On Efficient ML Model Training in Data Lakes

Wenbo Sun
Delft University of Technology
Delft, Netherlands
Supervisor: Rihan Hai
w.sun-2@tudelft.nl

ABSTRACT

In the era of big data, data lakes have become pivotal for storing vast amounts of diverse data types. These repositories are increasingly leveraged for machine learning (ML) applications, yet significant challenges remain, particularly in data integration and the efficient use of hardware resources. Traditional data preparation and integration processes are resource-intensive and time-consuming, creating bottlenecks that hinder ML training efficiency. This paper introduces novel methodologies to address these challenges by enabling direct ML training over columnar files, thus bypassing costly materialization processes. Additionally, it explores the use of GPUs to accelerate both data integration and ML training, enhancing overall efficiency. Our approach leverages factorized learning and matrix-represented metadata for effective ML training on columnar storage formats and employs autoencoders for multimedia data compression. This framework aims to significantly improve the speed and efficiency of ML model training within data lakes, paving the way for broader and more effective use in various domains.

VLDB Workshop Reference Format:
Wenbo Sun. On Efficient ML Model Training in Data Lakes. VLDB 2024 Workshop: VLDB Ph.D. Workshop.

1 INTRODUCTION

In the era of big data, the concept of a "data lake" [12, 21] has emerged as a pivotal architecture for storing vast amounts of unstructured and structured data. A data lake retains data in its native format, including files, multimedia, documents, and more, offering a scalable and flexible environment for data storage and analysis. As businesses continue to accumulate a wealth of raw data—from text documents and videos to complex tabular data—these repositories are increasingly recognized as invaluable resources for machine learning (ML) models aimed at driving digital transformation and enhancing business capabilities.

The trend of applying ML training over data lakes is gaining momentum [12, 24], particularly with the integration of classic ML models and advanced deep learning frameworks. These data lakes, often ubiquitous yet not always identified as such in both personal and business contexts, present a rich source for ML training. The raw data stored within, now commonly processed into embeddings, blur the traditional boundaries between different data formats. This transformation facilitates the advancement of deep learning and multimodal learning, harnessing the potential to revolutionize how businesses leverage their data assets.

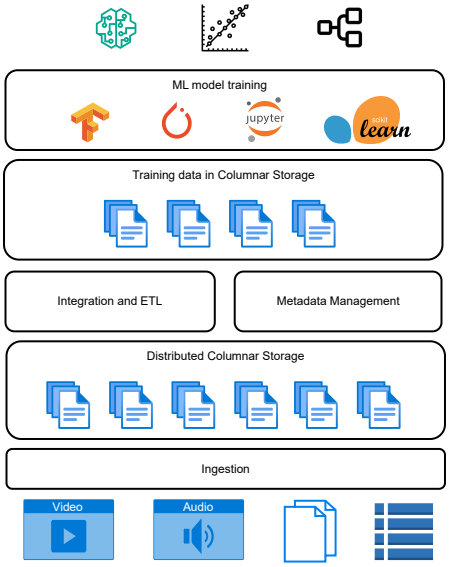

Figure 1: Workflow of ML model training in data lakes.

Despite these attractive opportunities, significant challenges impede the efficient utilization of data lakes for ML training. One of the most prominent issues is the high cost associated with data integration—preparing and unifying data from diverse sources for ML models. In a ML pipeline, substantial portion of resources is expended during data preparation phase [25] , overshadowing the actual model training efforts. The traditional paradigm necessitates a materialization process where data must be collected and related to create a unified view for training. This process, typically reliant on costly relational joins, involves intensive data access, movement, and replication. When dealing with large datasets, these operations become bottlenecks that severely limit the throughput of ML training tasks.

Furthermore, there exists a divergence in the hardware platforms used for data integration and ML training. While GPUs have become the standard infrastructure for modern ML computations, data lake operations are predominantly executed on CPU-based platforms [4–6]. This disparity necessitates frequent and intensive data transfers between heterogeneous hardware systems, further degrading the efficiency of ML training processes.

Addressing these challenges, this paper proposes novel methodologies for efficient ML training within data lakes, leveraging both the disparate and multimodal data in data lakes. We introduce an innovative approach that allows direct training over columnar files,

Proceedings of the VLDB Endowment. ISSN 2150-8097.

bypassing the need for costly data materialization. Additionally, we explore the utilization of GPUs not only for ML training but also for accelerating the data integration process, thereby minimizing the performance bottlenecks caused by hardware disparities.

