An Efficient Diversity-Aware Method for the Empty-Answer Problem

Yuto Ikeda[1,*], Chuan Xiao[1] and Makoto Onizuka[1]

[1]*Osaka University, 1-5 Yamadaoka, Suita, Osaka 565-0871 Japan*

Abstract

This study tackles the empty-answer problem in database queries, where no results meet all user-specified conditions, though some may satisfy individual ones. We explore query relaxation, which removes certain conditions for results, and a record search method that uses user preferences to evaluate records. In particular, we emphasize the importance of diversity in the results to better match user preferences, which has been ignored in existing approaches. To address this, we introduce the use of Maximal Marginal Relevance (MMR) – a ranking function balancing query relevance and record diversity – for query relaxation, proposing a method that searches for diverse record sets while maintaining many conditions. Experiments with real-world datasets demonstrated that the proposed method significantly increases search speed (up to 300 times faster) while maintaining high MMR scores, indicating an effective balance between efficiency and result diversity.

Keywords

empty-answer problem, query relaxation, maximal marginal relevance

1. Introduction

In various applications, setting desired conditions and searching for data is a fundamental operation. Ideally, searches should yield a small number of records that match the specified conditions. However, the number of records retrieved can vary significantly depending on the user's conditions, resulting in either too many records (the many-answer problem) or no records at all (the empty-answer problem) [1]. Whereas the many-answer problem can be addressed by preseting the top-k results (e.g., with a LIMIT clause), the empty-answer problem is more challenging. The causes of the empty-answer problem can be categorized into two scenarios: (1) records satisfy each condition individually but not collectively due to multiple conditions, and (2) the conditions are invalid, such as searching for records that do not meet pre-set constraints in the data. In this paper, we target the empty-answer problem and address the first scenario, where potentially all records in the database fall within the search scope.

To solve the empty-answer problem, existing methods are broadly classified into two approaches. The first is the query relaxation method [2, 3, 4], where conjunctive conditions set by the user are reduced to solve the empty-answer problem. The second is the ranking method [5], which translates user preferences into a function to evaluate records. It effectively reflects user preferences, especially when users can design the ranking function.

In the user's record set, the relevance to the query and the diversity within the recommended set are both important. Preserving diversity among recommended records is key to capturing user preferences [6]. While various approaches have been proposed, none have focused on the diversity of the record set post-solving the empty-answer problem in the database field. Meanwhile, Maximal Marginal Relevance (MMR) [7], a ranking function balancing query relevance and record diversity, has been proposed and widely used for information retrieval.

In this study, we aim to solve the empty-answer problem by considering diversity and utilizing MMR as the ranking function. We formalize MMR for relational database applications and propose a method for quickly finding a record set that maximizes MMR, Also, We devise a series of relaxed query search and record search techniques tailored to this objective. They broaden results by removing some conditions and evaluate records based on user-defined functions, respectively. In addition, we utilize cardinality estimation techniques to further optimize the search process. Experiments with real-world datasets show that our method significantly increases search speed (up to more than 300 times faster) while maintaining high MMR scores and outperforming baseline approaches.

2. Preliminaries

We denote the query provided by the user as q, and assume that this query is composed of m conjunctive conditions,

DOLAP 2024: 26th International Workshop on Design, Optimization, Languages and Analytical Processing of Big Data, co-located with EDBT/ICDT 2024, March 25, 2024, Paestum, Italy

*Corresponding author.

ikeda.yuto@ist.osaka-u.ac.jp (Y. Ikeda); chuanx@ist.osaka-u.ac.jp (C. Xiao); onizuka@ist.osaka-u.ac.jp (M. Onizuka)

https://sites.google.com/site/chuanxiao1983 (C. Xiao); http://www-bigdata.ist.osaka-u.ac.jp/professor/onizuka/onizuka_en.html (M. Onizuka)

0000-0001-5559-8300 (M. Onizuka)

with each condition represented as c_i for $0 \leq i \leq m-1$. Then, the query can be expressed as: $q = \bigwedge_{i=0}^{m-1} c_i$.

