

Article | Received 3 June, 2026; Accepted 12 September, 2026; Published 10 October, 2026
<https://doi.org/10.55092/pe20260012>



RAG-ReAct empowered large language models for localized intelligent bridge operation and maintenance: a knowledge-driven agentic approach

Yawu Su, Kechen Dong*, Yu Liu, Mosha Chen and Ruoyu Zhang

China Construction Digital Technology Co., Ltd., Beijing, China

* Correspondence author; E-mail: dongkechen@cohl.com.

Abstract: Bridges serve as critical transportation infrastructure hubs. The recent breakthroughs in large language models (LLMs) offer transformative potential for intelligent bridge operation and maintenance (O&M), but their direct application is hindered by domain-specific hallucination, black-box reasoning opacity, and passive response limitations. This manuscript proposes a knowledge-driven, edge-side bridge O&M agent based on LLMs, addressing those critical limitations of general-purpose LLMs in specialized structural health monitoring (SHM) domains. The methodology centers on three core technical components: (1) an automated knowledge base construction pipeline; (2) a standardized action space generated via LLMs, wherein each O&M tool is encapsulated with natural language descriptions; and (3) an integrated retrieval-augmented generation (RAG)–reasoning acting (ReAct) agentic workflow that executes closed-loop O&M tasks through “knowledge-driven thought–action execution–observation feedback” cycles. Systematic experiments on the Qwen3 series (0.6B–14B parameters) with a 60,000-character highway bridge O&M knowledge base demonstrate that domain knowledge base integration significantly enhances professional accuracy, and the proposed agent achieves effective closed-loop bridge O&M workflow execution on a single edge-side device, offering a lightweight, specialized, and proactive technical solution for intelligent bridge O&M.

Keywords: large language models; knowledge-driven agent; retrieval-augmented generation; reasoning acting; bridge operation and maintenance; structural health monitoring

1. Introduction

Bridges, as the core hub of transportation infrastructure, have their lifecycle safety and maintenance directly tied to public safety and the sustainable development of the social economy. However, as the scale of infrastructure expands and service conditions become increasingly complex, the conventional operation and maintenance (O&M) model, which relies heavily on manual inspection and empirical judgment, is facing severe challenges. According to reports from the American Society of Civil Engineers (ASCE) and infrastructure assessments from multiple countries, a large number of existing bridges have entered a “high-incidence period of deterioration” [1]. Manual inspection, meanwhile, suffers from low efficiency, high costs, and is highly susceptible to subjective influences. Although intelligent structural health monitoring (SHM) technologies have been widely adopted, the existing studies



Copyright©2026 by the authors. Published by ELSP. This work is licensed under Creative Commons Attribution 4.0 International License, which permits unrestricted use, distribution, and reproduction in any medium provided the original work is properly cited.

still struggle to transform massive volumes of heterogeneous monitoring data into actionable maintenance decisions, and they lack capabilities for linguistic interaction, reasoning, and adaptability [2–3]. Consequently, a more comprehensive, general-purpose, user-friendly, and intelligent paradigm and framework for bridge O&M has emerged as a major research focus.

In recent years, the groundbreaking advances in large language models (LLMs) [4] have brought about a paradigm shift in the intelligentization of civil engineering. Unlike traditional deep learning models, which focus narrowly on predictive accuracy for specific tasks, LLMs leverage their formidable capabilities in semantic comprehension, logical reasoning, and generalization to enable machines to interpret user instructions as well as complex textual data such as engineering standards and inspection reports [5–6]. This technological transformation has propelled bridge O&M research from mere “state recognition” toward “cognitive decision”, making it feasible to construct intelligent systems that emulate expert human reasoning and offering novel pathways for addressing unstructured data processing and complex decision logic [7]. Despite the immense potential demonstrated by LLMs, their direct application to civil engineering, particularly bridge O&M remains hindered by a pronounced domain chasm. First, because pre-training corpora typically lack domain-specific content, and the loss functions for LLMs training penalize incorrect and “I don't know” responses equally, generalist LLMs are highly liable to generate fabricated content—commonly referred to as model hallucination—when responding to specialized queries in bridge O&M. Second, as end-to-end semantic processing models, LLMs operate within a “black box”, making it difficult to identify the root causes of errors during long-chain reasoning and to implement external interventions. Third, LLMs are inherently passive, capable only of answering user queries without the ability to autonomously detect tasks or proactively interact with the external environment, thereby severely constraining their utility in bridge O&M. To overcome these limitations of general-purpose models in specialized domains, LLMs-augmentation techniques such as knowledge-driven methods, agents and agentic workflows have gained considerable momentum and are emerging as robust pathways for enhancing the engineering applicability of these LLMs.

