Knowledge-Graph-Grounded Multimodal Explanations for Trustworthy Time-Series Forecasting in DevOps Systems

Łukasz Sobczak[1,*], Nur Keleşoğlu[1] and Joanna Domańska[1]

[1]Institute of Theoretical and Applied Informatics, Polish Academy of Sciences, Bałtycka 5, 44-100, Gliwice, Poland

Abstract

Knowledge Graphs (KG) are increasingly used to improve the factual reliability of large language models, yet their role in explainable AI has largely been limited to either grounding generated responses or evaluating them. In this paper, we propose a knowledge-graph-grounded multimodal framework for explainable time-series forecasting in DevOps systems, where a single domain knowledge graph serves as the shared semantic foundation for explanation generation, evaluation, and programmatic verification. The framework integrates numerical forecasts, visual representations, SHAP-based feature attribution, temporal dependency analysis, uncertainty estimates, and symbolic domain knowledge within a multi-agent architecture. Operational rules and model-behavior concepts are explicitly represented in a RDF knowledge graph and retrieved dynamically during inference, enabling explanations that are simultaneously model-aware, semantically grounded, and externally auditable. We further introduce a KG-grounded evaluation protocol combining an ensemble of LLM judges with an independent graph-based verifier that automatically validates citations and factual claims. Experiments on a production-scale GPU cluster forecasting task demonstrate that semantic grounding substantially improves explanation groundedness while reducing KG contradictions by more than half and achieving 99.5% citation precision. The results show that even a compact, carefully engineered domain knowledge graph can significantly enhance the trustworthiness, traceability, and verifiability of LLM-generated explanations.

Keywords

Knowledge Graphs, Explainable AI, Multimodal LLMs, Time Series Forecasting, LLM-as-a-Judge, AI Agents, DevOps

1. Introduction

Recent progress in large language models (LLMs) has substantially reshaped the development of intelligent systems by enabling capabilities such as natural language interaction, reasoning, and decision support. More recently, these models have been expanded to support multimodal applications, integrating textual reasoning with structured data, visual information, and contextual cues.

One such domain is DevOps and cloud infrastructure management, where large-scale time-series data must be continuously monitored and analyzed to support tasks such as workload forecasting, anomaly detection, and resource allocation. Despite the high predictive performance achieved by modern forecasting models, their predictions are often difficult to interpret, as they provide limited insight into the underlying reasoning, thereby restricting operator trust and informed decision-making.

This limitation has led to increasing attention toward explainability and interpretability in machine learning and decision-support systems. Explainable AI (XAI) aims to generate post-hoc explanations for model predictions, helping users understand why a particular outcome was produced without explicitly exposing the model's internal reasoning process. By contrast, Interpretable AI (IAI) prioritizes transparency of the underlying model mechanisms, illustrating how predictions are generated, although it does not necessarily establish whether the reasoning itself is meaningful or well justified [1].

Existing approaches to explainability in time-series systems typically fall into two categories: (i) language-based reasoning, where LLMs generate textual descriptions of trends, and (ii) post-hoc

XAI-KG@ISWC2026 - 2nd International Workshop on eXplainable AI and Knowledge Graphs, co-located with ISWC 2026, October 25-29, 2026, Bari, Italy

*Corresponding author.

lsobczak@iitis.pl (Ł. Sobczak); nkelesoglu@iitis.pl (N. Keleşoğlu); joanna@iitis.pl (J. Domańska)

0000-0001-9439-1812 (Ł. Sobczak); 0000-0002-0306-7281 (N. Keleşoğlu); 0000-0002-1935-8358 (J. Domańska)



© 2026 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR Workshop Proceedings ceur-ws.org ISSN 1613-0073

published 2026-11-07

CEUR-WS.org/Vol-4318/paper2.pdf

analytical methods, such as Local Interpretable Model-agnostic Explanations (LIME) [2], Shapley Additive Explanations (SHAP) [3], ANCHORS [4], and saliency-based methods [5].

A third, increasingly prominent direction grounds explanations in *knowledge graphs* (KGs), which represent domain knowledge as explicit, semantically typed entities and relations. KGs have long served as a vehicle for explainable AI by letting systems justify decisions in terms of structured background knowledge [6, 7]. More recently, KGs have been paired with LLMs to constrain generation, improve factual reliability, and reduce hallucination [8, 9]. This makes KGs a natural substrate for trustworthy explanations because they anchor generated claims in explicit, auditable facts.

However, applying KGs to explain predictive time series models remains largely unexplored, particularly in settings where explanations must jointly integrate static domain knowledge and model-derived signals. Current systems often produce fragmented explanations: language-based reasoning lacks grounding in model behavior, analytical methods are rarely integrated into interactive workflows, and existing KG-based approaches typically use the graph either for generation or evaluation [10, 11], but not both. Consequently, no existing framework uses a single domain KG to ground explanation generation, define evaluation ground truth, and enable automatic verification of generated claims.

Such a framework naturally requires reasoning across heterogeneous modalities. Multimodal LLMs (MLLMs) offer a promising direction by enabling joint reasoning across textual, numerical, and visual modalities. However, integrating these capabilities into a coherent KG-grounded interaction pipeline remains largely unexplored.

In this paper, we propose a knowledge-graph-grounded, multimodal multi-agent framework for explainable time-series forecasting in DevOps systems. Instead of considering explainability solely as a post-hoc analysis stage, our approach incorporates model-aware information into an interactive multimodal reasoning pipeline including SHAP based feature attributions, temporal relationships, and uncertainty estimates. Both system knowledge and these model-aware signals are represented within an explicit KG, which is queried by the agents during runtime to support the reasoning process.

The framework is evaluated using four progressively richer reasoning configurations, allowing us to isolate the contribution of multimodal reasoning, model awareness, and KG grounding. To improve reliability, we introduce a cross-modal grounding strategy that aligns numerical forecasts, visual representations, model-derived signals, and KG facts within a unified reasoning process. Furthermore, KG-grounded citations enable LLM-free verification of generated claims, making system outputs externally auditable.

