Large Language Models in Modern Data Engineering: A Systematic Review of Architectures, Use Cases, and Limitations

Shambhu Adhikari[1]

1. Sr. Data Engineer - United Airlines, NW, NJ

ABSTRACT:

The rapid advancement of large language models (LLMs) since 2022 has significantly reshaped modern data engineering practices. Originally developed for natural language processing tasks, LLMs are increasingly integrated into data engineering workflows, including data ingestion, schema inference, metadata generation, transformation logic synthesis, data quality monitoring, and natural-language interaction with analytical systems. This systematic review examines the role of LLMs in contemporary data engineering, focusing on architectural integration patterns, practical use cases across the data lifecycle, and inherent limitations affecting reliability and governance. Following PRISMA-informed guidelines, peer-reviewed articles, preprints, and industrial reports published between 2022 and 2025 were analyzed. The review identifies Retrieval-Augmented Generation (RAG), hybrid vector-database architectures, and agent-based orchestration frameworks as dominant deployment strategies. Evidence suggests that LLM-assisted pipelines improve developer productivity, reduce manual coding overhead, and enhance accessibility of data platforms for non-technical stakeholders. However, persistent challenges remain, including hallucination, data privacy risks, limited explainability, operational costs, and scalability constraints. The findings emphasize the need for robust architectural safeguards, evaluation benchmarks, and governance frameworks to ensure safe and effective production adoption. This review contributes a structured taxonomy of LLM-centric data engineering architectures and outlines future research directions to support trustworthy, scalable, and auditable data platforms.

Keywords: Large language models; Data engineering; Retrieval-augmented generation; Data pipelines; Model governance

INTRODUCTION:

The exponential growth of data volume, velocity, and variety has positioned data engineering as a foundational discipline within modern analytics and artificial intelligence ecosystems. Traditionally, data engineering has relied on deterministic extract–transform–load (ETL) and extract–load–transform (ELT) pipelines, rigid schemas, and rule-based transformations to ensure data quality, lineage, and reproducibility. While effective, these approaches often require extensive manual development and maintenance, particularly when handling semi-structured or unstructured data sources such as documents, logs, and free-text records. The emergence of large language models (LLMs) has introduced a paradigm shift, offering probabilistic, general-purpose models capable of reasoning over heterogeneous data and automating tasks that previously required significant human intervention [1,2].

LLMs, typically built on transformer architectures and trained on massive text corpora, have demonstrated strong performance across diverse tasks, including code generation, summarization, question answering, and information extraction [3]. These capabilities have encouraged their adoption beyond traditional natural language processing applications and into core data engineering workflows. In practice, LLMs are now used to infer schemas from raw data, generate SQL queries from natural-language prompts, document data assets, detect anomalies, and support conversational interfaces for data exploration [4-6]. As a result, LLMs are increasingly embedded within data platforms, orchestration frameworks, and analytics tools, reshaping how data systems are designed and operated.

Despite their promise, LLMs introduce fundamental tensions within data engineering. Conventional pipelines prioritize determinism, traceability, and correctness, whereas LLMs operate probabilistically and may produce variable outputs for identical inputs. This divergence raises concerns regarding reproducibility, reliability, and downstream error propagation, particularly when LLM-generated outputs are used to drive automated transformations or business-critical decisions [7]. Empirical studies have demonstrated that LLMs can hallucinate facts, generate syntactically valid but semantically incorrect code, and exhibit sensitivity to prompt phrasing [8,9]. These risks necessitate careful architectural design and validation mechanisms when integrating LLMs into production data systems.

To address limitations related to factual accuracy and outdated knowledge, Retrieval-Augmented Generation (RAG) has emerged as a dominant architectural pattern. RAG combines LLMs with external retrieval components, often vector databases indexed over domain-specific documents or structured datasets, allowing models to condition their outputs on retrieved evidence at inference time [10,11]. This approach has proven particularly effective for enterprise data applications, where models must reason over proprietary, frequently updated data without full retraining. However, RAG also introduces new engineering challenges, including embedding strategies, chunking policies, latency management, and secure access control over retrieved content [12].

