Comparison of Cardinality Estimation Techniques Utilizing Machine Learning

Hastin Himanshubhai Modi
Arizona State University
hmodi5@asu.edu

Harshil Hemantbhai Patel
Arizona State University
hpate143@asu.edu

Smit Ashokbhai Jasani
Arizona State University
sjasani2@asu.edu

I. Introduction

Cardinality estimation serves as a cornerstone in the realm of query optimization within database systems. Effective query optimization is critical as it directly influences the performance of database systems by determining the most efficient execution plans. Traditionally, this process relies on statistical approaches to estimate the "cardinality", or the number of tuples that a query will return. These estimates are pivotal for selecting the most efficient join orders, join methods, and whether indexes should be used.

Traditional cardinality estimation methods, such as histograms, sampling, and the use of statistical synopses, suffer from several drawbacks. Primarily, they struggle to maintain accuracy in dynamic environments where data distributions change frequently due to updates. Additionally, these methods often fail to capture the complexities of multi-table joins, especially when there are correlations between columns across tables, which are common in real-world databases.

With the integration of machine learning (ML) techniques into this field, new horizons have opened up. ML models, capable of learning from data patterns, offer the potential to significantly outperform traditional methods in both static and dynamic contexts. These models can adapt to changes in data distributions and capture complex relationships within the data, potentially leading to more accurate and robust cardinality estimates. This report will examine the performance of three models across two datasets to analyze the effectiveness of ML techniques for cardinality estimation.

II. Related Work

This section delves into the traditional and contemporary methods for cardinality estimation. Traditional methods have primarily relied on heuristic-based approaches, where estimations are made based on pre-computed statistics such as histograms and samples. These techniques, while effective in controlled scenarios, often degrade in performance when dealing with complex queries or rapidly changing data [1]–[4]. In recent years, several research initiatives have focused on leveraging machine learning to improve cardinality estimation and it has been an active area of research [5], [6] since many years and has recently seen a rise in popularity. The methods for cardinality estimation can be divided in three categories: data-driven estimators, query-driven estimators, and hybrid estimators.

Data-driven Cardinality Estimation: Methods of data-oriented cardinality prediction are built upon models derived from the actual data. Numerous unsupervised learning methods are utilized for this type of cardinality prediction. For instance, probabilistic graphical models (PGM) [7]–[10] leverage Bayesian networks to depict the joint distribution of data, although they depend on assumptions of conditional independence. On the other hand, methods based on kernel density estimation (KDE) [11], [12] avoid these independence assumptions but often struggle with subpar accuracy issues stemming from challenges in tuning the bandwidth parameter. NeuroCard [13] is a joint cardinality estimator that creates a single neural density estimator over an entire database and is one of the models which will be used for the comparative analysis in this report.

Query-driven Cardinality Estimation: In the realm of query-based cardinality prediction, supervised techniques use the query workload to develop predictive models. Advances in deep learning have led to its application in this area. Research by Ortiz et al. [6] shows the application of multi-layer perceptron neural networks and recurrent neural networks on processed queries for effective cardinality estimation. Another approach by Sup [14] employs a method where queries are transformed into a feature set, with a neural network learning the feature weights to predict selectivity.

Hybrid Cardinality Estimation: Some recent methodologies combine both the query workload and data insights for more accurate cardinality predictions. Results from data-driven models are now integrated with query-encoded features in machine learning models to enhance prediction accuracy. Dutt et al. [1] have incorporated histogram-based cardinality estimates as additional features along with query attributes, employing both neural networks and tree-based models for estimation. Similarly, Kipf et al. [15] utilize results from sampling methods as supplementary features alongside query attributes in convolutional neural networks to refine cardinality estimation. MSCN [15] is a multi-set convolutional network which is tailored to representing relational query plans and builds on sampling-based estimation and is also one of them odels which will be used for the comparative analysis.

III. Algorithms and Datasets

The models discussed herein, namely FCN (Fully Connected Network) [16], MSCN (Multi-Set Convolutional Net-

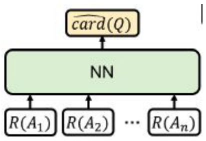

Fig. 1. FCN Model

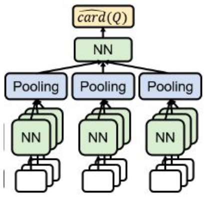

Fig. 2. MSCN Model

work), and NeuroCard, represent cutting-edge approaches in learned cardinality estimation, leveraging deep learning for improved accuracy and robustness over traditional histogram or sampling-based approaches. The architectures of the models are explained below:

- FCN: The FCN model in Fig. 1 [16] employs a deep neural network architecture that processes query features through multiple layers, enabling it to capture complex relationships between query parameters. This model is particularly adept at understanding inter- and intra-table predicate correlations, making it suitable for databases with intricate query patterns. The input to an FCN consists of features derived from SQL queries. These features typically include encoded representations of tables, joins, selection predicates, and other query components. Each feature might be one-hot encoded or transformed through embedding layers if they are categorical, such as table names or operation types. The FCN processes the input features through each layer, where every neuron in a layer is connected to all neurons in the previous layer, hence the term ”fully connected.” The weights of these connections, which are learned during training, determine the significance of each input feature concerning the cardinality estimate.
- MSCN: MSCN in Fig. 2 [16] utilizes a convolutional approach to handle sets of features from different parts of a SQL query (e.g., tables, joins, and predicates). This model is designed to capture weaker predicate correlations, which often occur in more straightforward query scenarios or less complex database schemas. MSCN also takes features derived from SQL queries. However, these features are organized into sets corresponding to different components of the query, such as sets of tables, joins, and predicates. The architecture features convolutional layers tailored to handle sets of input features. The convolutional layers apply filters to the feature sets to capture local patterns and relationships, which are crucial for understanding the impact of specific query components on the overall cardinality.

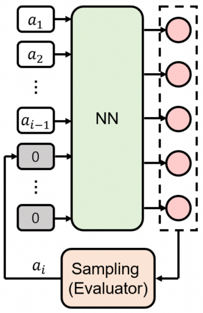

Fig. 3. NeuroCard Model

- NeuroCard: NeuroCard in Fig. 3 [16] extends the capabilities of cardinality estimation by employing a deep autoregressive model. This model predicts the number of result tuples (cardinality) by learning the joint distribution of table columns, thus capturing dependencies across the database schema. It’s particularly beneficial in environments where data distributions are not well understood or are highly variable. Unlike FCN and MSCN, NeuroCard primarily models the data distributions rather than the query structure. The input to NeuroCard is the data from the database tables, particularly focusing on the joint distribution of table columns. NeuroCard employs a deep autoregressive model, which models each attribute conditioned on previous attributes in a predefined order. This approach allows the model to learn the complex dependencies between different columns across one or more tables.

The 2 databases, i.e., IMDB and TPC-DS have been selected since they are used as standard databases in academia and also provide sufficient complexity to accurately gauge the performance of the models.

- IMDB: We have used a subset of the larger IMDB dataset, to maintain a manageable size but realistic complexity. It includes various tables typical of an online movie database, such as movies, actors, directors, and ratings. This database is ideal for testing cardinality estimation models because it represents real-world, complex query scenarios involving multiple joins and subqueries.
- TPC-DS: Even here, we have used a subset of the larger TPC-DS dataset. It is a widely recognized benchmarking tool designed to evaluate the performance of data warehousing solutions. It simulates a decision support system that provides answers to business-oriented ad-hoc queries, including those involving aggregations, joins, and nested queries. The complexity and variety of TPC-DS’s schema and queries make it an excellent candidate for assessing the performance and scalability of advanced cardinality estimation models.

The rationale behind comparing these models using the IMDB and TPC-DS databases lies in the distinct characteristics of these datasets. IMDB offers a real-world scenario with potentially unpredictable query patterns and data distributions, which challenges the models' ability to generalize from training data. In contrast, TPC-DS, with its structured and well-defined query workload, tests the models' efficiency and accuracy in a controlled environment. These comparisons aim to highlight the strengths and weaknesses of each model under different data characteristics and query complexities.

IV. Experimental Setup

This section delineates the experimental framework employed to assess the performance of the machine learning models designed for cardinality estimation. The evaluation is structured around rigorous training and testing protocols, leveraging extensive computational resources to ensure robustness and validity of the results.

The experimental evaluations are conducted on a high-performance computing instance on Amazon Web Services (AWS). The primary specifications of the system include:

- CPU: Intel Xeon E5-2686 v4 (Broadwell) with 8 virtual CPUs. This configuration provides a robust computing backbone suitable for handling multiple threads and processes efficiently.
- GPU: NVIDIA Tesla V100 with 16 GB memory. The NVIDIA Tesla V100 is one of the most powerful GPUs available in the cloud and is designed specifically for high-end computations required in machine learning and scientific computing. It significantly accelerates the training of neural network models.

The experimental framework is organized into two main stages: model training and testing protocol. Each stage is crafted to extract the most comprehensive understanding of each model's performance and characteristics.