Next, we define the set of conditions of the query q as C_q, and the power set of C_q as $Pow(C_q)$. A relaxed query q', derived from q, is formulated using a proper subset $C \subseteq Pow(C_q) \setminus \{C_q\}$ as follows:

$$q' = \bigwedge_{c_i \in C} c_i. \tag{1}$$

Since the number of elements in $Pow(C_q)$ is $2^m - 1$, there are $2^m - 2$ candidates for q', excluding q itself. We denote the total number of records in the database as n, the total number of columns in a record as l, and the number of records the user wishes to obtain as a query result as k.

MMR. MMR is a combined score comprising (1) the relevance of the recommended record set to the user-specified query, and (2) the diversity within the recommended record set. When the recommended record set is defined as D', MMR is defined as follows [7]:

$$MMR(q, D') = rel(q, D') + \lambda\, div(D') \tag{2}$$

where λ is a parameter that adjusts the balance between relevance and diversity.

We define relevance and diversity as follows:

$$rel(q, D') = \frac{1}{k} \sum_{r \in D'} \frac{\sum_{j=0}^{m-1} \delta(c_j, r)}{m} \tag{3}$$

$$div(D') = \begin{cases} 1 & \text{if } len(D') = 1 \\ \min_{r, r' \in D', r \neq r'} \left(\frac{dist(r,r')}{l} \right) & \text{if } len(D') \neq 1 \end{cases} \tag{4}$$

where $\delta(c_i, r)$ returns 1 if the record r satisfies the query condition c_i, and 0 otherwise. $len(D')$ denotes # records in D' Intuitively, Equation 3 represents the average ratio of the number of matching conditions to the total number of conditions for each record in D'. Additionally, the *dist* function in Equation 4 defines the distance between records, and any distance function can be applied. In this study, the Manhattan distance is used for numerical data, and the Hamming distance for categorical or binary data.

Problem Definition. The problem is defined as follows: given a query q that yields an empty-answer for single table dataset D, the objective is to identify a subset of records $D' \subseteq D$, consisting of k records, where k is the number that user designated, that maximizes $MMR(q, D')$.

MMR is a metric designed for evaluating a set of records. To identify the recommended record set that maximizes MMR, it is necessary to calculate and compare the distances between all pairs of the k records. This process incurs $O(mn^k)$-time, which becomes impractical, particularly for large values of k. As a result, methods have been proposed to search for the recommended record set in a greedy manner [8] for the sake of efficiency.

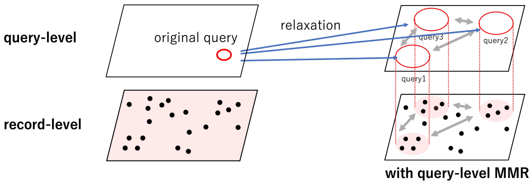

Figure 1: An overview of query-level MMR.

3. Proposed Method

3.1. Base Algorithm

To identify a recommended record set that maximizes the MMR score, it is necessary to calculate the distance between every pair of records (as per Equation 4), a task that is computationally intensive. Consequently, existing methods [8, 9] employ heuristic approaches to find approximate answers, including the state-of-the-art method [9] which incurs a time complexity of $O(nlk)$ to find these approximate answers.

Our approach also utilizes a heuristic method to efficiently find approximate solutions. Considering that a query can be viewed as an abstraction or specification of its resulting records, we introduce the concept of query-level MMR as a preprocessing step for record-level MMR. Figure 1 provides an overview of this process. Initially, we generate multiple relaxed queries from the original query, aiming to maximize query-level MMR (as shown in the top right corner of the figure). Subsequently, we acquire the results of these relaxed queries (depicted at the bottom right corner) and select a recommended record set from these results that maximizes the record-level MMR.