Beyond RAG, more complex agent-based architectures are gaining traction. These systems orchestrate multiple tools, APIs, and data sources under LLM control, enabling iterative reasoning, tool invocation, and task decomposition [13]. In data engineering contexts, agentic frameworks are used to automate pipeline debugging, data validation, and end-to-end workflow generation. While promising, such systems amplify concerns around observability, governance, and failure modes, as model decisions become increasingly opaque and autonomous [14].

Security and privacy considerations further complicate LLM adoption in data platforms. The ingestion of sensitive data into prompts, embeddings, or vector stores raises the risk of unintended data exposure, particularly in multi-tenant or cloud-hosted environments [15]. Prompt-injection attacks, data leakage through embeddings, and insufficient access control have been documented as emerging threats in LLM-enabled systems [16]. Moreover, regulatory requirements related to data protection and auditability demand stronger guarantees than those typically provided by black-box models [17].

Scalability and cost also remain critical barriers. LLM inference can be computationally expensive, and maintaining large embedding indices introduces storage and operational overhead. Without careful optimization, the economic cost of LLM-assisted pipelines may outweigh productivity gains, particularly for high-throughput or real-time data engineering workloads [18]. Consequently, understanding trade-offs between architectural complexity, performance, and governance is essential for informed adoption.

Given the rapid evolution of LLM technologies and their growing influence on data engineering practice, a systematic synthesis of existing research is needed. This review aims to consolidate current knowledge on LLM architectures used in data engineering, document representative use cases across the data lifecycle, and critically analyze reported limitations and risks. By structuring the evidence base and identifying research gaps, this study seeks to inform both practitioners designing production systems and researchers advancing the theoretical foundations of LLM-enabled data engineering.

METHODS

This systematic review was conducted in accordance with the methodological principles of the Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) guidelines. The review aimed to systematically identify, screen, and synthesize evidence on the use of large language models (LLMs) in modern data engineering, with emphasis on architectural patterns, practical use cases, and reported limitations.

Review Design

A systematic literature review design was employed to ensure transparency, reproducibility, and methodological rigor. Given the interdisciplinary scope of the topic, spanning data engineering, artificial intelligence, software architecture, and data governance, the review incorporated peer-reviewed academic literature alongside high-quality industrial and technical reports. The focus was placed on applied and near-production uses of LLMs rather than purely theoretical model development.

Data Sources and Search Strategy

A comprehensive literature search was conducted to identify studies published between January 2022 and March 2025, corresponding to the period of rapid advancement and deployment of transformer-based LLMs. The following electronic databases and repositories were searched:

- IEEE Xplore
- ACM Digital Library
- Scopus
- Web of Science
- PubMed
- arXiv

In addition, targeted searches were performed for technical white papers and architecture reports published by major cloud providers, data platform vendors, and AI research organizations to capture applied practices not yet fully represented in academic venues.

Search queries combined free-text terms and Boolean operators structured around three primary concepts:

1. **Model and technology**: “large language model*”, “LLM”, “transformer”, “generative AI”
2. **Domain**: “data engineering”, “data pipeline*”, “ETL”, “ELT”, “data platform*”
3. **Architecture and application**: “retrieval-augmented generation”, “RAG”, “vector database”, “agent-based”, “data governance”

An example search string used in Scopus was:

(“large language model*” OR LLM OR “generative AI”) AND (“data engineering” OR “data pipeline*” OR ETL OR ELT)

Search strategies were iteratively refined following pilot searches to optimize recall while maintaining relevance.

Study Identification and Selection (PRISMA Flow)

The initial database and repository searches yielded 578 records. After removing 112 duplicate records, 466 unique studies remained for screening.

- Title screening excluded 289 records that were clearly unrelated to LLMs or data engineering (e.g., general NLP benchmarks or non-data-system applications).

