Journal of Computer Technology and Software

ISSN:2998-2383

Vol. 3, No. 4, 2024

Trustworthy LLM-Oriented Enterprise ETL Orchestration via Neural Execution Verification

Weican Wang[1*], Jiajia Cheng[2]
[1]University of Rochester, Rochester, USA
[2]Columbia University, New York, USA
*Corresponding author: Weican Wang; christywg@outlook.com

Abstract: To address the challenges of heterogeneous data sources, complex business semantics, dense task dependencies, and insufficient execution reliability in enterprise ETL processes, this paper proposes an automated orchestration and reliable execution method for enterprise ETL based on an LLM orchestration module and a neural verifier. This method first models data tables, fields, transformation relationships, quality rules, and task dependencies into a unified ETL state diagram, fusing structural features, semantic descriptions, and quality profiles to form a reasonable data state representation. Then, the LLM orchestration module decomposes enterprise data processing requirements into tasks, selects operations, generates parameters, and plans dependencies, producing candidate ETL orchestration schemes with executable structures. To reduce logical biases and constraint omissions in the automatically generated process, a neural validator is further designed to judge the semantic consistency, structural legality, constraint satisfaction, and execution reliability of candidate operations, and an execution gating mechanism is used to complete operation screening, process correction, and reliable control. Comparative experimental results show that the proposed method exhibits good overall performance in terms of process planning accuracy, verification, and discrimination ability, constraint satisfaction degree, and execution success rate. It can effectively improve the quality of automated orchestration and reliable execution of enterprise ETL tasks, and provide effective methodological support for the intelligent governance of complex enterprise data pipelines.

Keywords: Enterprise data pipeline; LLM orchestration; neural validator; constraint-aware execution

1. Introduction

As enterprise digital transformation deepens, data has become a crucial foundational resource for business operations, risk control, operational decision-making, and intelligent services. Enterprise data is typically dispersed across business systems, log platforms, data warehouses, data lakes, and external interfaces, exhibiting significant differences in structure, update frequency, quality level, and business semantics[1, 2]. The ETL process, as a core component of data acquisition, transformation, cleaning, fusion, and loading, directly impacts the reliability of subsequent data analysis, model training, and business decisions. However, traditional ETL processes often rely on manual rule configuration, fixed task templates, and static scheduling strategies, making it difficult to adapt to the frequently changing data sources, complex dependencies, and dynamic business needs within enterprise data environments[3]. Therefore, building enterprise ETL methods with automatic understanding, orchestration, and reliable execution capabilities is crucial for improving enterprise data governance efficiency and the intelligence level of data infrastructure.

In complex enterprise scenarios, ETL processes are no longer simply data transfer tasks but involve multiple stages, including data pattern recognition, field semantic alignment, quality constraint checks, task dependency reasoning, anomaly handling, and execution strategy optimization[4]. Traditional methods typically break down these steps into relatively independent rule modules, resulting in high process maintenance costs, limited cross-scenario migration capabilities, and a lack of flexible semantic understanding when facing ambiguous requirements, unstructured descriptions, and complex business constraints. The introduction of LLM-based orchestration provides a new technical path for automated orchestration of enterprise ETL, enabling process planning, tool invocation, and step generation based on natural language requirements, data structure information, and task context[5]. However, relying solely on large language models to generate execution processes may still lead to issues such as logical inconsistencies, missing constraints, and unverifiable execution. Therefore, it is necessary to combine neural verifiers to constrain the rationality, completeness, and credibility of the generated processes. Table 1 summarizes the core problems in enterprise ETL scenarios and their capability requirements for intelligent methods.

Table 1. Core Issues and Intelligent Capability Requirements in Enterprise ETL Scenarios

Core Issue	Specific Manifestation	Corresponding Capability Requirement
Heterogeneous data sources	Inconsistent structures across databases, interfaces, logs, files, and data lakes	Cross-source data understanding and schema recognition capability
Complex business	Implicit dependencies among field meanings,	Semantic parsing and contextual reasoning

semantics	indicator definitions, and business rules	capability
Static orchestration rules	Task dependencies rely on manual configuration and are difficult to adjust dynamically.	Automatic planning and adaptive orchestration capability
Opaque execution process	Generated steps lack explainable evidence, making errors difficult to locate in time.	Process tracing and trusted verification capability
Diverse quality constraints	Missing values, duplicates, outliers, and consistency issues are intertwined.	Data quality detection and constraint validation capability
Difficult anomaly recovery	Scheduling failures, schema drift, and task interruptions affect downstream pipelines.	Anomaly identification and automatic recovery capability

As shown in Table 1, the key challenges of enterprise ETL have expanded from automating single tasks to intelligent governance across the entire process, addressing complex business semantics and dynamic data environments[6]. The LLM-based orchestration module provides strong semantic understanding, task decomposition, and tool-use planning capabilities, enabling it to transform enterprise users' data processing needs into executable ETL orchestration logic, thereby reducing manual configuration costs and enhancing the flexibility of process generation. The neural validator, on the other hand, can learn to discriminate data constraints, task dependencies, transformation logic, and result consistency before and after process execution, providing a reliable boundary for automatically generated ETL processes. The combination of these two technologies forms a closed-loop mechanism from requirement understanding, process planning, execution control, to result verification, enabling the ETL system to gradually shift from passively executing fixed rules to proactively understanding business needs, dynamically generating processing links, and continuously verifying execution reliability-a truly intelligent execution system.