We evaluate the proposed system in a realistic DevOps scenario using a large-scale GPU cluster dataset. The four progressively richer configurations are compared using an ensemble of LLM judges and programmatic KG verification. Results show that multimodal grounding improves explanation quality, while KG grounding provides the largest gains in groundedness, factual consistency, and auditability.

The rest of the paper is organized as follows. Section 2 reviews the state of the art about works on multimodal, language-based, and knowledge-graph-based grounding explanability in AI-driven systems. Section 3 presents the proposed system architecture. Section 4 describes the experimental setup, including the dataset, preprocessing steps, forecasting model, and evaluation methodology. Section 5 reports the experimental results. Finally, Section 6 concludes the paper and outlines directions for future work.

2. Related Works

In this section, we review prior work on explainability for data intensive AI systems, ranging from language-based reasoning to knowledge-graph-based grounding, and position our contribution with respect to these research directions. Existing approaches can be broadly grouped according to how interpretability is achieved: (i) language-based reasoning for time-series analysis, (ii) LLM-based textual explanation, (iii) structured and knowledge-grounded explainability, and (iv) knowledge-graph-based explainability and grounding, which is most closely related to our work.

2.1. Language-Based Reasoning for Time-Series Analysis

A recent line of work has explored the use of LLMs for time-series analysis by reformulating temporal data into natural language representations [12]. These approaches enable tasks such as time-series interpretation and question answering, including trend detection, seasonal pattern analysis, similarity evaluation, and causal reasoning [13, 14, 15, 16]. While such methods improve the accessibility and flexibility of time-series analysis, they primarily rely on language-based reasoning and do not explicitly provide model-aware or quantitatively grounded explanations of forecasting behavior.

2.2. LLM-Based Textual Explanation

Early efforts focused on leveraging LLMs to generate natural language explanations for model predictions. For instance, MLlm-DR [17] introduces a multimodal framework for depression recognition, where explanations are generated as clinician-style rationales accompanying diagnostic predictions. Similarly, TimeXL [18] employs an iterative prediction–reflection–refinement loop to improve forecasting quality while producing more interpretable textual reasoning. Although these methods enhance human readability, the generated explanations remain largely implicit, providing limited insight into the internal behavior of predictive models.

2.3. Structured and Knowledge-Grounded Explainability

Recent research has moved beyond purely textual explanations toward richer and more structured forms of interpretability. MMRepAgent [19] generates comprehensive financial reports by integrating structured market data, news-derived events, predictive models, visualizations, and knowledge graphs into coherent multimodal explanations. In the medical domain, HEAL-LLMF [20] combines multimodal data fusion with Bayesian causal inference and multi-instance learning to produce rationale-aligned diagnostic predictions, while DiagLLM [21] integrates physiological signals with expert knowledge to enable knowledge-grounded diagnostic reasoning. Complementary concept-based approaches interpret predictions using human-understandable semantic concepts shared across modalities, improving transparency through structured semantic representations rather than free-form textual explanations [22]. Collectively, these methods demonstrate that incorporating structured knowledge, causal reasoning, or semantic concepts substantially improves explanation quality. However, they generally remain focused on reasoning over domain knowledge rather than exposing the behavior of the underlying predictive model itself, such as feature contributions, temporal dependencies, or uncertainty estimates.

Across these directions, explanations remain either primarily language-based or focused on reasoning over external knowledge, with only limited access to the internal mechanisms of predictive models. In contrast, our approach combines analytical and data-driven explanation components with LLM-based reasoning. Specifically, we integrate SHAP-based feature attribution for fine-grained contribution analysis, temporal dependency structures to capture sequential dynamics, forecast behavior analysis across prediction horizons, uncertainty estimation for reliability assessment, and multimodal visual representations grounded in a domain knowledge graph. This enables explanations that are simultaneously human-readable, quantitatively grounded, verifiable, and directly linked to model behavior.

2.4. Knowledge-Graph-Based Explainability and Grounding

A complementary and long-standing line of research uses KGs as a vehicle for explainability rather than merely as a data store. Structured, semantically typed background knowledge enables AI systems to justify decisions in terms of explicit entities and relations, which is particularly valuable for otherwise opaque models [6]. Survey work characterizes how KGs support explainable machine learning by supplying background knowledge, enabling symbolic reasoning, and exposing human-interpretable structure [7]. A systematic review by Rajabi *et al.* [23] organizes KG usage in XAI along two dimensions: the function served (feature extraction, relation extraction, KG construction, and KG reasoning) and the stage of the pipeline (pre-, in-, and post-model). Their analysis shows that KGs are predominantly

employed before model inference for feature engineering and after inference for reasoning, particularly in transparency-critical domains such as healthcare. Likewise, a broader survey of comprehensible AI on KGs identifies grounding predictions in explicit domain knowledge as a key mechanism for trustworthy explanations [24]. Our work follows this perspective by representing operational rules and model-behavior concepts as first-class graph entities, while extending it to the comparatively unexplored setting of explaining predictive time-series forecasting models.

More recently, integrating KGs with LLMs has become a prominent strategy for grounding generated content and mitigating hallucinations. Roadmap studies argue that symbolic knowledge and neural language models are complementary, advocating architectures in which graph-structured knowledge constrains and verifies neural generation [8], while dedicated surveys summarize KG augmentation techniques and their impact on factual reliability [9]. Retrieval over graph structures, as exemplified by GraphRAG [25], and explicit linking of reasoning steps to graph-supported evidence [26] improve both faithfulness and traceability compared with conventional prompting. A closely related research direction uses KGs not to generate explanations but to evaluate them. GraphEval represents LLM outputs as knowledge graphs and identifies unsupported triples to provide fine-grained hallucination detection [10], while provenance-verification pipelines such as ProVe automatically validate KG-derived claims against supporting evidence [11]. These approaches typically separate generation from evaluation and focus on open-domain question answering. In contrast, our framework employs a single domain knowledge graph as the shared source of truth for explanation grounding, explanation evaluation, and citation verification without relying on an additional LLM verifier. Furthermore, we extend graph grounding beyond domain facts to encompass model-aware signals, including SHAP feature groups, temporal dependencies, forecast behavior, and predictive uncertainty, allowing explanations of why a forecast was produced to be audited with the same level of traceability as factual claims about the underlying system.

