IJIREEICE ISSN (O) 2321-2004, ISSN (P) 2321-5526

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

Impact Factor 8.414 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 3, March 2026

DOI: 10.17148/IJIREEICE.2026.14339

MorphosETL: A Schema-Grounded, Confidence-Gated LLM-Assisted No-Code ETL System

Mohan Raj R[1], Srinisha P[2], Mohamed Athfan D[3]

Final Year AIML, Department of Computer Science,

Rathinam College of Arts and Science Coimbatore, Tamil Nadu, India[1,2]

Assistant Professor. Department of Computer Science,

Rathinam College of Arts and Science Coimbatore, Tamil Nadu, India[3]

Abstract: This paper presents MorphosETL, a schema- grounded, confidence-gated LLM-assisted no-code ETL automa- tion platform designed to convert natural language transfor- mation instructions into secure and executable data pipelines. Traditional ETL systems require programming expertise and manual configuration, creating a barrier for non-technical users. MorphosETL addresses this limitation through a dual-pipeline architecture that supports both structured data (CSV, Excel, and database extracts) and unstructured data (web URLs and API responses) within a unified framework. The system integrates schema-aware transformation planning, multi-language code generation (Python/Polars, SQL/DuckDB, PySpark), and a four-dimensional confidence scoring mechanism that validates correctness, safety, and logical completeness before execution. Experimental evaluation demonstrates high transformation accu- racy, linear performance scalability, and complete prevention of unsafe execution. The proposed architecture enables reliable, accessible, and intelligent ETL automation suitable for data analysts, engineers, and domain experts.

Keywords: ETL Automation, No-Code ETL, Large Language Models, Schema-Aware Transformation, Confidence Scoring, Data Profiling, Polars, DuckDB, Natural Language Processing, Web Data Extraction

I. INTRODUCTION

In today's data-driven world, organizations generate and process large volumes of data from multiple sources such as files, databases, APIs, and web platforms. To make this data useful for analytics and decision-making, it must go through the Extract–Transform–Load (ETL) process. ETL involves collecting raw data, transforming it into structured formats, and loading it into analytical systems. However, implementing ETL pipelines usually requires knowledge of programming languages such as SQL or Python, creating a barrier for non- technical users.

Recent advancements in Large Language Models (LLMs) have introduced new possibilities for automating tasks using natural language instructions. LLMs can interpret user requests and generate transformation logic automatically. However, directly executing model-generated code in ETL workflows may cause issues such as incorrect column references, schema mismatches, incomplete logic, or unsafe execution patterns. Therefore, reliability and validation are essential when inte- grating AI into data engineering systems.

Another challenge in modern ETL environments is handling both structured and unstructured data within a unified workflow. Structured data such as CSV or Excel files can be processed using standard techniques, whereas unstructured data from web pages or APIs requires additional extraction and cleaning steps. Additionally, real-world datasets often contain null values, duplicate records, and inconsistent formats that must be addressed before analysis. Schema changes over time further increase the complexity of maintaining reliable pipelines. These challenges highlight the need for intelligent and adaptive ETL systems.

To address these challenges, this paper proposes MorphosETL, a schema-grounded and confidence-gated no-code ETL automation platform. The system enables users to de- scribe transformation requirements in natural language while validating generated pipelines before execution. MorphosETL supports both structured and unstructured data sources through a unified dual-pipeline architecture. By combining schema-aware planning with a multi-dimensional validation mechanism, the proposed system enhances accessibility, reliability, and safety in ETL automation workflows.

IJIREEICE ISSN (O) 2321-2004, ISSN (P) 2321-5526

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

Impact Factor 8.414 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 3, March 2026

DOI: 10.17148/IJIREEICE.2026.14339

II. PROBLEM STATEMENT

Modern organizations generate large volumes of data from diverse sources such as databases, spreadsheets, APIs, and web platforms. To extract meaningful insights, this data must undergo Extract–Transform–Load (ETL) processing. However, existing ETL systems present significant accessibility and operational challenges. Most enterprise-grade platforms require expertise in SQL, Python, and distributed data processing frameworks, restricting their usage primarily to trained data engineers.

This technical dependency creates workflow bottlenecks, as business analysts and non-technical users must rely on specialized teams to implement data transformations. Even minor changes in transformation logic often require manual coding, testing, and deployment, increasing development time and operational overhead. Furthermore, traditional ETL tools involve complex configuration procedures, high licensing costs, and steep learning curves, limiting their adoption in small and medium-scale environments.