Research on automated orchestration and reliable execution of enterprise ETL based on LLM-guided orchestration and a neural validator has significant theoretical and practical value. From a theoretical perspective, this approach can promote the integration of large language models, LLM-guided planning, automated data engineering, and trusted machine learning in enterprise data pipeline scenarios, providing a new research paradigm for semantic modeling, task planning, and execution verification of complex data processes[7]. From an application perspective, this method helps reduce the manual development and maintenance costs in enterprise data engineering, improves the responsiveness of data processing chains to business changes, and enhances the reliability, traceability, and auditability of critical data tasks. As enterprises increasingly demand high-quality data assets and intelligent data infrastructure, automated orchestration and trusted execution for ETL processes will become a crucial supporting direction for enterprise data governance and the construction of intelligent decision-making systems.

2. Datasets and Dataset Preprocessing

2.1 Dataset

This paper selects AdventureWorks2022 and AdventureWorksDW2022 as the data foundation for research on automated orchestration and trusted execution of enterprise ETL. This dataset is a publicly available example of an enterprise-level relational database, containing AdventureWorks2022 for online transaction processing and AdventureWorksDW2022 for data warehouse analysis. The former covers multiple business entities such as sales, customers, products, orders, employees, and suppliers, while the latter provides fact tables and dimension tables after dimensional modeling, enabling a relatively complete simulation of the typical ETL process of an enterprise extracting data from business systems, completing field mapping, cleaning and transformation, dimension association, and warehouse loading. This dataset possesses both source transaction database and target data warehouse structures, making it suitable for scenarios involving task understanding, process planning, tool invocation, and transformation logic generation for building LLM-based ETL workflows. It is also suitable for a neural verifier to verify field consistency, primary and foreign key constraints, transformation rules, data integrity, and the trustworthiness of loading results. The AdventureWorks database provides OLTP, Data Warehouse, and Lightweight versions in its official samples. These sample databases can be downloaded from public repositories, and the samples and templates are licensed under the MIT license. Therefore, they can meet the needs of enterprise ETL research for data openness, structural integrity, explicit business semantics, and a reproducible experimental environment.

2.2 Dataset preprocessing

To evaluate the impact of data preprocessing on source data quality and its consistency with the target data warehouse in the enterprise ETL automatic orchestration process, this paper conducts a statistical analysis from four dimensions: field integrity, business key uniqueness, referential integrity, and loading stability. Field integrity is used to measure whether key source fields contain missing or invalid values, reflecting the completeness of the basic data representation. Business key uniqueness focuses on duplicate records in core business entities, which directly affects entity identification, table association, and downstream aggregation accuracy. Referential integrity evaluates whether primary-foreign key relationships and cross-table dependencies remain valid during data extraction and transformation. Loading stability measures the row count deviation between processed source data and target warehouse tables, indicating whether the loading process can preserve data consistency after transformation and integration.

This analysis provides a quantitative basis for judging whether the preprocessing mechanism can effectively improve the reliability of enterprise ETL data flows before automated orchestration and trusted execution. In complex enterprise data environments, problems such as missing fields, duplicate business keys, broken referential relationships, and inconsistent loading results may propagate through downstream tasks and further affect workflow planning, constraint verification, and warehouse availability. Therefore, preprocessing is not only a data cleaning step but also an important prerequisite for

trustworthy ETL execution. By comparing the data quality indicators before and after preprocessing, the effectiveness of the proposed preprocessing procedure in reducing data noise, repairing structural constraints, improving cross-table consistency, and enhancing warehouse loading reliability can be more clearly demonstrated. The experimental results are shown in Figure 1.

