HAL: A Centralized Multi-Agent Large Language Model System for Reliable Natural Language Hybrid Data Query and Decision Support in Power Grid Command Centers

Hua Li
Electric Power Research Institute, China Southern Power Grid Company Limited, China
lih@csg.cn

Yuan La
Electric Power Research Institute, China Southern Power Grid Company Limited, China
layuan@csg.cn

Wanci Zhang
Electric Power Research Institute, China Southern Power Grid Company Limited, China
WanciZhang126@outlook.com

Zhengrong Wu
lectric Power Research Institute, China Southern Power Grid Company Limited, China
wuzr@csg.cn

Song Wang*
lectric Power Research Institute, China Southern Power Grid Company Limited, China
wangsong@csg.cn

Abstract: *Command centers of modern power transmission and distribution systems ingest massive heterogeneous data-structured measurements e.g., Supervisory Control and Data Acquisition (SCADA), topological models Geographic Information System (GIS), and unstructured operational logs-creating a critical challenge for timely, accurate, and explainable decision support. Existing single-tool or siloed approaches fail to jointly interpret natural language queries over hybrid data while ensuring reliability and traceability. This study introduces Hybrid Data Agent (HAL), a centralized multi-agent Large Language Model (LLM) system designed for natural language hybrid data querying and decision support in power grid command environments. HAL employs an LLM-based central coordinator to decompose user queries into subtasks and dispatch them to specialized expert agents-such as Structured Query Language (SQL) execution, semantic document retrieval, time-series anomaly detection, and causality reasoning-then fuses their outputs into coherent, verified responses. The system is evaluated on a high-fidelity simulated benchmark reflecting realistic grid operational scenarios, including electrical-physical events (e.g., current surges and protection switching). Comparative experiments against baselines (generic LLM agents and traditional pipelines) show that HAL substantially improves query success rate, answer accuracy, and robustness, while maintaining explainability via structured "think-act-observe" execution logs. Contributions include: 1) a distributed yet centrally coordinated architecture combining LLM planning with deterministic domain tools; 2) a hybrid data fusion mechanism aligning structured and unstructured evidence for natural language interfaces; 3) an explainable decision support engine tailored to power grid command centers. HAL reduces cognitive load on operators, accelerates fault diagnosis, and strengthens real-time operational intelligence, making it suitable for deployment in intelligent information systems for critical infrastructure.*

Keywords: *Natural language query, hybrid data fusion, large language model agent system, decision support system, power grid command and control, distributed intelligent agents, explainable information retrieval.*

Received August 08, 2025; accepted April 01, 2026
https://doi.org/10.34028/iajit/23/5/10

1. Introduction

The power transmission and distribution network is the core hub connecting the power generation side and the power consumption side, and its safe, stable, and economic operation is crucial for social and economic development. As the "Nerve Center" of power grid operation, the power transmission and distribution production command center undertakes key functions such as power grid status monitoring, fault emergency handling, and resource optimization scheduling. In recent years, with the wide application of sensor technology, the Internet of Things, and the Advanced Metering Infrastructure (AMI), the production command system has gathered a large amount of multi-source and heterogeneous data, including structured data such as real-time measurements from Supervisory Control and Data Acquisition (SCADA) and geographical topologies from Geographic Information System (GIS), as well as a large number of unstructured texts such as operation logs and fault reports.

In the face of such a complex and heterogeneous data environment, production commanders face the

severe challenge of "Information Overload" in the decision-making process. For example, when it is necessary to locate a complex problem, such as "Query the 10kV line that was faulty due to strong wind weather last week and affected key industrial users, and find the relevant emergency repair records", the commander must perform a series of cumbersome manual operations: write Structured Query Language (SQL) statements in the database, associate weather data from the meteorological system, identify the user scope in GIS, and finally search for emergency repair reports in the document library.

This series of manual and cross-system operation processes not only consume time and effort, but also are extremely prone to information omission or errors due to human negligence, seriously affecting the efficiency and accuracy of emergency response.

To solve this problem, there are obvious limitations in existing technical means. Traditional query analysis methods, such as keyword-based text retrieval or fixed SQL query templates, lack the ability to deeply understand the semantic meaning of natural language and cannot handle complex queries with colloquial and fuzzy expressions. In recent years, the emergence of Large Language Models (LLMs) represented by the Generative Pre-Trained Transformer (GPT) series has provided new possibilities for solving this problem. In theory, LLMs can directly translate natural language into executable code such as SQL. However, directly applying a single LLM to the data-driven and safety-critical industrial field faces four core bottlenecks:

Factual Hallucination: LLMs may generate answers that seem reasonable but do not match real-time and actual data, which is unacceptable in the power system that requires absolute accuracy Ji *et al*. [7].

Limited Real-time Data Access: The internal knowledge of LLMs is based on their static pre-trained corpus and cannot directly access and process the internal and real-time private databases of enterprises.

Lack of Domain-Specific Accuracy: For the professional terms, operation procedures, and complex equipment topology associations in the power transmission and distribution field, the understanding of general LLMs may deviate, resulting in incorrect query logic Cheng *et al*. [4].

Low Transparency: The "Black Box" feature of LLMs makes their reasoning process difficult to trace and verify, and cannot meet the requirements of production commanders for high reliability and audibility Gholamzadeh Khoee *et al*. [6].

To systematically address the above bottlenecks, we propose a solution based on the collaboration of Artificial Intelligence (AI) agents. We designed and implemented a new multi-agent framework called Hybrid Data Agent (HAL) (HAL-based LLM). The core idea of this framework is to use a powerful LLM as the "central coordinator" for task planning, and direct a group of "domain expert agents" with different specialties such as database query and document retrieval agents to work together, so as to build a system that can not only deeply understand the intention of natural language, but also accurately and reliably execute data operations. The main contributions of This study introduces can be summarized as the following three points:

A novel multi-agent framework HAL designed specifically for querying hybrid power transmission and distribution data is proposed. Through a structured design, this framework systematically aims to alleviate core bottlenecks such as factual hallucinations and limited real-time data access of a single LLM in industrial applications.

A dynamic task decomposition and professional agent scheduling mechanism is designed. Through a central coordination LLM, complex natural language queries are transformed into logically clear execution plans and dispatched to the most suitable domain expert agents, thus ensuring the accuracy and efficiency of the queries. Through comprehensive experimental verification and in-depth ablation/scalability studies, not only the superiority of the HAL framework compared to current mainstream methods is demonstrated, but also its zero-shot learning ability of the modular architecture in dealing with new business requirements is verified.

The experiments were conducted on a newly built high-fidelity benchmark dataset, and the results were excellent in multiple dimensions (Accuracy, Robustness, Scalability).

This work lies at the intersection of AI, natural language processing, and distributed information systems, addressing the challenge of querying and reasoning over hybrid structured and unstructured data in critical infrastructure command centers. By combining multi-agent LLM planning with deterministic domain tools, HAL delivers reliable, explainable decision support-directly aligning with The International Arab Journal of Information Technology (IAJIT) scope on intelligent information systems, hybrid data mining, knowledge-engineered query interfaces, and practical AI applications in complex operational environments.

