Computers in Industry 169 (2025) 104289

Contents lists available at ScienceDirect

Computers in Industry

journal homepage: www.sciencedirect.com/journal/computers-in-industry

# An ontology-based retrieval augmented generation procedure for a voice-controlled maintenance assistant☆

Heiner Ludwig *, Thorsten Schmidt, Mathias Kühn

*Faculty of Mechanical Science and Engineering, Institute of Material Handling and Industrial Engineering, TUD Dresden University of Technology, Dresden, 01062, Germany*

ARTICLE INFO

*Keywords:*
Retrieval augmented generation (RAG)
Large Language Model (LLM)
Web Ontology Language (OWL)
Maintenance assistant

ABSTRACT

This paper presents a novel approach to support complex maintenance procedures through a dialogue-driven digital assistant using an ontology-based retrieval augmented generation method. The core of the proposed system relies on the strong formalisation capabilities of the graph-based Web Ontology Language (OWL), combined with various retrieval algorithms and different Large Language Models (LLMs) to determine the most useful context for answering user queries. To do this, we use the popular principle of Retrieval Augmented Generation (RAG). Graph traversal enriches the contextual knowledge, enabling more accurate and context-aware responses. An evaluation using an OWL example ontology and an extensive Q&A dataset demonstrates the improved retrieval quality achieved by combining classical and vector-based semantic matching methods. The community-driven analysis of generation quality illustrates the usability of an OWL-based assistant for maintenance procedures on the basis of contexts and LLMs of varying configurations.

## 1. Introduction

As industrial machinery becomes increasingly complex, the need for expert maintenance personnel has grown significantly. This has led to high costs and slow response times when addressing maintenance issues (Silvestri et al., 2020). One promising solution is the use of voice-based assistance, which can leverage the advantages of speech-based interaction, such as hands-free operation and the potential for integration with augmented reality (AR) or virtual reality (VR) technologies (Ludwig et al., 2023).

To enable effective voice-assisted maintenance, a flexible and customisable semantic database is required. There has been substantial research in the area of industrial ontologies, with initiatives such as the Industrial Ontology Foundry (IOF) providing a foundation for modelling the industrial domain, including maintenance-related concepts and processes (Kulvatunyou et al., 2022).

Additionally, the rapid advancements in Large Language Models (LLMs) have opened up new possibilities for natural language processing and natural language generation in industrial applications. By combining the strengths of semantically structured knowledge bases with the powerful language understanding and generation capabilities of LLMs, we present a flexible digital assistant based on a novel Retrieval Augmented Generation (RAG) approach. RAG is a paradigm in LLMs and enhances generative tasks by first retrieving relevant information from external sources before answering questions or generating text. It exploits the complementary strengths of LLMs and semantic graph structures to provide a versatile, transparent and customisable language-assisted maintenance solution (Gao et al., 2023). The proposed system can be adapted to different LLMs and user interfaces through various (retrieval) parameter configurations, benefitting from the formal grammar and reasoning capabilities of the underlying ontological knowledge base.

*Graph RAG* is currently a much-noticed field of research and is attracting attention in various scientific fields as well as in industry (Corporation, 2024; Edge et al., 2024; Matsumoto et al., 2024). Web Ontology Language (OWL) extends the expressive power of simple knowledge graphs by typing entities and relations and allows them to be specified using predefined properties (such as hierarchies, symmetrical relations or cardinality rules) (W3C, 2004).

The paper is structured as follows: In Section 2, we provide an overview of the application of digital (voice) assistants for maintenance purposes and current developments in the field of RAG. In addition, we briefly describe the structure and advantages of using OWL as the semantic data structure. Section 3 presents a novel approach for RAG based on OWL that can be adapted to technical conditions and applications by parameter customisation. In Section 4, we evaluate the approach with respect to its performance in different configurations.

☆ The research is financially supported by the german Industrielle Gemeinschaftsforschung (IGF) under project number 22927 BG.

* Corresponding author.
*E-mail address:* heiner.ludwig@tu-dresden.de (H. Ludwig).

https://doi.org/10.1016/j.compind.2025.104289
Received 7 December 2024; Received in revised form 15 February 2025; Accepted 23 March 2025
Available online 3 April 2025

Finally, we discuss the results and give an outlook on further potentials in Sections 5 and 6.

## 2. Background

### 2.1. Voice assistants in the maintenance area

Maintenance describes the planned or unplanned execution of actions to restore the functionality of operating equipment (European Committee for Standardization (CEN), 2018). A fast, complete service is essential to minimise downtime and additional costs and thus keep companies competitive. Maintenance processes in manufacturing are complex and information-intensive, so the availability of relevant information plays an essential role in providing the best possible support for maintenance personnel (Ting Zheng and Glock, 2024). Various studies have shown that digital assistants are suitable for supporting employees in a maintenance context (Fleiner et al., 2021; Serras et al., 2020; Wellsandt et al., 2023, 2020). Voice assistants in particular offer advantages such as hands-free and eyes-free operation, high interaction speed and intuitive usability (Ludwig et al., 2023).

### 2.2. OWL in the maintenance area

The formalisation and consolidation of knowledge and terminology for the maintenance sector has been ongoing for decades. Numerous studies and initiatives have produced ontologies for mapping industrial maintenance processes (Karray et al., 2019; Woods et al., 2023a,b; Montero Jiménez et al., 2023; Kulvatunyou et al., 2022; Polenghi et al., 2022). OWL, as a formalisation language specified by the *W3C* (W3C, 2004), serves as a uniform basis language for modelling directed semantic graphs. It extends the less expressive *Resource Description Framework Schema* (RDFS). This enables the creation of formalised, generalised and reusable knowledge bases (“top-level ontologies”), which serve as the basis for modelling specific (maintenance) use cases (“lower-level ontologies”). Entities and relations can be labelled in different languages so that OWL is suitable for multilingual usage. An OWL ontology $O$ provides the following entities, relationships and axioms:

$$O = (TBox, ABox)$$

*where*

$$TBox = (C, R_O, R_D, A)$$
$$ABox = (I, R_{O,A}, R_{D,A})$$

The *Terminological Box* (TBox) contains domain concepts and defines description rules. $C$ denotes a collection of classes representing entity concepts (e.g. *Machine*, *Maintenance Routine*, *Maintenance Step*). $R_O$ are object properties that define relationships between entities (e.g. Maintenance Routine *contains step* Maintenance Step). $R_D$ are relations that assign attributes to entities (e.g. Maintenance Step *has safety instruction* `<string>`). $A$ are axioms for establishing constraints that can be evaluated using reasoning algorithms (e.g. a Maintenance Routine contains *at least one* Maintenance Step). $A$ does not contain the axioms of the ABox. One or more labels (possibly in different languages) can be added to each element of the TBox to act as synonyms.

The *Assertional Box* (ABox) contains concrete expressions of the concepts of $C$, $R_O$ and $R_D$ of the TBox using assertion axioms. Individuals $I$ have at least one type $C$ (e.g. *Renew filter* is of type Maintenance Routine, *Replace primary filter cell* is of type Maintenance Step) and can also be described by (several) labels. Object Property Assertions $R_{O,A}$ describe relations between two Individuals (e.g. Renew filter *contains step* Replace primary filter cell). Data Property Assertions $R_{D,A}$ add attributes to Individuals (e.g. Renew filter *has safety instruction “Wear protective clothing”*).

As can be seen from the examples given, the OWL entities and their attributes and relations are structured in so-called subject–predicate–object triples, whereby each component can be a single word, a group of words or several sentences.

### 2.3. Retrieval Augmented Generation (RAG)

RAG is a young, growing field of research that provides an alternative to LLM adaptation through fine-tuning and includes various retrieval-based and generation-based combination approaches (Gao et al., 2023). A retrieval mechanism searches an external data source for relevant information based on the user input, which is then injected as context into the prompt for a generative language model. This reduces hallucinations of the LLM and increases the transparency of contextual knowledge, as the data source is visible. In contrast to fine-tuning, small datasets can be taken into account, which is essential in the specialised maintenance context. While various methods for incorporating text documents, relational databases and knowledge graphs exist (Wang et al., 2023; Sen et al., 2023; Siddharth and Luo, 2024), the use of OWL structured graphs represents a research gap. We present a RAG procedure that benefits from the higher degree of formalisation of OWL in both the retrieval and generation phases. The LLM relies on the stored OWL ontology as a context for the interaction with the maintenance employee. This enables the worker to retrieve the relevant knowledge as quickly as possible without needing the technical understanding to query OWL.

We compose the prompt according to Gao et al. (2023) (see Fig. 1 “Prompt Generation”), consisting of the following four text sections:

1. *User input:* Voice input converted from speech to text.
2. *Static context statement:* A static command to instruct the LLM (how) to use the following OWL context. Only appears if the extracted OWL context is not empty.
3. *Extracted OWL context:* The context for interacting with the user input, formulated and arranged by the retrieval process using the OWL ontology. The maximum permissible context length varies depending on the LLM used.
4. *Instruction for the answer style (optional):* Domain-, output device and/or user-dependent, e.g. to provide a particularly detailed or stepwise response.