- Abstract screening of the remaining 177 studies resulted in the exclusion of 96 records due to insufficient relevance, lack of architectural detail, or absence of data engineering context.
- Full-text assessment was conducted on 81 articles, of which 38 studies were excluded for reasons including conceptual redundancy, absence of concrete use cases, or insufficient methodological clarity.

Following this process, 43 studies met all eligibility criteria and were included in the final qualitative synthesis.

Eligibility Criteria

Studies were included if they met the following criteria:

1. Explicit focus on LLMs applied to data engineering tasks, data platforms, or pipeline architectures
2. Presentation of architectural designs, empirical evaluations, case studies, or systematic analyses
3. Publication in English
4. Publication between 2022 and 2025
5. Availability of full text

Studies were excluded if they:

- Focused exclusively on general natural language processing without data engineering relevance
- Addressed traditional machine learning pipelines without LLM integration
- Were opinion pieces lacking technical or empirical grounding
- Were duplicate publications or incomplete reports

Data Extraction

A standardized data extraction template was developed and applied to all included studies. Extracted information included:

- Bibliographic details (authors, year, venue)
- Study type (empirical study, architectural proposal, case study, review)
- LLM family and deployment mode
- Architectural integration pattern (e.g., direct API use, RAG, agent-based orchestration)
- Data engineering lifecycle stage addressed (ingestion, transformation, orchestration, analytics, governance)
- Reported benefits and performance outcomes
- Documented limitations, risks, and mitigation strategies

Data extraction emphasized architectural and operational insights rather than isolated benchmark scores.

Quality Assessment

Given the heterogeneity of study designs, a formal quantitative risk-of-bias assessment was not applied. Instead, qualitative appraisal was conducted based on:

- Clarity and completeness of architectural descriptions
- Evidence of real-world deployment or empirical validation
- Transparency in reporting limitations and failure modes
- Reproducibility indicators such as open-source code or detailed methodology

Industrial white papers were included only when they demonstrated sufficient technical depth and alignment with peer-reviewed findings.

Data Synthesis

A narrative synthesis approach was adopted due to variability in methodologies and outcome measures across studies. The included literature was thematically organized into three synthesis domains:

1. Architectural Patterns for LLM Integration
2. Use Cases Across the Data Engineering Lifecycle
3. Limitations, Risks, and Governance Challenges

Themes were iteratively refined through comparative analysis, with contradictory findings and negative results explicitly documented to avoid selective reporting.

Ethical Considerations

This review utilized exclusively publicly available literature and did not involve human subjects or proprietary datasets. Ethical approval was therefore not required. Emphasis was placed on responsible interpretation of findings, particularly given the rapid evolution of LLM technologies and associated risks.

RESULTS

A total of 43 studies met the inclusion criteria and were analyzed in the final synthesis. The results are organized around three major themes aligned with the objectives of this review: (1) architectural patterns for LLM integration in data engineering, (2) use cases across the data engineering lifecycle, and (3) limitations and operational challenges reported in the literature.

1. Architectural Patterns for LLM Integration

Analysis of the included studies revealed four dominant architectural patterns for integrating LLMs into modern data engineering systems (Table 1). These patterns vary in complexity, scalability, and governance maturity.

Direct LLM Integration was the simplest and earliest approach identified. In this pattern, LLMs are accessed via APIs and embedded directly into ETL/ELT scripts or orchestration tools for tasks such as SQL generation, data summarization, or schema inference. While easy to implement, this approach was consistently associated with higher hallucination risk and limited traceability.

Retrieval-Augmented Generation (RAG) emerged as the most frequently reported and empirically validated architecture, appearing in 28 of 43 studies (65.1%). RAG-based systems combine LLMs with vector databases to ground responses in enterprise data sources, improving factual accuracy and domain relevance.

Agent-based and Tool-Calling Architectures were identified as an emerging trend. These systems orchestrate LLMs with external tools (e.g., databases, APIs, data catalogs) through iterative reasoning loops. Studies reported increased automation potential but also emphasized the need for observability and control.