This paper is structured to methodically explore these issues and solutions. We begin with a review of related work, highlighting the current methodologies and their limitations. Following this, we detail our problem statement and break it down into more specific research questions. In the section on preliminary solutions, we present our proposed architecture and methodology for training models directly over columnar files using GPU acceleration. This setup promises a significant enhancement in the efficiency and speed of ML model training within data lakes, setting the stage for transformative impacts across various domains.

2 RELATED WORK

This section introduces related works on ML model training over disparate data sources in data lakes. We categorize the relevant studies into two main categories: ML training over disparate tables and ML training over columnar files.

2.1 ML training over normalized tables

The integration of data preparation and machine learning (ML) training is a significant challenge due to the costly data integration processes in traditional ML pipelines. Various studies have aimed to address these inefficiencies by proposing methods to train models directly over normalized tables, thereby avoiding the expensive materialization required for table joins and data movement.

Factorized learning [8, 15, 18] methods have been explored to address this challenge. For instance, some approaches focus on Generalized Linear Models (GLMs), decomposing data into manageable parts for independent processing. However, these methods are limited to GLMs and do not extend to more complex models. Tree-based models [13] have also been adapted to work without materialization, but these often require specific modifications to the underlying algorithms, limiting their general applicability across different types of ML models.

Advancements in factorized learning have aimed to support feature interactions, extending applicability beyond simple linear relationships. However, these methods remain limited to linear or slightly non-linear models and struggle with highly non-linear models such as deep neural networks (DNNs). One recent study [9] show that factorized methods are constrained to neural networks with specific architectures, such as those using ReLU activation functions, highlighting significant limitations.

A notable gap in existing research is the lack of utilization of Graphics Processing Units (GPUs) to bridge the divide between data lake systems and ML training. GPUs can accelerate both data integration [19, 23] and ML training processes, but current methodologies have not fully leveraged this potential.

One promising approach [19] proposes a metadata representation that unifies relational joins and linear model training using linear algebra operations. This method enables the use of GPUs to accelerate the entire pipeline, including both data integration and ML training. However, the effectiveness of this approach is highly dependent on data characteristics, and the speedup is not consistent across different datasets.

Moreover, the single-table granularity of data organization poses additional challenges. In real-world scenarios, especially with large-scale tables, data is often stored in columnar file formats such as Parquet [3] and ORC [2]. Even if an ML model can be trained on a single table, the aggregation of data from these columnar files remains a necessary and costly step, especially when data is distributed across multiple storage systems.

Despite these efforts, significant gaps remain in integrating data lakes and ML training. Current solutions either do not effectively utilize GPUs or require extensive modifications to ML algorithms, limiting their usability and performance. Additionally, these approaches primarily focus on structured data, overlooking the demand for ML training with multimodal data in data lakes.

2.2 ML training over columnar storage

ML training over columnar storage [1–3] is an emerging field that integrates data processing and ML training. Torcharrow, for example, adapts tensor processing operators to columnar storage, enhancing data preprocessing speed and enabling joint optimization with ML training. However, these methods typically focus on individual columns rather than entire columnar files, leading to costly file aggregation steps.

Research has also explored accelerating ML training by leveraging columnar file structures. ColumnML [14] optimizes SGD for columnar storage by basing weight updates on features rather than rows. Features are partitioned, and the algorithm processes each partition before moving to the next, reducing intermediate data size and improving data locality. FPGA accelerators are introduced to streamline decompression and decoding stages, enhancing performance.

A recent study [20] proposes a modified Parquet file format for efficient ML training with sparse features. This method improves compression and I/O performance and introduces binary metadata to identify frequently used features, thus enhancing feature projection in ultra-wide tables. Unlike ColumnsML, this approach modifies the file format itself, making it more suitable for large-scale, data-centric workloads. However, changing the file format necessitates reprocessing all historical data, which makes its industrial applicability questionable.

3 RESEARCH QUESTIONS

In light of the challenges and existing methodologies for ML model training over disparate data sources in data lakes, this paper aims to address the following research questions:

- **How can ML models be efficiently trained directly over columnar files without the need for costly data materialization processes?**
  - This question investigates the feasibility of bypassing the traditional materialization step, which involves expensive relational joins and data movement. It explores methods that enable direct ML model training on data stored in columnar file formats such as Parquet and ORC.