## 3. Context creation

Context generation takes place in three steps: First, we extract and rank matching OWL entities based on user input using different comparison methods, collecting relevant Individuals. Next, outgoing Object Property Assertions $R_{O,A}$ include neighbouring Individuals. Finally, all results are ordered and their information is formulated in natural language (see Fig. 2). The methods of each step have parameters to adapt to different LLMs and applications. To determine the relevance of OWL entities, we use the following matching methods:

1. *Exact matching* $sim_{ex}(s_1, s_2)$ checks the identity of strings $s_1$ and $s_2$. We tokenize, lemmatise and remove stop words from search strings and OWL labels by using the *spaCy* framework (GmbH, 2023). For multi-word strings, word order is ignored. This method is suitable for cases with a precise, well-defined vocabulary (including extensive synonyms definitions) and for users with a high level of expertise.
2. *Similarity-based matching* $sim_{vec}(s_1, s_2)$ uses sentence embeddings of $s_1$ and $s_2$ for matching. We use a pre-trained multilingual *SBERT* model (Reimers and Gurevych, 2020) to generate embeddings for OWL entity labels. These high-dimensional vectors capture the meaning of sentences for comparison in vector space. Cosine similarity $\cos(\theta)$ measures similarity, with values closer to 1 indicating higher similarity. OWL entity embeddings are derived from subject–predicate–object triples (e.g. “cover flap” “has screw type” “M10 hexagon head screw”), creating new embeddings per label. Semantic embeddings offer flexibility and natural language understanding, but run the risk of being inaccurate due to biased training data, leading to false positives. This method benefits inexperienced users and incomplete OWL ontologies.

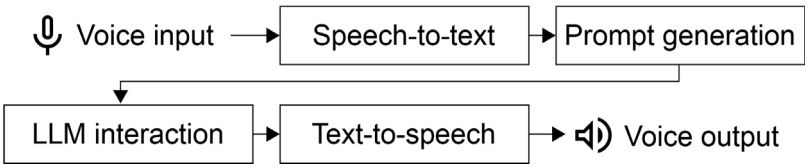


**Fig. 1.** Procedure for voice interaction with the maintenance assistant.

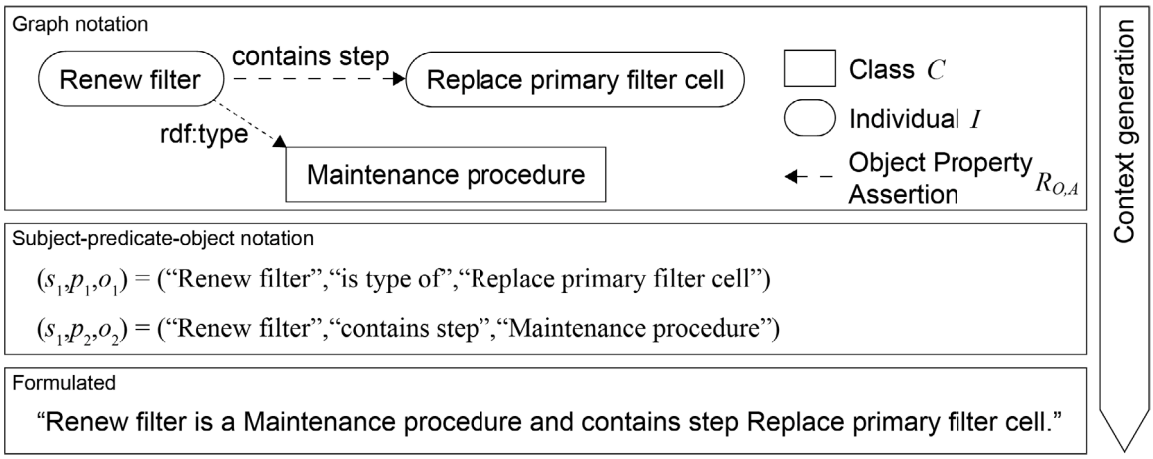


**Fig. 2.** Various notations for describing maintenance information.

3. *Phonetic matching* $sim_{phon}(s_1, s_2)$ builds on $sim_{ex}(s_1, s_2)$ to handle speech-to-text errors (see Step 2, Fig. 1). Phonetic algorithms convert syllables into codes based on pronunciation to identify similar-sounding words with different spellings. We use the *Cologne Phonetic Algorithm*, adapted from *soundx*, for German language (Jacobs, 1982). This method helps to analyse imprecise speech-to-text conversion, for example in noisy (industrial) environments. As this paper focuses on the OWL-based RAG method and not on speech-to-text accuracy, we list this method but do not evaluate it.

### 3.1. Retrieve OWL entry points

To extract named entities from user input $q$ in OWL, we apply matching techniques to identify relevant entities, relations and attributes. In addition, the OWL hierarchy structure infers further candidate individuals $I$ (see Algorithm 1). The hierarchy and typing in OWL allow implicitly matched individuals to be included. When a class label matches the query, all individuals of that class or subclass become startpoints. Hierarchical relationships are formalised by *rdf:type* and *rdfs:subClassOf*, allowing more comprehensive query handling. The modular retrieval system arranges matching procedures for different OWL entities sequentially. To optimise retrieval, we test three strategies that combine explicit precision $sim_{ex}$ with vector-based flexibility $sim_{vec}$ (see Section 4.1).

### 3.2. Context expansion

We use the extracted Individuals $I$ from 3.1 as starting points and use "hopping" to extend the context by walking through the ontology. The hopping depth $H_{depth}$ indicates how many Object Property Assertions are run through sequentially ("multi-hopping", see Fig. 3). Starting from the respective starting points, this process visits neighbouring Individuals step by step using linked Object Property Assertions $R_{O,A}$ in a breadth-first search manner. This is done until either: (a) the maximum number of hops $H_{depth}$ is reached, or (b) all linked entities have been visited. The edges can optionally be weighted on the basis

**Algorithm 1** Search relevant Individuals in OWL Ontology

**Description:** The structure of the algorithm illustrates the search mechanism. The order and algorithms of the matching procedures are flexible.

**Require:** $O$: OWL ontology, $q$: Search string, $sim_x(s_1, s_2)$: Matching procedure for entity labels with $x \in \{ex, vec, phon\}$ (depending on the retrieval configuration).

**Ensure:** $result_I$: Amount of Individuals considered relevant to the search term.

$result_I \leftarrow \emptyset$
**for** $I \in O.Individuals$ **do**
  # handle Individual retrieval
  **if** $\exists l \in I.labels : sim_x(q, l)$ **then**
    $result_I \leftarrow result_I \cup \{I\}$
  **end if**
  # handle hierarchical Individual retrieval
  **for** $C \in I.types \cup C.parentClasses$ **do**
    **if** $\exists l \in C.labels : sim_x(q, l)$ **then**
      $result_I \leftarrow result_I \cup \{I\}$
    **end if**
  **end for**
  # handle Data Property Assertion retrieval
  **for** $p \in I.dataPropertyAssertions$ **do**
    **if** $sim_x(q, p.labels, p.value)$ **then**
      $result_I \leftarrow result_I \cup \{I\}$
    **end if**
  **end for**
  # handle Object Property Assertion retrieval
  **for** $p \in I.objectPropertyAssertions$ **do**
    **if** $sim_x(q, p.labels, p.object.labels)$ **then**
      $result_I \leftarrow result_I \cup \{I\}$
    **end if**
  **end for**
**end for**
**return** $result_I$

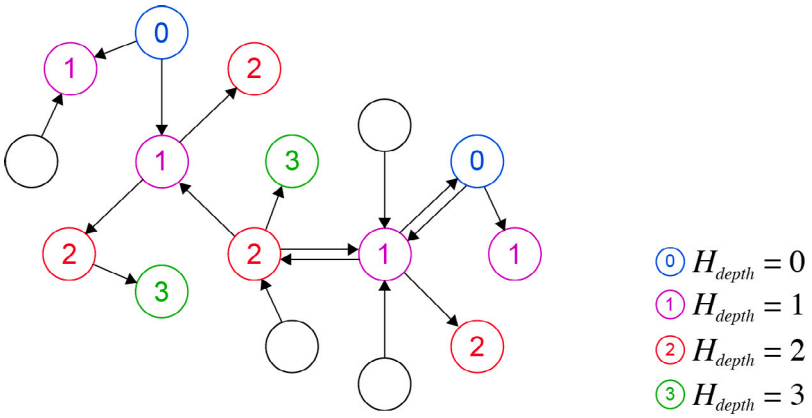


**Fig. 3.** Example of graph hopping for $H_{depth} \in \{0, 1, 2, 3\}$, taking into account the direction of the object property assertions. $H_{depth} = 0$ corresponds to the starting points.

