis, and implementation details of the model.**

Keywords—machine learning, index tuning, workload forecast, execution history

I. Introduction

Database performance is an area of extensive research due to its impact on almost all applications. The query request patterns vary with the changing business needs, making it necessary to develop dynamic database tuning techniques [1]. The different ways to improve the Database performance are by altering knob configurations, tuning indexes, partitioning, and following query guidelines. According to the existing researchers, indexing is an efficient solution to speed up query execution [2].

The first paper in the Automatic Database tuning series explains the automatic index-tuning model's theoretical framework in detail [3]. It contains two main modules as Index selection and workload forecast modules. According to the presented framework, the index selection module will use the Random Forest model due to its adaptability to variable dataset sizes and unbalanced datasets [3].

The workload forecast module forecasts the usage of a given column, table, or stored procedure [3]. This module provides the column's future usage for a given column to the index selection module. This module also helps report a given entity's usage to the database teams, enabling them to manage the resources efficiently [3]. This module uses Long Short Term Memory (LSTM) for workload forecast according to the theoretical framework.

The paper presents the research methodology of the Automatic Index Tuning model. The research methodology explains the reasons to choose qualitative analysis, target population, data collection instruments, data analysis, dataset construction, and implementation details. The paper presents the Research Methodology in Section II, expected results in Section III, and conclusion in Section IV.

II. Research Methodology

Most researchers created index tuning models using "What-If" plans, cost-based models, Heuristics, and Reinforcement Learning (RL). A research model used Machine Learning and "What-If" plans to perform index selection [4]. The main drawbacks with the existing models are that none of the models consider the future usage of a column, the use of expensive "What-If" plans, selection of index based on current query plan alone, and the long duration for the RL models to converge [5,6]. The research overcomes these drawbacks by considering the parameters like future workload forecast, query frequency, and column characteristics.

As the current research is the first of its kind in Database tuning, there is no prior dataset available. Hence, the first step in methodology focuses on identifying the features required for the index tuning dataset. The research follows a qualitative analysis approach to explore the domain experts' knowledge to look at index tuning and workload forecast in-depth and analyze the proposed model's possible consequences.

A. Qualitative Analysis

The qualitative analysis helps to understand: the real-time handling of database performance issues, scenarios where the Database professionals consider index tuning as a solution, different factors that drive the index creation decision, threshold values of fragmentation at which one needs to perform index defragmentation, and the target audience feedback on the proposed model. The outcome of this qualitative analysis will firstly help to make any additions or changes to the current features identified in the Dataset. Secondly, it helps assign values to the Dataset features, and finally, it helps align the proposed model to real-time needs.

1) *Data Collection Instruments:* The research uses Face-to-Face interviews to draw insights from the target individuals on a questionnaire. The questions posed to the interviewees will cover the topics such as index tuning, adaptability to changing workloads, and ML usage to improve performance. The subjects of interest here are Database professionals, and the interview format

is face-to-face using Skype video calls. The interviewers will record the calls for further analysis.

2) *Target Population and Sampling:* The research uses purposive sampling, as this technique concentrates on gaining an in-depth understanding of a phenomenon. The criterion to choose the target sample is the database developers and administrators with a minimum of five years of database experience and expertise in different business domains. The selection of an individual relies solely on their experience, as it plays a vital role in making Database performance tuning decisions. The initial sample is ten participants, and the number of participants will be increased if the interviews result in divergent views, making it difficult to conclude.

3) *Credibility, Dependability, Transferability, and Confirmability:* The criteria to judge qualitative research are credibility, dependability, transferability, and confirmability. Credibility ensures that the research is credible from the participant's perspective [7]. The different approaches to determine the research credibility are Persistent Observation, Prolonged Engagement, Member Check, and Triangulation [7]. The current research uses the Member Check strategy by presenting the analysis and conclusions drawn from the interview back to the participants. The participant's approval of the provided findings validates the analysis.

Dependability focuses on accountability to make the research adaptable to varying needs [7]. The dependability is ensured by keeping track of all the research steps and recording any research design changes due to external or internal factors. Confirmability concentrates on creating the results in such a state that others can confirm those results. The confirmability is ensured by conducting the data collection without any bias and getting the questionnaire reviewed by a domain expert to add value to the whole process.

Transferability determines if the research is extendable for other purposes in the domain. This research ensures transferability by throwing light on different aspects of index tuning and workload forecast. It also contributes to transferability by constructing the Dataset that is applicable for any index-tuning problem across different DBMS.