Knowledge-driven techniques focus on constructing domain-specialized reasoning databases or knowledge for data-driven models, endowing them with precise professional knowledge retrieval capabilities and logical constraints [8]. Existing research on knowledge-driven methodologies within civil and infrastructure engineering can be primarily categorized as follows: In terms of reviews and frameworks, Wan et al.[9] reviewed and highlighted the strong interpretability of knowledge-driven approaches; through the analysis and synthesis of thousands of papers, Jiang et al.[10] pointed out that bridge O&M requires a shift from data-driven to knowledge-driven thinking, and proposed a knowledge-oriented multi-source data framework to support intelligent holistic maintenance decision-making; Yang et al.[11] proposed a multi-layered intelligent bridge management framework based on the big-data knowledge engineering paradigm to enhance the sharing and service capabilities of domain knowledge; Jiang et al.[12] proposed a macro-level machine learning-enabled knowledge-driven framework that enhances holistic structural maintenance decision-making by dynamically linking real-time machine learning information with static knowledge bases. In terms of ontology construction and semantic reasoning, Jiang et al.[13] proposed a decision-making framework integrating static knowledge reasoning with dynamic finite element analysis, constructed a comprehensive bridge ontology encompassing maintenance standards, and achieved O&M decision support under complex rules; Jiang et al.[14] developed a knowledge base combining machine learning predictions with ontological semantic reasoning for assessing bridge corrosion

conditions; He et al.[15] addressed the "tacit rules" in bridge maintenance by proposing a framework integrating domain ontologies with smart contracts, improving the deployment efficiency and collaborative transparency of tacit decision rules; Banujan et al.[16] utilized Protégé to develop a standardized bridge maintenance ontology architecture, aiming to support explicit and tacit knowledge sharing among construction site stakeholders and researchers through formal taxonomies and ontological models. In terms of multimodal and data-fusion augmentation, Chen et al. [17] proposed an LLMs-based multimodal analysis framework that extracts entities from textual reports through fine-tuned LLMs to construct a bridge knowledge graph. By integrating Retrieval-Augmented Generation (RAG) with knowledge-enhanced prompting, this framework achieves cross-modal data association between text and imagery and context-aware maintenance decision support; Chai et al. [18] developed a domain-knowledge-driven image captioning generation framework that effectively reduces "hallucinations" in visual inspection and improves the accuracy of structural apparent damage descriptions. Nevertheless, despite the substantial progress achieved by existing works in knowledge-driven question-answering and reasoning for infrastructure domains, these studies remain confined to a passive information-processing paradigm, lacking proactive utilization and extraction of domain-specific knowledge. In particular, how to achieve knowledge-based deep semantic understanding, dynamic planning, and closed-loop execution remains a critical bottleneck in current research.

Agent or agentic workflow technologies endow LLMs with capabilities for autonomous planning, tool invocation, and multi-step reasoning, transforming them from passive respondents into proactive task executors [19]. Existing research on agents within civil and infrastructure engineering primarily encompasses the following areas: Intelligent modeling, Deng et al. [20] proposed BIMgent framework, which leverages multimodal LLM-driven agents to achieve autonomous building modeling through GUI operations; experimental results demonstrate its ability to automatically execute complex software workflows while adhering to design intent. Zhou et al. [21] investigated an architecture combining LLMs with agents, enabling the system to recognize two-dimensional drawings and invoke BIM software for automated three-dimensional substation modeling. Du et al. [22] proposed the Text2BIM multi-agent framework, which translates natural language instructions into executable code to generate editable BIM models, and introduced a rule-based model checker that iteratively optimizes model quality using predefined knowledge. Hellin et al. [23] proposed an LLM-based agentic workflow capable of retrieving BIM model information through natural language queries without requiring a specific ontology. Materials and energy consumption design, Chen et al. [24] constructed a code-compliant automated reinforced concrete design framework using LLMs, which coordinates multiple specialized agents to decompose tasks and cross-validate results. Zhang et al. [25] developed an LLM workflow comprising four core agents to automate the generation and debugging of error-free EnergyPlus building energy consumption models. Management and decision support, Yang et al. [26] proposed the IDM-GPT multi-agent framework, which connects users, traffic databases, and machine learning models to provide customized traffic analysis recommendations. Yan et al. [27] proposed a general-purpose AI agent framework based on Reasoning Acting (ReAct) strategy, utilizing the LLM as a "brain" for reasoning and invoking Internet of Things (IoT) tools to control smart building equipment. Kalyuzhnaya et al. [28] explored the application of LLM-based multi-agent systems in smart city management, combining RAG technology to enhance the accuracy of processing local regulations and knowledge. Nevertheless, these preview studies still exhibit significant

limitations. First, most methodologies demonstrate strong domain or case dependency, lacking generalizable adaptation to bridge O&M scenarios. Second, the knowledge-driven mechanism remains underdeveloped, with prevalent issues of insufficient knowledge empowerment or fractured “knowledge-tool” collaboration chains. Finally, some systems remain confined to the passive question-answering paradigm, failing to support proactive complex task planning and execution.

To address the aforementioned limitations, this study proposes a knowledge-driven LLMs-based localized highway bridge O&M agent. The proposed method first automates the construction of a local knowledge base from highway bridge O&M standard texts, achieving domain-specialized model question-answering and reasoning through RAG. Subsequently, O&M tools are encapsulated as registered functions or application programming interfaces (APIs), from which the LLMs reads the source code and automatically generates natural language descriptions encompassing names, functionalities, parameters, default values, and other attributes, thereby forming a standardized action space. An agent task execution strategy based on a “thinking – acting – observing” ReAct paradigm is then constructed. Upon receiving a user instruction in natural language, the agent generates O&M tasks through an instruction-knowledge dual-driving, dispatches high-relevance tools from the action space to complete O&M operations and aggregates execution results. Finally, the agent invokes LLMs to produce a task summary grounded in O&M knowledge, thereby realizing a knowledge-driven closed loop of “instruction–knowledge–action–summary”. The agent constructed through this method is capable of localized edge-side deployment, providing intelligent, stable, and secure technical support for bridge (representing for infrastructure engineering) O&M.