2. Related Work

This chapter aims to sort out the technical background and cutting-edge progress related to the work of This study to clarify the positioning and contributions of the authors The review first analyzes the inherent challenges of natural language queries for hybrid data in the industrial field, then review the data interaction technologies based on LLMs, then examines the current mainstream LLM agent frameworks, and finally, by analyzing their limitations in professional fields, identify the core motivation of This study

introduces and the gaps to be addressed.

2.1. Challenges of Natural Language Queries for Industrial Hybrid Data

In modern industrial scenarios such as power transmission and distribution, data exhibits significant "Hybrid Heterogeneous" characteristics. On the one hand, high-frequency time-series data, spatial topology information, and event records from systems such as SCADA, GIS, and Outage Management System (OMS) constitute a vast amount of structured data Li *et al*. [10]. On the other hand, a large number of fault analysis reports, equipment inspection records, operation logs, and operation procedures exist in the form of unstructured text Al Mtawa *et al*. [1]. This characteristic of data mixing brings three major technical bottlenecks to the realization of unified natural language queries: Incompatible interfaces: The underlying technologies and query languages for SQL queries for structured data and keyword or vector retrieval for unstructured text are completely different, making it difficult to integrate under a single interface. Complex semantic associations: To answer complex business questions, it is often necessary to perform in-depth semantic associations across different data sources. For example, to determine the root cause of a line fault, it may be necessary to perform logical reasoning on the fault time in the database and the meteorological warning information in the document library, which cannot be automatically completed by traditional systems. High reliability requirements: Safety-critical scenarios such as industrial production command have extremely high requirements for the accuracy and traceability of query results, and any errors caused by model "Hallucinations" or misunderstandings may lead to serious consequences.

2.2. Data Interaction Technologies Based on LLM

The emergence of LLMs has brought hope for breaking through the above bottlenecks. In the field of data interaction, LLMs mainly play a role in two ways:

2.2.1. Text-to-SQL for structured data:

This technology aims to directly translate users' natural language questions into executable SQL query statements Singh *et al*. [15]. Early research relied on complex semantic parsing and rule matching, while LLM-based Text-to-SQL methods have achieved significant performance improvements on multiple standard datasets through powerful in-context learning and instruction-following capabilities Zhang and Yang [21]. However, the accuracy of such methods still faces challenges when dealing with unseen databases with complex domain-specific knowledge.

2.2.2. Retrieval-Augmented Generation (RAG) for unstructured data:

To address the problem that LLMs cannot access private and real-time knowledge, the Retrieval-Augmented Generation (RAG) technology was proposed Wan *et al*. [17]. This technology first retrieves the most relevant context information to the question from an external knowledge base such as a document library, and then provides it together with the original question to the LLM to guide it to generate answers based on the provided facts. RAG has become a standard paradigm for improving the factuality of LLM answers and reducing the hallucination problem. Although these technologies have been successful in their respective fields, they are essentially "Point Tools" designed for a single data type and are difficult to independently handle complex industrial queries that require multi-step reasoning across structured and unstructured data. Garg *et al*. [5] further demonstrate that carefully configured hybrid RAG pipelines can strengthen semantic retrieval quality and real-time technical support across heterogeneous information sources.

2.3. LLM-based Agent Frameworks and Their Limitations

To solve more complex tasks that require multi-step and tool coordination, LLM-based Agent frameworks have emerged. Such frameworks use LLMs as the "Brain" and interact with the world by calling external "Tools", and have become a hot topic in cutting-edge research Wang *et al*. [19].

The React paradigm is a pioneering work in the field of agents. It enables LLMs to reason, plan, and call tools through an iterative loop of "Think-Act-Observe" Piccialli *et al*. [12]. However, its inherent serial execution mode is inefficient when dealing with subtasks that can be parallelized. For example, if a query requires both querying a database and searching for documents at the same time, serial execution will unnecessarily increase the response time. On this basis, popular open-source frameworks such as LangChain and LlamaIndex have emerged. They provide powerful modular components and a rich tool ecosystem, greatly reducing the development threshold for agent applications. However, the general planners of these frameworks often struggle when faced with highly specialized industrial domain knowledge Suri *et al*. [16]. For example, when generating complex join queries involving multiple special power transmission and distribution data tables, its accuracy is difficult to guarantee, and it is also difficult to autonomously decide when to call a specialized "Time Series Anomaly Detection" tool rather than a general "Data Plotting" tool.

Furthermore, autonomous agents represented by Autonomous Generative Pre-Trained Transformer

(AutoGPT) focus on long-term and open-ended task exploration and execution. Although they demonstrate remarkable autonomy, their high degree of uncontrollability is a fatal weakness in industrial production command scenarios that require precise, controllable, and verifiable execution.

It is worth noting that more complex multi-agent collaboration frameworks, such as AutoGen and CrewAI, have recently emerged. These frameworks often simulate social behaviors (e.g., role-playing and multi-turn dialogues) to achieve collaboration. However, they are typically designed for open-ended task exploration or creative generation, where the collaborative process has emergent and less predictable properties. In safety-critical domains like power production command, which demand high reliability, controllability, and auditable results, this unpredictability can be a significant drawback. In contrast, the proposed HAL framework adopts a deterministic, centralized dispatch model. This "central planning - expert execution" architecture ensures that every task decomposition and execution step is clear and traceable, forming a complete audit trail. This design choice allows HAL to meet complex query demands while better aligning with the strict requirements for system predictability and reviewability in industrial scenarios, thus forming the core motivation for the architectural selection by This study.

2.4. Research Gaps and Motivation of the HAL Framework

As mentioned above, when existing research is applied to the field of power transmission and distribution production command, which requires high reliability and high professionalism, there are obvious research gaps: Limitations of the collaboration mode: Most frameworks adopt general planners and serial execution modes, making it difficult to efficiently and accurately handle complex queries that require the collaboration of multi-domain expert knowledge. Insufficient domain adaptability: The general toolkit lacks in-depth optimization for specific industrial domains such as power transmission and distribution, resulting in poor performance in key tasks such as complex SQL generation Cen *et al*. [3]. Lack of controllability and reliability: The exploratory nature of autonomous agents runs counter to the high reliability and high reviewability requirements of industrial scenarios. To fill this gap, the motivation of This study is to design a collaborative framework that combines the powerful planning ability of LLM and the precise execution ability of domain tools. The HAL framework proposed by the authors. Conducts deterministic and traceable planning by introducing a central coordination LLM and designs a set of domain-specific agents for precise execution. This "Central Planning-Expert Execution" architecture aims to deeply couple the general reasoning ability of LLM with the reliability of domain tools, thereby effectively improving the complex query ability for the hybrid data of power transmission and distribution while ensuring the accuracy and controllability of task execution.

3. Design and Implementation of the HAL Framework

To effectively address the challenges of natural language queries for hybrid data in power transmission and distribution production command, we designed and implemented a Hybrid Agentic LLM framework, abbreviated as HAL. Section 3 details its overall architecture, core components, and workflow.