We define query-level MMR by substituting a recommended record set (D') with a relaxed query set (Q') in Equation 2 as follows:

$$MMR(q, Q') = rel(q, Q') + \lambda\, div(Q'). \tag{5}$$

Additionally, we introduce metrics for query relevance and diversity. Query relevance calculates the similarity between the original query and a relaxed query set, while query diversity measures the diversity within the relaxed query set. These metrics are defined as follows:

$$rel(q, Q') = \frac{1}{k} \sum_{q' \in Q'} \frac{len(q')}{len(q)} \tag{6}$$

$$div(Q') = \min_{q'', q' \in Q', q' \neq q''} \frac{len(q' \cap q'')}{len(q' \cup q'')}. \tag{7}$$

$len(q)$ denotes the number of conditions in a query q.

Our method comprises two stages: (1) searching for relaxed queries that maximize query-level MMR and obtaining their results; (2) selecting a recommended record set that maximizes record-level MMR from these results.

By relaxing the original query provided by the user, records that satisfy the relaxed query are considered as candidates for the recommended record set. These candidate records should ideally satisfy more conditions from the user's query while contributing to the diversity of the recommended record set. To achieve this, relaxed queries are selected to maximize query-level MMR.

As discussed in Section 2, there are $2^m - 2$ potential candidates for relaxed queries. Directly searching through all these candidates is impractical. In our proposed method, we record pairs of conditions used in previous relaxed queries and determine conditions greedily one by one for new relaxed queries. This approach ensures diversity in the records derived from relaxed queries by varying the conditions within each group, and it can be achieved with a time complexity of $O(m^2)$.

For specific processing, initially, from the conditions in the original query, we select one condition that has been used the least in previous relaxed query groups for the new relaxed query. For subsequent conditions, when the set of already determined conditions is $C_{q'}$, the condition c_{new} that minimizes $\sum_{c \in C_{q'}} count(c, c_{new})$ is chosen. Here, *count* is a function that records the frequency of condition pairs appearing in relaxed queries from past iterations. For instance, if a user's query contains conditions c_1, c_2, c_3, and the first recommended record search included c_1, c_2 in its relaxed query, then $count(c_1, c_2) = 1$ and $count(c_1, c_3) = 0$.

In case of multiple conditions minimizing the *count* function, priority is given to the condition least used in prior relaxed query groups. Finally, records satisfying all query conditions just before the relaxed query yields no results are considered as candidates. After determining the relaxed query, all condition pairs in this query are recorded.

When searching for a relaxed query, records are progressively narrowed down with each determined condition. Starting from the second condition, evaluation is conducted only on the record set that meets all previously established conditions. This strategy significantly reduces the number of records evaluated for each condition.

After identifying candidates for the recommended record set, we search for the record that maximizes the MMR from that set. In doing so, we maintain the minimum distance between the records selected in the recommended record set to reduce computational costs. However, unlike existing methods, our proposed method does not conduct a full search of records, rendering the avoidance of duplicate calculations for additional record candidates infeasible.

3.2. Complexity Analysis

The exact time complexity of the proposed method is influenced by the proportion of records satisfying each condition in the query and the dependency relationship between the sets of records satisfying multiple conditions, which makes it challenging to calculate precisely. We therefore consider the time complexity under the general assumption that there is no dependency relationship between the values of each column.

If the cardinality of c_i is $\sigma_i (0 \leq \sigma_i \leq 1)$, the time complexity of the proposed method is expressed as:

$$O(k(m^2 + kl + n \sum_{i=0}^{m-1} \prod_{j=0}^{i} \sigma_j)).$$

The first term accounts for the complexity of selecting conditions for the relaxed query and recording the conditions used in q', and the second term pertains to the complexity of calculating MMR for the record set obtained from q' and determining the recommended record set. The number of additional records is not factored in here. The third term relates to the complexity of searching q'. The sum inside represents the number of records evaluated for each condition.