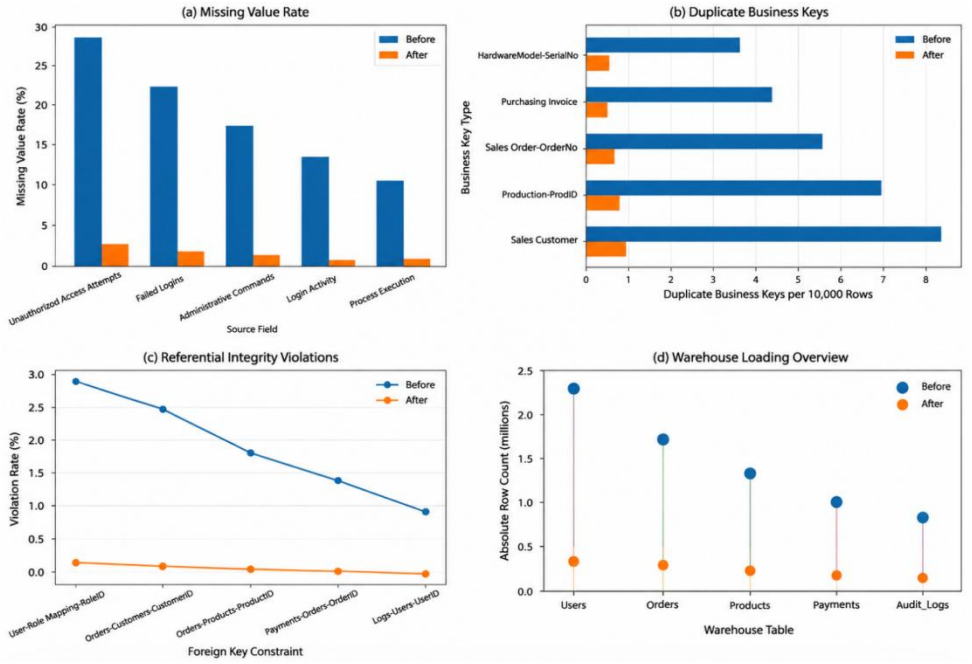

Figure 1. Comparison of dataset preprocessing results

Figure 1 presents the statistical comparison of dataset preprocessing results from four perspectives. Subfigure (a) reports the missing value rate of representative source fields, subfigure (b) shows the number of duplicate business keys in major source tables, subfigure (c) illustrates the referential violation rate of key foreign key constraints, and subfigure (d) compares the absolute row count deviation of different warehouse tables during data loading.

The results indicate that the preprocessing procedure effectively improves the overall quality and consistency of the ETL dataset. After preprocessing, missing values, duplicate keys, referential integrity violations, and warehouse loading deviations are all substantially reduced, demonstrating that the processed data become more complete, structurally consistent, and suitable for subsequent trusted execution verification.

3. Methodology

For enterprise-level ETL automatic orchestration tasks, the core objective of the proposed method is to unify distributed data sources, business semantics, quality constraints, and execution dependencies into a reasoning-oriented task space. In conventional enterprise data engineering, ETL workflows are often constructed through manually defined rules, fixed templates, and task-specific scripts, which makes the orchestration process highly dependent on prior configuration and difficult to adapt to changing business requirements. When data sources differ in schema design, field granularity, naming conventions, update frequency, and quality status, direct script generation based only on natural language instructions may easily lead to incomplete field mapping, unreasonable transformation logic, and missing dependency constraints. Therefore, the proposed method first establishes a structured state representation for the enterprise ETL environment, enabling the LLM orchestration module to perform workflow planning under explicit data context and constraint guidance rather than relying solely on textual task descriptions.

Enterprise data environments usually contain heterogeneous objects such as transactional databases, data warehouses, file interfaces, log records, and external services, and these objects are connected through complex extraction, cleaning, transformation, aggregation, and loading relationships. Relying only on field names or table schemas is insufficient to accurately characterize cross-source transformation relationships, especially when business indicators involve implicit semantic rules, multi-table dependencies, or data quality requirements. To address this issue, an ETL state graph is constructed to integrate data schemas, field semantics, quality rules, transformation dependencies, and execution constraints into a unified representation. This graph-based representation provides the LLM orchestration module with a structured understanding of source objects, target warehouse requirements, candidate operations, and dependency paths, while also offering the neural verifier a basis for checking semantic consistency, structural validity, and constraint satisfaction. Its model architecture is shown in Figure 2.