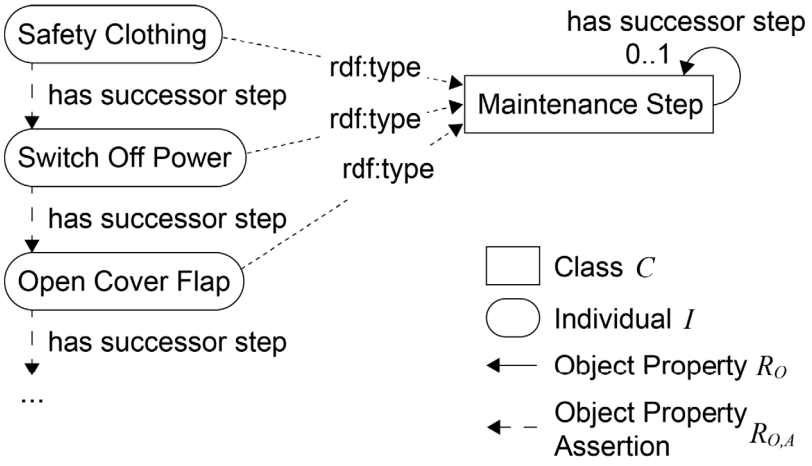


**Fig. 4.** Sequential OWL modelling of maintenance steps.

of semantic similarity to the user input. By defining minimum cos similarities $min_\theta cos(\theta)$ between the sentence embeddings of $R_{O,A}$ and user input, unsuitable edges are thus excluded.

### 3.3. Context arrangement and formulation

After extracting the relevant Individuals, their relationships and the ontological context, we order and formulate the context. We extract labels and values of $R_{O,A}$ and $R_{D,A}$ related to Individuals and concatenate them into sentences (see Fig. 2). The high density of information ensures that LLMs with smaller context sizes have access to the maximum amount of knowledge. To achieve this, we bundle $R_{O,A}$ by type $R_O$ and $R_{D,A}$ by type $R_D$, arranging different object values with commas. In addition, we list all relevant labels and types of individuals at the beginning to clarify synonyms and typification.

Liu et al. (2024) highlight the role of position and order in prompts for response generation. In maintenance, sequential steps are modelled in the ontology using Object Property Assertions $R_{O,A}$ to represent process chains (see Fig. 4). Less powerful LLMs infer step order from text arrangement rather than predecessor/successor relationships. To maintain order in context, if Individuals $I$ of type $C$ are linked by Object Property Assertions $R_{O,A}$ of the same type $R_O$, their relationships appear in the order given. If $R_{O,A}$ form a cycle, we start with a random Individual.

## 4. Evaluation

Gao et al. (2023) classify RAG evaluation into *Retrieval Quality*, which evaluates context extracted via OWL, and *Generation Quality*, which analyses LLM-generated responses. We first evaluate both metrics separately in different configurations and then compare the best configuration with state-of-the-art RAG systems.

**Table 1**
Structure of the example OWL filter maintenance ontology.

| Entity type | $\|C\|$ | $\|I\|$ | $\|R_D\|$ | $\|R_{D,A}\|$ | $\|R_O\|$ | $\|R_{O,A}\|$ | $\|A\|$ |
|---|---|---|---|---|---|---|---|
| Quantity | 93 | 222 | 42 | 315 | 36 | 376 | 244 |

Although the methodology can be generalised, we evaluate performance in the maintenance domain. We use a manually created OWL ontology describing the components and maintenance processes of a filter system (see Table 1). The ontology follows the *Ontology 101* method (Noy and Mcguinness, 2001), based on user manuals and staff experience. To classify maintenance activities, we use parts of the *IOF Maintenance Activity Ontology* (Woods et al., 2023b). We carried out the evaluation in English.

### 4.1. Retrieval quality

Retrieval quality is evaluated step by step using precision and recall. Precision measures the exclusion of irrelevant information, while recall measures the completeness of relevant OWL entities. The $F_\beta$ *score* with $\beta \in \{1, 2\}$ combines both metrics (see Eq. (1)). $F_1$ gives equal weight to precision and recall by taking their harmonic mean. $F_2$ emphasises recall, making it suitable for evaluating retrieval in LLMs that handle

**Table 2**
Retrieval quality questions of the evaluation Q&A data set.

| Content | Amount of questions by type | | | | | | |
|---|---|---|---|---|---|---|---|
| | What | Where | Why | Who | When | How | How many |
| Product | 568 | 65 | 7 | 8 | 8 | 92 | 6 |
| Process | 338 | 15 | 23 | 4 | 0 | 102 | 3 |
| Resource | 87 | 4 | 0 | 3 | 0 | 14 | 1 |

false positives more effectively.

$$F_\beta = \frac{(1+\beta^2)\cdot Precision \cdot Recall}{\beta^2 \cdot Precision + Recall} \quad (1)$$

To measure the retrieval quality, we semi-automatically created a Q&A dataset with questions about the content of the maintenance ontology. The dataset consists of 1348 questions, each expecting at least one relevant Individual $I$. We generated OWL triples for all Individuals, formulated questions using *GPT-4o*, and manually validated them. We project the widespread *product–process–resource* information modelling pattern onto the maintenance domain to form the content of the system requests (Cutting-Decelle et al., 2007). Questions about products relate to the maintenance object and its components and spare parts. Processes include maintenance routines and steps, handling and safety instructions. Consumables, tools and other maintenance equipment belong to resources. As question types, we differentiate between "what", "where", "why", "who", "when", "how" and "how many" (see Table 2). The Q&A data set and the corresponding OWL ontology of the filter system are available on Git.[1]

In the first step, we compare exact and vector-based matching procedures $sim_{ex}$ and $sim_{vec}$ for OWL component types. We evaluate precision and recall for each question and compute *Average Precision (AvgPrec)* and *Average Recall (AvgRec)*, then derive $F_1$ and $F_2$. If no matches are found, precision is set to 0. For vector-based matching $sim_{vec}$, we increment the minimum required cosine similarity $min_\theta cos(\theta)$ between the query embedding $q$ and each OWL component by 0.1. Table 3 shows that exact matching $sim_{ex}$, processing $C$ and $I$ labels (including OWL hierarchy), gives the best $F_1$ and $F_2$ results. For $cos(\theta) \geq 0.7$, vector-based methods do not find any OWL candidates. While $sim_{vec}$ results lag behind explicit methods, the evaluation confirms the viability of sentence embeddings from subject–predicate–object chains, despite deviations from natural language (e.g. missing articles or fillers).

In the second step, we combine all matching procedures in all configurations from the first step. We distinguish between three strategies for merging the respective matches:

1. *First match*: ends the further execution of matching procedures as soon as a matching procedure finds more than 0 matches.
2. *Union*: totals the results of all matching procedures and removes all duplicates. The order of the matching procedures is irrelevant.
3. *Majority*: selects all matches with the most occurrences in all matching procedures. This favours matches found by multiple matching procedures and includes all matches found if there is no overlap in the results (i.e. all matches only occur once). The order of the matching procedures is irrelevant.

Table 4 shows the top 3 combinations per strategy, measured by $F_1$ and $F_2$ scores. The *Majority* strategy significantly improves both scores using multiple matching configurations. The best configurations rely on at least one explicit comparison procedure, which also performs best in single retrieval evaluations.

The *First match* strategy offers no advantages over the use of single matching procedures. The explicit methods $sim_{ex}$ are dominant, the combination with other methods does not improve $F_1$ and $F_2$. $sim_{vec}(R_{O,A}, q)$ with $min_\theta cos(\theta) = 0.3$ in the second procedure of both top 3 rankings has minimal impact due to negligible matches (see Table 3), so $sim_{ex}(C \cup I, q)$ remains responsible for the majority of results.

The *Union* strategy gives slight precision and recall gains, improving $F_1$ and $F_2$. In particular, the combination of the explicit $sim_{ex}(C \cup I, q)$ method and the vector-based $sim_{vec}(R_{D,A}, q)$ method with $min_\theta cos(\theta) = 0.2$ proved to be the decisive basis. $sim_{vec}(R_{O,A}, q)$ with $min_\theta cos(\theta) = 0.3$ is also negligible here.

The *Majority* strategy performs best, improving precision and recall over the single methods. Overlapping results seems to compensate for uncertainties. The best $F_1$ and $F_2$ configurations differ, with $sim_{ex}(C \cup I, q)$ and $sim_{vec}(R_{D,A}, q)$ with $min_\theta cos(\theta) = 0.2$ in both top 3 rankings. The top 3 $F_1$ scores highlight the strong influence of explicit matching and the broader coverage of $\text{sim}_{\text{vec}}(R_{D,A}, q)$.

### 4.2. Generation quality

Generation Quality assesses the response quality of a selected LLM. Small, locally-deployable LLMs minimise third-party dependencies, potential data misuse and hardware requirements, while larger models are better suited to more extensive query handling (Chiang et al., 2024). We evaluate three models (Table 5): *openAI's GPT-4o*, the MMLU leader (Sept 2024); *Meta's Llama 3.1 8B*, a free model with low hardware requirements; and *Alphabet's Gemma 2B*, the top-ranked 2B model for mobile use.

We compose multiple contexts according to the methodology presented in Section 3.2 for 14 randomly selected questions from the Q&A dataset. The dataset is uploaded to Git.[2] To imitate a voice-based dialogue, answers can also be output as synthesised text-to-speech. We add an instruction to the prompt to give the shortest, most accurate answers to reduce cognitive load and speed up interaction. Inspired by *Chatbot Arena*,[3] we provide an online tool for pairwise comparison of LLM generation results with different hop depths (see Fig. 5) (Chiang et al., 2024). Users select the better answer, a tie, or mark both as unsatisfactory, while model and hop configuration remain anonymous. Different models compete in the same hop level (i.e. Single Round-Robin tournament). To compare hop depth effects, we add duels using the same LLM with varying contexts from $H_{depth} \in \{0, 1, 2\}$. Contexts that remain unchanged with increasing hop depth (due to no further outgoing $R_{D,O}$ for given $I$) are included once.