Considering these parameters, typically k and m are up to 100 or smaller, while n is often larger. Therefore, when $k, m, l \ll n$, the time complexity approximates to:

$$O(kn \sum_{i=0}^{m-1} (\prod_{j=0}^{i} \sigma_j)). \tag{8}$$

For estimating complexity, consider a scenario where the selection rate is identical for all conditions. If σ_j in Equation 8 is σ, we have

$$O(kn \sum_{i=0}^{m-1} (\prod_{j=0}^{i} \sigma_j)) \leq O(kn \sum_{i=0}^{\infty} (\sigma^j) = O(kn \frac{1}{1-\sigma}) \tag{9}$$

3.3. Further Optimizations

We revisit the proposed method. When determining the conditions of a query, the method does not explicitly address scenarios involving multiple relevant conditions. The primary objectives for obtaining a relaxed query, as mentioned earlier, are to maintain high diversity among queries and to retain as many conditions from the user's query as possible. In scenarios where conditions have the same priority in the co-occurrence matrix, selecting any of these conditions would similarly uphold the diversity among queries. Consequently, when prioritizing these conditions, the focus should be on obtaining a relaxed query that preserves more conditions from the user's query. For optimization, we propose to select from conditions with equal priority in the co-occurrence matrix those likely to yield more records after evaluation.

Methods for exploring the cardinality of conditions include: (1) direct evaluation of the condition to calculate the exact cardinality, and (2) utilizing cardinality estimation for an approximate value. Each of these methods presents its own advantages and disadvantages. Direct evaluation provides precise cardinality but may lead to longer execution times, especially when evaluating a few conditions over a large number of records. In contrast, cardinality estimation offers more consistent execution times regardless

Table 1
Result of exection time

k	5	10	15	20
proposed base	0.132	0.206	0.286	0.371
proposed optimized	3.264	12.016	23.768	39.651
greedy	22.922	47.472	72.196	94.803
random	0.002	0.002	0.002	0.002

of the number of conditions, but the accuracy diminishes as the number of conditions increases, leading to potential discrepancies between estimated and actual values (e.g., PostgreSQL's built-in cardinality estimation might yield low accuracy). Advanced machine learning-based approaches for cardinality estimation, such as DeepDB[10], Naru[11], and Scardina[12], may offer improved precision.

We recommend a hybrid approach that combines these methods to mitigate their respective drawbacks, aiming for both accuracy and efficiency. We suggest employing cardinality estimation when a few conditions are selected and a large number of records require evaluation, and transitioning to direct condition evaluation when the number of records to evaluate falls below a certain threshold.

4. Experiments

4.1. Experimental Settings

Dataset. We follow the existing research on query relaxation [4] to use the Cars dataset [13], which was released by Mottin et al. After removing duplicate records, this dataset comprises 128,443 records with 31 columns, all containing boolean values. We employed 167 of queries. These are the queries used in existing research [4] and these are consists of $4 - 10$ conditions.

Competitors. In addition to the method proposed in Section 3 (proposed base method) and the enhanced approach described in Section 3.3 (proposed optimized method), we included two comparison methods in our experiments: a greedy method targeting the entire dataset (greedy) and a random selection method (random). The threshold for switching from cardinality estimation to direct execution in the proposed optimized method is set at 10.

Environment. All experiments were performed on a MacOS Ventura 13.2.1 machine equipped with an Apple M2 CPU and 24 GB of main memory. For the implementation of all algorithms in the experiments, Python 3.8 was used. We employed PostgreSQL for data storage, ensuring a unified experimental environment between the proposed base method and the exhaustive search method. The reported execution times exclude the dataset loading times. To implement the cardinality estimation in the proposed optimized method, DeepDB [10] was utilized. The model used in this experiment was prepared beforehand.

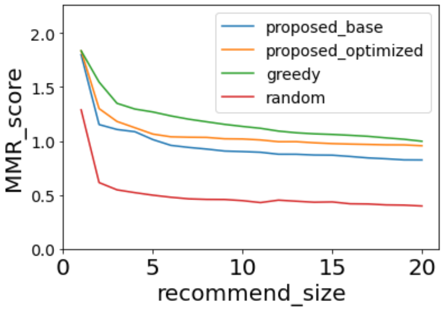

Figure 2: Result of MMR score

4.2. Experimental Results

Tables 1 present the results of measuring the execution times of each method for each query by varying the value of k. According to these findings, the proposed base method achieved speeds more than 100 times faster than the greedy search method, while the proposed optimized method was 3 – 7 times faster. These results demonstrate that our proposed base method achieves a remarkable speed-up, and that applying cardinality estimation incurs substantial computational costs, consuming a significant portion of the execution time. The execution time of our proposed base method scales almost linearly.