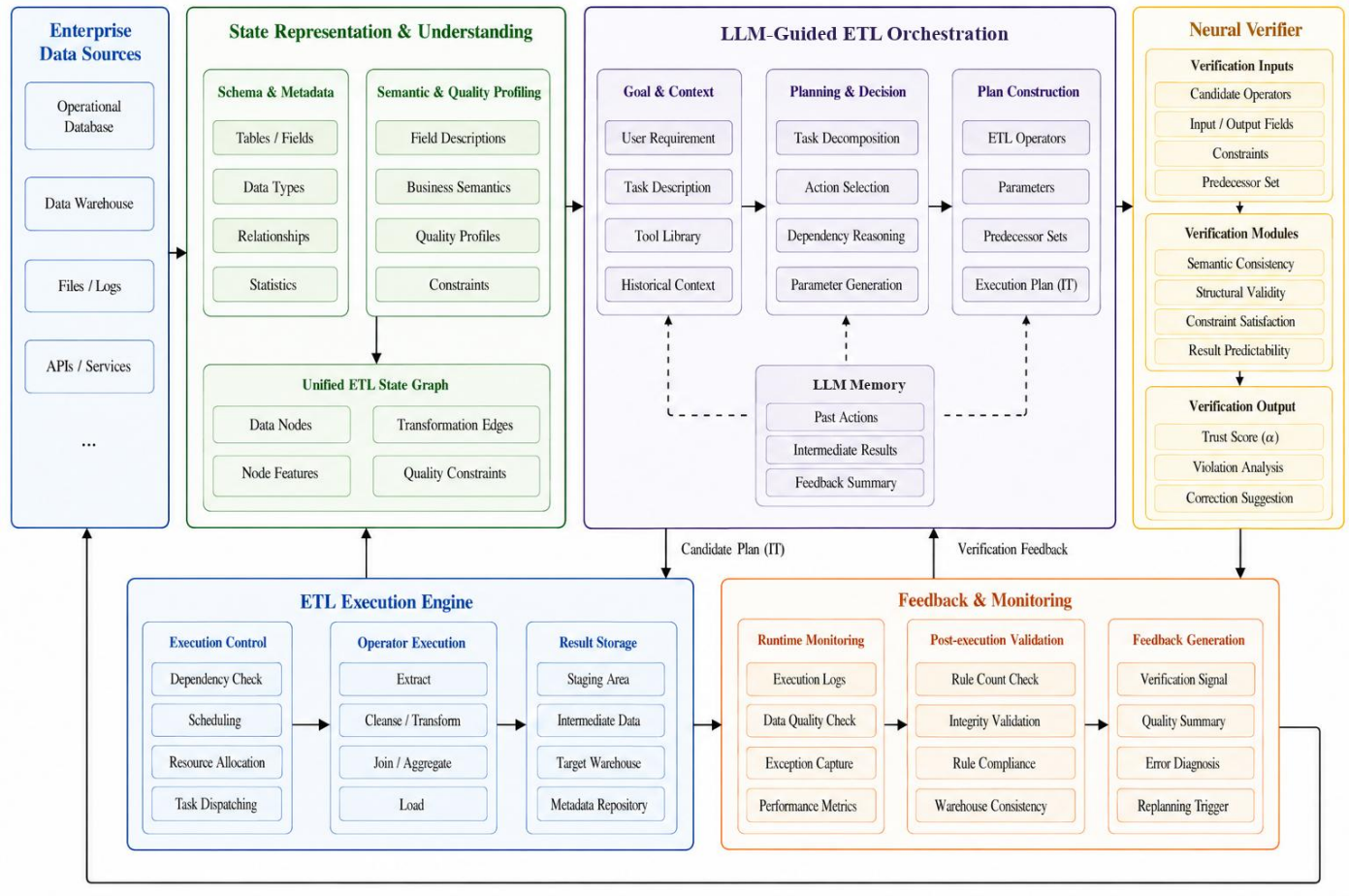

Figure 2. Overall model architecture

To ensure that the subsequent orchestration process can simultaneously perceive data objects and processing logic, the enterprise ETL scenario is represented as a directed graph composed of data nodes, transformation edges, node features, and constraint relations:

$$\mathcal{G}_e = (\mathcal{V}_d, \mathcal{E}_t, \boldsymbol{X}_d, \boldsymbol{C}_q)$$

In this graph, $\mathcal{V}_d$ carries data objects such as data tables, fields, files, and intermediate results, $\mathcal{E}_t$ describes transformation dependencies including extraction, cleaning, joining, aggregation, and loading, $\boldsymbol{X}_d$ encodes structural statistics, field types, and business descriptions as node features, while $\boldsymbol{C}_q$ represents quality rules such as completeness, consistency, uniqueness, and referential constraints. To reduce the differences among data sources in naming conventions, field granularity, and business definitions, field-level semantic representations are generated jointly from structural features, textual descriptions, and quality profiles, enabling the subsequent LLM-guided planning process to use more stable contextual cues when selecting transformation operators:

$$\boldsymbol{h}_i = \phi_\theta\left(\boldsymbol{x}_i^s, \boldsymbol{x}_i^m, \boldsymbol{x}_i^q\right)$$

The representation $\boldsymbol{h}_i$ integrates $\boldsymbol{x}_i^s$ derived from field type, value range, and schema position, $\boldsymbol{x}_i^m$ derived from field names, business annotations, and upstream-downstream descriptions, and $\boldsymbol{x}_i^q$ reflecting missing rate, duplicate rate, anomaly ratio, and constraint satisfaction status, so that each field node contains not only structural data meaning but also business semantic information oriented toward ETL decision-making.