3.1. Overall Architecture

The core design idea of the HAL framework is to separate complex reasoning tasks from deterministic tool execution. Its overall architecture is shown in Figure 1, which is a clear hierarchical modular system, consisting of a User Interface Layer, a Coordination and Planning Layer, an Agent Execution Layer, and a Data and Tools Layer from top to bottom.

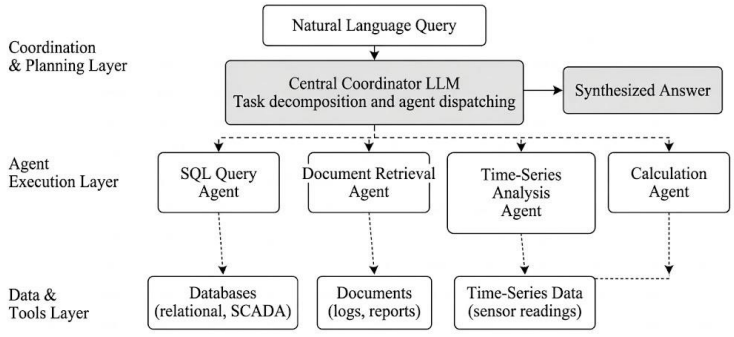

Figure 1. Overall architecture of the HAL (hybrid agentic LLM) framework.

This architecture adopts a hierarchical design, consisting of a user interface layer, a coordination and

planning layer, an agent execution layer, and a data and tools layer. The central coordination LLM, as the core, is responsible for task decomposition, agent scheduling, and answer integration, while each professional agent interacts with the hybrid data source by calling the underlying tools. The functions of the four core layers of the architecture are as follows:

The user interface layer, as the interactive front-end of the system, is responsible for receiving natural language queries input by the command personnel.

The Coordination and Planning Layer is the "brain" of the entire framework. Its core Central Coordinator LLM is responsible for deeply understanding the user's intent and decomposing complex problems (Task Decomposition) into a series of subtasks. Subsequently, it schedules (Agent Dispatching) the most suitable agents to execute according to the nature of the tasks, and finally integrates and refines (Answer Synthesis) the return results of each agent to generate the final answer.

The Agent Execution Layer is a group of expert agents each performing its own functions. The integrate four core agents: SQL Query Agent, Document Retrieval Agent, Time-Series Analysis Agent, and Calculation Agent. This specialized design ensures the accuracy and efficiency of task execution.

The Data and Tool Layer encapsulates all underlying data operations into a series of deterministic Application Programming Interface (API) functions, serving as a "toolset" that can be called by the upper-layer agents. As shown in Definition 1, this ensures the combination of the inference ability of the LLM and highly reliable data operations, guaranteeing the reliability and traceability of the entire query analysis process.

- **Definition 1: HAL Framework Core Toolset API.**

function execute_sql (query: str) returns List[Dict]
Input: A SQL query string.
Operation: Execute the query on the structured database (GIS, OMS).
Output: A list of dictionaries containing the query results, with each dictionary representing a row of data.

function vector_search(text: str, top_k: int) returns List[str]
Input: The query text and the expected number of returns.
Operation: Perform semantic similarity search in the document vector library.
Output: A list of the *k* most relevant document fragments.

function get_timeseries_data(device_id : str, start_time : str, end_time : str) returns List
Input: The device ID and the start and end times.
Operation: Extract time-series data from the SCADA database.
Output: Sequential data containing timestamps and measurement values.

function analyze_anomaly(data: List[Dict]) returns Dict
Input: A time series data in the format of get_timeseries_data.
Operation: Apply an anomaly detection algorithm for analysis.
Output: Structured information describing anomaly events (such as sudden drops or spikes).

Through the tools defined in Definition 1, the HAL framework separates the non-deterministic reasoning of the LLM from highly deterministic data operations, ensuring the reliability and traceability of the entire query analysis process.

3.2. Central Coordination of LLM and Task Planning

On top of the static architecture of HAL, its dynamic intelligence is reflected in the workflow of the central coordination LLM. To simulate the iterative reasoning process of human experts in solving complex problems, the researchers designed a cyclic decision-making framework for the LLM based on "Think-Act-Observe". The specific process of this framework is shown in Figure 2.

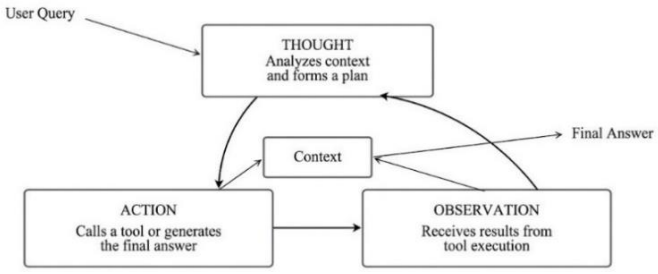

Figure 2. The "think-act-observe" cyclic workflow of the central coordination LLM.

This cycle is the core mechanism for the HAL framework to achieve complex task planning. The LLM dynamically solves problems through iterative "Thinking" (planning), "Acting" (tool invocation or

generating answers), and "Observing" (obtaining results), and corrects the execution path through the continuously updated context.

This framework does not generate all execution steps at once, but dynamically plans and adjusts through a continuous cycle:

Thought: The starting point of the cycle. The LLM analyzes all the information currently available (i.e., the context), conducts reasoning, and generates the next action plan.

Action: Based on the plan of "Thought", the LLM decides to invoke a specific tool or directly generate the final answer (Final Answer).

Observation: Receive the product of "Action" (such as the tool return result) and add it back to the context to provide a decision basis for the next round of "Thought".

The complete logic of this "Think-Act-Observe" cycle is formally defined in Algorithm 1. This design endows the HAL framework with the key advantages of robustness in handling unknown situations and transparency and interpretability of the solution process.

Algorithm 1. Task planning and execution process based on LLM.

Input: User Query UserQuery
Output: Final Answer FinalAnswer or Error Message

1. Initialize the context ← , including UserQuery
2. for i←1N do:
3. Thought ← LLM_Think (Context) // The LLM conducts thinking and planning
4. Action ← LLM_Act (Context, Thought) // The LLM decides the next action (invokes a tool or outputs an answer)
5. if is_FinalAnswer(Action) then:
6. FinalAnswer ← get_Answer_from (Action)
7. return FinalAnswer
8. end if
9. Observation - Execute_Tool (Action) // Execute the tool call and obtain the result
10. Context.append(Thought, Action, Observation) // Add the thought, action, and observation result to the context
11. end for
12. return "Failure: Max rounds reached"

3.3. Specialized Agent Design

After clarifying the macro-architecture and core workflow of HAL, this section will deeply analyze the internal design and implementation details of each specialized agent that constitutes the execution layer of the agent. the researchers adopt a collaborative mode of "general commander-domain expert". Inside some agents (such as the SQL query agent), another LLM fine-tuned with specific instructions is even encapsulated. This hierarchical and dedicated design is the key to ensuring the high precision and high efficiency of the entire framework.