3. System Overview

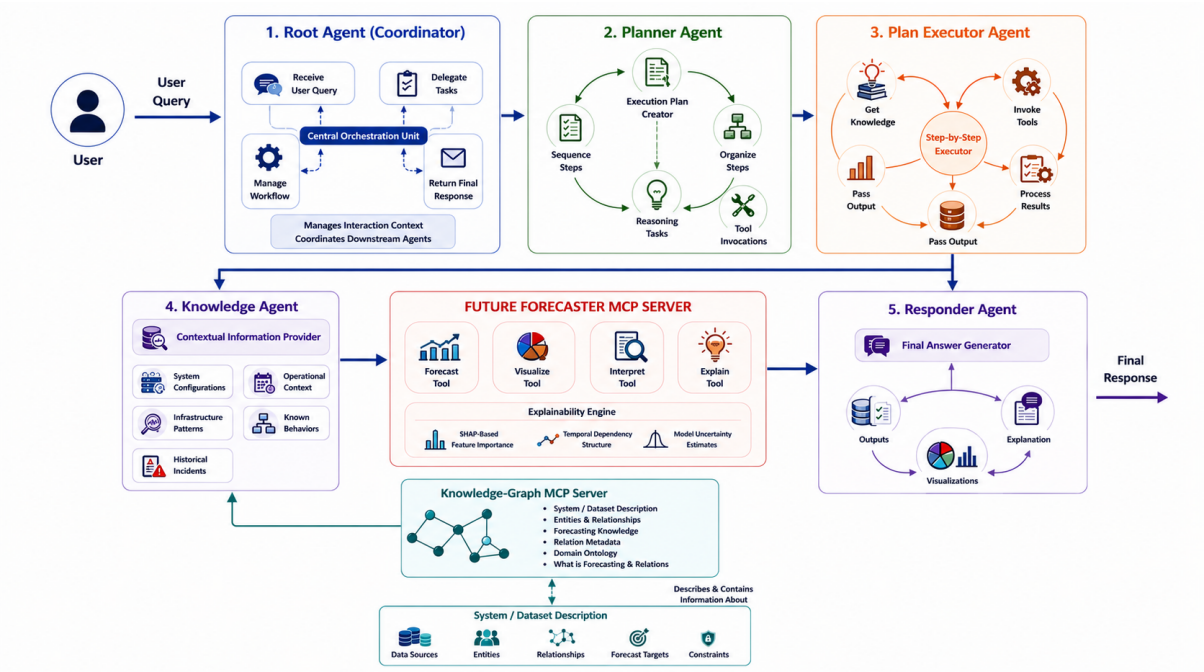


Figure 1: Overview of the proposed multi-agent architecture integrating analytical forecasting tools with semantic reasoning through a Knowledge Graph MCP server.

Figure 1 presents the architecture of the proposed explainable forecasting framework. The system combines a multi-agent orchestration layer with two complementary sources of reasoning: (i) analytical

forecasting services exposed through the Future Forecaster MCP server and (ii) symbolic domain knowledge provided by a dedicated Knowledge Graph (KG) MCP server. Together, these components enable generating explanations grounded simultaneously in numerical evidence, model-aware explainability signals, and explicit semantic knowledge.

The overall orchestration pipeline extends our previously proposed explainable forecasting framework [27]. While the original architecture focused on multimodal reasoning over forecasts, visualizations, and explainability signals, the present work introduces semantic grounding through a domain knowledge graph. Instead of relying solely on prompt-engineered background knowledge, factual information is retrieved dynamically from the KG during explanation generation.

The Root Agent coordinates the interaction workflow and delegates user requests to specialized downstream agents. The Planner Agent transforms natural language queries into executable plans, while the Plan Executor Agent carries out the required reasoning steps through MCP tool invocations. Analytical operations, including forecasting, visualization, interpretation, and explainability, are provided by the Future Forecaster MCP server. Whenever domain-specific knowledge is required, the Knowledge Agent retrieves relevant information from the KG MCP server. Finally, the Responder Agent integrates all collected evidence into the final natural-language response.

3.1. Knowledge Graph Construction

The KG was hand-engineered by the authors as a lightweight ontology expressed in the Web Ontology Language (OWL) (10 classes, $\sim$30 properties, all annotated with `rdfs:label`/`rdfs:comment`) populated with instance data, following a classical knowledge-engineering process over three authoritative sources:

1. **Dataset documentation.** Infrastructure and provenance facts (node count, physical card count, job/organization/trace statistics) were extracted from the documentation of the public cluster trace [28].
2. **Operational semantics.** The capacity, metric, and queueing/clipping semantics previously encoded as a free-text fact sheet in the knowledge agent's prompt were formalized as entities and reified operational rules. Reification was a deliberate design decision: each rule exposes its premise metric, comparator, threshold property, implied state, and consequence as separate assertions, so that a generated explanation can cite the exact inference path rather than the rule as an opaque whole.
3. **The forecasting pipeline itself.** The model-behavior layer (algorithm, hyperparameters, lag-group boundaries, SHAP configuration, uncertainty method) was extracted directly from the implementation, ensuring that the KG describes the deployed model rather than an idealized one.

Notably, the formalization step itself surfaced a latent ambiguity that free-text knowledge had masked: the source trace reports 11,702 physical GPU cards, while the operational saturation threshold used throughout the system is 10,412 GPUs. The KG resolves this by modeling both as distinct properties (`gc:gpuCardCount` vs. `gc:gpuCapacity`) with explicit semantics, a small but characteristic example of how KG construction enforces consistency that prompt engineering does not.