To evaluate the model's performance in duels, we calculate the *Elo* score. The updated Elo score $R_{\text{new}}$ for an LLM is calculated as:

$$R_{\text{new}} = R_{\text{old}} + K \cdot (S - E)$$

$$S = \begin{cases} 1 & \text{LLM gives the better answer} \\ 0.5 & \text{both LLMs provide adequate answers} \\ 0 & \text{LLM provides inadequate answer} \\ 0 & \text{both LLMs provide inadequate answers} \end{cases}$$

where $E$ is the expected score, calculated by the formula:

$$E = \frac{1}{1 + 10^{\frac{R_{\text{opponent}} - R_{\text{old}}}{400}}}$$

We set the constant $K = 32$ and the initial Elo value $R = 1000$ for each model. The higher the Elo score, the better the generation quality of the LLM compared to other models. In addition, we examine the answers of the models with regard to *Answer faithfulness* and *Answer relevance* (cf. Gao et al., 2023).

We analyse 960 duels of different LLMs and $H_{depth}$ evaluated by six test persons. Table 6 shows the proportion of decided and undecided duels between different model and $H_{depth}$ configurations. For example, GPT-4o with $H_{depth} = 1$ won 35%, lost 18% and drew 47% against

[1] https://tlscm.mw.tu-dresden.de/scm/repo/git/owl-rag-for-maintenance.

[2] https://tlscm.mw.tu-dresden.de/scm/repo/git/owl-rag-for-maintenance.

[3] https://lmarena.ai.

**Table 3**
Evaluation results of single matching procedures with a total of more than 0 matches targeting different OWL entities.

| $\min_\theta \cos(\theta)$ | AvgPrec | AvgRec | Avg amount of matches > 1 | Median amount of matches | $F_1$ score | $F_2$ score |
|---|---|---|---|---|---|---|
| $\text{sim}_{\text{vec}}(C \cup I, q)$ | | | | | | |
| 0.1 | 0.006 | **0.995** | 206.091 | 217 | 0.012 | 0.029 |
| 0.2 | 0.014 | 0.951 | 127.415 | 133 | 0.027 | 0.065 |
| 0.3 | 0.056 | 0.856 | 40.267 | 26 | 0.106 | 0.223 |
| 0.4 | 0.181 | 0.621 | 7.917 | 5 | 0.281 | 0.418 |
| 0.5 | 0.199 | 0.311 | 2.479 | 1 | 0.243 | 0.28 |
| 0.6 | 0.068 | 0.088 | 1.574 | 0 | 0.077 | 0.083 |
| $\text{sim}_{\text{vec}}(R_{D,A}, q)$ | | | | | | |
| 0.1 | 0.065 | 0.866 | 24.676 | 24 | 0.121 | 0.251 |
| 0.2 | 0.281 | 0.425 | 2.252 | 1 | 0.338 | 0.385 |
| 0.3 | 0.016 | 0.016 | 1 | 0 | 0.016 | 0.016 |
| $\text{sim}_{\text{vec}}(R_{O,A}, q)$ | | | | | | |
| 0.1 | 0.05 | 0.506 | 16.293 | 15 | 0.09 | 0.178 |
| 0.2 | 0.1 | 0.155 | 2.422 | 0 | 0.121 | 0.139 |
| 0.3 | 0.001 | 0 | 1 | 0 | 0 | 0 |
| $\text{sim}_{\text{ex}}(C \cup I, q)$ | | | | | | |
| – | **0.375** | 0.563 | 8.172 | 1 | **0.45** | **0.511** |
| $\text{sim}_{\text{ex}}(R_{D,A}, q)$ | | | | | | |
| – | 0.346 | 0.467 | 1.759 | 1 | 0.397 | 0.435 |

**Table 4**
Top 3 evaluation results of multiple matching procedures using different merging strategies by $F_1$ and $F_2$ score.

| | Configuration[a] | | | | | Evaluation results | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | $\text{sim}_{\text{ex}}(C \cup I, q)$ | $\text{sim}_{\text{ex}}(R_{D,A}, q)$ | $\text{sim}_{\text{vec}}(C \cup I, q)$ | $\text{sim}_{\text{vec}}(R_{D,A}, q)$ | $\text{sim}_{\text{vec}}(R_{O,A}, q)$ | AvgPrec | AvgRec | Avg amount of matches > 1 | Median mount of matches | $F_1$ score | $F_2$ score |
| | First match | | | | | | | | | | |
| Top 3 $F_1$ | $x_1$ | ~ | ~ | ~ | ~ | 0.375 | 0.563 | 6.911 | 1 | 0.45 | 0.511 |
| | $x_2$ | ~ | ~ | ~ | $0.3_1$ | 0.354 | 0.52 | 4.431 | 1 | 0.421 | 0.475 |
| | ~ | $x_1$ | ~ | ~ | ~ | 0.346 | 0.467 | 1.19 | 1 | 0.397 | 0.435 |
| Top 3 $F_2$ | $x_1$ | ~ | ~ | ~ | ~ | 0.375 | 0.563 | 6.911 | 1 | 0.45 | 0.511 |
| | $x_2$ | ~ | ~ | ~ | $0.3_1$ | 0.354 | 0.52 | 4.431 | 1 | 0.421 | 0.475 |
| | ~ | $x_1$ | ~ | ~ | ~ | 0.346 | 0.467 | 1.19 | 1 | 0.397 | 0.435 |
| | Union | | | | | | | | | | |
| Top 3 $F_1$ | x | | 0.6 | 0.2 | | 0.405 | 0.671 | 2.215 | 2 | 0.505 | 0.594 |
| | x | | 0.6 | 0.2 | 0.3 | 0.405 | 0.671 | 2.215 | 2 | 0.505 | 0.594 |
| | x | | | 0.2 | | 0.38 | 0.751 | 7.881 | 2 | 0.505 | 0.628 |
| Top 3 $F_2$ | x | | | 0.2 | | 0.38 | 0.751 | 7.881 | 2 | 0.505 | 0.628 |
| | x | | | 0.2 | 0.3 | 0.38 | 0.751 | 7.881 | 2 | 0.505 | 0.628 |
| | x | | 0.6 | 0.2 | | 0.375 | 0.751 | 7.915 | 3 | 0.501 | 0.626 |
| | Majority | | | | | | | | | | |
| Top 3 $F_1$ | x | x | 0.3 | 0.2 | 0.3 | **0.542** | 0.702 | 6.277 | 1 | **0.611** | 0.663 |
| | x | x | 0.3 | 0.2 | | **0.542** | 0.701 | 6.276 | 1 | 0.611 | 0.662 |
| | x | | 0.3 | 0.2 | 0.3 | 0.505 | 0.768 | 6.984 | 2 | 0.61 | 0.696 |
| Top 3 $F_2$ | x | | 0.2 | 0.2 | 0.3 | 0.496 | **0.786** | 16.258 | 2 | 0.608 | **0.704** |
| | x | | 0.2 | 0.2 | | 0.496 | 0.786 | 16.257 | 2 | 0.608 | 0.704 |
| | x | | 0.3 | 0.2 | 0.3 | 0.505 | 0.768 | 6.984 | 2 | 0.61 | 0.696 |

[a] Cell contents for matching procedures $sim_{vec}$ corresponds to $min_\theta cos(\theta)$. Empty cells mean that the respective matching procedure is not included. For the *First match* strategy, the subscripts indicate the order of the matching procedures. ~ marks procedures whose usage, position in the processing order and $min_\theta cos(\theta)$ are irrelevant for the evaluation results.

**Table 5**
LLMs used for generation quality evaluation.

| LLM | MMLU | Parameters | Context size |
|---|---|---|---|
| GPT-4o | 88.7 | 200B–1.8T[a] | 128K |
| Llama 3.1 8B | 66.7 | 8B | 128K |
| Gemma 2B instruct | 42.3 | 2B | 8192 |

[a] Based on estimates from different sources.