Model training:

- For FCN and MSCN, features are derived from SQL queries including attributes such as tables accessed, join conditions, and predicate values. Features are encoded using one-hot encoding for categorical variables (e.g., table names, join types) and normalized continuous values for numerical predicates. The MSCN model utilizes these three sets of features (table, join, and predicate) to capture a comprehensive view of the query's structure.
- NeuroCard, which learns distributions of data, does not use query-derived features directly but instead learns from the data distribution observed in the database columns. During training phase, model is trained on historical data where it learns the likelihood of attribute values given the values of other attributes. This is done by maximizing the probability of the observed data under the model.
- The datasets are split into training, validation, and test sets. The training set comprises 70% of the total data, used for the initial fitting of the models. The validation set accounts for 15% and is utilized for hyperparameter tuning and to prevent overfitting during the training process. The remaining 15% serves as the test set, used solely for evaluating the models' performance.

Model	IMDB	TPC-DS
FCN	8	15
MSCN	6	14
NeuroCard	5	10

TABLE I
LOG(Q-ERROR) RESULTS

Model	IMDB	TPC-DS
FCN	50	70
MSCN	40	60
NeuroCard	80	100

TABLE II
INFERENCE TIME(MS) RESULTS

Testing protocol:

- The primary metric for assessing model performance is the Q-error, which quantifies the ratio between the estimated cardinality and the actual cardinality, providing a measure of prediction accuracy.
- Each model is subjected to a series of tests against both the IMDB and TPC-DS datasets. The testing involves executing a predefined set of queries that are representative of typical workloads in real-world applications. The queries are designed to cover a wide range of scenarios, from simple lookups and aggregates to complex joins and nested queries.

V. Results

The evaluation of FCN, MSCN, and NeuroCard models on the IMDb and TPC-DS datasets provides significant insights into the potential of machine learning techniques to enhance cardinality estimation in database management systems. This section analyzes the results, focusing on the models' accuracy as measured by Q-error and their inference times, which are crucial for real-world deployment.

The Q-error metric in Table I, which represents the ratio of predicted to actual cardinality, serves as a primary indicator of estimation accuracy. A lower Q-error signifies higher accuracy and reliability of the model in practical settings. FCN demonstrates moderate accuracy across both datasets. While the model performs reasonably well, the relatively higher Q-error on TPC-DS suggests challenges in handling more complex or larger-scale data scenarios typical of decision support systems. MSCN shows improved accuracy over FCN, particularly with the IMDb dataset. The convolutional approach to handling feature sets from SQL queries appears to enhance its ability to understand and predict cardinalities more effectively than FCN, though it still faces difficulties with the complex queries in TPC-DS. NeuroCard outperforms both FCN and MSCN, achieving the lowest Q-error in both datasets. Its deep autoregressive approach, which models joint data distributions across table columns, provides a more nuanced understanding of underlying data patterns, leading to more accurate predictions.

Inference time in Table II is critical for the practical deployment of machine learning models in live database environments, where response time can significantly impact user experience and system efficiency. FCN offers the best inference times for IMDb among the models tested, suggesting it as a viable option for environments where response time is more critical than absolute accuracy. Despite its better accuracy profile, MSCN provides competitive inference times, slightly outperforming FCN in speed. This balance of accuracy and efficiency makes MSCN a strong candidate for real-world applications. NeuroCard, while providing the best accuracy, also incurs the longest inference times. This trade-off highlights the computational cost associated with its deeper and more complex analytical approach.

VI. Conclusion and Future Work

The findings from this study underscore the potential of machine learning models to revolutionize cardinality estimation. The improved accuracy and adaptability of these models can significantly enhance query optimization processes, leading to faster and more efficient database systems. The choice between FCN, MSCN, and NeuroCard should consider specific application requirements, including the need for accuracy versus computational resource constraints and response time requirements. NeuroCard offers the best accuracy but at the cost of slower response times, making it suitable for environments where prediction accuracy is paramount. In contrast, FCN and MSCN offer a better balance, potentially more suitable for scenarios with stringent performance requirements.

Potential areas for further research include exploring hybrid models that combine the strengths of the individual approaches discussed, implementing these models in distributed database environments, and enhancing their ability to incrementally learn from new data without full retraining. The integration of machine learning into cardinality estimation is a promising development that could lead to substantial improvements in database management technology. Continued research and development in this field are essential to fully realize the potential of these advanced ML models.

References

[1] Anshuman Dutt, Chi Wang, Azade Nazi, Srikanth Kandula, Vivek R. Narasayya, and Surajit Chaudhuri. 2019. Selectivity Estimation for Range Predicates using Lightweight Models. Proc. VLDB Endow. 12, 9 (2019), 1044–1057. https://doi.org/ 10.14778/3329772.3329780

[2] Jie Liu, Wenqian Dong, Qingqing Zhou, and Dong Li. 2021. Fauce: fast and accurate deep ensembles with uncertainty for cardinality estimation. Proceedings of the VLDB Endowment 14, 11 (2021), 1950–1963.

