DLR Projektträger

Projektträger für das
Bundesministerium
für Wirtschaft
und Energie

INDUSTRIELLE GEMEINSCHAFTSFORSCHUNG

Schlussbericht vom 30.09.2025

zum IGF-Vorhaben 01IF22927N

Thema

Entwicklung eines Assistenzsystems mit adaptiver sprachbasierter und visueller Benutzerschnittstelle zur optimalen Behebung von Störungen im Produktionsumfeld

Berichtszeitraum

01.01.2024 bis 30.09.2025

Forschungsvereinigung

Bundesvereinigung Logistik e. V.

Forschungseinrichtung(en)

TU Dresden, Professur für Technische Logistik [FE1]

TU Dresden, Professur für Wirtschaftsinformatik, insb. Business Engineering [FE2]

Inhaltsverzeichnis

1 Kurzfassung des Projektverlaufes und der Ergebnisse

Das Ziel des AVISSBA-Forschungsvorhabens war es zu untersuchen, wie sich die Interaktion eines Störungsbehebungsassistenzsystems mit einer visuellen (Smartphone/ Smartwatch/ Tablet) und sprachbasierten (Headset) Benutzerschnittstelle in Abhängigkeit verschiedener produktionslogistischer Kontextfaktoren (d.h. adaptiv) bestmöglich gestalten lässt. Die gewonnenen Erkenntnisse mündeten in die prototypische Umsetzung eines lernfähigen Systems. Lernfähig heißt, dass mittels Data-Mining gesammelter Daten und Erfahrungswerten eine Künstliche Intelligenz (KI) zur Auswertung natürlicher Spracheingaben und zur Generierung von Störungsbehebungen befähigt wird.

Dabei galt es drei Systemstufen im Praxiskontext zu konzipieren und prototypisch umzusetzen:

1. Wissensaufbau: Die Informationen zu Prozessen, Ressourcen und Produkten sollen im Wartungskonzept firmenspezifisch so modelliert werden, dass sie eine transparente, skalierbare und homogene Datenbasis bilden. Die Erstellung soll mit möglichst wenig Arbeitsaufwand verbunden sein.
2. Informationsbereitstellung: Die Wartungsinformationen müssen kontextabhängig bereitgestellt werden. Dabei soll auch sensibles, firmenspezifisches Wissen einbezogen werden. Das Ziel besteht in der Entwicklung eines Mechanismus, der Abfragen in natürlicher Sprache ermöglicht und je nach Mitarbeiterqualifikation und genutzter Hardware relevante Informationen ausgibt.
3. Einbezug visueller Informationen: Zum besseren Kontextverständnis soll der digitale Wartungsassistent Zugriff auf visuellen Echtzeitinformationen erhalten. Auf diese Weise muss der Wartungsmitarbeiter weniger umfangreiche Beschreibungen in das System eingeben.

Im Projekt wurden zunächst Kontextfaktoren (wie Wartungsabläufe, Qualifikation des Wartungspersonals, Einsatzumgebung, Datenschutz) für das Assistenzsystem in Zusammenarbeit mit den Projektpartner aufgestellt. Zudem wurden verschiedene Systeme für den Wissensspeicher untersucht, die eine ausreichende Flexibilität zur detaillierten Modellierung spezifischer Informationen bieten. Dabei hat sich ein Graph-basierter Ansatz in Form einer *Web Ontology Language* (OWL)-basierten Ontologie als vielversprechend erwiesen.

Die Entwicklungen wurden dabei stets in Praxisumgebungen (d.h. Versuchshalle der TU Dresden und vor Ort bei Partnern des projektbegleitenden Ausschusses) evaluiert. In verschiedenen Workshops mit den Projektpartner wurden Anforderungen an das System, Vorgehensweisen zur Modellierung der Datenstruktur, Prozessaufnahmestrategien und allgemein Notwendigkeit eines systematischen Störungsmanagements geklärt. Das KI-FineTuning zur Formalisierung des Unternehmenswissens wurde auf den Hochleistungsrechner der TU Dresden durchgeführt. Ergebnisse des Forschungsprojekts wurden auf mehreren Konferenzen vorgestellt und in Journals veröffentlicht.

2 Durchgeführte Arbeiten und Ergebnisse im Berichtszeitraum

2.1 Analyse (AP 1)

Unternehmensanalyse und erweiterte Literaturrecherche

Zur Gewinnung eines Überblicks über bestehende industriell genutzte digitale Assistenzsysteme und vorangegangene Forschungsprojekte wurden systematische Recherchen hinsichtlich Systemfunktionalität, technischer Umsetzung, möglichen Hürden und Hardwareoptionen durchgeführt (siehe Abb. 1). Durch die anhaltend rasante Entwicklung von großen KI-Sprachmodellen (Large Language Models, kurz LLM) seit der Einführung von ChatGPT-3.5 ab Ende 2022 wurden während der gesamten Projektlaufzeit neue Methoden und KI-Modelle recherchiert und deren Einsatz bewertet. Neben den KI-Modellen wurden zur Systemgestaltung potenzielle Kontextfaktoren gesammelt, die im Einsatz Einfluss auf die Systemkonfiguration haben. Dafür wurden während des Kick-Off-Meetings im Austausch mit dem Projektbegleitenden Austausch (PA) die aufgestellten Kontextfaktoren diskutiert und bewertet. Zur Verfeinerung und Klärung von domänenspezifischen Anforderungen wurden des weiteren Einzelgespräche mit Firmen des PA durchgeführt. Dabei wurden zunächst Firmen aus den Bereichen Keramik- und Metallverarbeitung, Logistikberatung, Logistiksoftware und Flugzeugbau befragt, im Projektverlauf kamen weitere Partner aus der Halbleiterbranche, Bahnbau, Werkzeugmaschinenbau, Lebensmittelproduktion und Militär hinzu.

Produkte:

Es existieren verschiedene kommerzielle Produkte, die Wartungsmaßnahmen unterstützen. Ziel der folgenden Auflistung ist die Schaffung eines Überblicks verschiedener Software- und digitaler Assistenzsysteme anhand von Beispielen (siehe Tabelle 1). Während der Projektlaufzeit wurde eine Vielzahl an neuen, KI-basierten Systemen veröffentlicht, sodass die Liste dauerhaft erweitert wurde.

Tabelle 1 Auswahl kommerzieller Produkte zur Unterstützung von Wartungsmaßnahmen

Confluence - Atlassian	Populäre, kommerzielle Wiki-Software
Eine webbasierte Kollaborations- und Wissensmanagement-Plattform (Corporate Wiki), mit der Teams Inhalte wie Seiten, Dokumentationen und Besprechungsnotizen erstellen, gemeinsam bearbeiten und verwalten können. Auch geeignet zum Zusammentragen von Wissen für Wartungsprozesse.	

IQX	Web-App für Wartung und Instandhaltung
Eine Web-basierte App zur Dokumentation von Wartungsvorgängen per Formulareingaben und Fotos. Enthält Checklisten und Terminplanung sowie Ersatzteilmanagement und Ersatzteillagerverwaltung.	