GPT-4o with $H_{depth} = 0$ (see first column, second row). Only eight answers were rated as poor overall. All models extract simple answers correctly (e.g. "Who is authorised to inspect the motor protection relays?"), regardless of hop level. For complex answers (e.g. "What steps are given for inspecting the fan motor?"), the model outputs differed more widely. The test subjects found GPT-4o to be the most faithful to the given context for answering the question, adding very little of its own information to the answer. Gemma 2B enrich the answer with own knowledge beyond the context information, whereby we did not detect any hallucinations in the test cases. Llama 3.1 only added "own" knowledge for small (incomplete) contexts. We evaluate the cross-model duels with the same hop level (see Fig. 6) and intra-model duels with different hop levels (see Fig. 7) separately. GPT-4o and Llama 3.1 achieved almost identical Elo scores in the cross-model duels. Gemma 2B performs worse at higher hop levels and occasionally includes irrelevant details. Overall, $H_{depth} > 0$ improves performance of

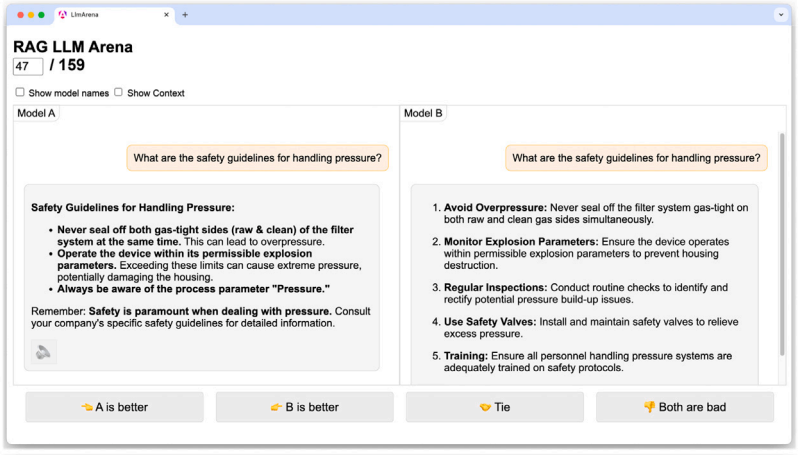


**Fig. 5.** Web-based RAG LLM Arena for anonymous, paired comparison of the generation quality of different LLMs.

**Table 6**
Proportion of duels won and tied in all evaluated LLM and $H_{depth}$ configurations.

| LLM | B | GPT-4o | | | Llama 3.1 8B | | | Gemma 2B instruct | | |
|---|---|---|---|---|---|---|---|---|---|---|
| A | $H_{depth}$ | 0 | 1 | 2 | 0 | 1 | 2 | 0 | 1 | 2 |
| GPT-4o | 0 | | 18\|47\|**35** | | 19\|51\|**30** | | | 27\|30\|**42** | | |
| | 1 | **35**\|47\|18 | | 30\|30\|**41** | | 25\|38\|**37** | | | 30\|18\|**52** | |
| | 2 | | **41**\|30\|30 | | | | 31\|24\|**44** | | | **52**\|22\|26 |
| Llama 3.1 8B | 0 | **30**\|51\|19 | | | | **27**\|45\|**27** | | **40**\|22\|34 | | |
| | 1 | | **37**\|38\|25 | | **27**\|45\|**27** | | **39**\|37\|22 | | **39**\|23\|38 | |
| | 2 | | | **44**\|24\|31 | | 22\|37\|**39** | | | | 31\|22\|**44** |
| Gemma 2B instruct | 0 | **42**\|30\|27 | | | 34\|22\|**40** | | | | 23\|47\|**30** | |
| | 1 | | **52**\|18\|30 | | | 38\|23\|**39** | | **30**\|47\|23 | | **37**\|28\|35 |
| | 2 | | | 26\|22\|**52** | | | **44**\|22\|31 | | 35\|28\|**37** | |

Cell content formatted as: *Duels won by LLM A* |*Tie* |*Won by LLM B* (in %). Empty cells mean that the respective duel was not carried out.

all LLMs (see Fig. 7). Gemma 2B and Llama 3.1 8B perform best with $H_{depth} = 1$, while GPT-4o benefits most from $H_{depth} = 2$.

### 4.3. Comparison with other RAG systems

In the last section, we compare the complete OWL-based RAG system with a state-of-the-art text-based RAG system and a non-OWL-based GraphRAG system. The document basis is again the user manual of the filter system, which is manually converted into text for the comparison systems in order to avoid PDF extraction errors.

- *Text-based RAG*: We implement a *langchain*-based system (Chase, 2025) with an embedding database and Gemma 2B, Llama 3.1 8B and GPT-4o for querying. We chunk the text into 1500-character segments with 100-character overlap using the recommended *RecursiveCharacterTextSplitter*, which does the split into units of meaning using line breaks. Smaller chunks show a deterioration in retrieval quality in the test.
- *Non-OWL-GraphRAG*: We use Microsoft's popular *GraphRAG* system (Corporation, 2024), which combines text and graph-based retrieval. During the indexing process, GPT-4o uses text chunks of 1500 characters. To avoid errors by reducing processing complexity, we need to limit chunks to 1000 characters for Llama 3.1 8B and 500 characters for Gemma 2B. The different context sizes are due to the use of the respective LLM during the pre-processing of the text content. We evaluate the query modes *local* and *global* and evaluate the better mode in each case. The recommended query mode that combines both modes, requires too much processing time for a speech-based interaction due to numerous internal LLM queries.
- *OWL-GraphRAG*: For OWL-RAG we use the best configuration from Section 4.1 with $H_{depth} \in \{0, 1\}$ as undirected hopping for context expansion.

We select 33 random questions each on product, process and resource content and compare both RAG methods on the following aspects:

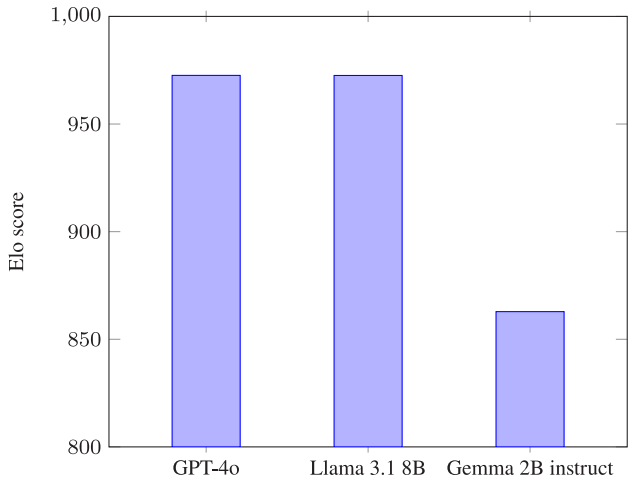


**Fig. 6.** Elo Scores of different LLMs and $H_{depth}$ against each other.

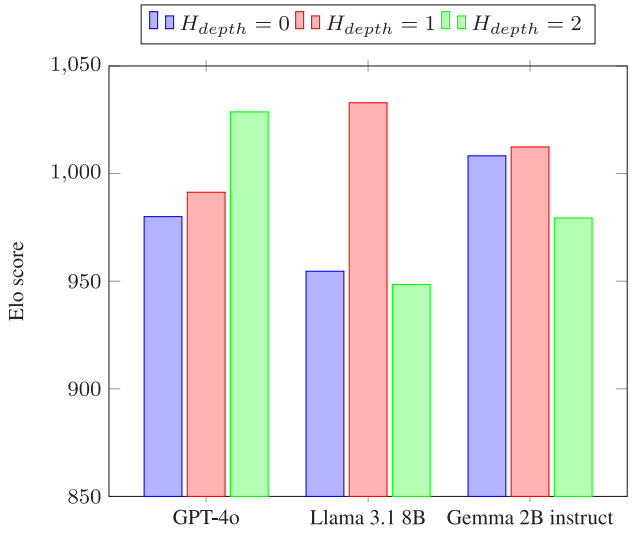


**Fig. 7.** Elo scores of intra LLM duels with different $H_{depth}$.

- *Completeness*: Relative amount of retrieval results that contain all necessary information to answer the question.
- *Context size*: The size of the formulated context, measured in characters. Affects processing time and accuracy. We only measure the context size of complete answer sets.
- *Accurate answers*: Relative amount of responses with correct information based on accurate contextualisation without hallucinations. Applies to complete search results only.

Table 7 compares retrieval and generation processes. OWL-based retrieval with $H_{depth} = 1$ achieves the best generation results and generally the best combined results for the two weaker LLMs. The OWL-based RAG without neighbouring entities finds less complete context information than the other RAG systems, especially for process questions. This is due to the fine-grained structure of OWL triples, which are therefore more dependent on neighbouring entities for information enrichment and provide less (textual) context for embeddings. However, the OWL-based system outperforms the text chunk-based system in generation, with the Gemma 2B instruct as the weakest model benefiting in particular. The clear entity structure and relationship descriptions reduce hallucinations and incompleteness. The differences within the non-OWL-based GraphRAG are striking, with the system tailored to the performance of GPT-4o with its large contexts and LLM-intensive indexing. While the performance of all three RAG methods for GPT-4o are close to each other, the OWL-based system clearly outperforms the other approaches when using Gemma 2B (see Fig. 8).

During the evaluation, we found that the text-based RAG search is more dependent on the wording of the question than the other RAG systems; similar questions sometimes give different results. All three LLMs struggled with the reverse error priority in the filter manual (error priority 3 instead of 1 as the most critical). For longer technical terms (e.g. “Ex II 1D TX”), the OWL-based system benefits from the explicit comparison $sim_{ex}$.

## 5. Discussion

### 5.1. OWL-based retrieval

The formalisation of OWL offers advantages for retrieval processes. The categorisation of entities and relations enables type-specific comparison methods to identify relevant elements. Explicit comparison methods $sim_{ex}$ are suitable for maintenance domains, using precise, well-defined vocabulary (e.g. IDs, unambiguous naming of components and routines). Vector-based procedures $sim_{vec}$ supplement the explicit comparison procedures and improve $F_1$ and $F_2$ results using the *Union* or *Majority* strategy. The $sim_{vec}(R_{O,A}, q)$ method is the least relevant, indicating that OWL entity relations are minor in the initial retrieval. Classes and Individuals $C \cup I$ and Object Property Assertions $R_{O,A}$ support explicit and vector-based matching. Including neighbouring individuals improves the completeness of the retrieved results, especially in maintenance processes (see Table 7). This is because process questions often relate to (neighbouring) successor or predecessor processes.