[3] Parimarjan Negi, Ryan Marcus, Andreas Kipf, Hongzi Mao, Nesime Tatbul, Tim Kraska, and Mohammad Alizadeh. 2021. Flow-Loss: Learning Cardinality Estimates That Matter. arXiv preprint arXiv:2101.04964 (2021).

[4] Yongjoo Park, Shucheng Zhong, and Barzan Mozafari. 2020. QuickSel: Quick Selectivity Learning with Mixture Models. In Proceedings of the 2020 International Conference on Management of Data, SIGMOD Conference 2020, online conference [Portland, OR, USA], June 14-19, 2020, David Maier, Rachel Pottinger, AnHai Doan, Wang-Chiew Tan, Abdussalam Alawini, and Hung Q. Ngo (Eds.). ACM, 1017–1033. https://doi.org/10.1145/3318464.3389727

[5] Max Halford, Philippe Saint-Pierre, and Franck Morvan. 2019. An Approach Based on Bayesian Networks for Query Selectivity Estimation. In Database Systems for Advanced Applications - 24th International Conference, DASFAA 2019, Chiang Mai, Thailand, April 22-25, 2019, Proceedings, Part II (Lecture Notes in Computer Science, Vol. 11447), Guoliang Li, Jun Yang, João Gama, Juggapong Natwichai, and Yongxin Tong (Eds.). Springer, 3–19. https://doi.org/10.1007/978-3-030-18579-4_1

[6] Jennifer Ortiz, Magdalena Balazinska, Johannes Gehrke, and S. Sathiya Keerthi. 2019. An Empirical Analysis of Deep Learning for Cardinality Estimation. CoRR abs/1905.06425 (2019). arXiv:1905.06425 http://arxiv.org/abs/1905.06425

[7] C Chow and Cong Liu. 1968. Approximating discrete probability distributions with dependence trees. IEEE transactions on Information Theory 14, 3 (1968), 462–467.

[8] Lise Getoor, Benjamin Taskar, and Daphne Koller. 2001. Selectivity estimation using probabilistic models. In SIGMOD. 461–472.

[9] Joshua Spiegel and Neoklis Polyzotis. 2006. Graph-based synopses for relational selectivity estimation. In SIGMOD. 205–216.

[10] Kostas Tzoumas, Amol Deshpande, and Christian S Jensen. 2013. Efficiently adapting graphical models for selectivity estimation. The VLDB Journal 22, 1 (2013), 3–27.

[11] Dimitrios Gunopulos, George Kollios, Vassilis J Tsotras, and Carlotta Domeniconi. 2000. Approximating multi-dimensional aggregate range queries over real attributes. Acm Sigmod Record 29, 2 (2000), 463–474.

[12] Dimitrios Gunopulos, George Kollios, Vassilis J Tsotras, and Carlotta Domeniconi. 2005. Selectivity estimators for multidimensional range queries over real attributes. The VLDB Journal 14, 2 (2005), 137–154.

[13] Zongheng Yang, Amog Kamsetty, Sifei Luan, Eric Liang, Yan Duan, Peter Chen, and Ion Stoica. 2020. NeuroCard: One Cardinality Estimator for All Tables. Proc. VLDB Endow. 14, 1 (2020), 61–73. https://doi.org/10.14778/3421424.3421432

[14] Shohedul Hasan, Saravanan Thirumuruganathan, Jees Augustine, Nick Koudas, and Gautam Das. 2020. Deep Learning Models for Selectivity Estimation of Multi-Attribute Queries. In SIGMOD. 1035–1050.

[15] Andreas Kipf, Thomas Kipf, Bernhard Radke, Viktor Leis, Peter A. Boncz, and Alfons Kemper. 2019. Learned Cardinalities: Estimating Correlated Joins with Deep Learning. In 9th Biennial Conference on Innovative Data Systems Research, CIDR 2019, Asilomar, CA, USA, January 13-16, 2019, Online Proceedings. www.cidrdb.org. http://cidrdb.org/cidr2019/papers/p101-kipf-cidr19.pdf

[16] Kyongmin Kim, Jisung Jeong, In Seo, Wook-Shin Han, Kangwoo Choi, and Jaehyok Chong. 2022. Learned Cardinality Estimation: An In-depth Study. In Proceedings of the 2022 International Conference on Management of Data (SIGMOD '22), June 12–17, 2022, Philadelphia, PA, USA. ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3514221.3526154.

[17] https://github.com/postechdblab/learned-cardinality-estimation