The research will consider the expert opinion on the interview questionnaire before starting the interviews to ensure the analysis's validity. This step ensures that the questions are related to the investigation theme and make necessary corrections. Selecting experienced Database developers and administrators ensures the reliability of the study. The reason to choose the database administrators and developers are their exposure to database performance issues daily.

4) *Interview Questionnaire:* The interview questionnaire focuses on getting the real-time scenarios and the resolution process followed in index tuning. It also helps to identify the areas of improvement that would improve the adaptability of the model to any business. The sample questions in the interview are presented below.

   a. Is there any specific criteria for index creation to ensure the indexes are usable and effective for database optimal operations?
   b. How often changes in database workloads affect the usage of existing indexes?
   c. Do you use any ML-based tools to automate DBMS tasks, and what are the challenges you face to customize it according to your work environment?

5) *Data Analysis:* Qualitative data analysis involves searching and ordering the transcripts acquired through interviews to enhance the system's understanding [8]. The research will use Thematic Analysis to extract the required information by themes. The research uses Dedoose to perform coding and extract the analysis outcome by themes.

In the first step, the researcher will load the interview transcripts into Dedoose to analyze the content. The next step contains the process of coding by assigning specific codes to the data. It helps to identify the different categories of opinions. The combination of codes into themes will assist in getting a broader picture of the collected data.

There are two approaches to identify the themes such as inductive and deductive. In the inductive approach, the data drives the theme creation, whereas, in the deductive approach, one analyzes the data with a predefined set of themes [9]. The data analysis followed a deductive approach to identify the themes, as the themes of interest are clear before the interview. The interview questionnaire designed targets to explore a predefined set of themes such as Index Selection, Defragmentation, Future Workload Forecast, and ML Models for Performance tuning. The analysis outcome in each theme acts as input for feature construction for index tuning and improve the model.

B. Implementation Details

The research contains two modules, such as Index selection and Workload Forecast Model, and both these modules use ML models, Python and SQL Server. The dataset construction and model will be developed and tested on SQL Server's Wide World Importers database. The below sections explains the implementation details of each of these modules in detail.

1) *Index Selection:* This module performs index selection by applying ML classification techniques on the Dataset constructed using the column properties and usage statistics. The qualitative analysis drives the dataset creation by finalizing the features. During the real-time implementation of the model, the existing Database indexes act as a basis for the dataset construction. The different instruments used are SQL Server, ML Models, and Python. The Index Selection model contains Actual Execution Plan Extractor (AEP Extractor), Actual Execution Plan Parser (AEP Parser) to parse the query execution plan, Feature Vector Constructor (FVC) to construct the feature vectors using execution plans and column properties, Workload Forecast Model (WFM), ML Classification Model, and Index recommendation Generator.

SQL Server is the DBMS used to test this model. The research uses SQL Server's Wide World Importers Database to test the model by running schedulers with random stored procedures. The model is platform-independent and can work on any DBMS software, as there is no dependency on DBMS components like Query Optimizers. The reason to choose SQL Server for the test is that it is a widely used DBMS across different domains. Python is the programming language used to parse the query execution plan and implement the ML models. The reason to choose Python is the support it offers in implementing a Machine Learning model (Scikit Learn) and parsing SQL queries (SQL parser).

The AEP extractor extracts the XML execution plan of the given query and query text from SYS.QUERY_STORE_PLAN using the Query IDs from SYS.QUERY_STORE_QUERY. The AEP Parser parses the actual execution plan of the existing indexes using Python. It parses the XML and gives the outcome as the table name, column name, operators, and table scan. It also uses SQL parser to parse the query text and extract predicate columns in the Where clause of the query. AEP sends each predicate column as input to the WFM that returns a future forecast for the column usage, and this is one of the features in the feature vector.

The Feature Vector Creation (FVC) combines the outcome from AEP Parser and WFM. Also, it retrieves column properties such as IsNullable, datatype, and constraints for each of the predicate columns using the system tables, such as SYS.COLUMNS and SYS.CONSTRAINTS. Finally, FVC adds the table's row count to the feature vector generated from the above steps. The feature vector for the query experiencing slowness will go through the steps followed in feature vector construction for training data. The outcome feature vector will serve as an input to the ML classification model.

The proposed model uses Machine Learning classification techniques, and the outcome of the model is a label that indicates whether to create an index or not. The classification model chosen for this purpose is the Random Forest model. The reason to pick this ensemble model is its flexibility to give efficient results with datasets of variable sizes and unbalanced data.

A table will be created to store the configuration parameters for the model. The decrease in the execution time of a query is a measure of performance gain added by this model. The database teams will use this model for slow running queries to get index recommendations. The comparison of the execution time before and after applying the index gives the performance improvement.