During the automatic orchestration stage, the role of the LLM orchestration module is not to directly generate an uncontrollable complete program, but to produce stepwise planning decisions according to the current state graph, task objective, and available tool set, decomposing complex ETL requirements into executable and traceable operation sequences that can be checked and rolled back by the execution control mechanism. Given the enterprise data processing objective $\mathcal{Q}$ and the tool space $\mathcal{A}$, the LLM proposes the next operation at step t according to historical actions, the current graph state, and constraint feedback, so that workflow planning can simultaneously satisfy business objectives and data quality boundaries:

$$a_t = \underset{a \in \mathcal{A}}{argmax}\, p_\psi\left(a | \mathcal{Q}, \mathcal{G}_{e'}^t, a_{<t}, \boldsymbol{f}_t\right)$$

The feedback vector $\boldsymbol{f}_t$ is jointly composed of execution logs, constraint checks, schema matching results, and validator-returned signals, allowing each orchestration step to be dynamically corrected according to the previous execution state. Unlike static ETL templates that only focus on predefined task orders, the candidate workflow generated by the LLM is further organized by the orchestration controller into an executable plan with dependency constraints, thereby avoiding premature execution of downstream tasks when field mapping has not been completed, cleaning results have not been generated, or upstream tables have not yet been loaded. Accordingly, the complete ETL orchestration plan is defined as a task sequence with dependency edges and operation parameters:

$$\Pi = \{(o_k, \boldsymbol{u}_k, \mathcal{P}_k)\}_{k=1}^{K}$$

In this formulation, o_k denotes the k-th ETL operation, $\boldsymbol{u}_k$ describes operation parameters such as source fields, target fields, join keys, and filtering conditions, and $\mathcal{P}_k$ stores the set of prerequisite tasks that must be satisfied before the operation is executed, allowing the orchestration result to be transformed from natural-language task understanding into a dependency structure that can be parsed by the execution engine.

To address field drift, schema conflicts, and omitted implicit business rules that commonly occur in enterprise ETL processes, a neural validator is introduced into the automatic orchestration process to evaluate the trustworthiness of the candidate operation plan proposed by the LLM. The validator does not merely check syntactic correctness, but jointly evaluates whether field mappings are reasonable, whether transformation logic satisfies constraints, whether task dependencies are closed, and whether loading results are consistent with the target warehouse schema. For an arbitrary candidate operation o_k, the validator outputs a trust score according to the operation context, input-output field representations, and constraint set:

$$s_k = V_\omega\left(o_k, \boldsymbol{h}_{src(k)}, \boldsymbol{h}_{tar(k)}, \boldsymbol{C}_q, \mathcal{P}_k\right)$$

The score s_k reflects the overall trustworthiness of the candidate operation in terms of semantic consistency, structural legality, and constraint satisfaction, and operations with low trust scores will trigger replanning, parameter correction, or manual audit marking. Since ETL execution requires not only trustworthy single-step operations but also global reachability, interpretability, and recoverability of the entire task chain, the proposed method further models process-level trustworthy execution as an optimization objective under multiple constraints:

$$\mathcal{J}(\Pi) = \sum_{k=1}^{K} \alpha\, s_k - \beta \mathcal{C}_{dep}(\Pi) - \gamma \mathcal{C}_{con}(\Pi)$$

This objective rewards highly trustworthy operations while penalizing dependency conflicts and constraint violations, thereby shifting the generation preference of the LLM orchestration module from merely completing tasks to generating execution schemes that are more stable, safer, and more consistent with enterprise data governance requirements.

The trustworthy execution stage emphasizes closed-loop control over the orchestration plan, so that the ETL workflow can continuously receive validation signals before, during, and after execution. Pre-execution validation is used to identify field mapping errors, missing dependencies, and illegal transformations. In-execution validation is used to capture anomalous records, schema drift, and task interruptions, while post-execution validation confirms data completeness, row-count consistency, and business rule satisfaction in the target warehouse. To feed validation results back into subsequent LLM planning steps, the system generates an execution gating signal according to the trust score and constraint losses:

$$g_k = \sigma\left(\lambda_1 s_k - \lambda_2 \mathcal{L}_q(o_k) - \lambda_3 \mathcal{L}_r(o_k)\right)$$

The gating value g_k determines whether the candidate operation is directly executed, whether it enters the correction stage, and whether the process needs to roll back to a previous state, while $\mathcal{L}_q(o_k)$ and $\mathcal{L}_r(o_k)$ characterize the quality constraint deviation and result consistency deviation, respectively, enabling the discriminative capability of the neural validator to be transformed into practical execution control. Finally, the ETL output after validation constraints is progressively generated by the operation sequence that passes the gating mechanism:

$$\boldsymbol{Y}^* = F_\Pi\left(\boldsymbol{D}_{src}; \{g_k o_k\}_{k=1}^{K}\right)$$

This design extends enterprise ETL automatic orchestration from one-shot workflow generation to a trustworthy execution process that is feedback-driven, verifiable, and traceable, thereby improving the automation level of data processing while enhancing the adaptability of the data processing chain to complex business semantics, dynamic data environments, and changing quality constraints.

4. Experimental Results and Analysis

4.1 Experimental setup

The experiments were conducted in a unified deep learning environment. The overall method consists of an LLM orchestration module, a state graph representation module, a neural validator module, and a trusted execution control module. The LLM orchestration module is responsible for generating candidate ETL operation sequences based on the task context. The state graph representation module encodes field semantics, inter-table dependencies, and quality constraints. The neural validator judges the semantic consistency, structural legality, and constraint satisfaction of candidate operations. The trusted execution control module determines whether to execute, correct, or replan based on the validation score. To ensure the stability of the training and orchestration processes, key hyperparameters such as the optimizer, learning rate, hidden dimension, number of planning steps, validation threshold, and number of replanning steps are kept constant, as shown in Table 2.