This manuscript is organized as follows: Section 1 presents the introduction, systematically elaborating on LLMs, agents, and their existing research and bottlenecks in the domain of bridge O&M; Section 2 details the methodology, comprehensively describing the construction and workflow of the proposed bridge O&M agent; Section 3 covers the experimental evaluation, in which the feasibility of the proposed method is systematically analyzed based on actual O&M knowledge bases and action space; Section 4 provides conclusions and future perspectives, with in-depth discussion and exploration of the capability boundaries and future outlook of the agent proposed in this study.

2. Methods

The workflow of the proposed bridge O&M agent, illustrated in Figure 1, is accomplished through a two-stage RAG [29]: firstly, the agent receives user’s O&M instruction and extracts knowledge pertinent to the instruction through RAG; secondly, based on the retrieved knowledge and the instruction, the agent dispatches relevant tools from the action space through RAG to execute the user’s instruction. The construction and workflow of this agent are elaborated in detail below.

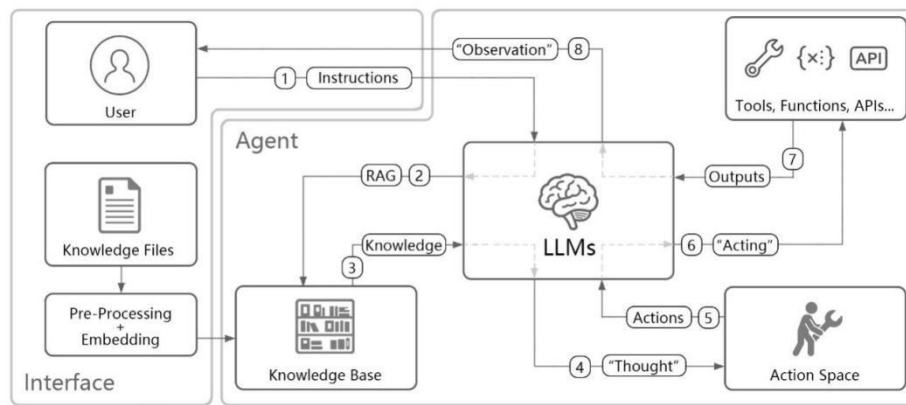


Figure 1. The workflow of bridge O&M agent based on LLMs.

2.1. Knowledge base and action space

The bridge O&M knowledge base and action space collectively delimit the operational scope of the LLMs agent. Within this boundary, the knowledge base furnishes the agent with a stable, externalized source of knowledge-driven intelligence, ensuring that the LLMs' generated content remains traceable and auditable, thereby mitigating the production of seemingly plausible yet factually erroneous or inaccurate "hallucinations". Concurrently, the action space serves as the execution capability carrier for the agent, enabling it to carry out relevant O&M directives rather than being confined to instructive natural language question-answering. This subsection discusses the construction methodologies for the LLMs-based knowledge base and action space proposed in this study.

Bridge O&M Knowledge Base. The knowledge base construction workflow primarily encompasses data acquisition and preprocessing, followed by text chunking. Specifically, knowledge documents such as O&M standards that support the agent are collected. Tables contained therein are converted into Markdown format using LLMs, while scanned documents and handwritten files are transformed into editable and archivable text through character recognition technologies such as optical character recognition (OCR) [30]. These are consolidated into unified knowledge text documents. The LLMs are then leveraged to perform data cleansing on this textual corpus, removing erroneous, garbled, duplicate characters, redundant punctuation, and formatting symbols (e.g., spaces, line breaks, page breaks), thereby furnishing a high-quality knowledge text repository for subsequent knowledge base construction.

Subsequently, the LLMs read the knowledge text documents and performs semantic chunking, the purpose of which is to ensure that the length of each text segment does not exceed the context window limits of both the semantic embedding model and the agent's LLMs, thereby preventing omissions in knowledge retrieval. The text chunking procedure comprises a three-stage process: first, preliminary segmentation is conducted according to natural semantic boundaries, using natural language structures such as chapters and paragraphs as delimiters to divide long documents into semantic chunks, thereby maximizing thematic coherence; if any semantic chunk exceeds the pre-set maximum length, a secondary segmentation is performed based on this pre-set maximum length, using structural separators and punctuation marks as boundaries (with priority given to natural paragraph delimiters, followed by sentence-boundary punctuation, and then intra-sentence pausing punctuation), rather than forcibly truncating mid-sentence; finally, an overlap zone is established—to prevent critical information loss at

chunk separation, adjacent chunks are configured such that the trailing characters of the preceding chunk coincide with the leading characters of the subsequent chunk (for instance, for a block of character length n , its terminal m characters will be identical to the initial m characters of the next block, where $m \ll n$), the purpose of which is to prevent critical information from being lost at chunk boundaries.

At this juncture, the construction of a knowledge base—from highly unstructured knowledge documents to an efficiently invocable repository—is complete, laying a robust foundation of externalized data for the downstream application to achieve precise semantic retrieval and high-quality question-answering generation.