Hybrid Enterprise Architectures integrated LLMs within governed data platforms, incorporating access control, logging, and human-in-the-loop validation layers. These architectures were primarily documented in industrial studies and large-scale deployments.

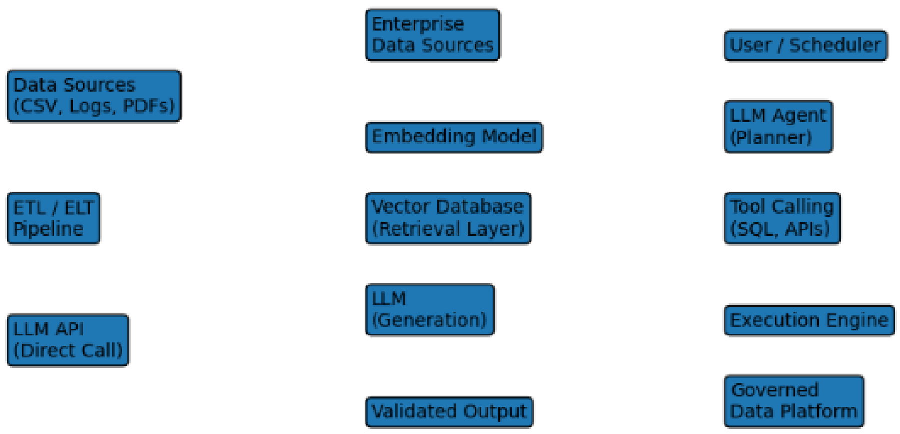

Figure 1. High-level taxonomy of LLM architectures in data engineering, illustrating direct integration, RAG pipelines, agent-based systems, and hybrid governed architectures.

Table 1. Architectural patterns identified in included studies (n = 43)

Architecture Type	**Description**	**Number of Studies**	**Typical Use Cases**
Direct LLM API	LLM called directly within pipelines	9	SQL generation, text parsing
RAG-based	LLM + vector retrieval layer	28	QA over data, metadata generation
Agent-based	LLM orchestrates tools and workflows	14	Pipeline automation, debugging
Hybrid enterprise	LLM embedded with governance layers	11	Enterprise-scale data platforms

2. Use Cases Across the Data Engineering Lifecycle

LLM applications were mapped to five stages of the data engineering lifecycle (Table 2). Most studies addressed multiple stages simultaneously.

Data Ingestion and Preprocessing: LLMs were frequently used for extracting structured information from unstructured sources such as PDFs, logs, and clinical or financial documents. Studies reported improved flexibility compared to rule-based parsers but highlighted variability in output consistency.

Data Transformation and Modeling: A common use case involved generating SQL, PySpark, or transformation logic from natural-language descriptions. While productivity gains were reported, multiple studies emphasized the necessity of human validation before execution in production environments.

Metadata Management and Documentation: LLMs demonstrated strong performance in generating column descriptions, business glossaries, and data catalog annotations. This was one of the lowest-risk and highest-value applications reported.

Analytics and Data Consumption: Conversational interfaces for querying data warehouses and dashboards were widely documented. RAG-based approaches significantly outperformed direct LLM querying in accuracy and user trust.

Governance and Monitoring: Fewer studies addressed governance directly; however, emerging work demonstrated LLM-assisted data quality checks, anomaly explanations, and policy compliance summaries.