With $H_{depth} = 1$ and a 27% smaller context, the OWL-based RAG method achieves an identical completeness retrieval performance as the text-based RAG. Only Microsoft’s GraphRAG method achieves better performance with GPT-4o, with a context that is 14 times larger. Both comparative RAG methods require a text-based database, so it is not possible to use existing OWL ontologies directly.

It should be noted that we analyse a sample OWL ontology and a Q&A dataset based on it. The ontology follows the product–process–resource subdivision and is based on a top-level ontology, but is not universally valid for all maintenance domains or filter systems. Different modelling styles may be appropriate depending on the application. However, due to OWL’s formalisation and domain-independent matching procedures, we believe that the RAG mechanism will produce similar results with other OWL ontologies.

### 5.2. Generation quality

Concatenation of subject–predicate–object triples from OWL retrieval confirms the suitability of OWL for RAG response generation. All LLMs tested produced meaningful results, demonstrating that smaller models can support this process for a maintenance assistant. GPT-4o outperforms Llama 3.1 8B in all configurations (see Table 7), but user ratings give similar Elo scores (see Fig. 6). This is partly due to the large number of undecided duels (>37% of the time both model answers are rated as good, see Table 6), and partly due to the different focus of user-centred evaluation which involves subjective judgements, taking into account irrelevant or supporting information, complex wording and special formatting (e.g. lists). This explains how Gemma 2B can outperform GPT-4o in user-centred evaluation at lower hop levels.

As LLM performance declines, so does answer faithfulness. Weaker LLMs add extraneous information (e.g. safety instructions) that is missing from the context. Incomplete contexts, especially for general machines and maintenance, allow for improvement. However, for specific machines and instructions, there is a risk of hallucination if the LLM deviates from the ontology (e.g. inverted priority levels, Section 4.3).

**Table 7**
Comparison of the OWL-based RAG method with a classic text-based RAG and a non-OWL-based GraphRAG method.

| Retrieval | | | Generation | | |
|---|---|---|---|---|---|
| Completeness (in %) | Avg context size (no. characters) | Median context size (no. characters) | Accurate answers GPT-4o (in %) | Accurate answers Llama 3.1 8B (in %) | Accurate answers Gemma 2B (in %) |
| Text-based RAG | | | | | |
| 85.9 | 4570.9 | 4537 | 92.9 | 79.1 | 60.5 |
| Non-OWL-based GraphRAG | | | | | |
| **100** | 41,032.5 | 41,513 | 89.9 | | [a] |
| 17.3 | 5455\|3925.6 | 5455\|3921 | | 88.2 | [b] |
| 11.2 | 5190\|3925.6 | 5190\|3921 | | | 72.7[b] |
| OWL-based RAG $H_{depth} = 0$ | | | | | |
| 70.7 | 601.5 | 442 | 94.2 | **90** | **85.7** |
| OWL-based RAG $H_{depth} = 1$ | | | | | |
| 85.9 | 2913.1 | 2074 | **97.6** | 87.1 | 80 |

[a] Local GraphRAG query mode.
[b] Global GraphRAG query mode. Uses two internal sequential LLM requests. Context sizes of both requests are separated by "|".

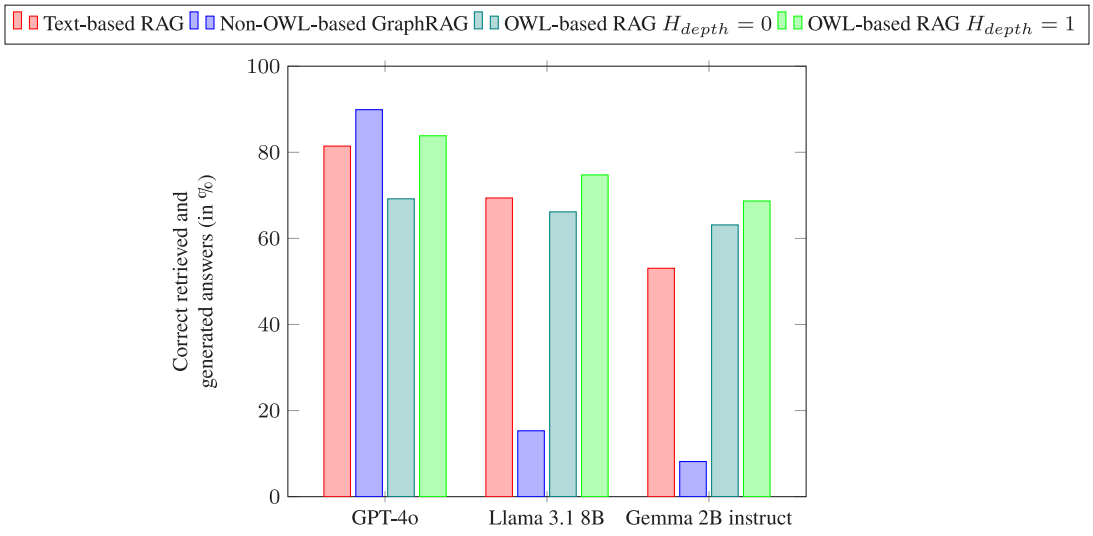


**Fig. 8.** Comparison of the overall performance of the RAG systems.

This can be mitigated by increasing the context size via $H_{depth}$ or by specialised prompting.

All three LLMs benefit from ontology hopping for context enrichment. Figs. 7 and 8 show that GPT-4o gains the most from additional information due to superior processing of larger contexts. This confirms that larger contexts improve responses even when they go beyond directly found OWL entities. However, $H_{depth} > 1$ introduces more irrelevant information that is misclassified as relevant by weaker LLMs.

The comparison with text-based RAG and non-OWL-GraphRAG suggests that the dense, structured formulation of OWL contexts improves LLM processing. In particular, uniform OWL structures improve response quality across models, with weaker LLMs benefiting most.

### 5.3. Suitability for voice based interaction

This paper presents a RAG method for chat and speech-based interaction. Freely available speech-to-text engines (e.g. OpenAI's *Whisper* (Radford et al., 2023)) and synthetic speech output enable speech-based operation (see Fig. 1). The use of less demanding LLMs ensures high processing speed and low latency, resulting in a good user experience by ensuring uninterrupted dialogue (Ludwig et al., 2023). If the ontology provides multilingual entity labels, the assistant supports multiple languages, with weaker LLMs performing best in English. The ease of customisation of OWL enables adaptive information delivery based on user qualification, authorisation and task-specific constraints of the OWL TBox (e.g. prioritisation of Data Property Assertions $R_{D,A}$ of a particular type $R_A$ that lead to the safety instructions). Withholding maintenance information due to insufficient qualification helps prevent accidents (e.g. excluding maintenance routines $I$ of type $C$ for non-qualified workers).

Further research is needed on the usability of speech-based dialogue systems. Large responses, such as multiple maintenance instructions, need to be broken down step by step with appropriate prompting. Studies have investigated the challenge of industrial noise in speech processing (Li et al., 2022; Birch et al., 2021) and confirmed its industrial suitability (Fleiner et al., 2021; Serras et al., 2020). However, further research needs to be evaluated for phonetic matching techniques $sim_{phon}$ in noisy environments.

## 6. Conclusion

In this paper, we present an OWL-based RAG procedure for a voice-controlled maintenance assistant, demonstrating the suitability of ontologies for knowledge modelling. The expressiveness of OWL and semantic retrieval mechanisms enable effective retrieval and information provision. This OWL-based Graph RAG approach provides a strong foundation for advanced digital maintenance assistants.

Given the extensive research on OWL, existing top-level OWL ontologies can be easily extended for new applications. The combination of classical and embedding-based retrieval methods proves to be the

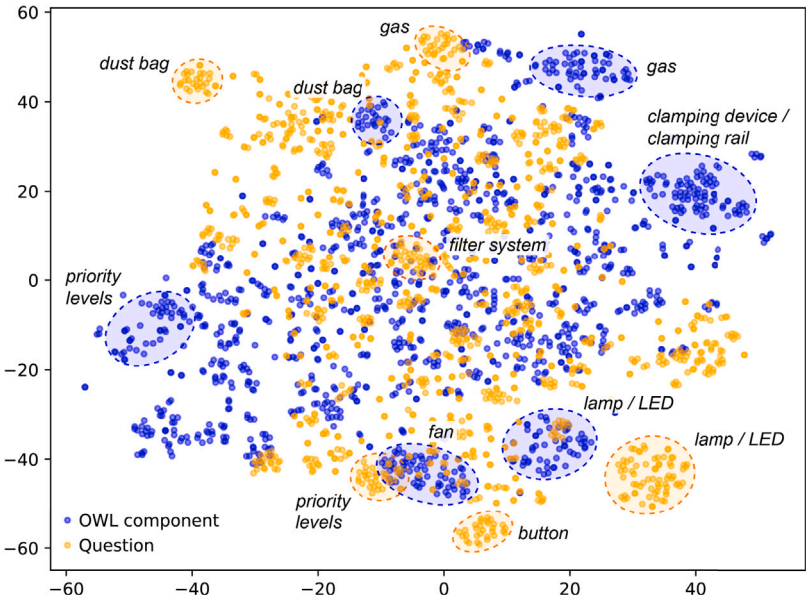


**Fig. A.9.** Visualisation of sentence embeddings with exemplary chunks using t-distributed stochastic neighbour embedding (t-SNE).