Bridge O&M Action Space. Action space is a standardized interface that specifies the agent's service realm. The action space in this study is constructed through LLM-based reading of source code or APIs. Specifically, source code or interfaces are collected for all downstream tools or modules that support the agent in providing O&M execution services; each tool, callable function, or interface is designated as an action. The LLMs read the source code or interface of each action and generates corresponding natural language descriptions based on prompt templates, the details of which are elaborated below:

$$\begin{cases} A = \{a_1, a_2, \dots, a_m\} \\ a_i = \{name_i, desc_i, P_i, eg_i\} = \text{LLM}(code_i | prompt) \\ P_i = \{p_1, p_2, \dots, p_n\} \\ p_j = \{pname_j, pdesc_j, \tau_j, \delta_j\} \end{cases} \quad (1)$$

The above Equation (1) delineates the composition of the action space A , wherein the description of each action a_i is generated by the LLMs under prompt-augmented conditions after reading the action source code $code_i$, comprising the action name $name_i$, action description $desc_i$, variable detail P_i , and invocation example eg_i . The length of the variable detail corresponds to the number of hyperparameters involved in the action; the description of each variable p_j includes its variable name $pname_j$, variable description $pdesc_j$, variable type τ_j , and default value δ_j when not explicitly specified in the instruction, where $\tau \in \{\text{str, int, float, bool, json}\}$, indicating that each variable belongs to one of the following categories: string, integer, floating-point number, Boolean value, or JSON format (dictionary or tensor). Subsequently, all O&M actions are transcribed into a specific standardized three-level “key-value” representation to complete the action space construction. The specific standardized structure of the action space is presented in Pseudocode 1.

Pseudocode 1. Standardized action space.

```

ActionSpace={‘function_0’: function_0, ‘function_1’: function_1, ..., ‘function_m’: function_m}
For i in {0, 1, 2, ..., m} do:
    ActionSpace[‘function_i’] = {‘name’: Name of the action.
                                ‘description’: Description of the action, including functions and writer’s name.
                                ‘parameters’: [para_0, para_1, ..., para_n]}

    For j in {0, 1, 2, ..., n} do:
        para_j = {‘name’: Name of the variable.
                 ‘type’: Type of the variable.
                 ‘description’: Description of variable, including physical meaning, default measurement, input example for
                 in-context learning.
                 ‘default’: Default of the variable.}

    End for
End for

```

It is noteworthy that, with the passage of time and technological advancement, the knowledge base and tools required for bridge O&M are subject to frequent updates. Unlike approaches that establish domain-specific LLMs cognition through fine-tuning, when knowledge and action contents undergo changes, the knowledge base and action space can accomplish content iteration merely through the addition, deletion, or revision of text segments and action descriptions, without necessitating incremental training of the LLM. This “plug-and-play” operational paradigm significantly enhances the capability and efficiency of practical applications.

2.2. RAG-ReAct agentic workflow

The proposed bridge O&M agent accomplishes multi-turn “thought”, “acting”, “observation”, and responding (i.e., ReAct) based on the RAG technical framework; the specific workflow is elaborated in detail in this subsection.

RAG-Based Knowledge Dispatch. Upon receiving user’s instruction, the agent is required to comprehend the connotation and execution methodology of the instruction based on the knowledge base—that is, to retrieve and organize the relevant O&M knowledge to generate a model “thought”, thereby providing guidance for subsequent task execution.

First, all text chunks in the knowledge base are subjected to vectorized embedding. Utilizing a pre-trained deep semantic embedding model, the cleansed and chunked text is transformed into dense vectors in a high-dimensional space, as formulated in Equation (2). These vectors serve not merely as mathematical representations of text, but more precisely capture their underlying semantic features, enabling indexing based on vector proximity.

$$p(\mathbf{Z}_{top-K} | x) \propto \exp(\text{Em}^T(\mathbf{Z})\text{Em}(x)) \quad (2)$$

$$p_{\text{RAG-Token}}(\mathbf{Y} | x) \approx \prod_i \sum_{z \in \mathbf{Z}_{top-K}} p(z | x) p(y_i | x, z, y_{1:i-1}) \quad (3)$$

Subsequently, the agent encodes the user instruction into a high-dimensional vector through the semantic embedding, after which distance metrics are computed between this vector and the segment vectors in the knowledge base to extract the top-K chunks with the highest relevance, as formulated in Equation (2), in which $\text{Em}(\bullet)$, $\mathbf{Z}$, $\mathbf{Z}_{top-K}$, x indicate the semantic embedding model, knowledge base, the top-K most relevant chunks, instruction, respectively.

Finally, due to the fact that multiple chunks is relevant to the instruction, the agent is urged to organize “thought” through RAG-Token [31] paradigm. Specifically, based on the retrieved top-K chunks, the LLMs generate a probability distribution for the next output token conditioned on each chunk; this procedure is then repeated for the subsequent tokens, as formulated in Equation (3), in which $\mathbf{Y}$, y_i respectively indicate “thought” and tokens of “thought”. This mechanism enables the agent to synthesize content from multiple knowledge segments during “thought” generation, thereby furnishing more comprehensive and reliable support.

RAG-Based Agent Acting. Upon completing “thought” regarding knowledge pertinent to the user instruction, the agent is required to retrieve and identify variables for the relevant O&M actions, thereby organizing the agent’s “acting” and aggregating execution results into the agent’s “observation” to fulfill the O&M instruction.

First, the action space is subjected to vectorized embedding. Utilizing a pre-trained deep semantic embedding model, the structurally standardized action space text is transformed into dense vectors in a high-dimensional space, as formulated in Equation (4). These vectors precisely capture the underlying semantic features of each action's name and functional description, hyperparameter definitions and exemplar descriptions and other attributes, enabling indexing based on vector similarity. Concurrently, the in-context learning capability of the LLMs is leveraged to identify the corresponding action variable values.

$$p(A_{a \geq Q} | Y) \propto \exp(\text{Em}^T(A_{a \geq Q}) \text{Em}(Y)) \quad (4)$$

$$p_{\text{RAG-Sequence}}(D | Y) \approx \prod_{a \in A_{a \geq Q}} p(a | Y) \prod_i p(d_i | x, a, d_{1:i-1}) \quad (5)$$

Second, the agent computes the distance metrics between the embedding of retrieved knowledge and action space, extracting the actions exceeding the relevance threshold, as formulated in Equation (4), in which Q indicates the threshold while $A_{a \geq Q}$ represents those related actions.