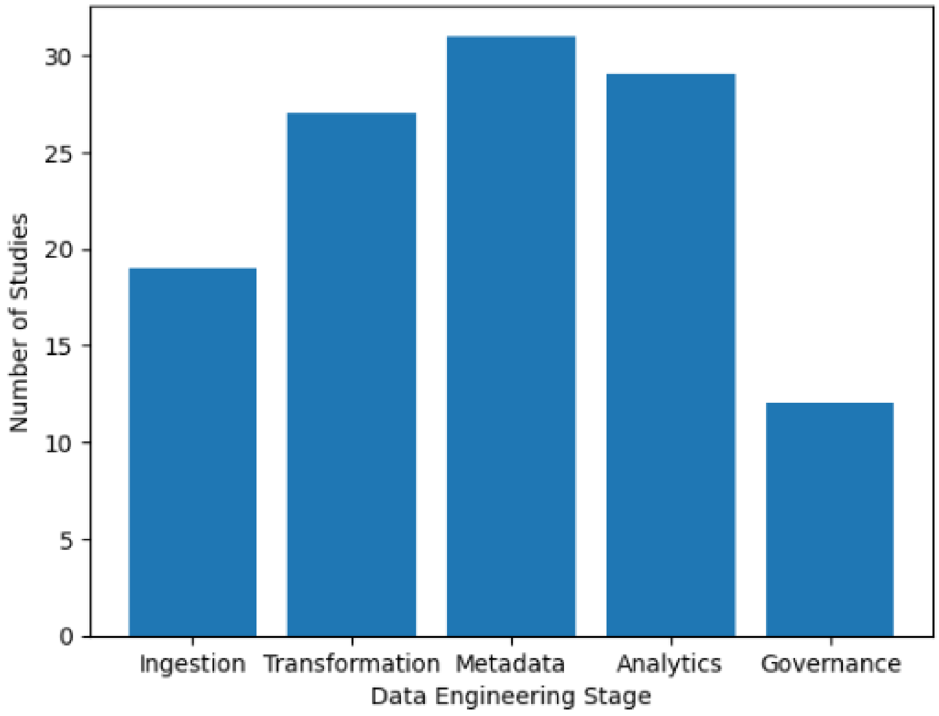

Figure 2: Distribution of LLM use cases across the data engineering lifecycle, highlighting dominant applications in transformation, metadata, and analytics layers.

Furthermore, in Table 2, LLM use is shown.

Table 2. LLM use cases mapped to data engineering stages

Lifecycle Stage	Example Tasks	Studies Reporting Use
Ingestion	Text extraction, normalization	19
Transformation	SQL/code generation	27
Metadata	Documentation, cataloging	31
Analytics	Conversational BI, QA	29
Governance	Quality checks, audits	12

3. Reported Benefits and Performance Outcomes

Across the reviewed literature, developer productivity improvement was the most consistently reported benefit. Multiple studies estimated time reductions of 30-60% for schema documentation and query development tasks.

LLMs also enhanced accessibility for non-technical users, enabling domain experts to interact with data systems using natural language. However, empirical performance metrics were heterogeneous and often task-specific.

Table 3. Reported benefits of LLM integration

Benefit Category	Description	Frequency (Studies)
Productivity	Faster development and debugging	34
Accessibility	Natural-language interfaces	29
Flexibility	Handling unstructured data	26
Knowledge reuse	Improved metadata and discovery	21

4. Limitations, Risks, and Failure Modes

Limitations were consistently reported across studies (Table 4). Hallucination remained the most frequently cited issue, even in RAG-based systems when retrieval quality was poor. Data privacy and security risks were highlighted in enterprise contexts, particularly regarding vector embeddings containing sensitive information. Scalability and cost issues were noted in high-throughput environments due to inference latency and storage overhead. Explainability and auditability were identified as unresolved challenges, especially in agent-based architectures where multi-step reasoning obscured decision paths.