MaintainX	KI-Assistent für Instandhaltung und Anlagenmanagement

Ein cloudbasiertes System für das Wartungsmanagement, das die Bereiche Work Orders, Anlagenverwaltung, Checklisten, Ersatzteile und Auswertungen zentral abbildet. Der integrierte KI-Assistent erstellt automatisch Wartungsanweisungen aus Dokumentationen, liefert kontextbezogene Hilfen für Techniker und ermöglicht das Erstellen von Berichten in natürlicher Sprache.

Timly	Cloud-basierte App für Instandhaltung und weiteres digitales Asset-Management
Eine cloudbasierte Software zur Verwaltung von Anlagen, Geräten und Wartungen. Sie erinnert automatisch an Wartungs- und Prüftermine und dokumentiert alle Maßnahmen zentral. Erlaubt das Hinterlegen von Anleitungen und Bildern.	

osapiens	Cloud-basierte Instandhaltungssoftware
Ermöglicht Anlagen, Wartungsaufträge, Ersatzteile und Lager zentral zu verwalten und Wartungsprozesse digital zu planen, zu dokumentieren und zu steuern. Bietet Methoden zur präventiven Instandhaltung.	

Wie sich aus der Recherche ergeben hat, existieren zahlreiche Softwareprogramme, die die Wartung unterstützten. Auch die Integration von KI-Chatbots nimmt zu. Insgesamt dominiert die rein visuelle Anleitung von Werkern für die Wartung per Menüstrukturen, Formularen und dokumentierten Wartungsschritten. Die Verknüpfung mit weiteren Informationen zu benötigten Werkzeugen und Verbrauchsmitteln ist teilweise möglich. Neben allgemein für die Wartung nutzbaren Programmen existieren auch branchenspezifische Anwendungen.

Forschungsprojekte:

Im Folgenden eine Auswahl an Forschungsprojekte, die sich ebenfalls mit der Schaffung von digitalen Wartungsassistenten beschäftigen, oder Grundlagen für den Aufbau legen (siehe Tabelle 2).

Tabelle 2 Auswahl von Forschungsprojekten zur Schaffung digitaler Wartungsassistenten

IMAIN	2012 – 2015
EU-Forschungsprojekt zur Kombination von realen und virtuellen Sensoren in Kombination mit FE-Simulationen zur Analyse mechanischer Belastungen. Aufbaue einer e-Maintenance-Cloud zur Kombination verschiedener Wartungsrelevanter Anwendungen, wie Dashboards und Alarmierung im Fall kritischer Maschinenausfälle.	

SERENA	2017 – 2021
EU-Forschungsprojekt zur Untersuchung von einfach zu bedienenden Schnittstellen für die Datenverwaltung und die Unterstützung menschlicher Bediener hinsichtlich Maschinenstatus und Wartungsanweisungen mithilfe von AR-Geräten.	

BaSys 4.2	2019 – 2022
Forschungsprojekt zur Erweiterung und dem besseren Zugang zur Verwaltungsschale von Industrie 4.0-Komponenten. Die ganzheitliche Betrachtung der Daten und die Verknüpfung von Prozess und Produktionsdaten sind auch für Wartungsroutinen relevant.	

SPAICER	2020 – 2023
Forschungsprojekt um Störungen frühzeitig zu erkennen und ihre Prozesse resilient und flexibel zu steuern. Kern dafür sind modulare „Smart Resilience Services“, die Datenanalysen, Machine Learning und wissensbasierte Verfahren kombinieren, um Handlungsempfehlungen und automatische Anpassungen zu ermöglichen. Dadurch wird Expertenwissen digitalisiert, Entscheidungsprozesse werden unterstützt und Produktionsqualität sowie Anlagenverfügbarkeit werden deutlich verbessert.	

KI-Reallabor (SmartFactoryOWLab)	Seit 2023
Ein Softwareprototyp, der anhand sequentieller Prozessschritte Hinweise zu typischen Problemen, deren Ursachen und Behebungsanleitungen ausgibt. Das System basiert dabei auf vorher eingetragenen Arbeitsroutinen und Erfahrungswerten. Die Software basiert auf einer rein visuellen Ausgabe.	

Zusammenfassend zeigt sich, dass die Wartung und Reparatur von technischen Geräten ein hochrelevantes Thema darstellt und in diesem Bereich viel geforscht wird. Die meisten Forschungsprojekte basieren dabei auf statisch vorgegebenen Abläufen, die nur eingeschränkt auf nicht dokumentierte Störungen reagieren können. Die Nutzung von semantischen Datenstrukturen zur Generierung einer homogenen Datenbasis ist vielversprechend.

Abbildung 1 Generationen von Assistenzsystemen

Kontextfaktoren:

Während im Projekt ursprünglich der Einsatz von kleineren KI-Sprachmodellen zur Analyse semantischer Ähnlichkeit und Kombination mit Wissensgraphen zur Bildung von vordefinierten Wartungsabläufen geplant war, hat der Durchbruch bei LLMs zu neuen Möglichkeiten komplexer und deutlich natürlicherer Dialoginteraktion geführt. Hierfür wurden sowohl praxisrelevante als auch technische Entscheidungsfaktoren zur Auswahl geeigneter Modelle aufgestellt. Zur Bewertung der allgemeinen Leistungsfähigkeit der Modelle wurde der *MMLU*-Score herangezogen. Dies hat direkte Auswirkungen auf die Kontextfaktoren des Systems:

Qualifikation und Sprache der Nutzer: Mitarbeiter sind je nach Erfahrungsgrad und Aufgabenfeld verschieden qualifiziert. Um eine hohe Akzeptanz gegenüber dem System zu garantieren, ist die Berücksichtigung der Erfahrungslevel notwendig. So sollten für unerfahrene Mitarbeiter umfangreichere Informationen ausgegeben werden, als bei Experten. In vielen KMUs arbeiten zudem Mitarbeiter, deren Muttersprache nicht deutsch ist. Das System sollte dementsprechend multilingual konfiguriert werden können.

Lautstärke und Staubbelastung: Im produktionslogistischen Arbeitsumfeld verlangen hohe Umgebungslautstärken und Staubbelastung robuste Hardware zur Bedienung des Assistenzsystems. So muss die sprachbasierte Interaktion auch bei hoher Umgebungslautstärke gewährleistet sein.

Datenschutz: Firmendaten enthalten oftmals sensible Informationen, die nicht an Dritte geteilt werden dürfen. Im Wartungskontext schließt dies insbesondere spezialisierte Geräte und Vorgehensweisen ein, aber auch Mitarbeiterdaten (wie Qualifikationen, Erfahrungslevel etc.). Entsprechend können entweder nur lokal ausführbare LLMs zum Einsatz kommen, sodass auf externe Cloud-Services verzichtet werden kann, oder ggf. Dienstleister mit einer entsprechenden Zertifizierung zum Schutz der Daten. Aus Gesprächen mit dem PA geht hervor, dass meist eine lokale Lösung bevorzugt wird.