Third, the agent leverages the LLMs' in-context learning capability to generate structured action dispatch text based on the RAG-Sequence [31] paradigm. Specifically, conditioned on a single retrieved action as the sole text chunk, the LLMs generate execution hyperparameters for each action variable based on the detailed action description therein and the user instruction. This procedure is then repeated for each retrieved action, and the outputs are composed into the execution dispatch text (JSON) for the current turn, thereby enabling the agent to initiate the “acting”—that is, to deploy locally registered functions, modules, and APIs for task execution, as formulated in Equation (5), in which D indicates the execution dispatch text and d_i indicates the dispatch tokens of action a .

At the conclusion of each interaction turn, the agent collects all downstream execution results and organizes them into a natural language report—the “observation”—which serves as the feedback, thereby completing one full ReAct cycle of “user instruction–knowledge dispatch–action execution–agent feedback”.

$$\begin{cases} Y_r = \text{LLM}(x_r | C, \text{prompt}) \\ C = \{c_1, c_2, \dots, c_{r-1}\} \\ c_i = \{x_i, Y_i, \text{obs}_i\} \end{cases} \quad (2)$$

Additionally, if the user appends an O&M instruction after the agent completes one ReAct cycle, the agent initiates multi-turn dialogue, incorporating the user instructions, model “thoughts”, and “observations” from all preceding turns into the generation conditions for generating the “thought” for a new ReAct cycle, and repeating the aforementioned ReAct steps until dialogue termination, as formulated in Equation (6), in which r denotes the current turn, C represents the dialogue context, and c_i ($i \in \{1, 2, \dots, r-1\}$) signifies the dialogue content of each historical cycle.

3. Experiments

This section evaluates the performance of the proposed localized O&M agent under real bridge O&M standard documents, encompassing the accuracy of knowledge dispatch and tool execution under the experimental hardware conditions of this manuscript, with pertinent discussion of observed phenomena.

3.1. Experimental data and condition

The experimental knowledge base in this study is constructed based on highway bridge O&M standard documents, encompassing multi-modal knowledge content including editable text, image-based text, and tables. Following data cleansing, format standardization, and text extraction, a plain-text knowledge base of approximately 60,000 characters is ultimately obtained. The agent's action space is defined and configured in accordance with bridge O&M standards.

The agent constructed in this study is deployed and executed in a local PyTorch + Ollama environment, with the hardware configuration specifically employing a single Inter i9 10900F CPU with 32GB RAM, and a single discrete GPU of NVIDIA GeForce RTX 3080 Ti (12 GB). Based on the aforementioned localized experimental environment, the LLMs adopted in the agent are the Qwen3 series [32], with parameter scales ranging from 0.6 billion to 14 billion. The pre-trained semantic embedding module employs all-MiniLM-L6-v2 model, which achieves efficient text representation and semantic matching based on the Sentence-BERT architecture [33].

Additionally, other key experimental hyperparameters are configured as follows: the pre-set chunk length is 300 characters, the top-K retrieval parameter for RAG is set to $K=10$, and the action screening confidence threshold is $\mathcal{Q}=0.7$.

3.2. Evaluation metrics

The experiments quantitatively evaluate both the “thought” and “acting” of the agent. Prior to introducing the evaluation metrics.

$$\text{Recall} = \frac{TP}{TP + FN} \quad (7)$$

$$\text{Faith} = \frac{TP}{TP + FP} \quad (8)$$

$$F1_{\text{BERT}} = 2 \times \frac{\text{Prec}_{\text{BERT}} \times \text{Recall}_{\text{BERT}}}{\text{Prec}_{\text{BERT}} + \text{Recall}_{\text{BERT}}} = \frac{2TP}{2TP + FP + FN} \quad (9)$$

$$F1_{\text{Act}} = 2 \times \frac{\text{Prec}_{\text{Act}} \times \text{Recall}_{\text{Act}}}{\text{Prec}_{\text{Act}} + \text{Recall}_{\text{Act}}} = \frac{2TP}{2TP + FP + FN} \quad (10)$$

$$\text{Acc}_{\text{Para}} = \frac{TP}{TP + FP} \quad (11)$$

Three metrics are employed to evaluate the “thought” of the agent: Recall, Faithfulness (Faith), and BERT-Score (F1BERT) [34]. The first examines the success rate of extracting the text chunks relevant to user instruction, as formulated in Equation (7). The second examines the proportion of segments in the generated “thought” that can be substantiated by knowledge (under the condition of successful knowledge) retrieval, as formulated in Equation (8). The third directly examines the similarity between the reference answer for the instruction and the agent's output “thought”, as formulated in Equation (9).

Two metrics are employed to evaluate the agent's “acting”: F1-score for action dispatch (F1Action) and action hyperparameter identification accuracy (AccPara). The former primarily examines whether the tools dispatched by the agent based on “thought” are sufficient and necessary, as formulated in Equation (10). The latter primarily examines the precision of action hyperparameters identified by the agent based on instruction, as formulated in Equation (11).