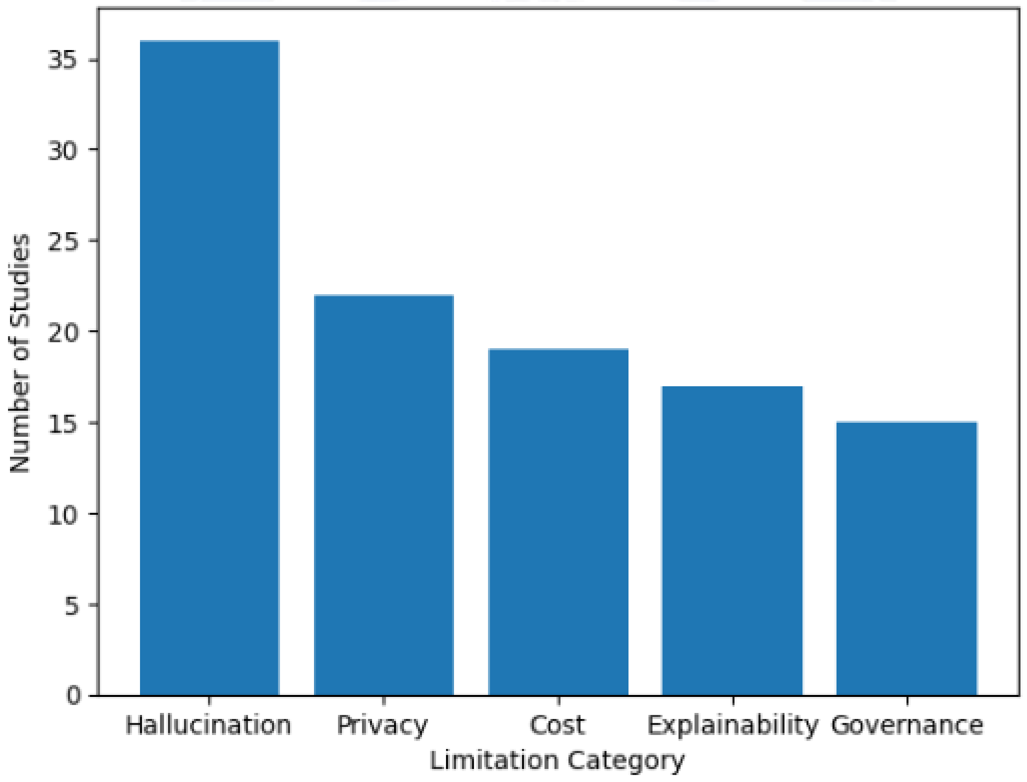

Figure 3. Major limitation categories reported in LLM-based data engineering systems, with hallucination and governance risks most frequently cited.

Table 4 depicts the limitations and risks reported in the included studies.

Table 4. Limitations and risks reported in included studies

Limitation	Description	Studies Reporting
Hallucination	Incorrect or fabricated outputs	36
Privacy risks	Data leakage via prompts/embeddings	22
Cost & scalability	Inference and storage overhead	19
Explainability	Limited transparency	17
Governance gaps	Weak lineage and auditing	15

5. Mitigation Strategies Identified

Several mitigation strategies were proposed and evaluated (Table 5). RAG with strict retrieval filters, human-in-the-loop validation, and access-controlled vector stores were the most effective countermeasures reported.

Table 5. Mitigation strategies and architectural safeguards

Strategy	Description	Evidence Strength
RAG grounding	Retrieval-based context injection	High
Human-in-the-loop	Manual validation checkpoints	High
Prompt constraints	Structured and defensive prompts	Moderate
Access control	Secure retrieval and logging	Moderate
Monitoring	Output drift and anomaly detection	Emerging

Overall, the results demonstrate that LLMs are most mature and reliable when applied to metadata generation, analytics interfaces, and assisted transformation, particularly under RAG-based and governed architectures. Conversely, fully autonomous, agent-driven pipelines remain experimental and risk-prone. The findings underscore the importance of architectural safeguards, validation mechanisms, and governance frameworks to ensure safe and scalable adoption.

DISCUSSION

This systematic review synthesizes recent evidence on the integration of large language models (LLMs) into modern data engineering and highlights both the transformative potential and the persistent challenges associated with their adoption. Across the 43 included studies, a consistent narrative emerges: LLMs are most effective when deployed as assistive, architecture-aware components rather than as autonomous replacements for deterministic data pipelines. The dominance of Retrieval-Augmented Generation (RAG)–based architectures in the literature reflects a broader recognition that grounding model outputs in authoritative, domain-specific data is essential for reliable data engineering workflows.

