A Prototyping Environment for Comparing Neuro-symbolic Diagnostic Assistants

Francesco Compagno[1,2][0000−0003−1002−608X], Jan-Christoph Kalo[1][0000−0002−5492−2292], and Paul Groth[1][0000−0003−0183−6910]

[1] University of Amsterdam, Science Park 904, Amsterdam, Netherlands
{f.compagno,p.t.groth,j.c.kalo}@uva.nl
[2] ASML**, De Run 6501, 5504 DR Veldhoven, Netherlands

Abstract. We present an environment that allows for the comparison of different diagnostic assistants for engineered systems that use large language models, symbolic reasoning, and ontology-based knowledge graphs in different combinations. Using toy physical systems and curated diagnostic scenarios, we explore strengths and limitations of these combinations.

Keywords: Equipment failure diagnosis · Neuro-symbolic architectures · Large language models

1 Introduction

Maintaining engineering systems is a challenging and important aspect of the modern economy [2]. A key part of maintenance involves resolving equipment failures and performing diagnosis (i.e. determining the failure cause). Computational approaches to diagnosis in engineered systems abound [7]. Recent work has explored the potential for knowledge graphs (KGs) both separately and in conjunction with Large Language Models (LLMs) for diagnosis [4, 8].

In this work, we develop an environment to compare different neuro-symbolic architectures for diagnostic tools. Specifically, we develop a general architecture for a diagnostic assistant in which different LLM and symbolic components (e.g., KGs, reasoners, etc.) can be used in a variety of combinations. The architectures are then instantiated in prototypes that can then be used to assess their strengths and weaknesses.

To better showcase the capabilities of the prototypes, we built a few toy engineered physical systems (one of these systems is shown in Figure 1, see the GitHub[3] repository for details on the systems), and designed different diagnostic scenarios, which highlight challenges in diagnosis.

** This work is partly sponsored by the company ASML in the context of a joint project of ASML and the University of Amsterdam on AI for diagnosis. We thank ASML research for the support in various brainstorming sessions.

[3] https://github.com/kataph/Diagnostic-Assistant-Demo

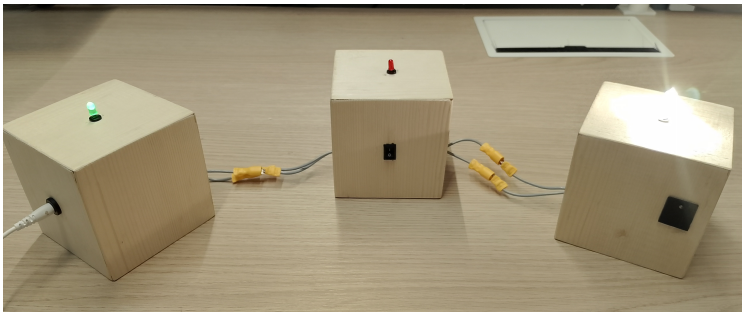

Fig. 1. A toy engineered system made of three cubes, which function respectively as power supply, control, and load subsystems.

2 Diagnostic Assistant Architecture and Environment

We model the diagnostic assistant's operating environment as follows (see Figure 2[4], left side): A physical engineered system is given, along with multimodal documentation (structured and unstructured). A diagnostic scenario begins when a saboteur agent selects and introduces a failure mode (the 'root cause'). A service agent then describes the system's current state to the assistant, initiating an iterative loop in which the assistant proposes diagnostic actions, the user executes them, and reports back the results. This loop continues until the root cause is identified.

Thus, diagnosis is abstracted as a sequence of diagnostic actions. Each diagnostic action is a verb-object couple where the verb belongs to a controlled list and the object is a component. Each action also has a cost, determined, for simplicity, by its verb.

For the diagnostic assistant architecture (Figure 2, right), we implement two options. The first is a standard LLM[5] with access to unstructured documentation and guided by a prompt.

The second is a neuro-symbolic architecture using a KG based on a recent ontology [5] describing components, functions, problems, and tests. The LLM only processes input: from each observation, it extracts mentioned components and labels their behavior as anomalous or nominal. If anomalous, the KG returns the components on which the mentioned components functionally depend; if nominal, the complementary set of components (all the components of the system on which the mentioned components do not functionally depend). In a lamp–battery system, 'the lamp does not turn on' yields $battery, lamp$, while 'the battery looks fine' yields $lamp$.

[4] All icons from flaticon.com, authors: Freepik, Nur syifa fauziah, Graphixpoint.

[5] We tested gpt-4.1 and gpt-5.1. More models will be studied in the future, for now model variety is secondary since our focus here is the evaluation environment itself.

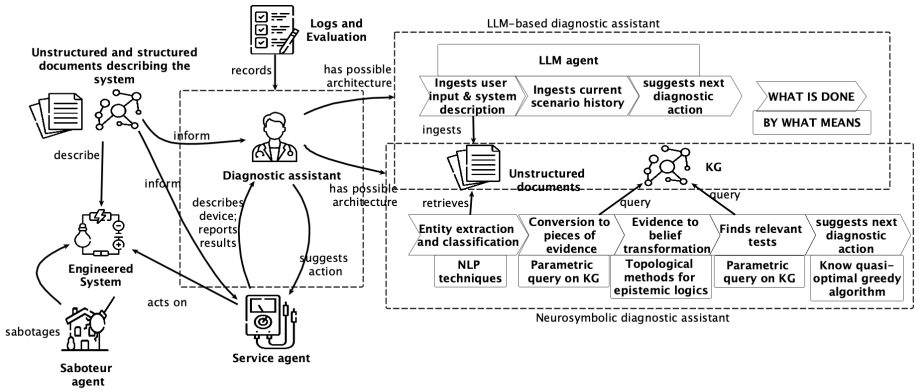

Fig. 2. Environmental setup of the diagnostic assistant with an actual physical system (left); and functional architectures of two diagnostic assistants (right).