Hardwareanforderung: Je nach Komplexität der LLMs schwankt auch die Hardwareanforderung bei der Ausführung. Mit steigender Anzahl an Parametern steigt gleichermaßen die Hardwareanforderung (abgesehen

von *Mixed-Expert-Modellen* (Cai, et al. 2025)), aber auch die Verarbeitungsgüte der Modelle. Ist ein lokaler Betrieb vorgesehen, sollten die Anforderungen so gering wie möglich sein, im Idealfall also weniger komplexe Modelle zum Einsatz kommen (bis max. 12B Parameter). Wird eine externe (Cloud-)Lösung genutzt, spielt die Hardwareanforderung keine Rolle. Um hohe Hardwarekosten zu umgehen, wurde das AVISSBA-System zudem so konzipiert, dass es firmenspezifische Daten einbeziehen kann, ohne dass zur Datenabfrage ein firmen- oder anwendungsspezifisches FineTuning von LLMs vorgenommen werden muss.

Gebühren: Externe Dienstleister verlangen entweder pauschale oder nutzungsbasierte Gebühren (i.d.R. Token-basiert). Diese sollten möglichst gering sein. Für eine lokale Nutzung fallen, bis auf den benötigten Strom, keine weiteren Gebühren an.

Lizenz: Einige LLMs besitzen Lizenzen, die die Nutzung nur für akademische Zwecke erlauben. Angestrebt sind Lizenzen, die eine vollumfängliche Nutzung erlauben, die Offenlegung der Gewichte der einzelnen Modelle wird nicht benötigt.

Verarbeitungsgeschwindigkeit: Mit steigender LLM-Komplexität sinkt bei gleichbleibender Hardware die Verarbeitungsgeschwindigkeit eingehender Informationen. Zum reibungslosen, natürlichen Ablauf von sprachgeführten Dialogen ist eine hohe Verarbeitungsgeschwindigkeit notwendig, sodass unterbrechungsfreie Interaktion zwischen Nutzer und digitalem Assistenten geschieht. Dies trägt zur zügigeren Durchführung der Arbeit bei und damit auch zur Systemakzeptanz.

Übersicht über nutzbare IT-Strukturen

Für den allgemeinen Systemaufbau wurden zunächst verfügbare Datenbanksysteme, Regelsysteme und sprachverarbeitende Modelle und Speech-to-Text und Text-to-Speech-Engines als verschiedene Komponenten verglichen. Im Projekt soll dabei nur frei verfügbare Software zum Einsatz kommen, die eine kostenfreie Anwendung für kommerzielle Zwecke erlaubt. Die Ausnahme bildet dabei ggf. der Einbezug von extern gehosteten LLMs. Um das Gesamtsystem für zukünftige technische Fortschritte zu öffnen, ist es modular mit fest definierten Schnittstellen aufgebaut, sodass einzelne Komponenten wahlweise ausgetauscht werden können.

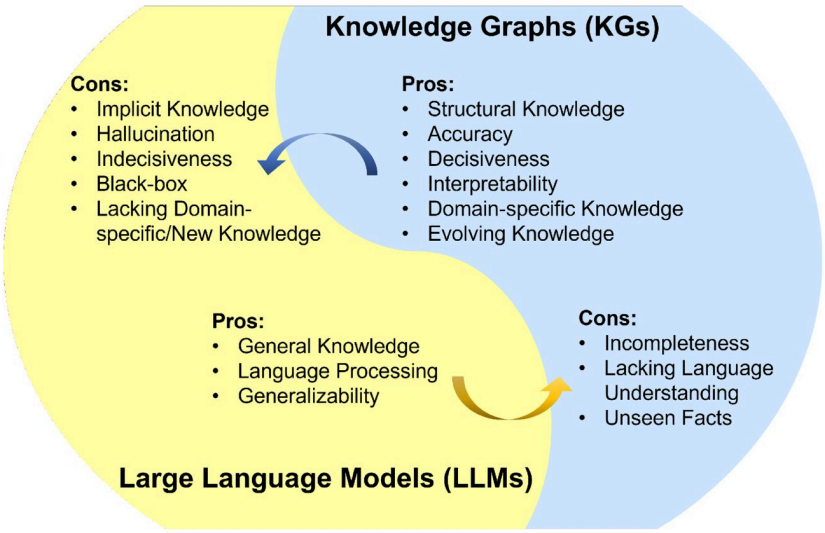

Abbildung 2 Synergie-Effekte bei der Kombination von Large Language Models und Knowledge Graphs (aus Pan, et al. 2024)

Datenbanksysteme: Für die Datenbanksysteme wurden vier Lösungen gegenübergestellt: Relationale Datenbanken, Graph-basierte Wissensspeicher, Vektordatenbanken und einfache Dokumentspeicher. Während relationale Datenbanken eine bewährte, meistgenutzte Lösung zur Datenspeicherung darstellen, haben sich in verschiedenen Voruntersuchungen Graph-basierte Wissensspeicher in Kombination mit Sprachmodellen als vielversprechende Option aufgetan. Pan et al. hebt die Synergieeffekte von LLMs und Wissensgraphen hervor und verweist auf die Aufhebung der jeweiligen Nachteile (siehe Abb. 2) (Reimers und Gurevych 2019; Pan, et al. 2024). Schlussendlich bildet eine Kombination aus Wissensgraph und Vektordatenbank die Systemgrundlage, wobei die Vektordatenbank automatisch anhand der Graphdatenstruktur mit Word- und Satzvektoren befüllt wird. Dabei kommen vortrainierte Sprachmodelle wie BERT oder SBERT zum Einsatz (Devlin, et al. 2019). Dies ermöglicht effiziente Retrievalprozesse innerhalb des Wissensgraphen anhand von semantischer Ähnlichkeit zur eingehenden Nutzeranfrage. Als Grammatik kommt der OWL-Standard zum Einsatz, der umfangreiche Formalisierungsoptionen bietet und die Nutzung von Reasoning-Verfahren zur Logiküberprüfung der Wissensstruktur erlaubt (W3 Semantic Web Standards 2012; Parsia, et al. 2017). Größere Datenmengen werden dabei in Triple-Form in einer Apache Jena-Datenbank abgespeichert (The Apache Software Foundation 2025). Zum besseren Vergleich der Leistungsfähigkeit wurde dazu in einem zweiten Ansatz ein Dokumentspeicher aufgebaut, der das Retrieval anhand der Umwandlung von Textabschnitten in semantische Vektoren durchführt.

Regelsystem: Zur Hinterlegung von Programmroutinen und -verhaltensweisen wird eine Regelsystem im Wissensgraph integriert. Dies schließt einerseits die Definition von Regelkonstrukten für das Assistenzsystem ein (Unterteilung in *Intents* und *Slots*), andererseits werden OWL- und SWRL-Axiome in die Auswertung mit einbezogen (wie bspw. Hierarchien, Kardinalitäten, Typisierung). So entsteht ein *model-driven* Softwaresystem, dass anhand der hinterlegen OWL-Ontologie ohne Programmierkenntnisse flexibel angepasst werden kann.