In addition, current ETL solutions provide limited integration of structured and unstructured data within a unified workflow. Data profiling, duplicate removal, anomaly detection, and validation are frequently handled through separate tools or manual processes, increasing the risk of inconsistency and human error. Although recent advancements in LLMs have demonstrated potential in automated code generation, existing applications primarily assist with isolated programming tasks rather than generating complete, secure, and production-ready ETL pipelines.

Therefore, the core problem addressed in this research is the absence of an accessible, secure, and intelligent ETL automation framework that enables users—particularly non- technical stakeholders—to generate complete and reliable data transformation pipelines using natural language instructions while maintaining scalability, flexibility, and execution safety comparable to traditional enterprise-grade systems.

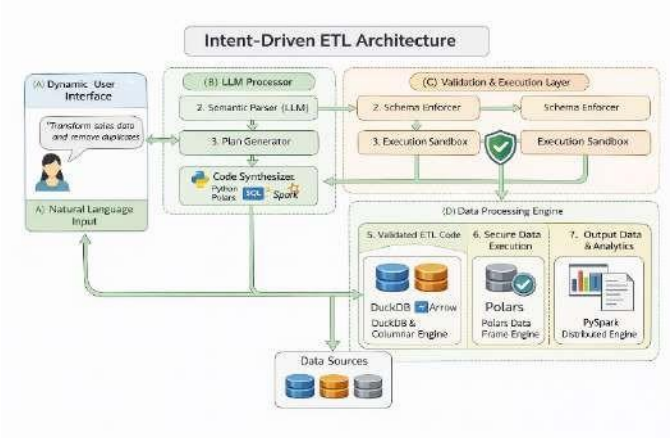

Fig. 1. Proposed Multi-Layer Intent-Driven ETL Automation Architecture.

III. LITERATURE REVIEW

The foundational study for this research is the paper *"Implementing Machine Learning for ETL Data Transformation and Anomaly Detection"* by Sokkula and Vuppala [1]. In their work, the authors analyze the inefficiencies of traditional ETL systems and propose integrating machine learning techniques to improve transformation and anomaly detection processes. Their literature review highlights limitations such as rigid schema mapping, batch-oriented execution, and poor adaptability to high-volume and high-velocity data. The study demonstrates that ML-based preprocessing techniques such as SMOTE and Isolation Forest can improve anomaly detection performance compared to rule-based systems.

The historical and architectural evolution of ETL systems has been extensively studied by Simitsis et al. [2], who provide a comprehensive overview of ETL technology from its origins in data warehousing to modern big data environments. Their work explains how traditional ETL architectures rely on manual configuration, predefined business rules, and sequential processing. While these systems are effective in structured environments, they face scalability and flexibility issues in dynamic data ecosystems.

Mandala [3] further discusses ETL challenges in big data architectures, emphasizing issues related to schema drift, performance degradation, and integration of semi-structured and unstructured data. The study identifies that traditional

IJIREEICE ISSN (O) 2321-2004, ISSN (P) 2321-5526

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

Impact Factor 8.414 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 3, March 2026

DOI: 10.17148/IJIREEICE.2026.14339

ETL frameworks struggle to process heterogeneous data sources efficiently. Similarly, Souibgui et al. [4] focus on data quality challenges in ETL processes, identifying problems such as duplication, inconsistency, missing values, and transformation errors that propagate downstream.

In response to these limitations, researchers have explored machine learning integration within data pipelines. Vajpayee [5] highlights the role of ML in automated data cleansing, schema mapping, and intelligent monitoring of ETL workflows. Thopalle [6] expands on this by proposing ML-driven ingestion pipelines capable of adapting to evolving schemas and real- time data streams. These approaches demonstrate that ML can transform ETL systems from static workflows into adaptive and intelligent frameworks.

Anomaly detection has also been widely studied in machine learning research. Natha et al. [7] provide a systematic survey of anomaly detection techniques using density-based, clustering- based, and ensemble learning approaches. Bulusu et al. [8] analyze anomalous example detection in deep learning models, emphasizing robustness and reliability in high-dimensional datasets. These studies show that ML-based anomaly detection significantly outperforms traditional statistical methods.