The graph was validated at three levels: *syntactically*, by verifying correct RDF/Turtle parsing; *semantically*, through competency questions evaluated using SPARQL (the standard query language for RDF graphs); and *functionally*, by ensuring consistent behavior across three independent consumers. Representative competency questions included "Which state is implied when the metric exceeds capacity?" and "Which feature groups does SHAP attribute to?" Because the same KG serves the knowledge agent, the judge context generator, and the programmatic verifier, modeling errors would immediately surface as citation-validation failures or judge–system disagreements. At 228 triples, the KG is deliberately compact. Our results indicate that even a small, carefully curated domain KG yields measurable grounding gains, which we consider an encouraging message for practitioners. (Semi-)automating its construction from configuration files and monitoring metadata is left for future work.

3.2. Knowledge Graph Integration

A key contribution of the proposed architecture is the introduction of a semantic reasoning layer based on a domain knowledge graph. Unlike the analytical MCP server, which operates on numerical time-series data, the KG MCP server provides structured symbolic knowledge represented in Resource Description Framework (RDF) using the Turtle serialization format, a W3C standard for encoding entities and their relationships in knowledge graphs. Rather than embedding domain knowledge directly into prompts, agents retrieve only the information required for the current reasoning task.

The KG MCP server acts as a semantic retrieval layer between the multi-agent framework and the underlying ontology. It exposes three MCP tools:

- `get_system_context`, returning the core facts describing the monitored infrastructure,
- `describe_entity`, providing detailed information about selected ontology entities,
- `sparql_query`, enabling arbitrary SPARQL queries over the knowledge graph.

This separation allows the domain knowledge to evolve independently of prompt engineering while ensuring that generated explanations remain explicitly grounded in the ontology.

3.3. Domain Knowledge Graph

The knowledge graph consists of RDF triples organized into three complementary layers; Figure 2 illustrates the main entities and relationships in the graph.

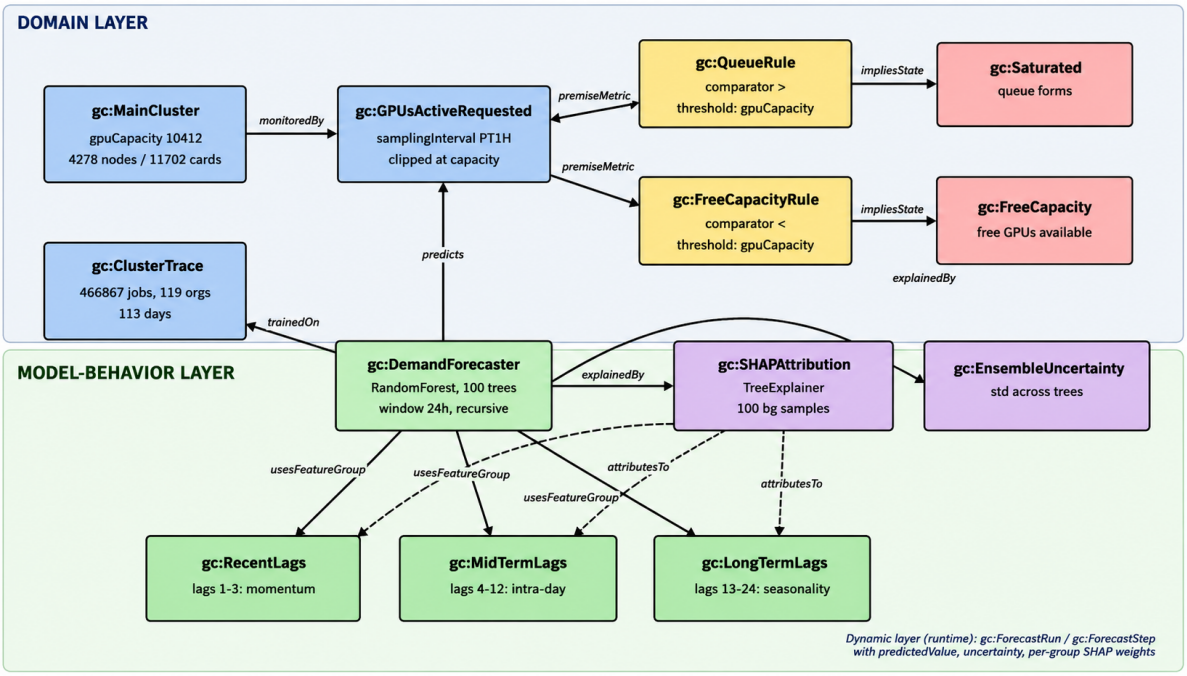


Figure 2: Main entities and relationships in the proposed domain knowledge graph. Operational rules are reified as first-class entities, while model-behavior concepts (SHAP feature groups and uncertainty) are explicitly represented in the ontology.

- **Domain layer.** Represents the monitored GPU infrastructure, including cluster characteristics, monitored metrics, and operational rules governing system behavior. In particular, operational rules are modeled as first-class entities rather than implicit prompt knowledge. Consequently, reasoning about states such as cluster saturation becomes an explicit graph traversal instead of an unsupported textual assumption.

- **Model layer.** Captures the forecasting model together with its explainability metadata, including SHAP feature groups, temporal dependency categories, and uncertainty estimates. By representing explainability concepts directly in the ontology, explanations referring to model behavior become semantically grounded.
- **Dynamic layer.** Stores runtime entities describing individual forecasting sessions, including predicted values, uncertainty estimates, and aggregated feature attribution scores generated during inference.

3.4. Grounded Explanation Generation

Previous versions of the framework relied on manually maintained prompt knowledge describing the monitored infrastructure. Although effective, this approach offered limited traceability and required manual updates whenever the system configuration changed.

In the proposed architecture, prompt based knowledge is replaced by semantic retrieval from the knowledge graph. During execution, the Knowledge Agent dynamically retrieves domain facts through the KG MCP server, while the Explain Tool combines these symbolic facts with numerical forecasts, SHAP attributions, temporal importance scores, and uncertainty estimates produced by the Future Forecaster MCP server.