3.3.1. SQL Query Agent

Core Competency: Text-to-SQL Conversion

Enhancement Mechanisms: To improve the accuracy of SQL generation in complex power transmission and distribution database schemas, the researchers employ the following key techniques: Dynamic Schema Injection: When passing the task of generating SQL to the LLM, not only the user's natural language question is passed, but also the table structure information (table names, field names, field types, field comments, foreign key relationships) most relevant to the question is dynamically extracted from the database metadata and provided to the LLM as part of the context. This provides the necessary knowledge for the LLM to generate syntactically correct and logically sound SQL.

Few-shot Prompting: Several high-quality examples of "natural language question-SQL query" pairs are built into the prompt. These examples cover common query types (such as single-table queries, multi-table joins, aggregation queries), and can effectively guide the LLM to learn the query patterns, naming conventions, and business logic of a specific database.

Query Verification and Repair Mechanism: Before the generated SQL is formally executed, it will pass through a fast syntax checker. If the verification fails, the system can feedback the error message together with the original SQL and the question to the LLM and request it to make corrections. This simple self-correction loop significantly improves the success rate of queries.

Mathematical Representation: Let the user's natural language question be Q_{user}, the relevant database Schema information be S_{db}, and the few-shot example set be $E_{few\text{-}shot}$. The process of generating the SQL query can be formally represented as a conditional generation model as Equation. (1):

$$SQL_{generated} = LLM\left(f\left(Q_{user}, S_{db}, E_{few-shot}\right)\right) \quad (1)$$

where f is a function that combines the question, Schema, and examples into a complete prompt.

3.3.2. Document Retrieval Agent

Core Competency: Dense Vector-based Semantic Retrieval Wu *et al.* [20].

Workflow: It is divided into two stages: offline indexing and online querying.

Offline Processing (Indexing): This stage is completed before the system is deployed. All unstructured documents (such as.txt,.pdf,.docx) are uniformly preprocessed, including text extraction, removal of irrelevant symbols, and then chunked according to meaningful paragraphs or a fixed length. Then, a pre-trained sentence embedding model (such as Sentence-BERT) is used to convert each text chunk into a high-dimensional dense vector (Embedding). Finally, all text chunks and their corresponding vectors

are stored in a dedicated vector database (such as FAISS, Milvus), and an efficient index is established.

Online Querying: When receiving the query text q from the central LLM, it is converted into a query vector $\vec{v}_q$ using the same embedding model. Then, in the vector database, the query vector is used to calculate the similarity (usually cosine similarity) with all document chunk vectors in the database, and the Top-K text chunks with the highest similarity are quickly retrieved and returned as the most relevant context information.

Mathematical representation: Let the set of document block vectors be $V=\{\vec{v}_1,\vec{v}_2,\ldots,\vec{v}_n\}$, where $\vec{v}_i$ is the vector obtained by transforming the document block d_i through the embedding function. For the query text q, its query vector is $\vec{v}_q$. The retrieval process aims to find a subset of document blocks D_{topK} such as Equation. (2):

$$D_{topK} = \underset{d_i \in D, |topK|}{argmax}\ similarity(\vec{v}_q, \vec{v}_i) \quad (2)$$

Among them, the cosine similarity calculation formula such as Equation. (3):

$$similarity(\vec{v}_q, \vec{v}_i) = \frac{\vec{v}_q \cdot \vec{v}_i}{\|\vec{v}_q\|\|\vec{v}_i\|} \quad (3)$$

4. Experiments and Result Analysis

To quantitatively evaluate the effectiveness of the proposed HAL framework designed a series of comparative experiments. This section will detail the simulation dataset these researchers constructed, the evaluation metrics adopted, the baseline models selected, and the final experimental results and analysis.

4.1. Experimental Setup

This section details the comprehensive experimental setup designed to rigorously evaluate the HAL framework. And describe the custom-built benchmark dataset, the specific implementation of the models and tools, the evaluation metrics, and the baseline models used for comparison.

4.1.1. High-Fidelity Hybrid Query Benchmark Dataset

To conduct a realistic and comprehensive evaluation, constructed a high-fidelity hybrid query benchmark dataset oriented towards the power transmission and distribution production command scenario Qiao *et al*. [13]. This dataset is designed to simulate the real-world query challenges faced by operators and is composed of the following components:

Structured Database: This database integrates core data from a GIS and an OMS Avcı [2].

Scale and Complexity: The database comprises 12 core tables with a total of 115 fields. Key tables include line-info, transformer-status, grid-topology, outage-records, and key-user-geodata. Complex relationships are established via foreign keys to support multi-table join queries.

Unstructured Document Library: We conducted collected and curated a substantial volume of domain-specific documents Ko *et al*. [8].

Scale and Content: The library contains 350 documents (in .pdf and .docx formats), totaling approximately 500,000 words. The content covers historical fault analysis reports, equipment operation procedures, grid emergency plans, and daily operational logs.

Time-Series Database: This component includes equipment measurement data collected from a SCADA system Wang and Zhao [18].

Data Characteristics: We conducted simulated time-series data for 200 key monitoring points over the past year with a 5-minute sampling frequency, including key metrics such as current, voltage, and active/reactive power.

Query-Answer Pairs: Based on the data sources above, we conducted meticulously designed 150 query-answer pairs as the evaluation benchmark. These queries are categorized into three levels of difficulty based on reasoning complexity and the number of required data sources:

Simple Queries (40%): Typically involve direct information retrieval from a single data source (e.g., "Query the current load rate of the No. 10 main transformer").

Medium Queries (40%): Require data aggregation or multi-table joins within a single data source (e.g., "List all lines that had an outage due to lightning last month"), or simple correlation across two data sources.

Complex Queries (20%): Demand multi-step reasoning and causal analysis across multiple heterogeneous data sources (e.g., "Analyze the sudden current increase on the 'South City Line' last Tuesday morning and determine its possible cause by consulting the operation logs").

Reproducibility Statement: Due to the sensitive nature of the simulated grid operational data, this dataset cannot be made fully public. However, to ensure the reproducibility of the work of the present study will provide the complete database schema, the topic distribution of the document library, and 20 anonymized query-answer pair samples as supplementary material upon acceptance of the manuscript. This will enable other research teams to construct benchmark datasets with similar complexity and feature distributions for fair comparative studies.

4.1.2. Model and Tool Implementation Details

To ensure the transparency and reproducibility of the experiments that were conducted this section details the specific implementation of the key components in

the HAL framework.

Large Language Model (LLM) configuration:

Model Selection: In the present study the Central Coordinator LLM within the HAL framework, as well as the LLMs used inside its specialized agents (e.g., the SQL Agent), is GPT-4-Turbo (gpt-4-1106-preview), accessed via the OpenAI API. All baseline models use the same LLM to ensure a fair comparison.

Prompt Engineering: We did not perform any fine-tuning on the LLM. The framework's domain adaptability and high performance are primarily achieved through sophisticated prompt engineering. The core prompt template for the Central Coordinator LLM is structured as follows:

Algorithm 2. Prompt-based reasoning step for central coordinator.