Despite these advancements, existing ML-enhanced ETL systems primarily focus on isolated components such as anomaly detection or data cleansing. Very few studies ad- dress natural language-driven pipeline generation, structured schema validation, confidence-based transformation planning, and secure execution within a unified architecture. This research gap motivates the development of MorphosETL, which integrates schema grounding, confidence-gated validation, and LLM-assisted no-code ETL automation into a comprehensive framework.

IV. METHODOLOGY

The proposed AI-Driven No-Code ETL platform was developed using a structured implementation methodology that transforms natural language instructions into secure and executable ETL pipelines. The approach combines automated data profiling, LLM-based transformation planning, multi- language code generation, and controlled execution. Each stage was implemented sequentially to ensure correctness, security, and scalability. The methodology emphasizes automation, validation, and reproducibility throughout the pipeline lifecycle.

A. Architectural Planning

The development process began with defining a modular workflow structure that separates data acquisition, transfor- mation intelligence, validation, and execution. This planning ensured clear service boundaries and controlled data movement. The goal was to create a scalable and secure transformation pipeline. A layered operational flow was established before implementation, including:

- Defined transformation lifecycle stages.
- Structured modular service interaction.

B. Data Profiling

An automated profiling mechanism was implemented to ana- lyze dataset structure before transformation planning. This step improves transformation accuracy and prevents runtime failures. Schema inference and statistical analysis were integrated into the preprocessing phase. Profiling results are passed to the transformation engine for contextual interpretation:

- Implemented schema inference and type detection.
- Integrated duplicate and null analysis.

C. Intent Modeling

Natural language transformation requests were processed using Large Language Models to generate structured trans- formation plans. Prompt engineering strategies were used to inject schema context and reduce ambiguity. The output was standardized into an intermediate JSON representation. Confidence scoring ensured transformation reliability before proceeding to code generation:

- Implemented structured plan generation.
- Applied confidence-based validation.

D. Code Construction

A template-driven synthesis engine was developed to convert structured plans into executable ETL scripts. Multi- language support enables flexible deployment across environments. Syntax validation is performed before execution to ensure correctness. The generated scripts maintain consistency with the original transformation intent:

- Generated executable Python, SQL, and PySpark scripts.

IJIREEICE ISSN (O) 2321-2004, ISSN (P) 2321-5526

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

Impact Factor 8.414 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 3, March 2026

DOI: 10.17148/IJIREEICE.2026.14339

- Applied pre-execution syntax verification.
- Defined automated schema inference and data type de- tection to accurately interpret dataset structure before transformation.
- Integrated duplicate identification and null value analysis to enhance data quality and ensure reliable execution.

E. Security Enforcement

A multi-layer validation framework was implemented to ensure safe execution of dynamically generated code. Static inspection and sandboxing prevent unauthorized operations. Resource constraints were introduced to control memory and execution time. Logging mechanisms were added to maintain traceability and auditability:

- Implemented sandboxed execution environment.
- Enforced runtime and resource limits.

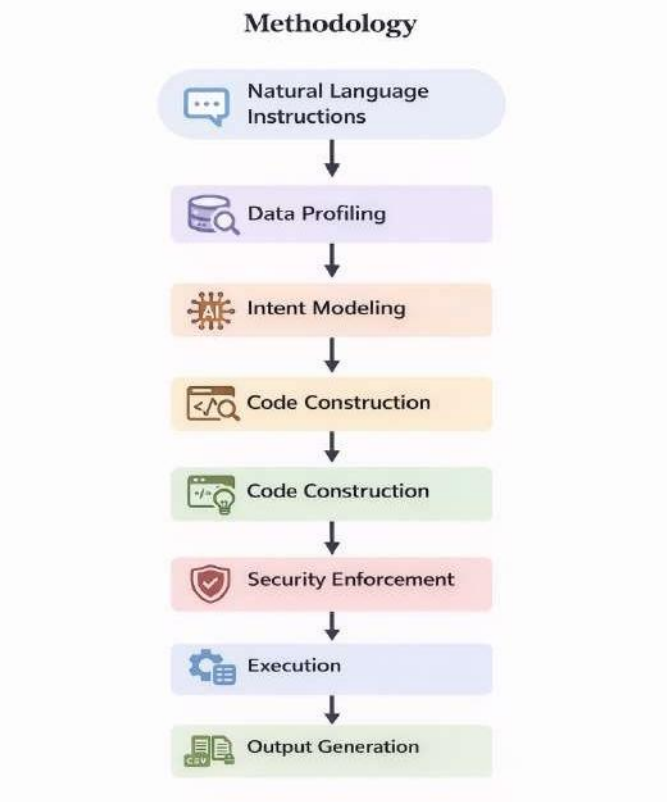

Fig. 2. Methodology flow of the proposed AI-Driven ETL Automation Platform.