Consequently, explanations are generated from two complementary evidence streams:

1. **Analytical evidence**, consisting of forecasts, visualizations, SHAP explanations, temporal dependencies, and uncertainty estimates;
2. **Semantic evidence**, consisting of infrastructure configuration, operational rules, forecasting model metadata, and other facts retrieved from the knowledge graph.

The Explain Tool fuses both evidence sources into a single grounded explanation. Static statements are supported by ontology entities, while analytical claims remain consistent with the underlying forecasting model, improving both traceability and factual consistency.

To make the notion of grounding explicit, we define the following concepts.

Definition 1 (Grounded Claim). A claim is considered grounded if it can be traced to one or more explicit evidence items originating either from the analytical forecasting pipeline or from the knowledge graph.

Definition 2 (Grounded Explanation). An explanation is grounded if every factual claim contained in the explanation is grounded.

The proposed grounding mechanism can be characterized independently of the implementation by defining the relationship between generated claims and their supporting evidence. Formally, let

$$E = \{c_1, \ldots, c_n\}$$

denote the set of claims contained in the generated explanation. For each claim $c \in E$, we define its supporting evidence as

$$support(c) \subseteq \mathcal{A} \cup \mathcal{K},$$

where $\mathcal{A}$ denotes analytical evidence and $\mathcal{K}$ semantic evidence retrieved from the knowledge graph.

A grounded explanation therefore satisfies

$$\forall c \in E, \qquad support(c) \neq \emptyset.$$

This formulation directly reflects the objective of the proposed framework: every generated statement should remain traceable either to numerical evidence produced by the forecasting model or to explicit semantic knowledge represented in the ontology.

3.5. Worked End-to-End Example

To make the grounding process concrete, we trace one query from our evaluation benchmark (query #1, overload-risk analysis) through the full KGF pipeline; all artifacts below are taken verbatim from the recorded system run and its verification output.

1. **Planning.** The Planner Agent matches the query to the forecasting plan template and emits a five-step plan: retrieve system knowledge, forecast, visualize, explain, respond.
2. **Semantic evidence.** The Knowledge Agent calls `get_system_context` on the KG MCP server and returns cited facts, including the saturation threshold (`gc:MainCluster gc:gpuCapacity` = 10,412) and the reified queueing rule (`gc:QueueRule` → `gc:Saturated`).
3. **Analytical evidence.** The Plan Executor invokes the `forecast` tool (48 h horizon, converted to 172,800 s) on the hourly trace; the tool persists the model artifacts required for explainability. The `visualize` tool renders a forecast plot and an animated simulation.
4. **Grounded explanation.** The `explain` tool computes SHAP attributions (aggregated into the three lag groups) and per-step ensemble uncertainty, and passes them — together with the forecast, the plot, and the cited KG facts (`kg_context`) — to the multimodal LLM, whose prompt enforces the citation rules of Section 3.4.
5. **Response.** The Responder Agent assembles the final markdown answer, preserving all citations.
6. **Verification.** Offline, the programmatic verifier resolves every citation and numeric claim against the graph.

The user query and a fragment of the generated explanation (lightly abridged; typography normalized to ASCII):

```
Query #1
GPU usage frequently approaches the cluster capacity [...] analyze whether the recent
usage pattern indicates a sustained risk of exceeding safe utilization thresholds (e.g.,
above 90% capacity). [...] forecast GPU usage for the next 48 hours and assess whether
similar peaks are likely to occur.
```

```
KGF response (fragment)
Operational saturation threshold = 10,412 GPUs
[gc:MainCluster gc:gpuCapacity].
90% threshold  = 9,371 GPUs (0.9 x 10,412)
[gc:MainCluster gc:gpuCapacity].
Main drivers:
- strong short-term momentum (lag_1/RecentLags)
- daily seasonality (lag_24/LongTermLags)
[gc:RecentLags gc:capturesPattern] [gc:LongTermLags gc:capturesPattern].
Ensemble uncertainty (std) mean  = 460 GPUs; uncertainty spikes near ramps (up to  = 1,069)
[gc:EnsembleUncertainty gc:method].
Point forecasts stay below capacity so the system would be 'FreeCapacity' by rule,
but uncertainty makes capacity exceedance plausible, implying a 'Saturated' state and
queue formation if realized. [gc:FreeCapacityRule gc:premiseMetric gc:GPUsActiveRequested]
[gc:QueueRule gc:impliesState gc:Saturated]
```

This example illustrates every element of the grounding contract of Section 3.4. Static claims draw their support from $\mathcal{K}$: the response contains 14 machine-checkable citations, all of which the verifier resolves to valid graph structure, and 4 anchored numeric claims (capacity, two 90% thresholds, one 85% threshold), all matching KG values. Dynamic claims draw their support from $\mathcal{A}$: predicted peaks ($\sim$9,959 and $\sim$10,205 GPUs) and exceedance probabilities are products of the forecasting pipeline and are asserted as such. Crucially, the two streams compose through the reified rule: the forecast value (analytical) is compared against `gc:gpuCapacity` (semantic), and the implied state is derived via the explicit path `gc:QueueRule` → `gc:Saturated` rather than asserted freely. Where no supporting entity exists, the response marks the statement explicitly (three `[no supporting KG entity]` markers)

instead of feigning grounding. The judge ensemble scored this response 4.75/5 on groundedness with a KG-contradiction rate of 0.

This example illustrates how analytical evidence and semantic knowledge jointly satisfy the grounding contract introduced in Section 3.4, while enabling both human-readable explanations and machine-verifiable provenance.

3.6. Progressive Grounding Levels

To quantify the contribution of successive reasoning capabilities, we compare four progressively richer grounding configurations:

- **Baseline Forecasting (BF).** Responses are generated solely from numerical forecasting outputs.
- **Interpretable Forecasting (IF).** Visual representations are incorporated, enabling perception-level multimodal reasoning over forecast trends.
- **Explainable Forecasting (EF).** Model-aware explainability signals, including SHAP feature attribution, temporal dependencies, and uncertainty estimates, are additionally integrated into the reasoning process.
- **Knowledge Graph Forecasting (KGF).** Extends EF by replacing prompt-based domain knowledge with semantic retrieval from the knowledge graph, producing explanations grounded simultaneously in model behavior and explicit domain knowledge.