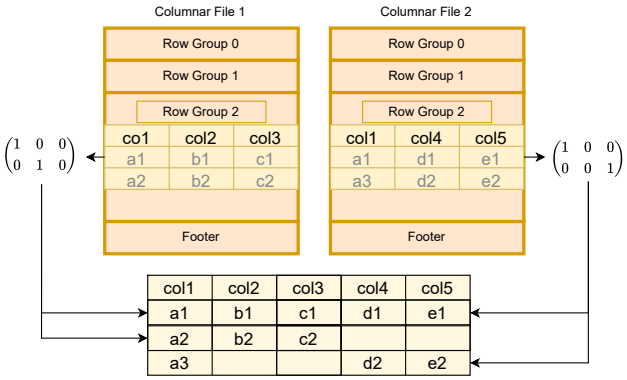

Figure 2: Illustration of how data in columnar files can be mapped to a virtual target table using matrix-represented metadata.

- **What modifications to file formats, such as Parquet, can make columnar storage more compatible with training a wide range of ML models?**
  - This question focuses on the design and implementation of optimized columnar file structures and compression methods that enhance compatibility with the training processes of popular ML models, such as neural networks, while maintaining comparable compression ratios.
- **What techniques can be employed to leverage GPU acceleration for both data integration and ML training tasks within data lakes?**
  - This question explores the utilization of GPUs not only for ML model training but also for accelerating the data integration process, addressing performance bottlenecks caused by data movement between heterogeneous devices.

By addressing these research questions, this paper aims to contribute to the development of more efficient and scalable ML training methodologies over columnar storage in data lakes.

4 PRELIMINARY SOLUTION

This section presents preliminary solutions to the research questions outlined in Section 3. The proposed solutions serve as a middleware that bridges data lakes and ML systems. Our objective in conceptualizing this middleware is to avoid the impracticality of introducing an entirely new columnar file format, which would necessitate the conversion of all historical data by practitioners seeking efficient ML training in data lakes. Conversely, developing new algorithms compatible with existing columnar storage formats would require extensive modifications to current ML training algorithms, resulting in a substantial workload. Therefore, our approach aims to balance innovation and practicality, facilitating efficient ML training without imposing excessive burdens on practitioners.

4.1 ML training over Parquet files

The core of training models over columnar files lies in understanding the relationships between these files. Similar to factorized learning over tabular data, which describes the relationships between

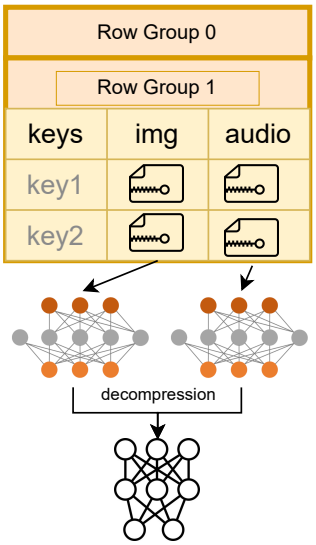

Figure 3: Autoencoders can be utilized as compression algorithms for multimedia data in place of traditional lossless algorithms. By stacking autoencoders with subsequent models, it is possible to enable multimodal model training over columnar files.

schemas and records in normalized tables, factorized learning can push ML training algorithms to operate on separated tables. Thus, if we comprehend the relationships between files, we can further push the training algorithms directly to the files without converting them into tables.

A notable study [19] proposes a matrix-represented data integration metadata, where schema and entity mappings between normalized tables are represented by binary sparse matrices. Specifically, suppose the target table after materialization is T, the schema mapping matrix M specifies how the columns of S are mapped to the columns of T.

Definition 4.1 (Mapping matrix).

$$M[i,j] = \begin{cases} 1, & \text{if the } j\text{-th column of } S \text{ is mapped to the } i\text{-th column of } T \\ 0, & \text{otherwise} \end{cases}$$

Similarly, the indicator matrix I can preserve row matching between the source table S and the target table T.