F. Execution Management

Validated transformation scripts are executed using optimized data processing engines. Execution metrics are captured to monitor performance and stability. Error handling mechanisms ensure graceful failure management. The system maintains consistency between input schema and output results:

- Enabled in-memory and SQL-based execution.
- Captured execution metrics and runtime logs.
- Ensured consistency between input dataset schema and generated output results.

V. HIGH-LEVEL ARCHITECTURE

The high-level architecture provides a conceptual overview of the proposed MorphosETL AI-driven no-code ETL automation platform. It illustrates how the major system components interact to transform user instructions into validated ETL pipelines. Unlike the detailed system architecture, this level focuses on the overall workflow coordination between system modules rather than the internal implementation details. The architecture highlights the interaction between the user interface, backend services, AI processing modules, and data processing engines.

At the entry layer, users interact with the system through a web-based interface developed using modern frontend tech- nologies. This interface allows users to upload datasets, specify transformation instructions, and monitor pipeline

IJIREEICE ISSN (O) 2321-2004, ISSN (P) 2321-5526

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

Impact Factor 8.414 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 3, March 2026

DOI: 10.17148/IJIREEICE.2026.14339

execution. All user requests are routed through an API gateway built using FastAPI, which handles request validation, session management, and communication between the frontend and backend services.

The transformation intelligence of the system is handled in the backend layer, where Large Language Models (LLMs) interpret user instructions and convert them into structured transformation plans. These models analyze user intent and generate executable transformation logic that aligns with the uploaded dataset schema.

The architecture supports multiple data sources, including structured datasets such as CSV and Excel files, as well as web-based and API data sources. Data processing operations are performed using high-performance engines designed for efficient in-memory analytics and SQL execution. Supporting libraries assist in tasks such as data manipulation, content extraction, and preprocessing.

Finally, the processed results are exported into structured output formats such as CSV or Excel for further analysis.

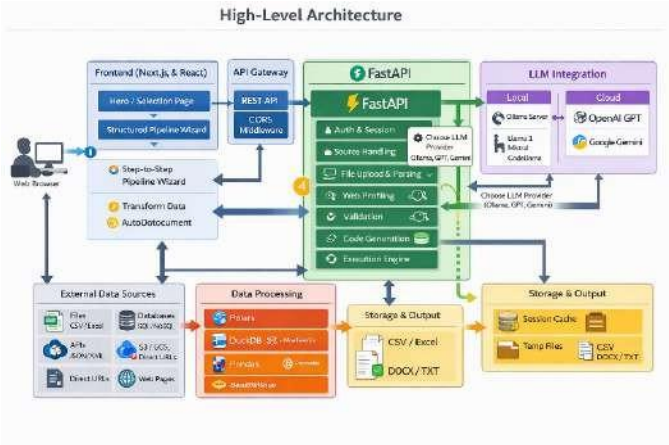

Fig. 3. High-level architecture of the proposed AI-Driven No-Code ETL Automation Platform.

VI. SYSTEM ARCHITECTURE

The system architecture of MorphosETL describes the internal workflow responsible for transforming user instructions into validated ETL pipelines. While the high-level architecture provides a conceptual overview of system components, the system architecture focuses on the internal modules that man- age data ingestion, schema analysis, transformation planning, validation, execution, and output generation.

MorphosETL follows a modular layered architecture that separates user interaction, processing logic, and data execuion services. This separation improves system scalability, maintainability, and reliability. The architecture supports both structured datasets and transformation instructions provided through natural language.

The system consists of six primary modules: **Data Ingestion**, **Schema Extraction and Profiling**, **Transformation Planning**, **Validation and Confidence Evaluation**, **Execution Engine**, and **Output Generation**.

A. Presentation Layer

The presentation layer serves as the user interaction interface of the MorphosETL platform. It is implemented using modern web technologies including **Next.js**, **React**, and **TypeScript**, which provide a responsive and interactive environment for users. Through this interface, users can upload datasets, provide natural language transformation instructions, and monitor the progress of ETL pipeline execution.