The BF→IF→EF progression isolates the contribution of multimodal perception and model-aware explainability, whereas the EF→KGF comparison quantifies the impact of semantic grounding introduced by the knowledge graph. This progressive design allows the contribution of each additional source of evidence to be evaluated independently.

4. Experimental Setup

4.1. Dataset, Forecasting Model, and Queries

We use the open GPU Cluster Spot Resource Dataset [28]: a production-grade trace of 466,867 AI job submissions from 119 organizations over ∼113 days on a heterogeneous cluster (4,278 GPU nodes with 10,412 available GPUs). The target metric `gpus_active_requested` is forecast by a Random Forest regressor over 24 lag features with recursive multi-step prediction; SHAP (TreeExplainer, 100 background samples) provides attributions aggregated into the three lag groups, and per-step ensemble standard deviation provides uncertainty. The evaluation uses 30 representative operator queries covering overload risk, temporal patterns, anomalies, capacity planning, and reasoning under uncertainty; each query is processed by all four pipelines under identical conditions.

4.2. KG-Grounded LLM-as-a-Judge Protocol

Each response is scored on ten criteria: clarity, helpfulness, insightfulness, trustworthiness, model awareness, consistency, and groundedness (0–5); hallucination (0–1, reported as *KG-contradiction rate*); actionability and uncertainty awareness (0–2). A weighted overall score emphasizes insightfulness, trustworthiness, model awareness, and groundedness. Two design choices distinguish the protocol:

KG-derived judge context. The judge's ground-truth system description is *generated from the knowledge graph* (57 facts serialized as triples), so the system under test and its evaluator share one source of truth.

Two-tier claim taxonomy. A preliminary evaluation round with a naive closed-world instruction ("use ONLY these facts") exposed a calibration failure: legitimate dynamic forecast values (predicted GPU counts, probabilities, horizons), which the static KG intentionally does not contain, were scored as unsupported — saturating the hallucination flag at 1.0 for *all* 120 responses and collapsing groundedness variance. The revised rubric instructs judges to (A) judge *static* system/model claims strictly against the KG, and (B) treat *dynamic* forecast outputs as legitimate unless they violate KG semantics (e.g., reporting the metric above capacity without queueing/clipping). A calibration probe confirms the fix: a correct uncited response receives hallucination 0 / groundedness 3; a corrupted response (wrong capacity, invented model and autoscaler) receives hallucination 1 / groundedness 0.

Multi-judge ensemble. To mitigate single-judge and same-family bias, four judges from different model families score all responses under identical prompts: GPT-5.4, Gemini 2.5 Pro, and two locally hosted open-weight models (gpt-oss-20B, Qwen3.6-35B). Scores are averaged per item; agreement is quantified with Krippendorff's α and pairwise Spearman correlations.

4.3. Programmatic KG Verification

Independently of any LLM, a verifier parses every response and checks against the graph: (a) *citation validity* — each `[gc:...]` citation is resolved and validated at subject/pair/triple/path level; (b) *static numeric claims* — numbers attached to unambiguous anchors (capacity, thresholds derived from it, tree count, input window, dataset statistics) are compared with KG values, with ground truth read from the graph rather than hardcoded. This complementary verifier provides an objective, LLM-independent assessment of factual correctness and citation validity, strengthening the overall evaluation methodology.

4.4. Statistical Analysis

Per-query judge-ensemble means are compared using paired Wilcoxon signed-rank tests (Pratt zero handling, $n = 30$) for all six system pairs, with Holm correction applied within each evaluation metric. We report matched-pairs rank-biserial correlations together with bootstrap 95% confidence intervals (10,000 resamples). The EF–KGF comparison is treated as the pre-specified primary analysis.

5. Results

All resources needed to reproduce the results (source code, KG, queries, responses, evaluation scripts) are publicly available at https://doi.org/10.5281/zenodo.21495764.

Table 1 summarizes the ensemble results. The BF→IF→EF progression replicates our earlier findings (multimodal grounding improves helpfulness and insightfulness; model-aware signals improve trustworthiness and model awareness). The new KGF variant improves *every* metric over EF, with the largest gains exactly where KG grounding should act: groundedness $+1.06$ (3.58→4.64) and a KG-contradiction rate reduced by $2.2\times$ (0.23→0.11).

5.1. Judge-Ensemble Comparison

For the primary comparison (KGF vs. EF), the overall-score difference of $+0.29$ is significant after Holm correction ($p < 0.001$) with a matched-pairs rank-biserial correlation of $r = +1.00$: KGF is at least as good as EF on *every one* of the 30 paired queries. Metric-level primary results: groundedness $+1.06$ ($p < 0.001$, $r = +1.00$), model awareness $+0.31$ ($p < 0.001$, $r = +1.00$), KG-contradiction -0.125 ($p = 0.006$, $r = -0.78$).

Table 1
Judge-ensemble means (4 judges, 30 queries per system). Higher is better except KG-contradiction. Best in bold.

Metric	BF	IF	EF	KGF
Clarity	4.43	4.49	4.48	**4.49**
Helpfulness	4.50	4.92	4.89	**4.96**
Insightfulness	3.78	4.16	4.36	**4.52**
Trustworthiness	3.79	3.83	4.22	**4.44**
Model awareness	2.39	3.68	4.66	**4.97**
Consistency	4.73	4.46	4.71	**4.91**
Groundedness	3.62	3.25	3.58	**4.64**
KG-contradiction ↓	0.26	0.32	0.23	**0.11**
Actionability	1.99	2.00	2.00	**2.00**
Uncertainty awareness	1.39	1.50	1.95	**2.00**
Overall	3.44	3.68	4.01	**4.29**

Ranking stability across judge families. Figure 3 shows the central robustness result: all four judges, spanning two commercial families and two open-weight models, yield the identical ranking BF $<$ IF $<$ EF $<$ KGF. Raw-scale agreement is modest (per-metric ordinal Krippendorff's α ranges from -0.26 for clarity to 0.73 for uncertainty awareness), reflecting the well-known scale-calibration differences between LLM judges; however, pairwise Spearman correlations of per-item overall scores are consistently high (0.67–0.85). LLM judges disagree on *absolute scores* but agree on *orderings*, which is precisely what the comparative conclusions rely on, and which motivates reporting rank-based statistics throughout.