Definition 4.2 (Indicator matrix [8]).

$$I[i,j] = \begin{cases} 1, & \text{if the } j\text{-th row of } S \text{ is mapped to the } i\text{-th row of } T \\ 0, & \text{otherwise} \end{cases}$$

In Fig. 2, we take Parquet files as an example. Based on this representation, we can further derive that if the meaning of matrices M and I is modified to reflect how columns and row groups are mapped to the target table T, the operators in ML models can be directly applied to columns and row groups in Parquet files. More importantly, with this representation, the relationships between rows are now extended to the content within row groups, which are not necessarily tabular data. This modification facilitates multimodal learning over Parquet files, significantly enhancing usability compared to factorized learning, which is limited to tabular data.

4.2 Autoencoder-based compression

Another obstacle in integrating columnar files with ML training arises from the necessity of compression during storage. Compression and decompression algorithms often do not preserve semantics [11, 17], posing a significant challenge, particularly when dealing with multimedia data such as videos and audios. For instance, modern computer vision models [7, 16] typically take image patches, represented by matrices, as input. However, compressed data in traditional columnar files cannot maintain the semantic integrity of image patches. As a result, directly applying ML training over compressed files is not feasible. Furthermore, decompression becomes a significant overhead, especially for large multimedia datasets. Therefore, the endeavor to train models over Parquet files also demands the development of semantic-preserving compression algorithms.

One potential solution, as illustrated in Fig. 3, is to utilize autoencoders as compression algorithms [10, 26] for multimedia data in data lakes. The nature of data compression achieved by autoencoders has been extensively explored in computer vision domains. Autoencoders use a lower-dimensional hidden space to represent raw data. Although autoencoders are inherently lossy compression algorithms, considering the general robustness of neural network-based models, the degradation in effectiveness may be acceptable. However, it is essential to empirically evaluate the extent of loss and establish mathematical frameworks to determine error bounds associated with this approach.

4.3 GPU-accelerated ML pipeline

Expanding upon the preliminary solutions discussed in Sections 4.1 and 4.1, we can envision a pathway where the ML pipeline, from Parquet file to model training, leverages modern hardware for integrated acceleration. The matrix-represented metadata transforms file operations into linear algebraic operations. Subsequently, after locating the data in specific locations within each file, autoencoders can be employed for decompression. Due to the linear algebraic nature of operations involved in handling files, autoencoders, and subsequent model training, leveraging GPUs becomes advantageous. Consequently, the entire pipeline can now be adapted to harness the computational power of GPUs. While some works [14, 22] have explored external accelerators to expedite the decompression stage of columnar files, these methods typically require specifically designed hardware or circuits rather than utilizing general-purpose computing units. Furthermore, the autoencoder itself can be stacked with other deep learning models, opening up additional opportunities to discover more effective autoencoders for compression while simultaneously mitigating loss incurred during compression.

5 CONCLUSION

This paper has explored the challenges and potential solutions related to ML model training over columnar files in data lakes. Through our investigation, we have proposed preliminary solutions, including leveraging factorized learning methodologies and matrix-represented metadata to facilitate direct ML training over columnar files. Additionally, we explored the potential of using autoencoders as compression algorithms for multimedia data in data lakes, thereby enabling the integration of modern hardware acceleration, particularly GPUs, into the ML pipeline.

Looking ahead, our aim is to consolidate these solutions into a cohesive middleware. This middleware will seamlessly integrate with existing data lakes and ML frameworks, providing a robust platform for efficient and scalable ML model training. By combining these solutions, we envision a future where ML practitioners can harness the full potential of columnar files in data lakes while leveraging the power of modern hardware acceleration