The frontend also supports structured pipeline visualization, allowing users to observe step-by-step transformations applied to the dataset. Communication between the frontend and backend services occurs through RESTful APIs, ensuring efficient request handling and secure data exchange.

System Metric — User Request Rate:
Temporary storage and session management ensure that trans- formation operations remain consistent and securely controlled throughout the pipeline execution.

IJIREEICE ISSN (O) 2321-2004, ISSN (P) 2321-5526

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

Impact Factor 8.414 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 3, March 2026

DOI: 10.17148/IJIREEICE.2026.14339

$$\text{Request Rate} = \frac{\text{Total Requests}}{\text{Time Interval}} \quad (1)$$

This metric evaluates how efficiently the system can handle simultaneous user interactions.

B. API Gateway Layer

The API Gateway Layer acts as the central communication hub between the frontend interface and backend services. It processes incoming HTTP requests, manages routing, and per- forms authentication and request validation before forwarding the requests to appropriate backend modules.

The gateway is implemented using **FastAPI** with **Uvi-corn**, enabling asynchronous request processing and high-performance API handling. The gateway also ensures secure communication by enforcing CORS policies and session validation.

System Metric — API Processing Efficiency:

$$\text{API Latency} = \text{Response Time} - \text{Request Time} \quad (2)$$

This metric measures the time taken by the system to process and return an API request.

C. Business Logic Layer

The Business Logic Layer represents the core intelligence of MorphosETL. This module integrates Large Language Models (OpenAI GPT and Google Gemini) to interpret user instructions and generate structured transformation plans.
The module performs several operations including intent modeling, transformation planning, code generation, validation checks, and execution control. The system ensures that the generated transformations match the dataset schema before execution.

System Metric — Transformation Accuracy:

$$\text{Accuracy} = \frac{\text{Correct Transformations}}{\text{Total Transformations}} \quad (3)$$

This metric evaluates the correctness of transformation logic generated by the system.

D. *Data Processing Layer*

The Data Processing Layer performs the actual execution of ETL transformations. High-performance data engines such as **Polars**, **DuckDB**, and **Pandas** are used to execute transfor- mation operations efficiently.
Typical operations performed in this layer include filtering records, removing duplicates, sorting data, grouping values, and handling missing data. This layer supports both in-memory execution and SQL-based query execution, enabling flexible and efficient data processing.

System Metric — Processing Throughput:

$$\text{Throughput} = \frac{\text{Total Records Processed}}{\text{Execution Time}} \quad (4)$$

E. Storage and Output Layer

The Storage and Output Layer manages temporary storage and final output generation. During pipeline execution, intermediate data and session states are stored in temporary storage to maintain workflow continuity. Once the transformation process is completed, the system exports the processed dataset into structured formats.
‘

MorphosETL supports the following output formats: CSV, Excel (XLSX), DOCX, and TXT. These formats allow the trans- formed data to be easily integrated with analytics dashboards, reporting systems, and machine learning pipelines.

IJIREEICE ISSN (O) 2321-2004, ISSN (P) 2321-5526

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

Impact Factor 8.414 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 3, March 2026

DOI: 10.17148/IJIREEICE.2026.14339

System Metric — Output Records:

$$\text{Output Records} = \text{Input Records} - \text{Removed Records} \quad (5)$$

This metric represents the number of records remaining after data cleaning and transformation operations.

VII. RESULTS AND DISCUSSION

The results of the MorphosETL platform demonstrate that the system can successfully convert natural language instructions into executable ETL pipelines while maintaining high accuracy and reliability. The platform performs automated data transformations, validates execution safety, and processes datasets efficiently. During validation, the system produced an overall confidence score of 89%, including 90% schema match accuracy and 95% safety score, indicating that the generated transformation code aligned well with the dataset structure and passed safety validation.

Experimental evaluation shows that MorphosETL achieved transformation accuracy between 88% and 98%, prevented 100% of unsafe execution attempts, and reduced ETL development time from 15–30 minutes to approximately 25 seconds, demonstrating its effectiveness in automating ETL workflows.