Table 2. Experimental Environment and Hyperparameter Settings

Parameter Category	Parameter Name	Setting Value
Basic Environment	Deep Learning	PyTorch

Basic Environment	Framework Programming Language	Python 3.10
Training Settings	Optimizer	AdamW
Training Settings	Initial Learning Rate	1e-4
Training Settings	Weight Decay	1e-4
Training Settings	Batch Size	32
Training Settings	Training Epochs	100
Training Settings	Dropout	0.10
Training Settings	Gradient Clipping Threshold	1.0
State Graph Encoding	Node Embedding Dimension	256
State Graph Encoding	Hidden Dimension	512
State Graph Encoding	Number of Graph Propagation Layers	2
State Graph Encoding	Number of Attention Heads	4
LLM Orchestration	Maximum Planning Steps	12
LLM Orchestration	Baseline LLM	Qwen14B
LLM Orchestration	Number of Candidate Operators	5
LLM Orchestration	Generation Temperature	0.20
LLM Orchestration	Top-p Sampling	0.90
Neural Verifier	Verifier Hidden Dimension	512
Neural Verifier	Trust Score Threshold	0.75
Trusted Execution Control	Execution Gate Threshold	0.60
Trusted Execution Control	Maximum Number of Replanning Attempts	3

4.2 Experimental Results and Analysis

To ensure sufficient horizontal reference for the performance analysis of enterprise-level ETL automatic orchestration and trusted execution, this paper selects representative studies in related areas such as data integration process generation, data preprocessing pipeline construction, data preparation feedback optimization, computation pipeline debugging, interactive data transformation, automated data cleaning, and concept model-driven ETL generation as comparison objects. The comparison process revolves around four indicators: PAcc, VAcc, CSR, and ESR, to characterize the comprehensive performance of different methods in process planning, verification, discrimination, constraint satisfaction, and execution completion. The experimental results are shown in Table 3.

Table 3. Comparative results of related methods and the proposed method

Author	PAcc	VAcc	CSR	ESR
Steiner et al.[8]	88.41	84.62	85.33	82.75
Giovanelli et al.[9]	84.37	80.52	82.16	78.94
Konstantinou et al.[10]	82.65	83.10	83.48	80.21
Lourenço et al.[11]	80.92	82.44	81.67	79.36
Kandel et al.[12]	78.34	75.21	76.58	74.12
Mahdavi et al.[13]	84.91	85.03	86.21	81.74
Muñoz et al.[14]	76.88	73.46	78.04	72.95
Ours	92.73	91.86	93.14	90.58

The proposed method demonstrates superior overall performance in terms of process planning accuracy, validation, discrimination capabilities, constraint satisfaction, and execution success rate. These results indicate that LLM-guided planning can effectively enhance the automatic orchestration capabilities of enterprise ETL tasks, while the neural validator further strengthens the credibility and execution stability of the generated process, enabling the model to possess more reliable processing capabilities under complex data dependencies, quality constraints, and task scheduling scenarios.

To further analyze the impact of key modules in this paper's method on the automatic orchestration and trusted execution process of enterprise ETL, this paper constructs an ablation setting based on a unified ETL state diagram representation, an LLM memory feedback mechanism, and a neural verifier and execution gating mechanism. Table 4 presents the performance under different module configurations, characterizing the role of each component in process planning, trusted verification, constraint satisfaction, and execution stability.

Table 4. Ablation results of key components

Ablation Setting	PAcc	VAcc	CSR	ESR
Full method	92.73	91.86	93.14	90.58
w/o Unified ETL State Graph	88.46	87.13	88.75	86.42
w/o LLM Memory Feedback	89.12	88.36	89.67	87.51
w/o Neural Verifier and Execution Gate	86.94	83.58	84.21	81.76

The ablation results in Table 4 demonstrate that the proposed complete method exhibits superior overall performance in terms of process planning accuracy, verification, discrimination capabilities, constraint satisfaction, and execution success rate. The unified ETL state diagram provides the LLM orchestration module with a structured foundation of data semantics and dependencies, while the memory feedback mechanism enhances the orchestration process's ability to utilize historical actions and intermediate states. The neural validator and execution gating mechanism further improve the reliable screening and execution control capabilities of candidate operations, thus enabling the method to demonstrate more stable and reliable task processing performance in complex enterprise ETL automated orchestration scenarios.

Figure 3 illustrates the discriminative performance of the neural validator in the constraint satisfaction and semantic consistency space. Acceptable operations are mainly concentrated in the high constraint satisfaction and high semantic consistency regions, reprogramming operations are distributed in the intermediate transition region, and blocked operations are mostly located in the low confidence region, indicating that the neural validator can clearly distinguish candidate ETL operations at different confidence levels.