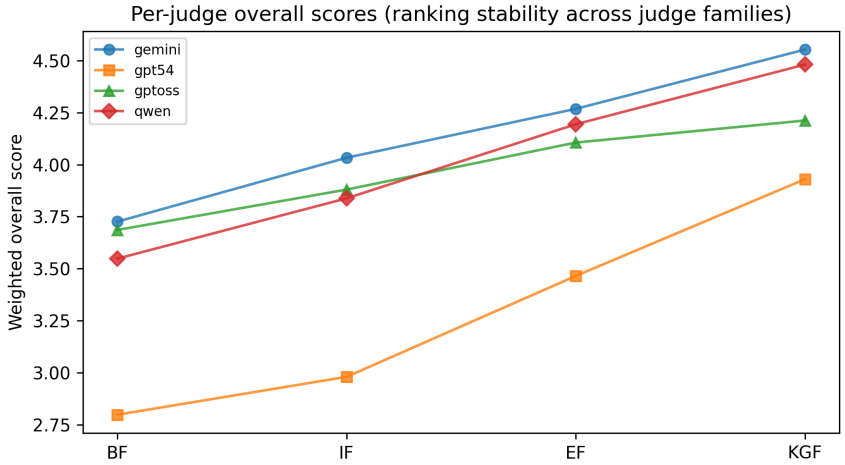


Figure 3: Weighted overall score per judge. Judges differ substantially in absolute calibration (GPT-5.4 strictest, Gemini most lenient), yet all four independently produce the same ranking BF $<$ IF $<$ EF $<$ KGF.

5.2. Programmatic Verification Against the KG

Table 2 shows the verification results. Three observations stand out. First, only KGF produces *machine-checkable provenance*: 432 citations across all 30 responses, of which 99.5% resolve to valid graph structure (the two rejects are malformed model outputs, correctly flagged). Second, KG grounding roughly *doubles the density of checkable static claims* (49 vs. 20-25): explanations become more anchored in explicit system facts rather than vague references. Third, the verifier caught a genuine reasoning error in a non-grounded variant, an IF response computing a 90% capacity threshold as $\sim$13,980 GPUs (from a capacity of 10,412), exactly the class of confident numerical error that KG citations render detectable, and that text-only evaluation misses.

Table 2
LLM-free verification against the knowledge graph (30 responses per system).

	BF	IF	EF	KGF
KG citations emitted	0	0	0	**432**
valid (citation precision)	—	—	—	**430 (0.995)**
Checkable static numeric claims	21	20	25	**49**
verified against KG	21	19	25	**49**

5.3. Lessons for KG-Based Evaluation

Our initial calibration run demonstrates a pitfall relevant to the evaluation of KG-based explanations in dynamic domains: a closed-world judging instruction over a *static* KG misclassifies legitimate *dynamic* model outputs as hallucinations, destroying metric variance (all 120 responses flagged). The two-tier claim taxonomy (static claims $\rightarrow$ KG-verifiable; dynamic outputs $\rightarrow$ semantics-constrained) restored discriminative power while keeping the KG as the single source of truth. We propose this taxonomy as a reusable protocol element for benchmarking KG-grounded explanations of predictive models.

6. Conclusion

This paper presented a knowledge-graph-grounded framework for explainable time-series forecasting in DevOps systems. Rather than treating the knowledge graph as an auxiliary source of background knowledge, we use it as a shared semantic layer that simultaneously grounds explanation generation, defines the evaluator's reference knowledge, and enables programmatic verification of generated claims. By integrating symbolic domain knowledge with analytical model outputs within a multimodal multi-agent architecture, the proposed framework produces explanations that are not only more informative but also explicitly traceable and externally auditable.

Experimental results demonstrate that the KG-grounded configuration consistently outperforms the baseline, interpretable, and explainable variants. In the primary comparison against the explainable forecasting (EF) configuration, groundedness increases from 3.58 to 4.64, model awareness further improves, and the KG-contradiction rate is reduced by a factor of $2.2\times$. Furthermore, citation precision of 0.995 shows that generated explanations can be verified automatically against the underlying ontology, providing an additional level of reliability beyond conventional LLM-based evaluation.

Future work will focus on extending the dynamic knowledge graph layer to capture runtime forecasting provenance, enriching the ontology with causal operational knowledge, automating knowledge graph construction from monitoring and configuration sources, and conducting a human evaluation to investigate the agreement between domain experts, LLM judges, and the proposed programmatic KG verification.

Acknowledgments

This work was partly supported by the National Centre for Research and Development (NCBR) and co-funded by the European Union under the European Funds for Modern Economy (FENG) Programme (SMART Konsorcja), project no. FENG.01.01-IP.01-A0GV/24-00, "Advanced maintenance and diagnostic tool for IT start-ups". This work was also supported by COST Action CA24121 – Knowledge Graphs in the Era of Large Language Models (KGELL).

Declaration on Generative AI

During the preparation of this work, the authors used Claude (Anthropic) in order to: Paraphrase and reword, improve writing style, and check grammar and spelling. After using this tool, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

References

[1] V. Vishwarupe, P. M. Joshi, N. Mathias, S. Maheshwari, S. Mhaisalkar, V. Pawar, Explainable ai and interpretable machine learning: A case study in perspective, Procedia Computer Science 204 (2022) 869–876.

[2] M. T. Ribeiro, S. Singh, C. Guestrin, " why should i trust you?" explaining the predictions of any classifier, in: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, 2016, pp. 1135–1144.

[3] S. M. Lundberg, G. Erion, H. Chen, A. DeGrave, J. M. Prutkin, B. Nair, R. Katz, J. Himmelfarb, N. Bansal, S.-I. Lee, From local explanations to global understanding with explainable ai for trees, Nature machine intelligence 2 (2020) 56–67.