Input: user_query, available_tools, context_history
Output: action (Tool call or Final Answer)
1: function GenerateNextStep(user_query, available_tools, context_history)
2: system_role ← "You are an expert dispatcher for a power grid command center…"
3: prompt ← FormatTemplate(system_role, user_query, available_tools, context_history)
4: // LLM generates both Thought and Action
5: thought, action ← LLM(prompt)
6: return action
7: end function
8:
9: function FormatTemplate (…)
10: // Constructs the full text prompt as shown in the original template
11: return formatted_text_prompt
12: end function

Core Toolset Implementation:

Vector-search (Document Retrieval): Unstructured documents were encoded into 768-dimensional vectors using the Sentence-BERT (all-mpnet-base-v2) model. These vectors are stored in a local vector index built with FAISS (Facebook AI Similarity Search) to enable efficient semantic similarity searches.

Analyze-Anomaly (Time-Series Analysis): This tool utilizes the Isolation Forest algorithm from the scikit-learn library. It efficiently detects anomalies in time-series data (such as current surges or voltage drops) and returns a structured dictionary containing the anomaly's timestamp, direction, and magnitude.

Execute-SQL and get-timeseries-data: These tools execute queries on a PostgreSQL relational database and an InfluxDB time-series database, respectively.

4.1.3. Evaluation Metrics

To comprehensively and objectively evaluate model performance, designed three core quantitative metrics:

Query Success Rate (QSR): Measures the robustness of the system Mahmud *et al*. [11]. It refers to the proportion of queries for which the model can successfully complete all internal steps (without code execution errors, getting stuck in infinite loops, or timing out) and generate the final answer in the total number of queries. This metric does not evaluate the correctness of the answer content.

Execution Accuracy (EA): Measures the quality of the executable code (mainly SQL) generated by the model Sequeda *et al*. [14]. Its calculation formula is: the number of SQL codes with correct logic generated/the total number of queries that require SQL generation×100%. This metric is only applicable to tasks involving code generation.

Answer Accuracy Score (AAS): Measures the comprehensive quality of the final answer Lee *et al*. [9]. We adopted the method of manual scoring. Two experts in the power transmission and distribution field scored each final answer "blindly". The scoring criteria are as follows:

2 Points: The answer is completely correct, the information is complete, and the expression is clear.

1 Point: The answer is partially correct, or contains key information but there are omissions or minor errors.

0 Points: The answer is completely wrong, or fails to answer the core of the question.

The final AAS is the average of the scores of the two experts.

4.1.4. Baseline Models

To verify the superiority of the framework, has been selected the following three representative methods for comparison:

Single LLM: Directly use the GPT-4 model (without any external tools), and require it to answer questions only based on the provided context information and its internal knowledge.

General-Purpose Agent (GPA): To ensure a fair comparison, we conducted implemented a generic React-style agent using the popular LangChain framework. Crucially, this GPA was configured with the exact same LLM (GPT-4-Turbo) and the exact same underlying toolset as the HAL framework. The only difference is its planning mechanism; the GPA relies on LangChain's general-purpose planner, which lacks the domain-specific optimizations of the HAL architecture. This baseline is designed to validate the superiority of the carefully designed "central coordination-expert execution" architecture over a generic agent counterpart.

Traditional approach: Simulate the traditional workflow. For simple queries, use fixed keyword matching and SQL templates; for complex queries, it is considered unable to handle.

4.2. Overall Performance Comparison and In-depth Analysis

We conducted comprehensively tested HAL and the baseline model on the evaluation set. First, Table 1

macroscopically shows the comprehensive performance of each model.

Table 1. Overview of the comprehensive performance of each model.

Model	QSR	AAS	EA
HAL	96.0%	1.85	92.1%
General-purpose Agent (GPA)	78.3%	1.33	68.4%
Single-LLM	78.0%	1.35	73.7%
Traditional	40.0%	0.80	100.0%

The results in Table 1 clearly show that HAL significantly leads in comprehensive performance. To further explore the performance differences of different methods on tasks of different difficulties, we conducted conducted a more fine-grained analysis, and the detailed results are shown in Table 2.

Table 2. Detailed performance comparison between hal and baseline methods on three types of query tasks.

Method	Query type	QSR (%)	EA (%)a	AAS (average score)
HAL	Simple query	100.0	95.0	1.95
	Medium query	95.0	90.0	1.80
	Complex query	90.0	-	1.75
	Overall	96.0	92.1	1.85
General-purpose agent (GPA)	Simple query	95.0	80.0	1.65
	Medium query	75.0	60.0	1.20
	Complex query	60.0	-	0.95
	Overall	78.3	68.4	1.33
Single-LLM	Simple query	95.0	85.0	1.70
	Medium query	80.0	65.0	1.30
	Complex query	50.0	-	0.80
	Overall	78.0	73.7	1.35
Traditional	Simple query	70.0	100.0	1.40
	Medium query	30.0	100.0	0.60
	Complex query	0.0	-	0.0
	Overall	40.0	100.0	0.80

Note: a) Execution Accuracy (EA): This metric is calculated only for query tasks that require the generation of executable code (mainly SQL).

Its calculation formula is: the number of codes with correct logic generated/the total number of queries that need to generate codes in this category×100%.

b) Not applicable (-): Complex queries require multi-step reasoning and tool calls across data sources (such as "query the database first and then search the documents"), and their execution process does not generate a single executable code. Therefore, the EA metric cannot be used for evaluation, and its final effect is comprehensively reflected by the AAS.

From the breakdown data in Table 2 and the intuitive comparison in Figure 3, the analysis provides deeper insights: Robustness of HAL: HAL maintains consistently high performance across tasks from simple to complex (AAS only drops slightly from 1.95 to 1.75), demonstrating its strong stability and adaptability. Bottlenecks of baseline models: All baseline models exhibit significant performance degradation when faced with complex queries. In particular, traditional methods completely fail on complex tasks (QSR is 0.0), highlighting the necessity of modern LLM-based methodologies. Challenges of complex tasks: The inapplicability of the EA metric for complex queries (Footnote b) itself illustrates the nature of such tasks-they require high-quality synthesis, analysis, and reasoning rather than simple step-by-step execution. The significant advantage of HAL in the AAS metric (1.75 vs 0.95/0.80) directly demonstrates its core competitiveness in this regard.

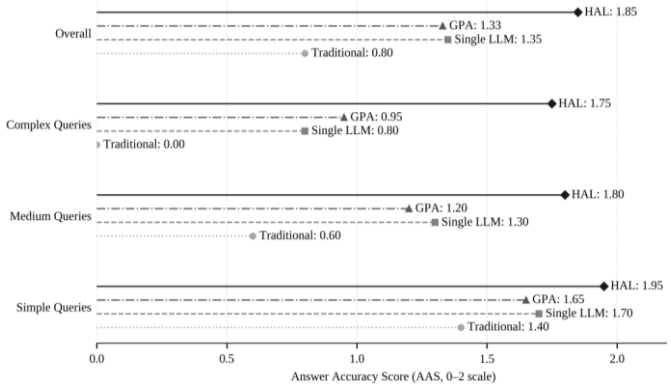

Figure 3. Comparison of AAS scores for different methods on the answers of various queries.

4.3. Case Study: End-to-End Parsing of Complex Queries

In order to go beyond pure quantitative metrics and more intuitively demonstrate the capabilities and advantages of the HAL framework in handling complex real-world problems, this section presents its complete workflow end-to-end through a specific, highly complex case. We selected a typical query task that requires causal analysis across data sources, which cannot be directly solved by a single model or traditional tools.