Table	Column	Where	GroupBy
OrderLines	PackageTypeID	0	0
OrderLines	OrderID	0	0
Orders	CustomerID	0	0
PackageTypes	PackageTypeID	0	0
Orders	OrderID	0	0
Customers	CustomerID	0	0
PurchaseOrderLines	LastEditedWhen	1	0
PurchaseOrders	LastEditedWhen	1	0

Table 1: Column Operations

Column	Data Type	Unique	Foreign Key
PackageTypeID	Int	1	1
OrderID	Int	1	1
CustomerID	Int	1	1
PackageTypeID	Int	1	0
OrderID	Int	1	0
CustomerID	Int	1	0
LastEditedWhen	Datetime2	0	0
LastEditedWhen	Datetime2	0	0

Table 2: Column Properties

The feature vector that serves as input to the random forest model is the vector constructed with features {Where, GroupBy, Join, TableScan, Nullable, Datatype, Unique Constraint, Foreign Key Constraint, Table Row Count} and the outcome of the model is an index recommendation. Figure 1 presents the sample dataset for the used to train the random forest model. The python utilities created as part of the proposed model takes the stored procedure name and constructs the feature vector without any manual intervention. The feature vector is input to the random forest model to get index recommendations for a given stored procedure.

2) *Future Workload Forecast Model:* This module forecasts future usage at a granular level,` such as a column, query, and table level. The Dataset for this model is constructed by running schedulers with different stored procedures on the SQL Server's Wide World Importers Database for 15 days. Capturing the Database usage is a mandatory step before using this model. The technologies used to implement this model are Autoregressive Integrated Moving Average (ARIMA), Python, and SQL Server.

The previous article's model in this series explains that the workload forecast module uses LSTM [3]. The model is changed to ARIMA due to the current dataset size. Besides, the dataset size, even in real-time, will be smaller in the initial days of the model setup, which will grow over time. Hence using ARIMA is the suitable approach for this module.

As mentioned earlier, SQL Server is one of the highly used Databases, because of which is chosen to implement and test the proposed model. However, this model is adaptable to different types of Databases. Aforementioned, Python has excellent support for ML libraries and SQL parsing, due to which it is selected to implement this model.

Query	Execution Date Time	Occurrences
GetOrderUpdates	2021-02-17 09:00:00.000	94
GetOrderUpdates	2021-02-17 10:00:00.000	87
GetOrderUpdates	2021-02-17 11:00:00.000	64
GetOrderUpdates	2021-02-17 12:00:00.000	82

Table 3: Query Usage Stats

The WFM uses the Extended Events functionality of SQL Server to track the calls to stored procedures along with the time of the call. Extended Events is a lightweight utility provided by SQL Server to collect the required data for performance tuning and monitoring [10]. It provides options to configure the required events, fields, and storage locations to save the tracked information. It is an alternative to SQL profiler, which keeps a significant overhead on the DBMS. This utility adds the details of an event with selected fields into a location configured in the Data Storage section that helps to specify the fields of interest. In this case, the fields of interest are the stored procedure name and the time of its execution.

The schedulers with all the available stored procedures in the Wide World Importers Database simulates the real-time database behavior. The procedures' calls are randomized, and Python calls the schedulers to run once every hour from morning 9 to evening 6. The Logon Scheduler runs the procedures to perform Login and Change Password functionalities, will run only from 9 to 11, as the login process usually happens in the early work hours.

The Python program will run at the end of the day to process the stats captured in the Extended Events. The processing will happen in two steps. The program captures the procedure and extracts the tables and column names from the query execution plan using AEP Parser in the first step. In the second step, python resampling techniques aggregate the number of calls per hour. The Dataset created from this process will act as training data for the time series forecast model, and this process will run until it gets the necessary training data.

Table	Execution Date Time	Occurrences
Orders	2021-02-17 09:00:00	98
Orders	2021-02-17 10:00:00	73
Orders	2021-02-17 11:00:00	100
Orders	2021-02-17 12:00:00	55

Table 4: Table Usage Statistics

Table	Column	Execution Date Time	Occurrences
Orders	OrderID	2021-02-17 09:00:00	94
Orders	OrderID	2021-02-17 10:00:00	87
Orders	OrderID	2021-02-17 11:00:00	64
Orders	OrderID	2021-02-17 12:00:00	82

Table 5: Column Usage Statistics

Time-series forecasting models play a crucial role in business across different domains due to their capability to foresee events. The Autoregressive Integrated Moving Average (ARIMA) is a well-known models in this domain and is an advanced model that overcomes the Autoregressive Moving Average (ARMA) model's limitation of being able to work only for linear data. ARIMA model offers excellent flexibility, solid statistical properties, and can handle nonstationary data [11]. The reason to choose this model is that the Dataset used in this research is free from noise as the data creation process happens at one source with predefined instructions. The training dataset creation occurs in the initial phase of implementing WFM and does not directly rely on the online data.