Speech-to-Text: Die fehlerfreie Wandlung der eingesprochenen Informationen zu Text ist essenziell für die weitere Informationsverarbeitung. Analog zur Betrachtung der Kriterien zur LLM-Auswahl spielt auch hier der Datenschutz, verfügbare Hardware und Verarbeitungsgeschwindigkeit eine wichtige Rolle. *Whisper* ist ein prominentes *OpenAI*-Modell, das kostenfrei in verschiedenen Komplexitätsstufen zur Verfügung steht (Radford, et al. 2022). In verschiedenen Tests wurde die Genauigkeit der Verarbeitungsgeschwindigkeit gegenübergestellt, sodass die Wahl letztendlich auf die *base*-Konfiguration gefallen ist (siehe Schmidt, Ludwig und Kühn 2025). Durch weitere String-Matching Verfahren wird die Erkennung ähnlicher Wörter im Wissensspeicher begünstigt, selbst wenn kein 100% exakte Übereinstimmung herrscht (u.a. durch den Einsatz der Stammformreduktion und *Levenshtein*-Distanz). Für die Leistungstests kam ein 3M Peltor Headset mit Noice-Cancelling-Technologie zum Einsatz, dass über einen integrierten Gehörschutz verfügt (siehe Abb. 3).

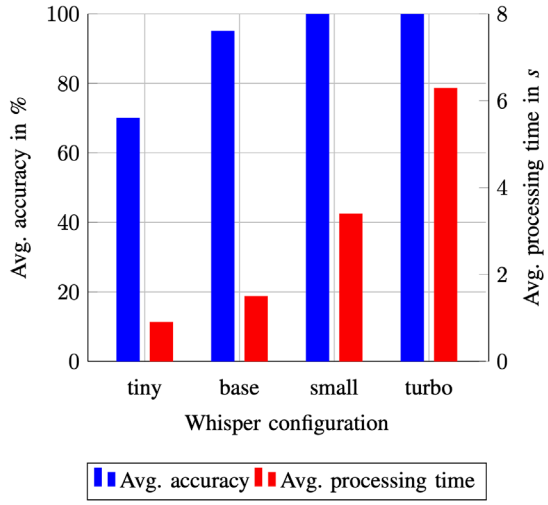

Abbildung 3 Vergleich der Whisper Sprach-zu-Text-Konfigurationen hinsichtlich Transkriptionsgenauigkeit und Verarbeitungszeit (aus Schmidt, Ludwig und Kühn 2025)

Text-to-Speech: Für die Umwandlung von Text zu Sprache zur Informationsausgabe wurden die Optionen *FreeTTS* für Erstellung der Audio-Datei im Backend und die *Web Speech API* für die Ausgabe im Frontend gewählt (FreeTTS 2025; Mozilla Foundation 2025). Werden sensible Informationen verarbeitet, ist FreeTTS vorzuziehen, da keine externe Verarbeitung der auszugebenden Texte stattfindet.

2.2 Identifikation Einsatzszenarien (AP 2)

Firmenspezifische organisatorische Ablauferfassung

In der organisatorischen Ablauferfassung zur Analyse von Wartungs- und akuten Reparaturmaßnahmen wurden einerseits produzierende KMUs aus dem PA gefragt, aber auch Beratungsfirmen, die von Abläufen von Kunden berichtet haben. Zudem wurden auch die groben Wartungsabläufe im Defence-Bereich einbezogen. Im Fokus standen dabei die Informationsquellen zur Beschreibung der Wartung, das Wartungspersonal selbst und auch die Wartungsumgebungen.

Wartungsinformationsquellen: Die Informationsquellen sind, je nach Branche, sehr unterschiedlich und teilen sich in implizite und explizite Quellen auf. Stark reglementierte Branchen, wie Aviation oder Defence, verfügen über zertifizierte Abläufe und Qualifikationen, die fixen, feingranular definierten Abläufen folgen und klare Verantwortlichkeiten definieren. Dem gegenüber stehen weniger reglementierte Abläufe, wie die Wartung von Bearbeitungsmaschinen. Dabei geben Betriebsanleitungen vor, welche Arbeiten selbst vorgenommen werden können und in welchen Bereichen Experten zugezogen werden sollen. Die Betriebsanleitungen sind dabei meist mit eigenen unternehmensinternen Erfahrungswerten angereichert, die entweder explizit (bspw. handschriftlich) hinzugefügt werden oder implizit im Wissensschatz von Mitarbeitern vorhanden sind. Zudem ist das Anlegen von eigenen Dokumenten mit spezifischen Problemlösungen üblich (bspw. eigene Excel-Tabellen mit beschriebenen Störung-Lösungs-Paaren). Zuletzt sind unternehmensintern konstruierte Maschinen und Apparaturen zu nennen, die über selbst verfasste Wartungsanleitungen in Textform und Konstruktionspläne verfügen.

Wartungspersonal: Je nach Umfang, Komplexität und potenziellen Gefahren bei Wartungsarbeiten kommt unterschiedliches Wartungspersonal zum Einsatz. Standardisierte, stark reglementierte Abläufe setzen meist

speziell qualifiziertes Personal voraus, das zuvor in speziellen Schulungen angelernt wurde. Hierbei wird teilweise auf Dienstleistungen Dritter zurückgegriffen, sodass es im Reparaturfall zu deutlichen Verzögerungen kommen kann. In diesen Fällen liegt das Wissen explizit in Form von gedruckten oder digitalen Dokumenten vor und ist umfassend beschrieben. Weniger komplexe Wartungen werden insb. in KMUs durch Arbeitskräfte der jeweiligen Bereiche vorgenommen, vornehmlich durch erfahrene Mitarbeiter, die schon jahrelang mit entsprechendem Gerät arbeiten. Ist ein solcher Mitarbeiter nicht verfügbar, bspw. durch abweichende Schichtzeiten oder Arbeitgeberwechsel kann dies eine starke Verzögerungen im Produktionsablauf bedeuten. Hierbei existieren nicht immer klar geregelte Abläufe, sodass Abstimmungen teilweise ad hoc telefonisch getätigt werden.

Wartungsumgebung: Zumeist findet die Wartung innerhalb der Räumlichkeiten der einzelnen Unternehmen statt, teilweise werden aber auch Remote- und Infield-Wartungen durchgeführt. Die Umgebungen unterliegen teilweise industrieller Lärm- oder Staubbelastung oder auch speziellen klimatischen Bedingungen (bspw. Kühlhallen). Dementsprechend existieren fix positionierte und mobile Wartungssysteme in verschiedenen Robustheitsgraden und Wartungsmethoden. Dies ist besonders hinsichtlich der nutzbaren Geräte sowie nutzbarer Hardware relevant.

Aus den Beobachtungen geht deutlich die Notwendigkeit eines modularen Systems hervor, das situativ je nach Anforderung konfigurierbar sein muss. Zudem ist ein übergreifender Wissensspeicher notwendig, der sowohl bestehendes Wissen aufnehmen und verknüpfen kann, als auch neues Wissen hinzufügen kann. Zudem muss eine hochspezialisierte Wissensmodellierung möglich sein, um detaillierte Informationen abrufen zu können. Für eine umfassende Wartungsunterstützung ist es dabei zusätzlich notwendig, neben den eigentlichen Wartungsprozessen und -prozessschritten auch Informationen zu Bauteilen und Arbeitsprozessen der technischen Geräte zu verknüpfen, um besser „Warum“-Fragen beantworten zu können. Ein generalisierter Systemansatz sorgt dabei für die Möglichkeit des domänenübergreifenden Einsatzes.