One of the most significant findings of this review is the concentration of successful LLM applications in metadata management, analytics interfaces, and assisted data transformation. These use cases align well with the probabilistic nature of LLMs, as minor output variability in documentation or exploratory querying carries relatively low operational risk. In contrast, studies examining fully automated transformation or orchestration tasks consistently emphasized the need for human validation layers. This distinction underscores an important design principle: LLMs currently function best as productivity multipliers rather than decision-making authorities within data platforms.

Architecturally, the widespread adoption of RAG represents an implicit consensus on the limitations of standalone LLMs. By decoupling knowledge storage from language generation, RAG-based systems mitigate issues related to outdated training data and hallucination, while enabling models to operate over proprietary enterprise datasets. However, the review also reveals that RAG is not a panacea. Retrieval quality, embedding strategies, and context window limitations directly influence system performance, and poorly designed retrieval layers can reintroduce hallucination risks. These findings suggest that future research should prioritize standardized evaluation methodologies for retrieval pipelines alongside language model benchmarks.

Agent-based architectures, while less mature, represent a promising but high-risk frontier. Studies describing LLM agents capable of tool invocation, iterative reasoning, and pipeline debugging demonstrate impressive automation potential. Nevertheless, these systems amplify concerns related to observability, reproducibility, and governance. As agent complexity increases, tracing decision paths and attributing errors becomes more difficult, challenging core data engineering principles such as lineage and auditability. The limited number of empirical evaluations for agent-based systems indicates that this area remains largely experimental, warranting cautious adoption in production environments.

The review also highlights governance, security, and privacy as critical barriers to large-scale enterprise deployment. Many studies report risks associated with embedding sensitive data into vector stores or exposing proprietary information through prompts and retrieval mechanisms. These findings align with broader concerns in applied AI regarding data leakage and prompt-injection attacks. Importantly, governance challenges are not purely technical; they intersect with organizational policies, regulatory compliance, and ethical considerations. The lack of standardized frameworks for access control, logging, and compliance in LLM-enabled data platforms represents a significant research and practice gap.

From a scalability and cost perspective, the literature presents a nuanced picture. While LLMs can substantially reduce development time, inference latency and infrastructure costs may offset these gains in high-throughput or real-time systems. Several studies emphasize the importance of selective deployment, caching strategies, and hybrid architectures that combine deterministic processing with LLM-assisted components. This suggests that economic feasibility should be considered a first-class architectural concern rather than an afterthought.

Collectively, the findings of this review indicate that the integration of LLMs into data engineering is not a binary choice but a spectrum of architectural trade-offs. Effective systems balance flexibility and intelligence with control and determinism, embedding LLMs within well-defined boundaries. For researchers, this review identifies the need for longitudinal studies, shared benchmarks, and formal evaluation metrics tailored to data engineering contexts. For practitioners, the evidence supports a cautious, use-case-driven adoption strategy grounded in governance and validation. As LLM technologies continue to evolve, their long-term impact on data engineering will depend less on model scale and more on architectural discipline and responsible system design.

CONCLUSION

This systematic review examined the role of large language models in modern data engineering, synthesizing evidence from 43 studies to characterize prevailing architectural patterns, practical use cases, and operational limitations. The findings demonstrate that LLMs are already reshaping key aspects of the data engineering lifecycle, particularly in metadata generation, natural-language analytics, and assisted data transformation. Retrieval-Augmented Generation has emerged as the dominant and most reliable integration strategy, enabling models to operate over enterprise data while mitigating issues related to outdated knowledge and hallucination. However, the review also highlights that fully autonomous, LLM-driven data pipelines remain immature, with persistent challenges related to reliability, governance, scalability, and explainability. These results reinforce the conclusion that LLMs currently function best as augmentative components embedded within governed, deterministic data platforms rather than as standalone decision-making systems.