Query task: "Analyze the sudden increase in current of the 'South City Line' around 10 am on June 15, 2025, and combine the operation logs to determine its possible causes." The challenge of this task is that it requires the system to not only process and analyze structured time-series data SCADA, but also understand the analysis results and use these results to find relevant evidence in unstructured text logs, and finally form a logical causal explanation. The detailed execution log chain of the HAL framework for handling this task is shown in Figure 4.

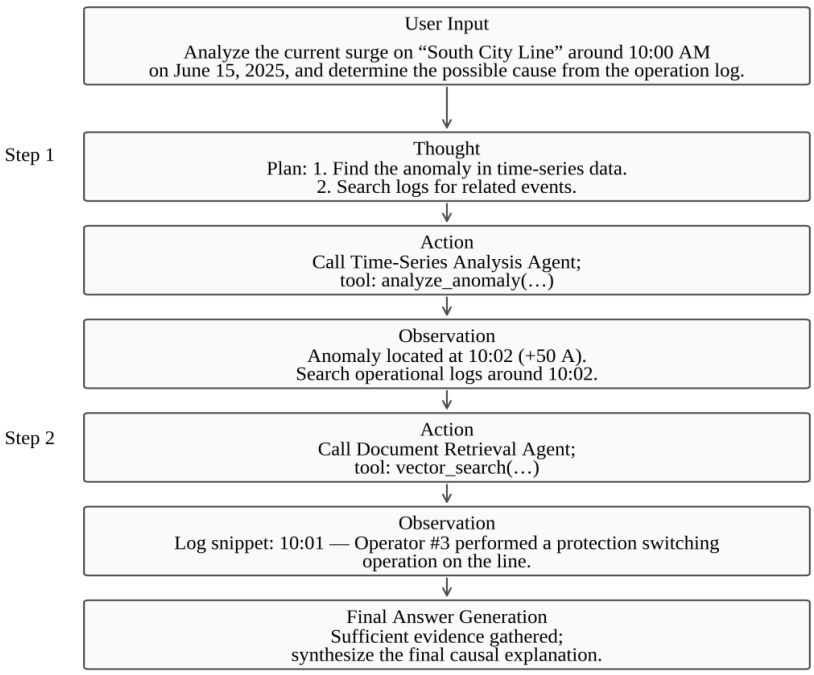

Figure 4. The "think-action-observe" execution log chain of a complex query case.

The following discussion breaks down the internal operation process of the HAL framework step by step in combination with the content of Figure 4.

Step 1. Task understanding and initial planning (cycle 1-thought). The user's natural language query is first passed to the central coordinating LLM. The LLM enters the first "Thought" stage of the "Thought-Action-Observation" cycle. Through in-depth semantic analysis of the query intent, it understands that this is a causal analysis task and autonomously decomposes it into two logically related subtasks: First, it must accurately locate the abnormal event of "sudden increase in current" in the time-series database and obtain its precise time and magnitude. Then, using the located event information, it searches the text log library for operation records that may have caused this event near that time point.

Step 2. Time-series data analysis and anomaly location (cycle 1-action and observation) Based on the above plan, the LLM enters the first "Action" stage. It determines that subtask 1 belongs to the category of time-series data analysis, so it decides to dispatch the Time-Series Analysis Agent. It sends instructions to this agent to call the get-timeseries-data and analyze-anomaly tools. After receiving the tool execution results, the system enters the "Observation" stage. The return information received by the central coordinating LLM is a structured data object: (Event: sudden increase in current, Time: 10:02, Magnitude: +50A). This observation result accurately answers the first part of the task.

Step 3. Re-Planning based on new information (cycle 2-thought). This precise observation result is automatically added back to the LLM's context, becoming the key basis for the next round of decision-making. The LLM enters the second "Thought" stage. At this time, it clarifies the next goal: to find relevant text evidence based on the two core pieces of information, "10:02" and "sudden increase in current of+50A".

Step 4. Associated log retrieval and evidence discovery (cycle 2-action and observation). The LLM's second "Action" is to dispatch the Document Retrieval Agent.

It uses the newly obtained context information to intelligently construct a query for semantic search's

Operation records of sudden increase in current on the South City Line around 10:02' and calls the vector-search tool. The Document Retrieval Agent performs an efficient semantic similarity search in the vectorized log library and returns the most relevant log fragment as the second "Observation" result: "Log fragment: 10:01-Operator #3 performed protection switching operation on the South City Line..."

Step 5. Answer Integration and Generation (Final Answer Generation). At this point, the LLM determines during the new "Thought" phase that it has collected two key information fragments sufficient to form a complete causal chain. The task can be terminated. It stops invoking tools and instead activates its "Answer Synthesis" ability. It logically concatenates and organizes the language of the two observations to generate a final, smooth, and accurate natural language answer, which is presented to the operator through the user interface: "At 10:02 on June 15, 2025, a sudden+50A current increase was detected on the 'South City Line'. This event was most likely caused by the protection switching operation performed by Operator #3 on this line at 10:01." Case Study Conclusion: This case clearly demonstrates the significant advantages of the HAL framework over other methods: Autonomous Planning and Dynamic Execution: Instead of executing a rigid preset script, HAL, like a human expert, dynamically plans the next action based on the available information, demonstrating a high degree of intelligence and flexibility. Fact-based Reasoning: Every component of the final answer is firmly anchored in the real data (+50A magnitude, 10:02 time, 10:01 operation log) obtained from the underlying data tools, effectively eliminating the common "hallucination of facts" problem in LLMs. Interpretability and Traceability: As shown in Figure 4, the entire complex problem-solving process is completely recorded as a clear and auditable "Thought-Action-Observation" log chain. This not only makes the source of the final answer completely transparent and traceable but also provides valuable basis for subsequent process optimization and fault troubleshooting.

4.4. Ablation Experiments

To verify the necessity of each component in the HAL framework, we conducted ablation experiments, and the results are shown in Table 3.

Table 3. Results of the ablation experiments on the HAL framework.

Configuration	Answer accuracy score (AAS)	Performance degradation relative to the complete framework
HAL (complete framework)	1.85	-
1. Remove vector search (document tool)	1.32	-0.53
2. Remove analyze anomaly (analysis tool)	1.59	-0.26
3. Replace the central coordinator with a general LLM	1.12	-0.73

The experimental results clearly show that removing any one of the specialized tools Configurations 1 and 2 leads to a significant performance decline. Replacing the optimized central coordinator with a general LLM Configuration 3 results in a precipitous drop in performance. This proves that the high performance of HAL does not come from a single component, but from its overall architecture-the inevitable result of a powerful central coordinator working in tandem with a complete set of specialized tool sets.

5. Discussion

The experimental results clearly show that when the proposed HAL framework processes complex mixed power transmission and distribution data query tasks, its performance significantly exceeds that of current mainstream methods. Section 5 aims to deeply analyze the internal reasons for the success of the HAL framework, explore the superiority of its design concept from three core dimensions: factual accuracy, system transparency, and architecture scalability, and objectively analyze its current limitations. First, to understand the fundamental reasons for the success of the HAL framework from a macro perspective, it is being re-examine its essential differences in design philosophy with the baseline method through Figure 5.