Konkrete Ableitung technischer Anforderungen

Anhand der Analyse der Wartungsprozesse und daraus hervorgegangenen Anforderungen wurden die technischen Systemanforderungen abgeleitet. Im Vordergrund stand dabei die Notwendigkeit eines modularen, flexiblen digitalen Assistenzsystems, das für verschiedene Domänen mittels anpassbarer Wissensdatenbasis geeignet ist. Dabei wird ein Model-Driven-Ansatz gewählt, bei dem sowohl die Dialoggestaltung, als auch das hinterlegte Wissen homogen in einer Graphdatenstruktur hinterlegt wird. Wie in AP1 recherchiert wurde, eignet sich dabei OWL als Modellierungssprache durch ihre klare Struktur. Zudem ist OWL etabliert, sodass einerseits diverse Frameworks zur Auswertung bereitstehen (vgl. *OWL API* für Java (University of Manchester 2025), *rdflib* für python (RDFLib Team 2025)) und auch LLMs das Format verarbeiten können.

Um maximale Konfigurierbarkeit zu ermöglichen, sind Front- und Backend klar getrennt. Das Backend stellt dabei Informationen und Funktionalität per fest definierten Schnittstellen über Webschnittstellen bereit. Zur Gewährleistung des Datenschutzes muss das Gesamtsystem dabei nicht zwangsläufig für das Internet geöffnet werden, sondern kann komplett innerhalb der Firmennetzwerke genutzt werden. Es handelt sich um eine klassische Client-Server-Architektur, bei der ein Server Informationen zentralisiert bereitstellt und modifiziert und eine beliebige Anzahl Clients Informationen abfragen und eintragen können (siehe Abb. 4).

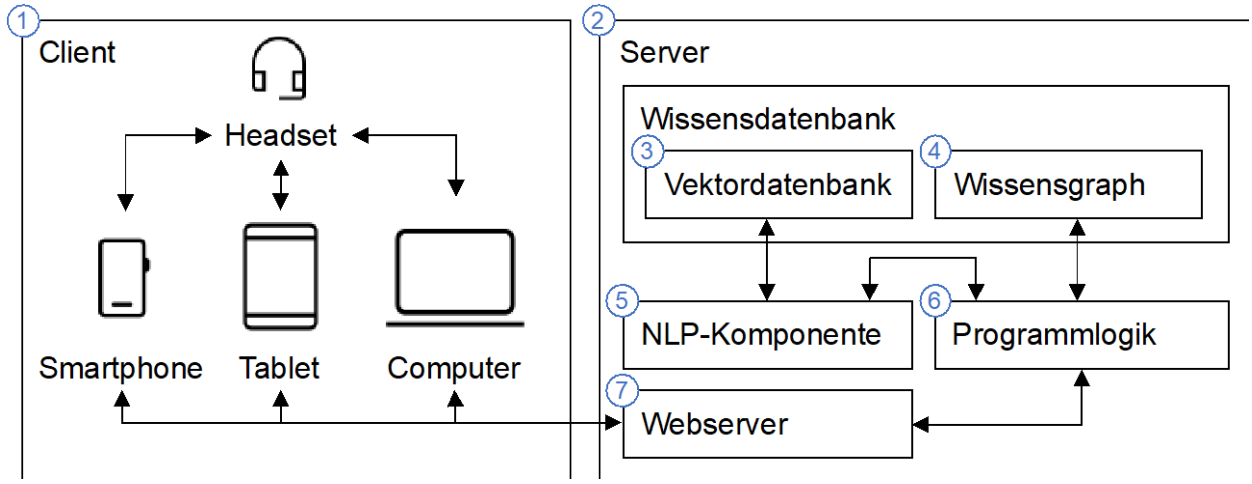

Abbildung 4 Client-Server-Architektur

Das Gesamtsystem lässt sich in 7 Bestandteile untergliedern, die verschiedene Verantwortlichkeiten besitzen und mittels fest definierter Schnittstellen gekoppelt sind:

1 Client: Je nach Einsatzbereich eignen sich verschiedene stationäre und mobile Endgeräte. Diese können entweder mittels Maus und Tastatur, Toucheingabe mittels Chat-Ansicht oder per Sprachinteraktion bedient werden. Das Frontend wird als *Angular*-Applikation bereitgestellt und kann per Browser im Web, oder aber als kompilierte App installiert werden. So wird Betriebssystem- und Geräteunabhängigkeit gewährleistet und die Entwicklung vereinheitlicht. Es kann eine beliebige Anzahl von Clientgeräten auf den Server zugreifen.

2 Server: Der Server stellt das Backend des Gesamtsystems dar und vereint einzelne Komponenten zum Speichern, Modifizieren und Bereitstellen von Informationen.

3 Vektordatenbank: Für das semantische Retrieval von kontextrelevanten Informationen aus dem Wissensgraphen wird die Vektordatenbank aufgebaut. Immer wenn es zu Änderungen im Wissensgraphen kommt, werden erneut Sentence-Embeddings mittels des SBERT-Modells für jegliche Beziehungen und Attribute des OWL-Graphen automatisch erstellt. Dafür werden die Triple des Wissensgraphen in Sätze ausformuliert (siehe Abb. 5).

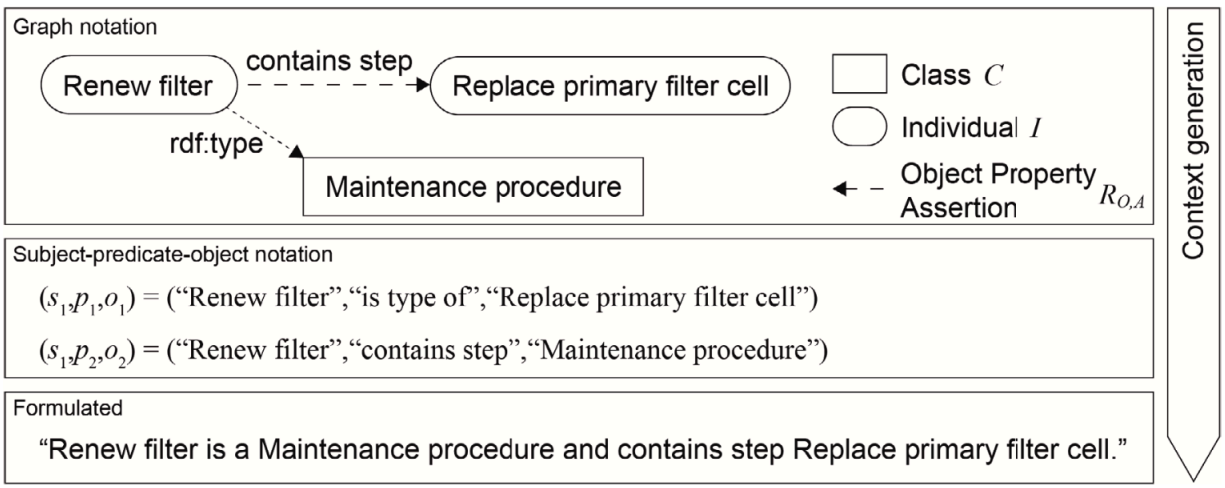

Abbildung 5 Verschiedene Abbildungsmöglichkeiten von Informationen (aus Ludwig, Schmidt und Kühn 2025)