3.3. Experimental results

The specific performance of the bridge O&M LLMs-based agent with varying parameter scales is presented in Table 1; a response instance excerpt of the agent based on Qwen3-14B + KB to users' instruction. The systematic experiments in this study reveal the following phenomena:

Table 1. Quantitative evaluation result of bridge O&M agent performance constructed from various llms.

Agent's LLM	Recall (%)	Faith (%)	F1 _{BERT} (%)	F1 _{Action} (%)	Acc _{Para} (%)
Qwen3-0.6B	N/A	N/A	3.76	48.24	43.09
Qwen3-0.6B + KB	34.00	31.30	10.08	61.18	45.74
Qwen3-1.7B	N/A	N/A	4.90	42.35	59.04
Qwen3-1.7B + KB	33.00	67.11	22.20	71.76	56.91
Qwen3-4B	N/A	N/A	9.22	64.71	72.77
Qwen3-4B + KB	51.00	84.20	40.95	71.76	72.34
Qwen3-8B	N/A	N/A	6.87	67.06	89.36
Qwen3-8B + KB	53.00	89.16	46.25	74.12	89.36
Qwen3-14B	N/A	N/A	6.96	60.00	91.49
Qwen3-14B + KB	54.00	92.13	48.84	72.94	89.89

*"+ KB" indicates a LLM integrated domain knowledge base.
 *"N/A" indicates the metrics do not exist in this agent.

First, LLM agents without knowledge base integration produce outputs that severely deviate from domain standards. Native LLMs unintegrated with the bridge O&M knowledge base exhibit varying degrees of knowledge bias in their responses to bridge O&M-related instructions, manifesting as answers that are partially or entirely non-compliant with bridge O&M professions. This is likely attributable to the lightweight, locally deployable LLMs selected in this study, whose pre-training corpora consist primarily of general-domain text, lacking specialized incremental pre-training and instruction fine-tuning targeted at the domain of bridge O&M. Consequently, their innate capabilities in domain knowledge memorization and professional constraint comprehension are inherently insufficient, making it difficult to generate compliant and accurate O&M decision text without external knowledge augmentation. Therefore, implanting a vertically specialized knowledge base into the agent is critically necessary, and the experimental results demonstrate that such implantation significantly enhances the professional accuracy of agent-generated outputs.

Second, LLMs parameter scale exhibits a significant positive correlation with overall agent performance. Under identical experimental settings and knowledge base conditions, agents constructed from LLMs with larger parameter scales demonstrate superior performance across dimensions including knowledge comprehension, action decision, and answer normativity. This is likely attributable to the fact that larger parameter scales endow models with stronger semantic representation capabilities, logical reasoning capabilities, and instruction-following capabilities [35]; concurrently, the Qwen3 series employed in this experiment all adopt dense architectures (dense models) without employing mixture-of-experts (MoE) sparse activation strategies [32], thereby engaging full parameters during the inference phase. Consequently, model capabilities exhibit stable gains with increasing parameter scale; for instance, on the 14-billion-parameter model (14B), common issues prevalent in smaller-scale LLMs—such as ambiguous semantic comprehension and uncontrolled format output—occur with substantially lower probability.

Third, the recall rate of knowledge dispatch constrains the reliability of “thought”. Under conditions where the RAG module successfully retrieves relevant O&M knowledge chunks, most agents ultimately generate “thoughts” that are highly congruent; however, across the overall experiments, the completion probability of the agent’s knowledge chunk matching based on instructions remains at a relatively low level. The underlying causes of this phenomenon may include the following aspects: limitations of the retrieval algorithm, such that methods based on TF-IDF [36] or simple semantic similarity may fail to capture complex semantic relationships in long texts; and limitations of vector representation—as with all LLMs employed in this study, the semantic embedding model utilized herein has likewise undergone pre-training exclusively on general-domain textual data, lacking targeted adaptation to vertical domain professional terminology, modes of thinking, polysemy, long texts, and other content. Consequently, the resulting word or sentence embeddings may prove insufficient to satisfy the matching requirements with user instructions.

Fourth, the hyperparameter identification performance of action dispatch is independent of the knowledge base. Experimental results indicate that, under the premise of correct action dispatch by the agents, there is no significant difference in the accuracy of hyperparameter identification from the action space across models, regardless of whether the bridge O&M knowledge base is integrated, and this accuracy exhibits an approximate positive correlation with LLMs’ parameter scale. This is highly likely attributable to the fact that the action space already contains complete contextual information encompassing hyperparameter definitions, data formats, invocation examples, and other attributes. Therefore, the task of extracting hyperparameters from natural language instructions is essentially equivalent to a generic in-context learning and slot-filling task, a capability primarily determined by the LLMs’ intrinsic semantic comprehension level rather than by the domain knowledge base.

Finally, a certain agent exhibits a secondary knowledge dispatch reasoning behavior. The agent constructed from Qwen3-14B invokes the knowledge base again during the final generation of the “observation” phase for verification and augmentation, a behavior not pre-defined through chain-of-thought prompting. The probable cause lies in the fact that larger-scale parameter LLMs possess stronger in-context and format learning capabilities, enabling them to learn the paradigm of “invoking the knowledge base before generating responses” from historical interaction trajectories. That is to say, this phenomenon may continue to emerge in larger-scale models within the same series; should this hold true, it would further corroborate the strong correlation between in-context learning capability and LLM scale. Additionally, this non-predefined agent behavior carries guiding significance for more complex model reasoning and warrants further research and analysis.