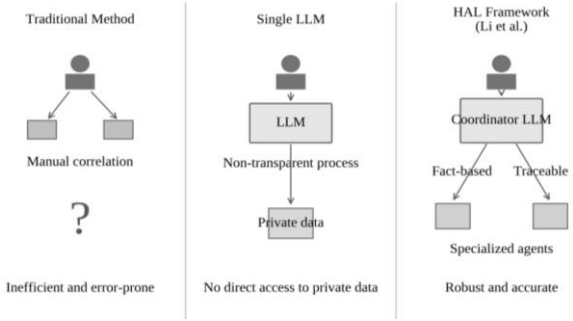

Figure 5. Comparison of the fundamental differences in the processing flow between the HAL framework and the baseline method.

As shown in Figure 5, the success of HAL is not accidental. Instead, its architecture design directly circumvents the inherent defects of traditional methods and single LLMs.

5.1. Ensuring Factual Accuracy: From Generation to Verification

The most core risk of a single LLM lies in "fact

hallucination". The HAL framework fundamentally alleviates this problem by reshaping the role of the LLM. In HAL, the core responsibility of the central coordinating LLM is planning rather than directly answering. It is responsible for generating deterministic instructions (such as SQL queries or API calls) to obtain facts, rather than generating the facts themselves. The final answer is completely constructed based on the raw data retrieved from real databases and document libraries, which mechanistically ensures a high degree of factuality of the answer.

The experimental results provide strong evidence: Table 2 shows that HAL has achieved extremely high AAS on all tasks, which is directly attributed to the fact that its answers are constructed on real data. At the same time, the ablation experiments in Table 3 show that removing any one of the data tools leads to a sharp decline in performance, which conversely proves that obtaining facts through precise tool execution is the key to ensuring the quality of the final answer.

Of course, this mechanism transfers the risk from uncontrollable "content hallucination" to more controllable "planning errors". Although the LLM may still make mistakes when understanding user intentions, its complete execution logs make such errors diagnosable and traceable. This is a significant improvement in ensuring system reliability compared to the unpredictable "black box" errors of a single LLM.

5.2. Enhancing System Transparency: From Traceability to Explainability

Different from the "black box" reasoning process of a single LLM, the HAL framework has made significant improvements in system transparency. We distinguish between two concepts: traceability and explainability.

The core advantage of HAL lies in its provision of complete traceability. The "Think-Act-Observe" log chain in the figure is a direct manifestation of this traceability. Any output result can be traced back to its source: which agent was called, which SQL statement was executed, and which document fragments were retrieved. For high-reliability scenarios such as power transmission and distribution production command, this deterministic and auditable execution record is a crucial technical foundation.

However, technical traceability is not equivalent to interpretability at the user cognitive level. Although users can see what the system decides to do, the decision logic behind it (Why) remains obscure to non-technical users. Acknowledging this limitation points the way forward for future research: developing an "explanation module" that automatically translates technical execution logs into a user-friendly natural language "decision summary", thus truly achieving the leap from "traceable" to "interpretable".

5.3. Enhancing System Scalability: Modularity and Zero-Shot Learning

The modular design of HAL endows it with excellent scalability and maintainability. When faced with new business requirements (e.g., introducing new data sources or analysis tools), developers do not need to conduct costly retraining of the large central LLM. They only need to follow the API specifications of the framework, develop a new specialized agent or tool, and register its function description in the system.

The central coordinating LLM can automatically incorporate new tools into future task planning in a zero-shot scenario by learning their function descriptions. This "plug-and-play" ability greatly reduces the long-term maintenance cost and the threshold of technology iteration of the system. The ablation experiments in Table 3 also confirm this from the side: Each tool is like an independent module, and its presence or absence has a clear and quantifiable contribution to the system performance, proving the effective modularity of the architecture.

5.4. Limitations and Future Work

Although the HAL framework performs excellently, we recognize that there are still some directions for optimization:

Planning robustness: For extremely ambiguous or queries with extremely long logical chains, the current planning mechanism still has the risk of failure. Introducing more advanced planning algorithms such as the Tree of Thoughts is a future research direction.

Collaboration efficiency: The current collaboration mode dominated by the central LLM is essentially serial in nature. Studying more complex collaboration strategies such as parallel execution and task graph scheduling is expected to further improve the response speed of complex queries.

Cost and latency: Multiple calls to the LLM incur computational costs and time delays. Exploring optimization strategies such as dynamic model selection (calling small models for simple tasks) and result caching is crucial for deployment in real-time emergency scenarios.

6. Conclusions

This study introduces address the problem of natural language query analysis in the power transmission and distribution production command system, which is caused by heterogeneous data sources and complex information associations. An innovative intelligent agent collaborative large language model framework-HAL is proposed and implemented. This framework deeply understands user queries and plans tasks through a central coordinating LLM, and intelligently schedules a group of specialized intelligent agents (such as SQL query, document retrieval, time series

analysis agents) to collaboratively execute data operations, and finally integrates multi-source information to generate accurate, comprehensive and traceable natural language answers.

We conducted comprehensive experimental verification on a highly simulated power transmission and distribution hybrid dataset. The results show that the query success rate of the HAL framework reaches 96.0%, and the answer accuracy score is as high as 1.85 (on a 2-point scale). Its performance is significantly better than that of a single LLM, general intelligent agents and traditional methods in all dimensions. Especially when dealing with complex queries that require multi-step reasoning across data sources, the architecture advantages of HAL are particularly prominent.

The core contribution of the present study is to confirm that the intelligent agent collaborative architecture of "central planning-expert execution" is an effective way to solve the problem of query analysis of mixed data in professional fields. It not only provides a powerful, robust and scalable technical solution for building the next generation of power transmission and distribution production command intelligent assistant systems, but also is expected to liberate commanders from cumbersome and inefficient data query work, enabling them to focus more on high-value analysis, judgment and decision-making, thus comprehensively improving the intelligent level and safe operation efficiency of power grid operation.