4 Wissensgraph: Die OWL-basierte Ontologie dient als Wissensgrundgerüst zur Beschreibung von Wartungsprozessen, zu wartenden Objekten und dafür benötigten Werkzeugen. Zum systematischen Aufbau einer Ontologie existieren verschiedene Vorgehensweisen wie *Ontology 101* oder *METHONTOLOGY* (Fernández, Gómez-Pérez und Juristo 1997; F. Noy und McGuinness 2001). Dabei wird der Vorteil der Wiederverwendbarkeit von *Upper Ontologies* hervorgehoben, die bereits grundsätzliche Gegebenheiten, wie technische Zusammenhänge und/oder Prozesse, definieren, auf denen domänenspezifisch aufgebaut werden kann (vgl. CDM-Core (Mazzola, et al. 2016)). Der Protégé-Editor bietet eine graphische Oberfläche zur händischen Modellierung und Verknüpfung bestehender Ontologien, ohne dass Programmierkenntnisse benötigt werden (National Institute of General Medical Sciences 2025). Das System kann beliebige Ontologiestrukturen verarbeiten, während des Projektes hat sich eine Aufteilung in die vielgenutzte Produkt-Prozess-Ressource-Aufteilung bewährt (Cutting-Decelle, et al. 2007), wobei die Wartungsgegenstände als Produkte, die Wartungsroutinen als Prozesse und die Werkzeuge und Verbrauchsmittel als Ressourcen definiert sind. Für große Datenmengen kann ein *Jena Fuseki Triple*-Store genutzt werden.

5 NLP-Komponente: Natural Language Processing (NLP) beschreibt die computergestützte Verarbeitung von natürlicher Sprache. In der NLP-Komponente sind alle dafür notwendigen Funktionen gebündelt. Diese können über Webschnittstellen abgerufen werden und sind in *Python* implementiert. Die Funktionalität umfasst neben KI-gestützten Methoden, wie LLMs, Embedding Modellen und Klassifizierungsmodellen auch klassische Matching-Methoden, wie *Levenshtein*- und *Jacchard* Distanz, und probabilistische *Stemming*-Methoden. Dabei sind alle Methoden multilingual konfigurierbar (vgl. *spaCy* Framwork (spaCy 2025)).

6 Programmlogik: Die Programmlogik verbindet alle Komponenten und organisiert die internen Datenströme. Das *Java*-basierte Programm steuert die Dateneingabe, -modifikation und -ausgabe.

7 Webserver: Der Webserver dient zur Kommunikation mit den Client-Geräten. Er stellt das Frontend als *Angular*-Anwendung bereit (Google 2025). Die Angular-Anwendung ist kompatibel für verschiedene Browser und responsive Bildschirmgrößen und ermöglicht die rein sprachbasierte, textbasierte oder hybride Interaktion. Zudem können Bilder, Videos und Audio-Clips ausgegeben werden (um bspw. Wartungsanleitungen multimedial zu gestalten).

Im Projektverlauf wurden zwei weitere Komponenten zum Einbezug visueller Informationen und zur automatischen Generierung von Wissensgraphen hinzugefügt, diese sind näher in Abschnitt 2.4 (AP 4) beschrieben.

Neben der Eigenentwicklung wurde zum besseren Vergleich ein *Microsoft Copilot Studio*-basiertes System für Wartungszwecke implementiert. Dies bietet insbesondere den Vorteil der Verfügbarkeit und einfachen Nutzbarkeit, kann jedoch nicht mehrere Dokumente verknüpfen und ist nicht lokal ausführbar, sodass der Datenschutz geprüft werden muss. Tabelle 3 stellt Vor- und Nachteile beider Ansätze gegenüber.

Tabelle 3 Vergleich OWL-basierter und MS Copilot Studio-basierter digitaler Assistent

OWL-basiert	MS Copilot Studio
+ Hohe Transparenz	+ Geringer Einrichtungsaufwand
+ Geringe Halluzinationen bei der Nutzung mit LLMs	+ Keine Einarbeitung in Modellierungssprache notwendig
+ Lokale Ausführbarkeit	- Nutzung externer Dienste

+ Einbezug verschiedener Datenquellen	- Kombination mehrerer Dokumente nicht möglich
- händische Modellierung von OWL komplex und zeitaufwändig	- Keine multimodale Ausgabe

Aufsetzen der Interaktionsdatenbank

Zur Nutzung der Vorteile des Wissensgraphens ist die Interaktion ebenfalls in OWL modelliert (siehe Abb. 6). Dafür wurde ein eigenes Vokabular angelegt, mit dem neben der Informationsabfrage weitere Aktionen („Intents“) definiert werden können. Jeder Intent besteht aus mindestens einem „Slot“, der jeweils einen Eingabeparameter vorgibt. Dabei werden Eingabetypen direkt mit inhaltlichen Elementen der Ontologie verknüpft.