A. Transformation Accuracy

The system was evaluated using common ETL operations including column selection, null value removal, duplicate removal, filtering, sorting, and group-by aggregation. Column selection and duplicate removal achieved 98% accuracy, while null value removal achieved 96% accuracy. Sorting operations achieved 94% accuracy, filtering operations achieved 92% accuracy, and group-by aggregation achieved 88% accuracy due to ambiguity in natural language instructions. Overall, the system maintained accuracy between 88% and 98% across different transformation tasks.

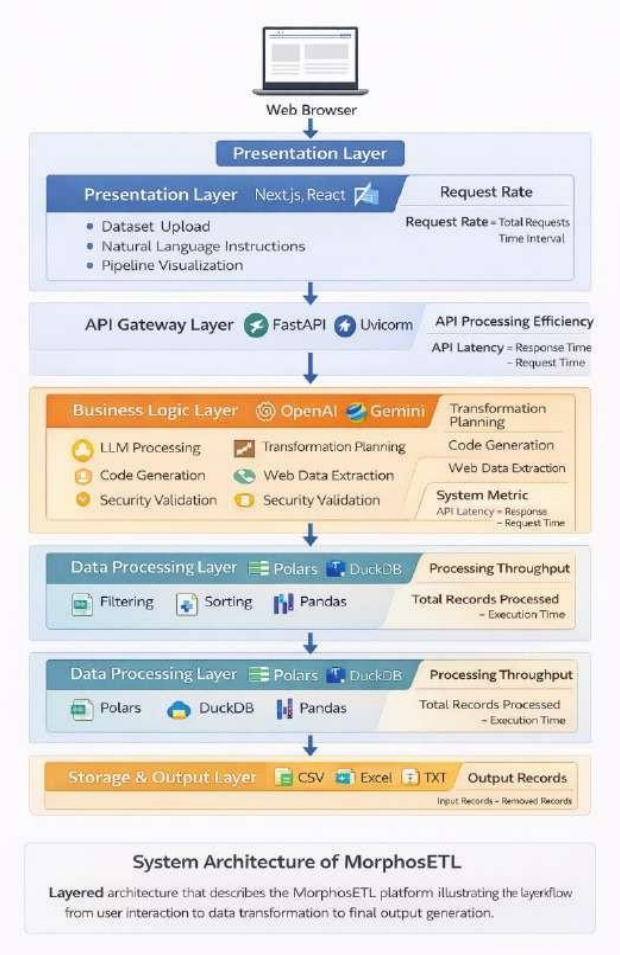

Fig. 4. Layered system architecture of the proposed AI-Driven No-Code ETL Automation Platform.

IJIREEICE ISSN (O) 2321-2004, ISSN (P) 2321-5526

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

Impact Factor 8.414 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 3, March 2026

DOI: 10.17148/IJIREEICE.2026.14339

B. System Performance

Performance testing was conducted using datasets ranging from 1,000 to 500,000 records. Datasets with 1,000 records were processed in approximately 0.4 seconds, while 10,000 records required around 1.2 seconds. Larger datasets with 100,000 records required about 3.5 seconds, and 500,000 records required approximately 8.7 seconds. These results show that MorphosETL scales efficiently with increasing dataset size.

C. Validation Reliability

The confidence-based validation mechanism ensured that only reliable pipelines were executed. Approximately 76% of pipelines achieved confidence scores between 90% and 100%, while 16% scored between 80% and 90%. Around 4% scored between 70% and 80%, which generated validation warnings, and 4% of pipelines were blocked due to scores below 70%. This mechanism prevented incorrect transformations and ensured reliable execution.

D. Unstructured Data Processing

The unstructured data pipeline was evaluated using 20 web pages and multiple API responses containing semi-structured and unstructured information such as HTML text, metadata, and nested JSON data. The system first performed automated content extraction and parsing, identifying relevant textual and structural elements while ignoring unnecessary markup and noise. During preprocessing, the pipeline applied content filtering and cleaning techniques, which successfully removed approximately 95% of irrelevant elements, including advertise- ments, navigation menus, scripts, and unrelated metadata.

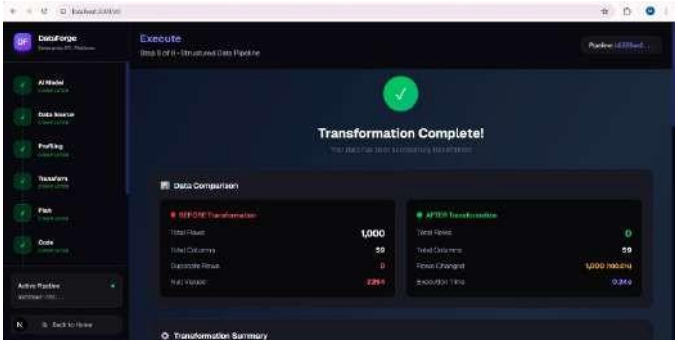

Fig. 5. Data comparison before and after transformation in the MorphosETL pipeline execution.