Figures 2 display the MMR score results. The weighting factor λ was set to 1 for both diversity and relevance. Both the diversity and relevance terms were normalized to the range $[0, 1]$, resulting in MMR scores ranging from $[0, 2]$. The experimental results indicate that the proposed base method approached the performance of the greedy search method and achieved higher MMR scores compared to the baseline random method.

5. Conclusion

In this study, we formulated an evaluation metric that considers both diversity and relevance in the field of databases. We proposed a method that searches for a record set that can be presented quickly compared to existing methods, solving the empty-answer problem. We further improved the method by employing cardinality estimation. In addition, we conducted an empirical evaluation on a dataset and queries used in previous research, confirming that our method achieves significant speed improvements while maintaining accuracy.

Acknowledgements

This work is supported by JSPS Kakenhi 22H03903, 23H03406, 23K17456, and CREST JPMJCR22M2.

References

[1] S. Basu Roy, H. Wang, G. Das, U. Nambiar, M. Mohania, Minimum-effort driven dynamic faceted search in structured databases, in: CIKM, 2008, pp. 13–22.

[2] N. Koudas, C. Li, A. K. Tung, R. Vernica, Relaxing join and selection queries, in: VLDB, 2006, pp. 199–210.

[3] C. Mishra, N. Koudas, Interactive query refinement, in: EDBT, 2009, pp. 862–873.

[4] D. Mottin, A. Marascu, S. B. Roy, G. Das, T. Palpanas, Y. Velegrakis, A holistic and principled approach for the empty-answer problem, The VLDB Journal 25 (2016) 597–622.

[5] S. Agrawal, S. Chaudhuri, G. Das, A. Gionis, Automated ranking of database query results, CIDR (2003).

[6] M. Kaminskas, D. Bridge, Diversity, serendipity, novelty, and coverage: a survey and empirical analysis of beyond-accuracy objectives in recommender systems, ACM Transactions on Interactive Intelligent Systems 7 (2016) 1–42.

[7] J. Carbonell, J. Goldstein, The use of mmr, diversity-based reranking for reordering documents and producing summaries, in: SIGIR, 1998, pp. 335–336.

[8] I. Catallo, E. Ciceri, P. Fraternali, D. Martinenghi, M. Tagliasacchi, Top-k diversity queries over bounded regions, ACM Transactions on Database Systems 38 (2013) 1–44.

[9] K. Hirata, D. Amagata, S. Fujita, T. Hara, Solving diversity-aware maximum inner product search efficiently and effectively, in: CIKM, 2022, pp. 198–207.

[10] B. Hilprecht, A. Schmidt, M. Kulessa, A. Molina, K. Kersting, C. Binnig, Deepdb: Learn from data, not from queries!, Proceedings of the VLDB Endowment 13 (2020) 992–1005.

[11] Z. Yang, E. Liang, A. Kamsetty, C. Wu, Y. Duan, X. Chen, P. Abbeel, J. M. Hellerstein, S. Krishnan, I. Stoica, Deep unsupervised cardinality estimation, arXiv preprint arXiv:1905.04278 (2019).

[12] R. Ito, Y. Sasaki, C. Xiao, M. Onizuka, Scardina: Scalable join cardinality estimation by multiple density estimators, arXiv preprint arXiv:2303.18042 (2023).

[13] D. Mottin, S. B. Roy, A. Marascu, , G. Das, T. Palpanas, Y. Velegrakis, A holistic and principled approach for the empty-answer problem, https://helios2.mi.parisdescartes.fr/~themisp/queryrelaxation, 2023.