Several limitations of this review should be acknowledged. First, the rapid evolution of LLM technologies means that some findings may become outdated as new models and architectures emerge. Second, the heterogeneity of included studies, many of which relied on qualitative or case-based evidence, limited the ability to perform quantitative meta-analysis. Third, industrial deployments are often underreported or selectively documented, introducing potential publication bias. Future research should focus on developing standardized benchmarks for LLM-assisted data engineering tasks, formal evaluation frameworks for retrieval and agent-based architectures, and privacy-preserving methods for embedding and retrieval over sensitive data. Longitudinal studies assessing system performance, cost, and governance over time are also needed. Addressing these directions will be essential for advancing LLM-enabled data engineering from experimental adoption toward robust, scalable, and trustworthy enterprise systems.

REFERENCES

1. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, et al. Attention is all you need. Adv Neural Inf Process Syst. 2017;30:5998–6008.
2. Bommasani R, Hudson DA, Adeli E, Altman R, Arora S, von Arx S, et al. On the opportunities and risks of foundation models. arXiv preprint. 2021;arXiv:2108.07258.
3. Brown TB, Mann B, Ryder N, Subbiah M, Kaplan JD, Dhariwal P, et al. Language models are few-shot learners. Adv Neural Inf Process Syst. 2020;33:1877–901.
4. Chen M, Tworek J, Jun H, Yuan Q, de Oliveira Pinto HP, Kaplan J, et al. Evaluating large language models trained on code. arXiv preprint. 2021;arXiv:2107.03374.
5. Li J, Tang J, Han J. Data management for machine learning: A survey. IEEE Trans Knowl Data Eng. 2023;35(1):1–22.
6. Zhou Z, Liu Y, Xu J, Tang J. Natural language interfaces for databases: A survey. ACM Comput Surv. 2024;56(3):1–39.
7. Sculley D, Holt G, Golovin D, Davydov E, Phillips T, Ebner D, et al. Hidden technical debt in machine learning systems. Adv Neural Inf Process Syst. 2015;28:2503–11.
8. Ji Z, Lee N, Frieske R, Yu T, Su D, Xu Y, et al. Survey of hallucination in natural language generation. ACM Comput Surv. 2023;55(12):1–38.
9. Zhang Y, Sun S, Galley M, Chen Y, Brockett C, Gao X, et al. DialoGPT: Large-scale generative pre-training for conversational response generation. ACL. 2020;270–80.
10. Lewis P, Perez E, Piktus A, Petroni F, Karpukhin V, Goyal N, et al. Retrieval-augmented generation for knowledge-intensive NLP tasks. Adv Neural Inf Process Syst. 2020;33:9459–74.
11. Karpukhin V, Oguz B, Min S, Lewis P, Wu L, Edunov S, et al. Dense passage retrieval for open-domain question answering. Proc EMNLP. 2020;6769–81.
12. Izacard G, Grave E. Leveraging passage retrieval with generative models for open domain question answering. EACL. 2021;874–80.
13. Yao S, Zhao J, Yu D, Du N, Shafran I, Narasimhan K, et al. ReAct: Synergizing reasoning and acting in language models. arXiv preprint. 2022;arXiv:2210.03629.
14. Schick T, Dwivedi-Yu J, Dessì R, Raileanu R, Lomeli M, Zettlemoyer L, et al. Toolformer: Language models can teach themselves to use tools. Adv Neural Inf Process Syst. 2023;36.

15. Carlini N, Tramer F, Wallace E, Jagielski M, Herbert-Voss A, Lee K, et al. Extracting training data from large language models. USENIX Security Symposium. 2021;2633–50.
16. Zou A, Wang Z, Kolter JZ, Fredrikson M. Universal and transferable adversarial attacks on aligned language models. arXiv preprint. 2023;arXiv:2307.15043.
17. Mittelstadt B, Russell C, Wachter S. Explaining explanations in AI. Proc FAT. 2019;279–88.
18. Patterson D, Gonzalez J, Le Q, Liang C, Munguia LM, Rothchild D, et al. Carbon emissions and large neural network training. arXiv preprint. 2021;arXiv:2104.10350.