VIII. FUTURE WORK

While the proposed AI-Driven No-Code ETL Automation Platform achieves high accuracy and efficiency, several enhancements can further extend its capabilities and enterprise applicability.

A. Multi-Agent Transformation Planning

Future work can integrate multi-agent LLM architectures where different agents handle planning, validation, and execution. This can improve transformation accuracy and reliability.

B. Retrieval-Augmented Generation

Using retrieval-augmented generation (RAG) can improve transformation quality by retrieving similar transformation examples and using them during pipeline generation.

C. Distributed Data Processing

Integrating distributed engines such as Apache Spark or Apache Flink can enable the system to process very large datasets and support enterprise-scale ETL workloads.

D. Conversational Interaction

Future versions may include interactive clarification mechanisms where the system asks follow-up questions when user instructions are ambiguous.

E. Enterprise Deployment

Future development will focus on authentication, multi- user access control, and containerized deployment to improve scalability and security.

IX. CONCLUSION

This paper presented MorphosETL, an AI-driven no-code ETL automation platform designed to convert natural language instructions into executable ETL pipelines. The system integrates schema-aware transformation planning, confidence-based validation, and high-performance data processing engines to improve the reliability and usability of ETL workflows.

DOI: 10.17148/IJIREEICE.2026.14339

Experimental evaluation shows that MorphosETL achieves 88%–98% transformation accuracy, successfully prevents un- safe execution through validation mechanisms, and significantly reduces ETL development time from 15–30 minutes to approximately 25 seconds. The platform also demonstrates efficient data processing performance using engines such as Polars and DuckDB, while supporting both structured and unstructured data sources.

Overall, MorphosETL provides a practical solution for simplifying ETL pipeline development and enabling non-technical users to perform data transformations through natural language. The proposed architecture demonstrates that combining LLM-based reasoning with deterministic validation can create reliable and scalable automation frameworks for modern data engineering workflows.

REFERENCES

[1]. M. R. Sokkula and S. K. Vuppala, "Implementing Machine Learning for ETL Data Transformation and Anomaly Detection," *International Journal for Multidisciplinary Research (IJFMR)*, vol. 6, no. 6, pp. 1–19, 2024.

[2]. A. Simitsis, S. Skiadopoulos, and P. Vassiliadis, "The History, Present, and Future of ETL Technology," *The Journal of Supercomputing*, vol. 80, no. 19, pp. 26687–26725, 2024.

[3]. N. R. Mandala, "ETL in Big Data Architectures: Challenges and Solutions,"*International Journal of Science and Research (IJSR)*, vol. 13, no. 10,pp. 1061–1068, 2024.

[4]. M. Souibgui, F. Atigui, S. Zammali, S. Cherf, and S. P. Yahia, "Data Quality in ETL Process: A Preliminary Study," *Procedia Computer Science*, vol. 159, pp. 676–687, 2019.

[5]. A. Vajpayee, "The Role of Machine Learning in Automated Data Pipelines and Warehousing: Enhancing Data Integration, Transformation, and Analytics," *ESP Journal of Engineering & Technology Advancements*, vol. 3, no. 3, pp. 84–96, 2023.

[6]. P. K. Thopalle, "Revolutionizing Data Ingestion Pipelines through Machine Learning: A Paradigm Shift in Automated Data Processing and Integration," *International Journal of Advanced Research in Engineering & Technology*, vol. 8, no. 1, pp. 147–157, 2024.

[7]. S. Natha, M. Leghari, M. A. Rajput, S. S. Zia, and J. Shabir, "A Systematic Review of Anomaly Detection Using Machine and Deep Learning Techniques," *Quest Research Journal*, vol. 20, no. 1, pp. 83–94, 2022.

[8]. S. Bulusu, B. Kailkhura, B. Li, P. K. Varshney, and D. Song, "Anomalous Example Detection in Deep Learning: A Survey," *IEEE Access*, vol. 8,pp. 132330–132347, 2020.