REFERENCES

[1] 2009. Apache Avro. https://avro.apache.org/. Accessed: 2024-05-29.
[2] 2013. Apache ORC. https://orc.apache.org/. Accessed: 2024-05-29.
[3] 2013. Apache Parquet. https://parquet.apache.org/. Accessed: 2024-05-29.
[4] 2017. Apache Iceberg. https://iceberg.apache.org/. Accessed: 2024-05-29.
[5] 2019. Apache Hudi. https://hudi.apache.org/. Accessed: 2024-05-29.
[6] Michael Armbrust, Tathagata Das, Liwen Sun, et al. 2020. Delta lake: high-performance ACID table storage over cloud object stores. *Proc. VLDB Endow.* 13, 12 (aug 2020), 3411–3424.
[7] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. 2020. Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934* (2020).
[8] L. Chen, A. Kumar, J. Naughton, and J. M. Patel. 2017. Towards Linear Algebra over Normalized Data. *PVLDB* 10, 11 (2017).
[9] Zhaoyue Cheng, Nick Koudas, Zhe Zhang, et al. 2021. Efficient Construction of Nonlinear Models over Normalized Data. 1140–1151.
[10] Yoojin Choi, Mostafa El-Khamy, and Jungwon Lee. 2019. Variable Rate Deep Image Compression With a Conditional Autoencoder. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.
[11] Patrick Damme, Dirk Habich, Juliana Hildebrandt, et al. 2017. Lightweight Data Compression Algorithms: An Experimental Survey (Experiments and Analyses). In *Proceedings of the 20th EDBT 2017*. 72–83.
[12] Rihan Hai, Christos Koutras, Christoph Quix, and Matthias Jarke. 2023. Data Lakes: A Survey of Functions and Systems. *IEEE Transactions on Knowledge and Data Engineering* 35, 12 (2023), 12571–12590.
[13] Zezhou Huang, Rathijit Sen, Jiaxiang Liu, and Eugene Wu. 2023. JoinBoost: Grow Trees over Normalized Data Using Only SQL. *Proc. VLDB Endow.* 16, 11 (jul 2023), 3071–3084.
[14] Kaan Kara, Ken Eguro, Ce Zhang, and Gustavo Alonso. [n.d.]. ColumnML: Column-Store Machine Learning with On-The-Fly Data Transformation. *Proceedings of the VLDB Endowment* 12, 4 ([n. d.]).
[15] Mahmoud Abo Khamis, Hung Q. Ngo, XuanLong Nguyen, Dan Olteanu, and Maximilian Schleich. 2018. AC/DC: In-Database Learning Thunderstruck. In *Proceedings of the Second Workshop on Data Management for End-To-End Machine Learning*. ACM, 1–10.
[16] A. Kirillov, E. Mintun, N. Ravi, et al. 2023. Segment Anything. In *2023 IEEE/CVF ICCV*. 3992–4003.
[17] Maximilian Kuschewski, David Sauerwein, Adnan Alhomssi, and Viktor Leis. 2023. BtrBlocks: Efficient Columnar Compression for Data Lakes. *Proc. ACM Manag. Data* 1, 2, Article 118 (jun 2023), 26 pages.
[18] Side Li, Lingjiao Chen, and Arun Kumar. 2019. Enabling and Optimizing Non-Linear Feature Interactions in Factorized Linear Algebra. In *SIGMOD*. 1571–1588.
[19] Ziyu Li, Wenbo Sun, Danning Zhan, Yan Kang, Lydia Chen, Alessandro Bozzon, and Rihan Hai. 2024. Amalur: Data Integration Meets Machine Learning. *IEEE Transactions on Knowledge and Data Engineering* (2024), 1–14.
[20] Gang Liao, Ye Liu, Jianjun Chen, and Daniel J Abadi. 2024. Bullion: A Column Store for Machine Learning. *arXiv preprint arXiv:2404.08901* (2024).
[21] Fatemeh Nargesian, Erkang Zhu, Renée J Miller, Ken Q Pu, and Patricia C Arocena. 2019. Data lake management: challenges and opportunities. *Proceedings of the VLDB Endowment* 12, 12 (2019), 1986–1989.
[22] David Sidler, Zsolt Istvan, Muhsen Owaida, Kaan Kara, and Gustavo Alonso. 2017. doppioDB: A Hardware Accelerated Database. In *Proceedings of the SIGMOD 2017*. Association for Computing Machinery, 1659–1662.
[23] Wenbo Sun, Asterios Katsifodimos, and Rihan Hai. 2023. Accelerating Machine Learning Queries with Linear Algebra Query Processing. In *Proceedings of the 35th SSDBM*. Association for Computing Machinery, Article 13, 12 pages.
[24] Merlinda Wibowo, Sarina Sulaiman, and Siti Mariyam Shamsuddin. 2017. Machine Learning in Data Lake for Combining Data Silos. In *Data Mining and Big Data*. 294–306.
[25] Doris Xin, Hui Miao, Aditya Parameswaran, and Neoklis Polyzotis. 2021. Production Machine Learning Pipelines: Empirical Analysis and Optimization Opportunities. In *Proceedings of the 2021 SIGMOD*. 2639–2652.
[26] Lei Zhou, Chunlei Cai, Yue Gao, et al. 2018. Variational Autoencoder for Low Bit-rate Image Compression. In *Proceedings of the CVPR Workshops*.