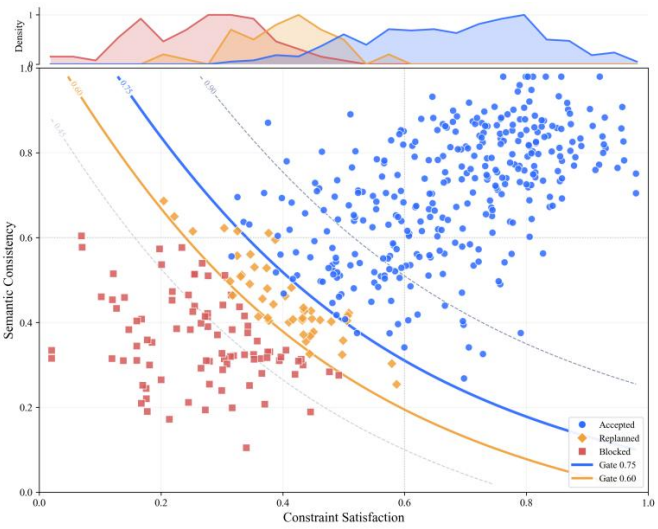

Figure 3. Neural validator constraint discrimination boundary visualization experiment

These results demonstrate that the proposed method can effectively identify candidate operations that satisfy business semantics, structural constraints, and execution conditions, and can screen and control operations with potential risks. The combination of the neural validator and the execution gating mechanism enhances the confidence discrimination capability of the ETL orchestration process, enabling automatically generated task plans to have better execution reliability and process security.

To analyze the impact of LLM generation temperature on enterprise ETL orchestration quality, this paper further examines the changes in candidate operation generation stability and constraint satisfaction capability under different temperature settings. Generation temperature directly affects the diversity of LLM-generated operation selection; too low a temperature may limit the flexibility of task planning, while too high a temperature may increase the probability of generating unstable candidate operations. Sensitivity analysis of this hyperparameter further illustrates the adaptability and robustness of the proposed method to key generation parameters in the automatic orchestration stage, as shown in Figure 4.

Experimental results demonstrate that the proposed method maintains good constraint satisfaction under appropriate LLM generation temperature settings. Reasonable temperature parameters help balance the determinism and exploratory nature of the candidate ETL operation generation process, enabling the LLM orchestration module to generate operation plans that better meet structural constraints and business semantic requirements while maintaining orchestration stability. This enhances the reliability and trustworthy execution capability of the enterprise's automated ETL orchestration process.

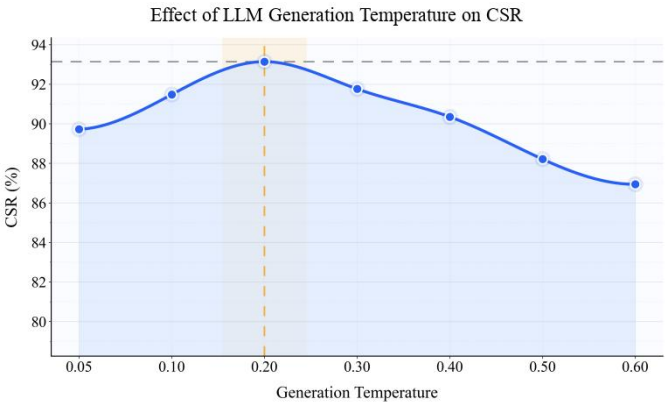

Figure 4. The effect of LLM generation temperature on CSR

5. Conclusion

This paper proposes an intelligent framework based on an LLM orchestration module and a neural verifier to address the needs of enterprise-level ETL automatic orchestration and trusted execution. This framework aims to solve problems such as heterogeneous data sources, implicit business semantics, complex task dependencies, diverse quality constraints, and difficulty in verifying execution processes in complex enterprise data environments. The method unifies the representation of enterprise data objects, field semantics, transformation relationships, and quality rules into a reasonable ETL state space. This allows the automatic orchestration process to move beyond direct generation from natural language to scripts, shifting towards process planning based on structured state understanding, task decomposition, dependency modeling, and constraint awareness. In this way, the LLM orchestration module can generate more executable ETL operation sequences in multi-source data and complex business contexts, while the neural validator further judges the semantic consistency, structural legality, and constraint satisfaction of candidate operations, thereby improving the reliability of the automatically generated process.

From a methodological design perspective, the main value of this paper lies in combining the task understanding and planning capabilities of a large language model with the trusted judgment capabilities of a neural verification mechanism, forming a closed-loop execution paradigm for enterprise data engineering scenarios. A unified ETL state diagram provides the LLM orchestration module with a comprehensive representation of data schemas, field semantics, task dependencies, and quality constraints, enabling it to utilize more complete contextual information during orchestration. The LLM memory feedback mechanism enhances the system's ability to utilize historical actions, intermediate results, and validation feedback, helping to reduce redundant planning and unstable operations. The neural verifier and execution gating mechanisms further transform trustworthiness judgments into execution control signals, enabling the ETL process to identify potential risks before execution, implement constraint control during execution, and ensure consistency between data results and target warehouse requirements after execution.