The configuration table created while initializing the model stores the configuration information related to this module. This configuration capability makes the model dynamic, and Python handles the necessary communication between the database and the ML model.

The dataset with execution time and number of calls serves as the training dataset for the ARIMA model. The data shown in table 1, 2, and 3 are the sample datasets to train the ARIMA models. The input to the trained model is the execution time and it will output the number of calls expected to happen at the given time.

III. Expected results

The research results are of two sections. They are qualitative analysis and implementation results. Qualitative analysis will throw light on the factors influencing index tuning decisions, the importance of workload forecast models, and feedback on the proposed model. The implementation will result in the index recommendations and workload forecast at a future point in time.

The qualitative analysis will answer the below questions

- What factors influence the Database professional's decision to create an index on a particular column?
- What parameters are used by the Database professionals to determine the need for index defragmentation?
- How does granular workload forecast assist in decision-making in Database performance tuning?
- What are the benefits and areas for improvement pointed out by the Database professionals in the proposed model?

The index selection model should suggest the correct columns as indexes, resulting in a decrease in the execution time. The variation in the execution time acts as a metric for the model accuracy. The usage values predicted by the workload forecast model will present the future column/table/stored procedure usage in future time and evaluate the model accuracy using MASE. The results will project the performance improvement of a database without manual intervention.

IV. Conclusion

The automatic index tuning using the ML model will overcome the drawbacks of existing models. It will open new research avenues to address Database tuning tasks using a promising technology like ML. The research also presents the factors that impact index tuning decisions based on database professionals' views, making it a baseline for further index tuning research. The future work is to create the features using the qualitative analysis, implement the proposed model, and study the model's feedback given by the target population. Addressing the concerns raised by the database professionals will pave the way for further research in this area.

References

[1] Kamatkar, S. J., Kamble, A., Viloria, A., Hernández-Fernandez, L., & Cali, E. G. (2018). Database Performance Tuning and Query Optimization. Data Mining and Big Data, 3–11. https://doi.org/10.1007/978-3-319-93803-5_1

[2] Qi, C. (2016). On index-based query in SQL Server Database. 2016 35th Chinese Control Conference (CCC), 9519–9523. https://doi.org/10.1109/chicc.2016.7554868

[3] Valavala, M., & Alhamdani, W. (2021). Automatic Database Index Tuning Using Machine Learning. 2021 6th International Conference on Inventive Computation Technologies (ICICT), 523–530. https://doi.org/10.1109/icict50816.2021.9358646

[4] Ding, B., Das, S., Marcus, R., Wu, W., Chaudhuri, S., & Narasayya, V. R. (2019). AI Meets AI. Proceedings of the 2019 International Conference on Management of Data - SIGMOD '19, 1241–1258. https://doi.org/10.1145/3299869.3324957

[5] Tizhoosh, H. R. (2006). Opposition-Based Reinforcement Learning. Journal of Advanced Computational Intelligence and Intelligent Informatics, 10(4), 578–585. https://doi.org/10.20965/jaciii.2006.p0578

[6] Naik, S. (2017). TIER: Table index evaluator and recommender — A proposed model to improve transaction performance in distributed heterogeneous Databases. 2017 International Conference on Soft Computing and Its Engineering Applications (IcSoftComp), 1–8. https://doi.org/10.1109/icsoftcomp.2017.8280085

[7] Korstjens, I., & Moser, A. (2017). Series: Practical guidance to qualitative research. Part 4: Trustworthiness and publishing. European Journal of General Practice, 24(1), 120–124. https://doi.org/10.1080/13814788.2017.1375092

[8] Wong L. (2008). Data analysis in qualitative research: a brief guide to using nvivo. Malaysian family physician: the official journal of the Academy of Family Physicians of Malaysia, 3(1), 14–20.

[9] Caulfield, J. (2020, August 14). How to do thematic analysis. Scribbr. https://www.scribbr.com/methodology/thematic-analysis/

[10] Microsoft (2019, July 23). XEvents overview - SQL Server. Microsoft Docs. https://docs.microsoft.com/en-us/sql/relational-databases/extended-events/extended-events?view=sql-server-ver15

[11] Liu, C., Hoi, S., Zhao, P., & Sun, J. (2016). Online ARIMA Algorithms for Time Series Prediction. AAAI.