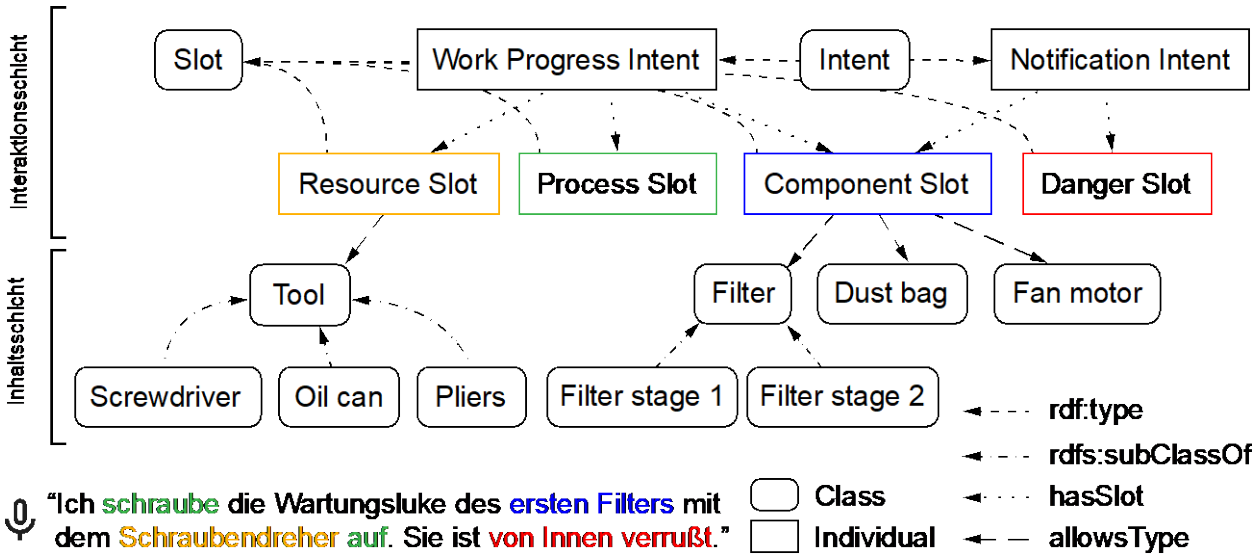

Abbildung 6 Beispielkonfiguration der Interaktionsdatenbank

Im Beispiel wurden zwei Intents zur Dokumentation der Wartungsdurchführung modelliert: *Work Progress Intent* und *Notification Intent*. Der Work Progress Intent nimmt Daten zum durchgeführten Arbeitsprozess (*Process Slot*) am betroffenen Bauteil (*Component Slot*) mit entsprechenden Hilfsmitteln (*Resource Slot*) auf. Ersichtlich wird auch die Verknüpfung zwischen Interaktions- und Inhaltsschicht, indem Stammdaten direkt mit den Slots verknüpft wird, sodass Eingabetypen festgelegt werden. So können ungenaue Informationen durch Rückfragen seitens des digitalen Assistenten konkretisiert werden (bspw. „Ich kontrolliere den Filter“ – „Welchen Filter? Filterstufe 1 oder 2?“). Des Weiteren können Slots auch freie Texte aufnehmen, bspw. bei Gefahrenmeldungen, die vorher unbekannt waren und so nicht im Wissensgraph hinterlegt sind. Hierfür werden die eingegebenen Informationen anhand semantischer Ähnlichkeit zu den Freitext-Slots ausgewertet und entsprechend zugeordnet.

Um die Interaktionsgeschwindigkeit möglichst natürlich und zeiteffizient zu gestalten, ist eine *Multi-Intent-Detection* implementiert, die die Erkennung mehrerer Intents (und Slots) erlaubt. So können mehrere Informationen mit einer einzelnen Eingabe getätigt werden, wie in einem natürlichen Gespräch. Dabei ist die Erkennung der Intents im System konfigurierbar und basiert auf einem Methoden-Ensemble aus einem vortrainierten *zero-shot multi-label* KI-Klassifikationsmodell (Schmidt, Ludwig und Kühn 2025), das die Ähnlichkeit

von Labels der OWL Intent-Individuals und der Nutzereingabe berechnet, der Menge potenziell gefüllter Slots (je höher die Menge, desto wahrscheinlicher der Intent) und der Menge an ähnlichen Intents, gemessen anhand von direkt übereinstimmenden Informationen und vektor-basierter semantischer Ähnlichkeit.

2.3 Modellierung Wissensdatenstruktur zur Störungsklassifizierung und -behebung (AP 3)

Datenextraktion

Zur inhaltlichen Modellierung der Wissensdatenstruktur wurden verschiedene Wartungsdokumente analysiert (siehe Abb. 7). Neben klassischen, vom Hersteller zur Verfügung gestellten Dokumenten gibt es dabei oftmals firmenspezifische Informationen, die entweder handschriftlich oder in digitalen Dokumenten ergänzende Hinweise und Best-Practises enthalten.

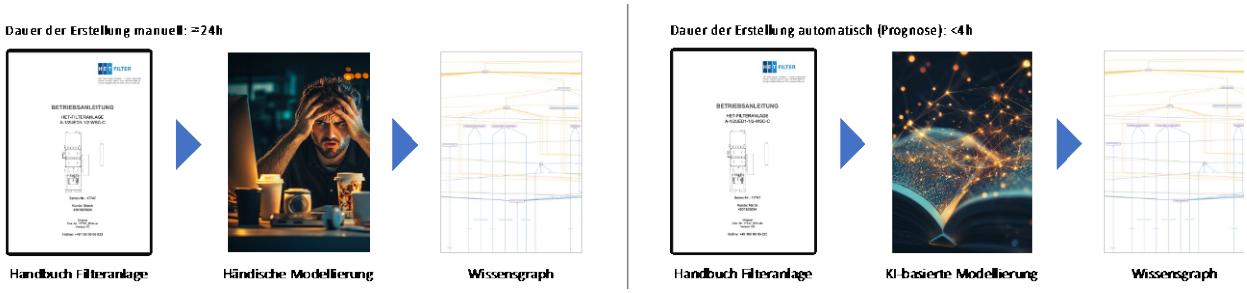

Abbildung 7 OWL-Modellierung anhand von Bedienungsanleitungen

Für weitere Untersuchungszwecke wurde u.a. die Bedienungs- und Wartungsanleitung einer Filtermaschine modelliert (siehe Abb. 8). Dabei hat sich gute Übersicht und Erweiterbarkeit, aber auch ein hoher Arbeitsaufwand und die Komplexität bei der Modellierung von OWL-Graphen gezeigt. Hinsichtlich eines geeinten Terminologie-Managements zeigt OWL Vorteile, da Synonyme und Hierarchien definiert werden können. Im Gegensatz zu textuellen Bedienungsanleitungen wird zudem auf redundante Texte verzichtet (wie bspw. die mehrfache Ausformulierung von Sicherheitshinweisen), da diese in der Graphstruktur mehrfach verlinkt werden können. Wirkungszusammenhänge zwischen Prozessen, technischen Komponenten und Fehlermeldungen können übersichtlich visualisiert und erweitert werden. Zudem wurde eine *Upper Ontology* zur Beschreibung von Maintenance-Aktivitäten einbezogen, die elementare Wartungsaktivitäten beschreibt (vgl. Mazzola, et al. 2016). Die Modellierung selbst wurde anhand des *Ontology 101*-Leitfadens in *WebProtégé* vorgenommen (F. Noy und McGuinness 2001).

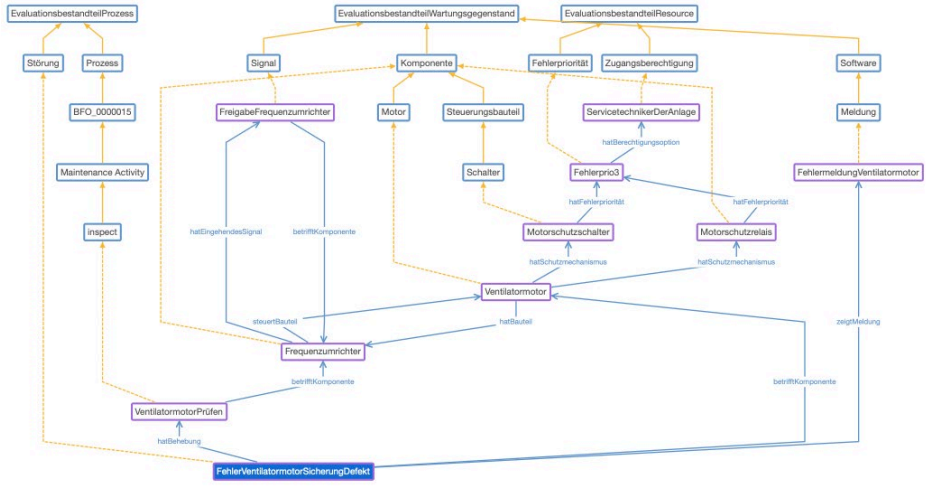

Abbildung 8 Ausschnitt OWL-Wissensgraph einer Filtermaschine; ausgehend vom Fehler „Ventilatormotor Sicherung Defekt“

Der hohe Arbeitsaufwand zur Informationsmodellierung motiviert die Schaffung von Automatismen zur unterstützenden Modellierung. Dieses Untersuchungsgebiet wird in der Wissenschaft als „Knowlegde Graph Generation“ oder „Ontology Learning“ bezeichnet und existiert seit Jahrzehnten (Khadir, Aliane und Guessoum 2021). Durch deutliche Leistungssprünge von KI-Sprachmodellen und damit der Verarbeitbarkeit von komplexen, ungeordneten Informationen ist die Nutzung von LLMs zur Schaffung eines Text-to-OWL-Prozesses vielversprechend. Detailliertere Informationen sind im Abschnitt „Aufsetzen der Wissensdatenbank“ zu finden.