**Table B.8**
Evaluation of $\text{sim}_{\text{vec}}(R_{D,A}, result_1)$ for $\min_{\theta\,:\,\theta\in[-0.1,0.9]} \cos(\theta)$ for the Q&A dataset.

| $min_\theta cos(\theta)$ | AvgPrec | AvgRec | $F_1$ | $F_2$ |
|---|---|---|---|---|
| −0.1 | 0.703 | **1** | 0.8257 | 0.922 |
| 0 | 0.703 | 0.999 | 0.826 | 0.922 |
| 0.1 | 0.707 | 0.999 | 0.828 | **0.923** |
| 0.2 | 0.714 | 0.995 | **0.832** | 0.923 |
| 0.3 | **0.72** | 0.982 | 0.831 | 0.916 |
| 0.4 | 0.717 | 0.964 | 0.822 | 0.902 |
| 0.5 | 0.695 | 0.912 | 0.788 | 0.857 |
| 0.6 | 0.655 | 0.825 | 0.73 | 0.784 |
| 0.7 | 0.553 | 0.639 | 0.593 | 0.62 |
| 0.8 | 0.309 | 0.329 | 0.318 | 0.324 |
| 0.9 | 0.039 | 0.057 | 0.046 | 0.052 |

most powerful, allowing a practical RAG application for maintenance. Regulation of context size via minimum sentence embedding similarity $min_\theta cos(\theta)$ and hop depth $H_{depth}$ adapts to LLM performance. In particular, smaller, freely available LLMs can process OWL-based knowledge and ensure accessibility, scalability and privacy through local execution and show better performance compared to text-based RAG. This is also where the OWL-based RAG shows the greatest performance advantages over existing RAG processes, providing the basis for a robust framework for a digital text- or speech-based maintenance assistant.

Looking ahead, further research and system optimisation is needed in ontology modelling, retrieval quality and usability. Automation of ontology creation (*ontology learning*) is desirable, as manual methods are time-consuming and error-prone (Yang et al., 2021). While ontology learning has been studied, it remains unexplored in industrial maintenance. The integration of graph theory could improve retrieval by evaluating OWL entities via graph structures (e.g. Graph Attention Networks (Vrahatis et al., 2024)). The refinement of embedding-based similarity methods for maintenance-specific embeddings is also promising. The hybrid retrieval architecture and the inclusion of LLMs in the data indexing of the non-OWL-based GraphRAG system is a worthwhile addition to the RAG method presented. Finally, voice-based interaction needs to support longer, staged responses to mitigate cognitive overload, applying principles of voice user interfaces.

## CRediT authorship contribution statement

**Heiner Ludwig:** Writing – original draft, Supervision, Software, Resources, Methodology, Investigation, Data curation, Conceptualization. **Thorsten Schmidt:** Writing – review & editing. **Mathias Kühn:** Writing – review & editing.

## Declaration of generative AI and AI-assisted technologies in the writing process

During the preparation of this work the author(s) used *DeepL Translator* in order to improve the English wording. After using this tool/service, the author(s) reviewed and edited the content as needed and take(s) full responsibility for the content of the publication.

## Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Heiner Ludwig reports financial support was provided by Industrielle Gemeinschaftsforschung (IGF). If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Acknowledgement

The research is financially supported by the german Industrielle Gemeinschaftsforschung (IGF) under project number 22927 BG.

## Appendix A. Visualisation sentence embeddings

In order to analyse the meaningfulness of the sentence embeddings, we use the *t-distributed stochastic neighbour embedding* (t-SNE) method to show the distribution of the encoded OWL components and questions of the Q&A dataset with regard to clusters (see Fig. A.9). We thus perform a dimensional reduction of the 384-dimensional SBERT sentence embeddings into 2-dimensional space by initialising t-SNE with a perplexity of 50 and a maximum of 5000 iterations. The visualised clusters contain of similar text content (for both the questions and the OWL components). This shows that the sentence embeddings adequately represent the formulated OWL structures and questions in the vector space by their meaning (and not because of the sentence structures) and are therefore suitable for context retrieval tasks. It should be noted that the distances between the clusters are not necessarily meaningful, as t-SNE focuses on the preservation of local structures.

**Table C.9**
Full comparison of the OWL-based RAG method with a classic text-based RAG and a non-OWL-based GraphRAG method.

| | Retrieval | | | Generation | | |
|---|---|---|---|---|---|---|
| | Completeness (in %) | Avg context size (no. characters) | Median context size (no. characters) | Accurate answers GPT-4o (in %) | Accurate answers Llama 3.1 8B (in %) | Accurate answers Gemma 2B (in %) |
| | Text-based RAG | | | | | |
| Product | 84.9 | 4806.3 | 5120 | 89.3 | 71.4 | 60.7 |
| Process | 78.8 | 4526.7 | 4396 | 96 | 84.6 | 84.6 |
| Resource | 93.8 | 4379.7 | 4177 | 96.8 | 83.9 | 41.9 |
| Overall | 85.9 | 4570.9 | 4537 | 92.9 | 79.1 | 60.5 |
| | Non-OWL-based GraphRAG (GPT-4o)[a] | | | | | |
| Product | 100 | 41,392.8 | 41,804 | 90.9 | | |
| Process | 100 | 41,196 | 41,721 | 93.9 | | |
| Resource | 100 | 40,508.8 | 40,598 | 84.8 | | |
| Overall | **100** | 41,032.5 | 41,513 | 89.9 | | |
| | Non-OWL-based GraphRAG (Llama 3.1 8B)[b] | | | | | |
| Product | 21.2 | 5455\|3947.1 | 5455\|4006 | | 62.5 | |
| Process | 21.2 | 5455\|3990.5 | 5455\|3962 | | 100 | |
| Resource | 9.4 | 5455\|3747 | 5455\|3746 | | 100 | |
| Overall | 17.3 | 5455\|3925.6 | 5455\|3921 | | 88.2 | |
| | Non-OWL-based GraphRAG (Gemma 2B)[b] | | | | | |
| Product | 3 | 5190\|4031 | 5190\|4031 | | | 100 |
| Process | 18.2 | 5190\|4003.7 | 5190\|3974.5 | | | 75 |
| Resource | 12.5 | 5190\|3847 | 5190\|3769.5 | | | 66.7 |
| Overall | 11.2 | 5190\|3925.6 | 5190\|3921 | | | 72.7 |
| | OWL-based RAG $H_{depth} = 0$ | | | | | |
| Product | 78.8 | 723.6 | 615 | 100 | 88 | 88.5 |
| Process | 48.5 | 691 | 491 | 93.8 | 100 | 75 |
| Resource | 84.4 | 428.7 | 252 | 89.3 | 89.3 | 89.3 |
| Overall | 70.7 | 601.5 | 442 | 94.2 | **90** | **85.7** |
| | OWL-based RAG $H_{depth} = 1$ | | | | | |
| Product | 84.4 | 3323.4 | 2697 | 100 | 89.3 | 89.3 |
| Process | 75.8 | 3284.7 | 2799 | 96 | 88 | 76 |
| Resource | 97 | 2252.1 | 1036 | 96.9 | 84.3 | 75 |
| Overall | 85.9 | 2913.1 | 2074 | **97.6** | 87.1 | 80 |

[a] Local GraphRAG query mode.

[b] Global GraphRAG query mode. Uses two internal sequential LLM requests. Context sizes of both requests are separated by "|".

## Appendix B. Data property assertion filtering

The high informative value of sentence embeddings of Data Property Assertions $R_{D,A}$ (see Table 3) suggests potential for excluding irrelevant assertions when formulating retrieved Individuals. For the 1348 questions in the Q&A dataset, we manually extracted relevant $R_{D,A}$ of individuals $I$ and compared them with those classified as relevant by $sim_{vec}$. Table B.8 shows that $min_\theta cos(\theta) = 0.3$ gives the highest precision, while $min_\theta cos(\theta) = 0.1$ and 0.2 minimally improve the $F_1$ and $F_2$ scores. Overall, improvements are negligible, but may be higher for ontologies with more and/or less domain-specific $R_{D,A}$.

## Appendix C. RAG comparison

See Table C.9.

## Data availability

Data provided in Git repository, link provided.

## References


Birch, B., Griffiths, C., Morgan, A., 2021. Environmental effects on reliability and accuracy of MFCC based voice recognition for industrial human-robot-interaction. Proc. Inst. Mech. Eng. Part B: J. Eng. Manuf. 235 (12), 1939–1948. http://dx.doi.org/10.1177/09544054211014492, arXiv:https://doi.org/10.1177/09544054211014492.

Chase, H., 2025. Langchain. URL: https://www.langchain.com/.

Chiang, W.-L., Zheng, L., Sheng, Y., Angelopoulos, A.N., Li, T., Li, D., Zhang, H., Zhu, B., Jordan, M., Gonzalez, J.E., Stoica, I., 2024. Chatbot arena: An open platform for evaluating LLMs by human preference. arXiv:2403.04132.