[4] M. T. Ribeiro, S. Singh, C. Guestrin, Anchors: High-precision model-agnostic explanations, in: Proceedings of the AAAI conference on artificial intelligence, volume 32, 2018.

[5] K. Simonyan, A. Vedaldi, A. Zisserman, Deep inside convolutional networks: Visualising image classification models and saliency maps, arXiv preprint arXiv:1312.6034 (2013).

[6] F. Lecue, On the role of knowledge graphs in explainable ai, Semantic Web 11 (2020) 41–51.

[7] I. Tiddi, S. Schlobach, Knowledge graphs as tools for explainable machine learning: A survey, Artificial Intelligence 302 (2022) 103627.

[8] S. Pan, L. Luo, Y. Wang, C. Chen, J. Wang, X. Wu, Unifying large language models and knowledge graphs: A roadmap, IEEE Transactions on Knowledge and Data Engineering 36 (2024) 3580–3599.

[9] G. Agrawal, T. Kumarage, Z. Alghamdi, H. Liu, Can knowledge graphs reduce hallucinations in LLMs?: A survey, in: Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), Association for Computational Linguistics, Mexico City, Mexico, 2024, pp. 3947–3960. doi:10.18653/v1/2024.naacl-long.219.

[10] H. Sansford, N. Richardson, H. P. Maretic, J. N. Saada, GraphEval: A knowledge-graph based LLM hallucination evaluation framework, in: Workshop on Knowledge-infused Learning (KiL) at KDD 2024, 2024. ArXiv:2407.10793.

[11] G. Amaral, O. Rodrigues, E. Simperl, ProVe: A pipeline for automated provenance verification of knowledge graphs against textual sources, Semantic Web 15 (2024) 2159–2192. doi:10.3233/SW-233467.

[12] Y. Kong, Y. Yang, S. Wang, C. Liu, Y. Liang, M. Jin, S. Zohren, D. Pei, Y. Liu, Q. Wen, Position: Empowering time series reasoning with multimodal LLMs, arXiv preprint arXiv:2502.01477 (2025).

[13] J. Chen, A. Feng, Z. Zhao, J. Garza, G. Nurbek, C. Qin, A. Maatouk, L. Tassiulas, Y. Gao, R. Ying, MTBench: A multimodal time series benchmark for temporal reasoning and question answering, arXiv preprint arXiv:2503.16858 (2025).

[14] Y. Kong, Y. Yang, Y. Hwang, W. Du, S. Zohren, Z. Wang, M. Jin, Q. Wen, Time-MQA: Time series multi-task question answering with context enhancement, in: Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2025, pp. 29736–29753.

[15] C. Wang, Q. Qi, J. Wang, H. Sun, Z. Zhuang, J. Wu, L. Zhang, J. Liao, ChatTime: A unified multimodal time series foundation model bridging numerical and textual data, in: Proceedings of the AAAI Conference on Artificial Intelligence, volume 39, 2025, pp. 12694–12702.

[16] Z. Xie, Z. Li, X. He, L. Xu, X. Wen, T. Zhang, J. Chen, R. Shi, D. Pei, ChatTS: Aligning time series with LLMs via synthetic data for enhanced understanding and reasoning, Proceedings of the VLDB Endowment 18 (2025) 2385–2398.

[17] W. Zhang, J. Chen, E. Zhu, W. Cheng, Y. Li, Y. Li, Y. J. Wang, Mllm-dr: towards explainable depression recognition with multimodal large language models, ACM Transactions on Multimedia Computing, Communications and Applications 22 (2026) 1–23.

[18] Y. Jiang, W. Yu, G. Lee, D. Song, K. Shin, W. Cheng, Y. Liu, H. Chen, Timexl: Explainable multimodal time series prediction with llm-in-the-loop, Advances in Neural Information Processing Systems 38 (2026) 135431–135467.

[19] X. Li, Y. Zeng, X. Xing, J. Xu, X. Xu, MMRepAgent: Explainable stock earnings forecasting via multimodal report agent framework, Knowledge-Based Systems (2025) 115197.
[20] M. A. Haq, A. A. Mohamed, R. K. Mahendran, HEAL-LLMF: A hierarchical explainable AI framework using LLM fusion for multi-modal and personalized healthcare decision making, Information Systems (2026) 102696.
[21] J. Wang, T. Li, Y. Yang, S. Chen, W. Zhai, DiagLLM: multimodal reasoning with large language model for explainable bearing fault diagnosis, Science China Information Sciences 68 (2025) 160103.
[22] J. Parekh, P. Khayatan, M. Shukor, A. Newson, M. Cord, A concept-based explainability framework for large multimodal models, in: Advances in Neural Information Processing Systems, volume 37, 2024, pp. 135783–135818.
[23] E. Rajabi, K. Etminani, Knowledge-graph-based explainable ai: A systematic review, Journal of information science 50 (2024) 1019–1029.
[24] S. Schramm, C. Wehner, U. Schmid, Comprehensible artificial intelligence on knowledge graphs: A survey, Journal of Web Semantics 79 (2023) 100806.
[25] D. Edge, H. Trinh, N. Cheng, J. Bradley, A. Chao, A. Mody, S. Truitt, J. Larson, From local to global: A graph RAG approach to query-focused summarization, arXiv preprint arXiv:2404.16130 (2024).
[26] A. Amayuelas, J. Sain, S. Kaur, C. Smiley, Grounding LLM reasoning with knowledge graphs, arXiv preprint arXiv:2502.13247 (2025).
[27] N. Keleşoğlu, Ł. Sobczak, J. Domańska, From plots to words: Model-aware multimodal explanations as a foundation for accessible, non-visual interaction, arXiv preprint arXiv:2608.24910 (2026).
[28] S. I. Ovi, GPU cluster spot resource dataset analysis, https://github.com/sultanul-ovi/GPU-Cluster-Spot-Resource-Dataset-Analysis, 2025. Accessed: 2026-04-23.