These sets are taken as pieces of evidence and combined into beliefs over root-cause candidates via topological evidence models [1][6]. The KG is queried again to build a map from the possible problems for candidate components to the corresponding tests. Tests are then ranked by a greedy information-gain heuristic [3], and the top test is suggested as the next diagnostic action. The role of the KG in the neuro-symbolic assistant is twofold: (i) it encodes the functional knowledge required to expand the extracted entities in a multi-hop manner, and (ii) it captures knowledge about possible failure modes and associated tests required to construct a diagnostic procedure.

3 Evaluation by Scenarios and Preliminary Insights

We prepared diagnostic scenarios with toy physical systems covering core diagnosis challenges, including resource constraints, misleading evidence, and unforeseen component interactions.

Preliminary small-scale experiments suggest complementary strengths and weaknesses of the two architectures.

The LLM-based assistant tends to propose reasonable test sequences, tolerates minor contradictions or missing evidence, and can use user suggestions beyond simple action outcomes. However, it sometimes misunderstands component functions, proposes plausible but contextually nonsensical actions, and may use inefficient testing strategies.

The neuro-symbolic assistant instead reliably constructs near-optimal procedures when relevant components, failure modes, and diagnostic actions are

[6] Implemented with the `evidence_belief_models` Python package (Pinto Prieto, 2026): `https://github.com/dpp-research/evidence_belief_models` , plus custom implementations of algorithms from [6].

represented in the KG. Apart from its reliance on entity extraction, it is also less sensitive to documentation phrasing. Its main limitation is domain coverage: if a failure mode, interaction, or test is absent from the knowledge base, it cannot reason about it.

4 Demo Setup

We will have multiple physical electronic systems on site, and failures can be injected in a variety of ways. Demo attendees will be able to interact with the systems, to diagnose them, and to make use of the diagnostic assistants (either via command line or voice interface), and see the potential of our setup for evaluation and comparison of diagnostic assistants. This interaction will consist of supplying an initial description of the system state to the assistant, and then following (or refusing to follow) the suggestions of the assistant and reporting their results.

The codebase of the diagnostic assistant and a video showcasing its use in a diagnostic scenario are available at `https://github.com/kataph/Diagnostic-Assistant-Demo` (the video is also available at `https://youtu.be/beh18vLPm30`).

5 Conclusion and Future Work

Neuro-symbolic approaches that combine KGs and LLMs offer a new opportunity to address the challenges of diagnostics of engineered systems. Our work aims to provide an accessible environment to test and explore different neuro-symbolic architectures for this problem domain.

Based on this, our preliminary comparison shows that LLM-based and neuro-symbolic assistants have complementary strengths (flexibility and robustness versus fast determination of a good testing procedure). Thus, we plan further work in the direction of merging the two approaches.

This comparison also assessed the evaluation environment itself. We identified several challenges and possible improvements. Currently the employed engineered systems are simple and limited in variety. We plan to generate a broader class of systems by aggregating simple modules, aiming to balance simplicity with generality for diagnostic applications. We also lack guarantees on scenario completeness. To address this, we plan to develop quality metrics for diagnostic systems and design scenarios that stress-test specific quality dimensions. Programmatic scenario execution requires simulating the effects of diagnostic actions on engineered systems. We initially used LLMs for this, but later switched to traditional engineered system simulation, due to better correctness and stability, albeit with higher development cost. Finally, evaluating qualitative metrics is expensive when many long scenario logs require manual review. For example, studying reasoning patterns of LLM-based assistants required extensive inspection of generated text. We plan to investigate semi-automatic methods for this task.

References

1. Baltag, A., Bezhanishvili, N., Özgün, A., Smets, S.: Justified belief, knowledge, and the topology of evidence. Synthese (2022)
2. Dhillon, B.S.: Engineering maintenance: a modern approach. CRC press (2002)
3. Johnson, R.A.: An information theory approach to diagnosis. In: Proceedings of The Sixth National Symposium on Reliability & Quality Control in Electronics, Washington, D.C., January 11-13 (1960)
4. Naghdipour, A., Kruit, B., Chen, J., Schlobach, S.: Scalable knowledge representation for fault diagnosis of cyber physical systems: a systematic literature review. Semantic Web Journal (2025)
5. Pour, A.N., Kruit, B., Chen, J., Webers, G., Kruizinga, P., Schlobach, S.: Knowledge representation and engineering for smart diagnosis of cyber physical systems. In: ISWC 2024 Posters, Demos and Industry Tracks (2024)
6. Prieto, D.P.: Combining Uncertain Evidence, Ph.D. Thesis (2024)
7. Soleimani, M., Campean, F., Neagu, D.: Diagnostics and prognostics for complex systems: A review of methods and challenges. Quality and Reliability Engineering International **37**(8), 3746–3778 (2021)
8. Yuhan, L., Yuan, Z., Yufei, L., Zhen, X., Yixin, H.: Intelligent fault diagnosis for CNC through the integration of large language models and domain knowledge graphs. Engineering (2025)