4. Conclusion

This study constructs a knowledge-driven, localized bridge O&M agent based on LLMs. Through automated construction of O&M domain-specific knowledge bases, standardized action space, and adoption of an integrated RAG–ReAct workflow, the agent achieves closed-loop execution of bridge O&M tasks encompassing “knowledge-based reasoning–O&M execution–action observation”.

Through systematic experimentation, the core findings of this manuscript are as follows: native large language models, lacking domain knowledge constraints, are prone to content deviation and hallucination in bridge O&M tasks. Upon integration with the dedicated O&M knowledge base, the agent achieves significant improvements across key metrics including knowledge recall rate, content

faithfulness, and BERT-score, substantiating the necessity of domain knowledge augmentation. Within the model series adopted in this study (Qwen3), overall agent performance exhibits a positive correlation with LLMs' parameter scale, with larger-scale LLMs demonstrating stronger semantic comprehension and in-context learning capabilities. Under the standardized action space construction methodology and format employed in this manuscript, hyperparameter identification accuracy within the action space depends primarily on the model's intrinsic in-context learning capability and is minimally affected by the knowledge base. The dispatch precision of RAG-based knowledge retrieval in the O&M agent of this study remains suboptimal, constraining overall reasoning reliability. Large-scale parameter LLMs, in the absence of pre-set rules, can spontaneously generate secondary knowledge dispatch behavior, proactively invoking the knowledge base to accomplish result verification and content augmentation, thereby revealing the latent proactivity of LLM-based agents.

The O&M agent construction methodology proposed in this manuscript can effectively suppress domain hallucination, ameliorate the "black-box" reasoning problem, and support localized edge deployment on a single GPU with "plug-and-play" iterative updates. It possesses certain feasibility and engineering application value in practical bridge O&M intelligentization scenarios, and can provide lightweight, specialized, and proactive technical solutions for infrastructure safety O&M.

In response to the core findings of this study, future research will focus on the mechanism and performance of domain-specific knowledge dispatch, and conduct in-depth analysis of the emergence mechanism of non-predefined agent behavior, thereby guiding and harnessing such proactive behaviors to achieve further enhancement of decision reliability and complex task adaptability for bridge O&M agents.

Data availability statement

No supplementary or additional data were generated in this study.

Declaration of Generative AI and AI-assisted Technologies

During the preparation of this manuscript, the authors used generative AI tools only to improve language and readability. The authors take full responsibility for the content of the manuscript.

Acknowledgments

The work was supported by Technology R&D Project of China Construction Digital Technology Co., Ltd, Development Planning and Scenario Validation Project of Large Language Model for the Construction Industry.

Authors' contribution

Conceptualization, Yawu Su and Kechen Dong; methodology, Yawu Su, Kechen Dong and Yu Liu; software, Yu Liu and Mosha Chen; validation, Yawu Su, Mosha Chen and Ruoyu Zhang; formal analysis, Kechen Dong and Yu Liu; investigation, Yawu Su and Ruoyu Zhang; resources, Kechen Dong and Mosha Chen; data curation, Yu Liu and Mosha Chen; writing—original draft preparation, Yawu Su and Yu Liu; writing—review and editing, Kechen Dong, Mosha Chen and Ruoyu Zhang; visualization,

Yu Liu and Mosha Chen; supervision, Kechen Dong; project administration, Kechen Dong. All authors have read and agreed to the published version of the manuscript.

Conflicts of interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

- [1] Ogun E, Voern YA, Lee D. A real-time mobile robotic system for crack detection in construction using two-stage deep learning. *Sensors* 2026, 26(2):530.
- [2] Li X., Zhu Y, Yu W. Data-driven structural health monitoring through echo state network regression. *Information* 2025, 16(8), 678.
- [3] Entezami A, Sarmadi H, Behkamal B, Mariani S, *et al.* Health monitoring of large-scale civil structures: an approach based on data partitioning and classical multidimensional scaling. *Sensors* 2021, 21(5):1646.
- [4] Achiam J, Adler S, Agarwal S, Agarwal S, Ahmad L, *et al.* GPT-4 technical report. *arXiv* 2023, arXiv:2303.08774.
- [5] Höltingen L, Zentgraf S, Hagedorn P, König M. Utilizing large language models for semantic enrichment of infrastructure condition data: a comparative study of GPT and Llama models. *AI Civ. Eng.* 2025, 4(1):14.
- [6] Chen X, Lei Y, Li Y, Parkinson S, Li X, *et al.* Large models for machine monitoring and fault diagnostics: opportunities, challenges, and future direction. *J. Dyn. Monit. Diagn.* 2025, 4(2):76–90.
- [7] Kumar D, Agrawal A. Advancing bridge infrastructure management through artificial intelligence: a comprehensive review. *International Journal of Bridge Engineering. Waste Manage. Res.* 2025, 2(3):214250021–1:18.
- [8] Liu M, Zhang L, Chen J, Chen W, Yang Z, *et al.* Large language models for building energy applications: opportunities and challenges. *Build. Simul.* 2025, 18(2):225–234.
- [9] Wan S, Guan S, Tang Y. Advancing bridge structural health monitoring: insights into knowledge-driven and data-driven approaches. *J. data sci. intell. syst.* 2024, 2(3):129–140.
- [10] Jiang Y, Yang G, Li H, Zhang T. Knowledge driven approach for smart bridge maintenance using big data mining. *Autom. Constr.* 2023, 146:104673.
- [11] Yang J, Xiang F, Li R, Zhang L, Yang X, *et al.* Intelligent bridge management via big data knowledge engineering. *Autom. Constr.* 2022, 135:104118.
- [12] Jiang Y, Yang G, Li H, Zhao K, Zhang T. Machine learning informed knowledge driven framework supporting holistic bridge maintenance. In *Proceedings of the 30th EG-ICE: International Conference on Intelligent Computing in Engineering*, London, UK, July 4–7, 2023.
- [13] Jiang Y, Yang G, Li H, Zhang T, Khudhair A. Physics-informed knowledge-driven decision-making framework for holistic bridge maintenance. *J. Constr. Eng. Manage.* 2024, 150(9):04024105.
- [14] Jiang Y, Li H, Yang G, Zhang C, Zhao A. Machine learning-driven ontological knowledge base for bridge corrosion evaluation. *IEEE Access* 2023, 11:144735–144746.
- [15] He C, Liu M, Hsiang SM, Pierce N, Megahed S, *et al.* An ontological knowledge-driven smart