Comparative experimental results show that the proposed method achieves better overall performance in terms of process

planning accuracy, validation discrimination capability, constraint satisfaction, and execution success rate, demonstrating that this method can effectively adapt to the complex task requirements of enterprise ETL automated orchestration. Compared to processing methods that rely solely on static rules, fixed templates, or single model generation, this method emphasizes the synergistic relationship between business semantic understanding, structural dependency modeling, and trustworthiness verification, giving enterprise data processing processes stronger automation capabilities, stability, and traceability. For applications such as enterprise data governance, data warehouse construction, intelligent report generation, data quality management, and business decision support, this research can reduce manual configuration and maintenance costs, improve the reliability of data flow processes, and provide methodological support for building a more intelligent, controllable, and trustworthy data infrastructure.

Future research can be further expanded to enterprise data environments with larger scales, higher dynamism, and greater business complexity. On the one hand, subsequent work can introduce more granular business rule modeling and cross-system semantic alignment mechanisms, enabling the LLM orchestration module to better adapt to field drift, changes in indicator definitions, and multi-department data collaboration. On the other hand, the trusted execution process can be further combined with audit logs, access control, data lineage tracing, and anomaly recovery strategies to form an automated governance closed loop covering the data lifecycle. As enterprise data platforms gradually develop towards intelligence, autonomy, and trustworthiness, the ETL automatic orchestration method based on an LLM orchestration module and a neural verifier is expected to be further extended to scenarios such as real-time data pipelines, cross-cloud data integration, intelligent data platforms, and enterprise-level AI data preparation, providing a more reliable intelligent execution foundation for complex data engineering tasks.

References

[1] Y. Ge, Y. Liu, Z. Ye, et al., "Text-to-pipeline: Bridging natural language and data preparation pipelines, " arXiv preprint arXiv:2505.15874, 2025.

[2] M. Di Profio, M. Zhong, Y. Sripada, et al., "FlowETL: An Autonomous Example-Driven Pipeline for Data Engineering, " arXiv preprint arXiv:2507.23118, 2025.

[3] L. Liu, S. Hasegawa, S. K. Sampat, et al., "Autodw: automatic data wrangling leveraging large language models, " in Proceedings of the 39th Institute of Electrical and Electronics Engineers/Association for Computing Machinery International Conference on Automated Software Engineering, 2024, pp. 2041-2052.

[4] M. Parciak, B. Vandevoort, F. Neven, et al., "Schema matching with large language models: an experimental study, " arXiv preprint arXiv:2407.11852, 2024.

[5] H. Zhang, Y. Dong, C. Xiao, et al., "Large language models as data preprocessors, " arXiv preprint arXiv:2308.16361, 2023.

[6] M. Spreafico, L. Tassini, C. Sancricca, et al., "Lost in the Pipeline: How Well Do Large Language Models Handle Data Preparation?, " arXiv preprint arXiv:2511.21708, 2025.

[7] H. Foidl, V. Golendukhina, R. Ramler, et al., "Data pipeline quality: Influencing factors, root causes of data-related issues, and processing problem areas for developers, " Journal of Systems and Software, vol. 207, p. 111855, 2024.

[8] Y. Zhang, A. Floratou, J. Cahoon, S. Krishnan, A. C. Müller, D. Banda, F. Psallidas, and J. M. Patel, "Schema Matching using Pre-Trained Language Models," in Proc. IEEE 39th International Conference on Data Engineering (ICDE), 2023, pp. 1558-1571.

[9] J. Giovanelli, B. Bilalli, and A. Abello, "Data pre-processing pipeline generation for AutoETL, " Information Systems, vol. 108, p. 101957, 2022.

[10] N. Konstantinou and N. W. Paton, "Feedback driven improvement of data preparation pipelines, " Information Systems, vol. 92, p. 101480, 2020.

[11] R. Lourenço, J. Freire, and D. Shasha, "Bugdoc: A system for debugging computational pipelines, " in Proceedings of the 2020 Association for Computing Machinery Special Interest Group on Management of Data International Conference on Management of Data, 2020, pp. 2733-2736.

[12] S. Kandel, A. Paepcke, J. Hellerstein, et al., "Wrangler: Interactive visual specification of data transformation scripts, " in Proceedings of the Association for Computing Machinery Special Interest Group on Computer-Human Interaction Conference on Human Factors in Computing Systems, 2011, pp. 3363-3372.

[13] M. Mahdavi and Z. Abedjan, "Baran: Effective error correction via a unified context representation and transfer learning, " Proceedings of the Very Large Data Base Endowment, vol. 13, no. 12, pp. 1948-1961, 2020.

[14] L. Muñoz, J. N. Mazón, and J. Trujillo, "Automatic generation of ETL processes from conceptual models, " in Proceedings of the Association for Computing Machinery Twelfth International Workshop on Data Warehousing and Online Analytical Processing, 2009, pp. 33-40.