Corporation, M., 2024. Graphrag: A modular graph-based retrieval-augmented generation (RAG) system. https://github.com/microsoft/graphrag.

Cutting-Decelle, A., Young, R., Michel, J.-J., Grangel, R., Cardinal, J., Bourey, j.-p., 2007. ISO 15531 MANDATE: A product-process-resource based approach for managing modularity in production management. Concurr. Eng.: R A 15, 217–235. http://dx.doi.org/10.1177/1063293X07079329.

Edge, D., Trinh, H., Cheng, N., Bradley, J., Chao, A., Mody, A., Truitt, S., Larson, J., 2024. From local to global: A graph RAG approach to query-focused summarization. arXiv:arXiv:2404.16130.

European Committee for Standardization (CEN), 2018. EN 13306:2018-02 Maintenance - Maintenance terminology. Technical Report, European Committee for Standardization (CEN), http://dx.doi.org/10.31030/2641990.

Fleiner, C., Riedel, T., Beigl, M., Ruoff, M., 2021. Ensuring a robust multimodal conversational user interface during maintenance work. In: Proceedings of Mensch Und Computer 2021. MuC '21, Association for Computing Machinery, New York, NY, USA, pp. 79–91. http://dx.doi.org/10.1145/3473856.3473871.

Gao, Y., Xiong, Y., Gao, X., Jia, K., Pan, J., Bi, Y., Dai, Y., Sun, J., Guo, Q., Wang, M., Wang, H., 2023. Retrieval-augmented generation for large language models: A survey. ArXiv abs/2312.10997, URL: https://api.semanticscholar.org/CorpusID:266359151.

GmbH, E., 2023. Spacy dependency parser. URL: https://spacy.io/api/dependencyparser.

Jacobs, J., 1982. Finding words that sound alike. The SOUNDEX algorithm.. Byte 7 473–474.

Karray, M.H., Ameri, F., Hodkiewicz, M., Louge, T., 2019. ROMAIN: Towards a BFO compliant reference ontology for industrial maintenance. Appl. Ontol. 14, 155–177. http://dx.doi.org/10.3233/AO-190208, 2.

Kulvatunyou, B., Drobnjakovic, M., Ameri, F., Will, C., Smith, B., 2022. The Industrial Ontologies Foundry (IOF) Core Ontology. Formal Ontologies Meet Industry (FOMI) 2022, Tarbes, FR, URL: https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=935068.

Li, S., Yerebakan, M.O., Luo, Y., Amaba, B., Swope, W., Hu, B., 2022. The effect of different occupational background noises on voice recognition accuracy. J. Comput. Inf. Sci. Eng. 22 (5), 050905. http://dx.doi.org/10.1115/1.4053521, arXiv:https://asmedigitalcollection.asme.org/computingengineering/article-pdf/22/5/050905/6869354/jcise_22_5_050905.pdf.

Liu, N.F., Lin, K., Hewitt, J., Paranjape, A., Bevilacqua, M., Petroni, F., Liang, P., 2024. Lost in the middle: How language models use long contexts. Trans. Assoc. Comput. Linguist. 12, 157–173. http://dx.doi.org/10.1162/tacl_a_00638, URL: https://aclanthology.org/2024.tacl-1.9.

Ludwig, H., Schmidt, T., Kühn, M., 2023. Voice user interfaces in manufacturing logistics: a literature review. Int. J. Speech Technol. 26 (3), 627–639. http://dx.doi.org/10.1007/s10772-023-10036-x.

Matsumoto, N., Moran, J., Choi, H., Hernandez, M.E., Venkatesan, M., Wang, P., Moore, J.H., 2024. KRAGEN: a knowledge graph-enhanced RAG framework for biomedical problem solving using large language models. Bioinformatics 40 (6), btae353. http://dx.doi.org/10.1093/bioinformatics/btae353, arXiv:https://academic.oup.com/bioinformatics/article-pdf/40/6/btae353/58186419/btae353.pdf.

Montero Jiménez, J.J., Vingerhoeds, R., Grabot, B., Schwartz, S., 2023. An ontology model for maintenance strategy selection and assessment. J. Intell. Manuf. 34 (3), 1369–1387. http://dx.doi.org/10.1007/s10845-021-01855-3.

Noy, N., Mcguinness, D., 2001. Ontology development 101: A guide to creating your first ontology. Knowl. Syst. Lab. 32.

Polenghi, A., Roda, I., Macchi, M., Pozzetti, A., Panetto, H., 2022. Knowledge reuse for ontology modelling in maintenance and industrial asset management. J. Ind. Inf. Integr. 27, 100298. http://dx.doi.org/10.1016/j.jii.2021.100298, URL: https://www.sciencedirect.com/science/article/pii/S2452414X21000947.

Radford, A., Kim, J.W., Xu, T., Brockman, G., Mcleavey, C., Sutskever, I., 2023. Robust speech recognition via large-scale weak supervision. In: Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., Scarlett, J. (Eds.), Proceedings of the 40th International Conference on Machine Learning. In: Proceedings of Machine Learning Research, vol. 202, PMLR, pp. 28492–28518.

Reimers, N., Gurevych, I., 2020. Making monolingual sentence embeddings multilingual using knowledge distillation. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, URL: https://arxiv.org/abs/2004.09813.

Sen, P., Mavadia, S., Saffari, A., 2023. Knowledge graph-augmented language models for complex question answering. In: Dalvi Mishra, B., Durrett, G., Jansen, P., Neves Ribeiro, D., Wei, J. (Eds.), Proceedings of the 1st Workshop on Natural Language Reasoning and Structured Explanations. NLRSE, Association for Computational Linguistics, Toronto, Canada, pp. 1–8. http://dx.doi.org/10.18653/v1/2023.nlrse-1.1, URL: https://aclanthology.org/2023.nlrse-1.1.

Serras, M., García-Sardiña, L., Simões, B., Álvarez, H., Arambarri, J., 2020. AREVA: Augmented reality voice assistant for industrial maintenance.

Siddharth, L., Luo, J., 2024. Retrieval augmented generation using engineering design knowledge. Knowl.-Based Syst. 303, 112410. http://dx.doi.org/10.1016/j.knosys.2024.112410, URL: https://www.sciencedirect.com/science/article/pii/S095070512401044X.

Silvestri, L., Forcina, A., Introna, V., Santolamazza, A., Cesarotti, V., 2020. Maintenance transformation through industry 4.0 technologies: A systematic literature review. Comput. Ind. 123, 103335. http://dx.doi.org/10.1016/j.compind.2020.103335, URL: https://www.sciencedirect.com/science/article/pii/S0166361520305698.

Ting Zheng, S.M., Glock, C.H., 2024. A review of digital assistants in production and logistics: applications, benefits, and challenges. Int. J. Prod. Res. 1–27. http://dx.doi.org/10.1080/00207543.2024.2330631, arXiv:https://doi.org/10.1080/00207543.2024.2330631.

Vrahatis, A.G., Lazaros, K., Kotsiantis, S., 2024. Graph attention networks: A comprehensive review of methods and applications. Futur. Internet 16 (9), http://dx.doi.org/10.3390/fi16090318, URL: https://www.mdpi.com/1999-5903/16/9/318.

W3C, 2004. Owl web ontology language reference. URL: https://www.w3.org/TR/owl-ref/. (Accessed 19 April 2024).

Wang, X., Yang, Q., Qiu, Y., Liang, J., He, Q., Gu, Z., Xiao, Y., Wang, W., 2023. KnowledGPT: Enhancing large language models with retrieval and storage access on knowledge bases. ArXiv abs/2308.11761. URL: https://api.semanticscholar.org/CorpusID:261076315.

Wellsandt, S., Foosherian, M., Bousdekis, A., Lutzer, B., Paraskevopoulos, F., Verginadis, Y., Mentzas, G., 2023. Fostering human-AI collaboration with digital intelligent assistance in manufacturing SMEs. In: Alfnes, E., Romsdal, A., Strandhagen, J.O., von Cieminski, G., Romero, D. (Eds.), Advances in Production Management Systems. Production Management Systems for Responsible Manufacturing, Service, and Logistics Futures. Springer Nature Switzerland, Cham, pp. 649–661.

Wellsandt, S., Rusák, Z., Ruiz Arenas, S., Aschenbrenner, D., Hribernik, K., Thoben, K.-D., 2020. Concept of a voice-enabled digital assistant for predictive maintenance in manufacturing. http://dx.doi.org/10.2139/ssrn.3718008.

Woods, C., French, T., Hodkiewicz, M., Bikaun, T., 2023a. An ontology for maintenance procedure documentation. Appl. Ontol. 18, 1–38. http://dx.doi.org/10.3233/AO-230279.

Woods, C., Selway, M., Bikaun, T., Stumptner, M., Hodkiewicz, M., 2023b. An ontology for maintenance activities and its application to data quality. Semant. Web Preprint, 1–34. http://dx.doi.org/10.3233/SW-233299, Preprint.

Yang, L., Cormican, K., Yu, M., 2021. Ontology learning for systems engineering body of knowledge. IEEE Trans. Ind. Inform. 17 (2), 1039–1047. http://dx.doi.org/10.1109/TII.2020.2990953.