- contract framework for implicit bridge preservation decision making. *J. Constr. Eng. Manage.* 2025, 151(4):04025008.
- [16] Banujan K, Vasanthapriyan S. Bridge ontology architecture for knowledge management in bridge maintenance. In *Proceedings of the 2020 Moratuwa Engineering Research Conference (MERCon)*, Moratuwa, Sri Lanka, July 28–30, 2020, pp. 1–6.
- [17] Chen Q. Domain-specific adaptation of large language models and integration with knowledge graph analytics for enhanced bridge maintenance decision making. Ph.D. thesis, University of Illinois Urbana-Champaign, 2025.
- [18] Chai C, Gao Y, Xiong G, Liu G, Li H. Domain knowledge-driven image captioning for bridge damage description generation. *Autom. Constr.* 2025, 174:106116.
- [19] Durante Z, Huang Q, Wake N, Gong R, Park JS, *et al.* Agent AI: surveying the horizons of multimodal interaction. *arXiv* 2024, arXiv:2401.03568.
- [20] Deng Z, Du C, Nousias S, Borrmann A. BIMgent: towards autonomous building modeling via computer-use agents. *arXiv* 2025, arXiv:2506.07217.
- [21] Zhou L, Huang Y, Yan T, Li D, Cai H, *et al.* Research on the automatic BIM modeling method of substation based on LLM agent. In *Proceedings of the 6th International Conference on Artificial Intelligence and Advanced Manufacturing (AIAM 2024)*, Hybrid Conference, Frankfurt, Germany, October 20–21, 2024, pp. 686–692.
- [22] Du C, Esser S, Nousias S, Borrmann A. Text2BIM: generating building models using a large language model-based multiagent framework. *J. Comput. Civ. Eng.* 2026, 40(2):04025142.
- [23] Hellin S, Nousias S, Borrmann A. Natural language information retrieval from BIM models: an LLM-based agentic workflow approach. In *Proceedings of the 2025 European Conference on Computing in Construction*, Porto, Portugal, July 14–17, 2025.
- [24] Chen J, Bao Y. Multi-agent large language model framework for code-compliant automated design of reinforced concrete structures. *Autom. Constr.* 2025, 177:106331.
- [25] Zhang L, Ford V, Chen Z, Chen J. Automatic building energy model development and debugging using large language models agentic workflow. *Energy Build.* 2025, 327:115116.
- [26] Yang F, Liu X, Lu L, Wang B, Liu C. Independent mobility GPT (IDM-GPT): a self-supervised multi-agent large language model framework for customized traffic mobility analysis using machine learning models. *arXiv* 2025, arXiv:2502.18652.
- [27] Yan X, Yang X, Jin N, Chen Y, Li J. A general AI agent framework for smart buildings based on large language models and ReAct strategy. *Smart Constr.* 2025, 2(1):1–2.
- [28] Kalyuzhnaya A, Mityagin S, Lutsenko E, Getmanov A, Aksenkin Y. LLM agents for smart city management: enhancing decision support through multi-agent AI systems. *Smart Cities* 2025, 8(1):19.
- [29] Arslan M, Ghanem H, Munawar S, Cruz CM. A survey on RAG with LLMs. *Procedia Comput. Sci.* 2024, 246:3781–3790.
- [30] Mithe R, Indalkar S, Divekar N. Optical character recognition. *Int. J. Recent Technol. Eng.* 2013, 2(1):72–75.
- [31] Lewis P, Perez E, Piktus A, Petroni F, Karpukhin V. Retrieval-augmented generation for knowledge-intensive NLP tasks. *Adv Neural Inf Process Syst* 2020, 33:9459–9474.
- [32] Yang A, Li A, Yang B, Zhang B, Hui B, *et al.* Qwen3 technical report. *arXiv* 2025,

arXiv:2505.09388.

- [33] Reimers N, Gurevych I. Sentence-BERT: sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Hong Kong, China, November, 2019, pp. 3982–3992.
- [34] Zhang T, Kishore V, Wu F, Weinberger KQ, Artzi Y. BERTScore: evaluating text generation with BERT. *arXiv* 2019, arXiv:1904.09675.
- [35] Kaplan J, McCandlish S, Henighan T, Brown TB, Chess B, *et al.* Scaling laws for neural language models. *arXiv* 2020, arXiv:2001.08361.
- [36] Aizawa A. An information-theoretic perspective of TF–IDF measures. *Inf. Process. Manage.* 2003, 39(1):45–65.