References

[1] Al Mtawa Y., Haque A., and Halabi T., "A Review and Taxonomy on Fault Analysis in Transmission Power Systems," *Computation*, vol. 10, no. 9, pp. 1-19, 2022. DOI: 10.3390/computation10090144

[2] Avcı E., "Hybrid Artificial Intelligence Techniques for Enhanced Electricity Outage Prediction and Management in Distribution Networks," *Turk J Electr Power Energy Syst*, vol. 4, no. 2, pp. 63-73, 2024. DOI:10.5152/tepes.2024.24008

[3] Cen J., Liu J., Li Z., and Wang J., "SQLFixAgent: Towards Semantic-Accurate Text-to-SQL Parsing via Consistency-Enhanced Multi-Agent Collaboration," *in Proceedings of the AAAI Conference on Artificial Intelligence*, Philadelphia, pp. 49-57, 2025. DOI: 10.1609/aaai.v39i1.31979

[4] Cheng Y., Zhao H., Zhou X., Zhao J., and et al., "A Large Language Model for Advanced Power Dispatch," *Scientific Reports*, vol. 15, no. 1, Article 8925, 2025. DOI: 10.1038/s41598-025-91940-x

[5] Garg M., Namboodiri N., Joshi K., Krishna K., and Vora D., "Optimizing Multimodal RAG Systems for Multilingual Product Support," *The International Arab Journal of Information Technology*, vol. 23, no. 3, pp. 505-520, 2026. DOI: 10.34028/iajit/23/3/9

[6] Gholamzadeh Khoee A., Wang S., Yu Y., Feldt R., and Parthasarathy D., "GateLens: A Reasoning-Enhanced LLM Agent for Automotive Software Release Analytics," *arXiv Preprint*, arXiv:2503.21735v4, 2025. DOI: 10.48550/arXiv.2503.21735

[7] Ji Z., Lee N., Frieske R., Yu T., and et al., "Survey of Hallucination in Natural Language Generation," *ACM Computing Surveys*, vol. 55, no. 12, pp. 1-38, 2023. DOI: 10.1145/3571730

[8] Ko K., Nyein T., Oo K., Oo T., and Zin T., "Retrieval Augmented Generation for Document Query Automation Using Open Source LLMs," *in Proceedings of the 5th International Conference on Advanced Information Technologies*, Yangon, pp. 1-6, 2024. DOI: 10.1109/ICAIT65209.2024.10754919

[9] Lee U., Kim Y., Lee S., Park J., and et al., "Can We Use GPT-4 as a Mathematics Evaluator in Education?: Exploring the Efficacy and Limitation of LLM-Based Automatic Assessment System for Open-Ended Mathematics Question," *International Journal of Artificial Intelligence in Education*, vol. 35, no. 3, pp. 1-37, 2024. DOI: 10.1007/s40593-024-00448-4

[10] Li Y., Wang C., Li J., Liu J., and Chen Y., "Big Data-Driven Smart Energy Management: From Applications to Challenges," *Journal of Cleaner Production*, vol. 317, no. 128369, pp. 215-225, 2021. https://doi.org/10.1016/j.rser.2015.11.050

[11] Mahmud S., Nakamura M., and Zilberstein S., "Maple: A Framework for Active Preference Learning Guided by Large Language Models," *in Proceedings of the AAAI Conference on Artificial Intelligence*, Philadelphia, pp. 27518-27528, 2025. https://doi.org/10.1609/aaai.v39i26.34964

[12] Piccialli F., Chiaro D., Sarwar S., Cerciello D., and et al., "AgentAI: A Comprehensive Survey on Autonomous Agents in Distributed AI for Industry 4.0," *Expert Systems with Applications*, vol. 291, no. C, pp. 1-18, 2025. DOI: 10.1016/j.eswa.2025.128404

[13] Qiao J., Zhou A., Peng L., Zhu L., and et al., "Research and Implementation of Hybrid Storage Method for Multi-Source Heterogeneous Data of Electric Distribution Network," *in Proceedings of the 3D Imaging-Multidimensional Signal Processing and Deep Learning: Images, Augmented Reality and Information Technologies*, Springer, pp. 31-40, 2023. DOI: 10.1007/978-981-99-1230-8_3

[14] Sequeda J., Allemang D., and Jacob B., "A Benchmark to Understand the Role of Knowledge Graphs on Large Language Model's

Accuracy for Question Answering on Enterprise SQL Databases," *in Proceedings of the 7th Joint Workshop on Graph Data Management Experiences and Systems and Network Data Analytics*, New York, pp. 1-12, 2024. https://doi.org/10.1145/3661304.3661901

[15] Singh A., Shetty A., Ehtesham A., Kumar S., and Khoei T., "A Survey of Large Language Model-Based Generative AI for Text-to-SQL: Benchmarks, Applications, Use Cases, and Challenges," *in Proceedings of the IEEE 15th Annual Computing and Communication Workshop and Conference*, Las Vegas, pp. 15-21, 2025. DOI:10.1109/CCWC62904.2025.10903689

[16] Suri S., Das S., Singi K., Dey K., and et al., "Software Engineering Using Autonomous Agents: Are We There Yet?," *in Proceedings of the 38th IEEE/ACM International Conference on Automated Software Engineering*, Echternach, pp. 1855-1857, 2023. DOI: 10.1109/ASE56229.2023.00174

[17] Wan Y., Chen Z., Liu Y., Chen C., and Packianather M., "Empowering LLMs by Hybrid Retrieval-Augmented Generation for Domain-Centric Q and A in Smart Manufacturing," *Advanced Engineering Informatics*, vol. 65, no. PB, pp. 1-16, 2025. https://doi.org/10.1016/j.aei.2025.103212

[18] Wang J. and Zhao C., "Robust Control Performance Monitoring for Varying-Dimensional Time-Series Data Based on SCADA Systems," *IEEE Transactions on Instrumentation and Measurement*, vol. 71, pp. 1-14, 2022. DOI: 10.1109/TIM.2022.3177217

[19] Wang L., Ma C., Feng X., Zhang Z., and et al., "A Survey on Large Language Model Based Autonomous Agents," *arXiv Preprint*, arXiv:2308.11432v7, pp. 1-42, 2023. https://doi.org/10.1007/s11704-024-40231-1

[20] Wu S., Wei W., Zhang M., Chen Z., and et al., "Generative Retrieval as Multi-Vector Dense Retrieval," *in Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, New York, pp. 1828-1838, 2024. DOI: 10.1145/3626772.3657697

[21] Zhang B. and Yang Y., "MediaHG: Rethinking Eye-Catchy Features in Social Media Headline Generation," *in Proceedings of the Conference on Empirical Methods in Natural Language Processing*, Singapore, pp. 5766-5777, 2023. DOI: 10.18653/v1/2023.emnlp-main.352

Hua Li is a Senior Manager at China Southern Power Grid Company Limited. He holds a Master of Engineering degree and received a Bachelor's degree from Xi'an Jiaotong University in 2016. Research interests focus on Digital Grid Planning and Construction, Intelligent Power-Grid Information Systems, and Data-Driven Decision Support.

Yuan La is a Senior Manager at China Southern Power Grid Company Limited. He received a Bachelor's degree from Xi'an Jiaotong University in 2017. Research interests focus on Digital Grid Planning and Construction, Intelligent Power-Grid Systems, and Operational Data Analytics.

Wanci Zhang is a Senior Manager at China Southern Power Grid Company Limited. She received a Master's degree from North China Electric Power University in 2021. Research interests focus on power-grid transmission, transmission engineering, and Digital Technologies for Power-System Operations.

Zhengrong Wu is a Senior Manager at China Southern Power Grid Company Limited. He received a Master's degree from North China Electric Power University in 2022. Research interests focus on power-grid construction, Digital Grid Engineering, and Intelligent Infrastructure Management.

Song Wang is a Senior Manager at China Southern Power Grid Company Limited. He received a Bachelor's degree from Guangxi University in 2018. Research interests include Mechanical Automation, Intelligent Equipment, and Digital Technologies for Power-Grid Operation and Maintenance.