Anwendung von KI-Klassifikationsverfahren

Ergänzend zum ursprünglichen Projektplan kommen neben KI-Sprachmodellen zur Bildung von Wort- und Satzvektoren (wie BERT oder SBERT) dem Einsatz von LLMs eine zentrale Bedeutung zu. Zur Abfrage von Informationen aus dem Wissensgraphen kommt dabei ein *Retrieval Augmented Generation* (RAG)-Verfahren zum Einsatz. Dabei handelt es sich um eine Methode, bei der im ersten Schritt anhand der Nutzerangabe eine Datenbasis nach relevanten Informationen durchsucht wird (Retrieval-Schritt) und im zweiten Schritt mitsamt der ursprünglichen Nutzereingabe in einem Prompt formuliert wird und in ein LLM eingegeben wird (Generation-Schritt). Auf diese Weise wird dem LLM ein zu nutzender Kontext anhand der Datenbasis vorgegeben. Vorteile sind dabei, dass spezifisches (Fimen-)Wissen durch LLMs verarbeitet werden kann, ohne dass ein zeit-, daten- und hardwareaufwändiges Training von LLMs stattfinden muss (Gao, et al. 2024).

OWL-basiertes RAG-System

Zur möglichst präzisen Informationsextraktion aus der gegebenen OWL-Ontologie wurde dafür im Projekt ein neues RAG-System konzipiert und evaluiert, dass im Retrieval-Schritt verschiedene Ähnlichkeitsverfahren kombiniert und mittels Nachbarbeziehungen im Graphen („Hops“) einen bestmöglichen Kontext für den Generation-Schritt bereitstellt (siehe Abb. 9). Das OWL-basierte RAG-System ist besonders bei Nutzung von weniger leistungsfähigen, lokal lauffähigen LLMs den bisherigen RAG-Systemen überlegen, da es auf die Generierung möglichst kleiner Kontextgrößen ausgelegt ist, sodass ausschließlich die relevantesten Informationen zur Beantwortung der Nutzerfrage einbezogen werden.

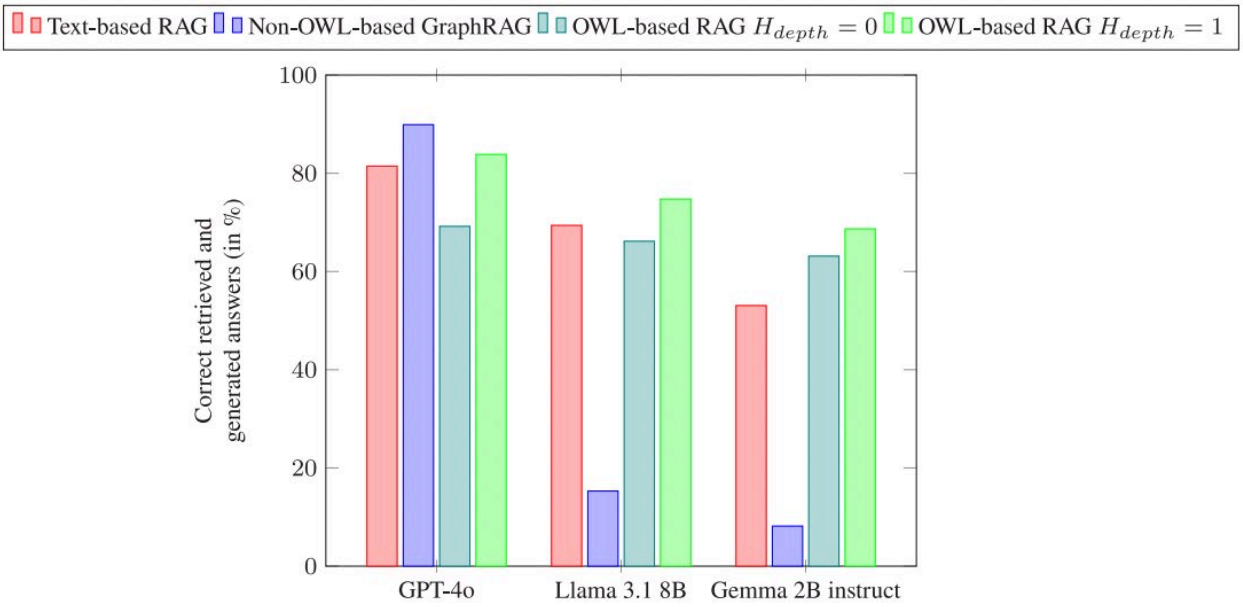

Abbildung 9 Leistungsfähigkeit verschiedener RAG-Systeme anhand von jeweils 33 Fragen zu Produkt, Ressource und Prozess im Wartungskontext (aus Ludwig, Schmidt und Kühn 2025)

Im Retrieval-Schritt kommen drei verschiedene Verfahren zum Einsatz, die in der technischen Domäne unterschiedliche Stärken besitzen (siehe Tabelle 4):

Tabelle 4 Retrievalverfahren für OWL-basiertes RAG-Verfahren

Explizites Retrieval	
Beschreibung	Dieses Verfahren nutzt die Labels und Attribute der OWL-Entitäten und verschiedene String-Matching-Verfahren mit der Nutzereingabe zur Herausarbeitung relevanten Elementen. Es sind keinerlei zusätzliche Datenbanksysteme für die Abfrage notwendig.
Methoden	Tokeninzing, Stemming, Jacchard Distance, Levenshtein Distance
Vorteile	+ Hohe Transparenz + Präzise bei technischem Vokabular (wie Bauteile-Ids)
Nachteile	- Umfangreiche Synonym-Definition notwendig - Inhaltliche Bedeutung ggf. nicht abgebildet
Nutzer	Experten, die genau das technische Vokabular kennen

Vektor-basiertes Retrieval	
Beschreibung	Dieses Verfahren wandelt die Nutzereingabe und ausformulierte OWL-Triples in hochdimensionale Vektoren um. Für eine zügige Interaktion werden dabei alle vektorisierten OWL-Informationen initial in einer Vektordatenbank gecached. Mittels cos-Distanz wird die Ähnlichkeit zwischen der Nutzereingabe und den Vektoren ermittelt.

Methoden	Word- und Sentence Embeddings, Vektordatenbank, cos-Distanz
Vorteile	+ ermöglicht unscharfe Suche + hohe Toleranz gegenüber ungenauen Eingaben + berücksichtigt Kontext der Eingabe → sinnvolles Homonym-Handling
Nachteile	- vortrainierte Embedding-Modelle z.T. fehleranfällig bei technischen Bezeichnungen
Nutzer	Unerfahrene, die Gegebenheiten nicht konkret benennen können

Phonetisches Retrieval	
Beschreibung	Dieses Verfahren nutzt phonetische Algorithmen zur Steigerung der Robustheit bei ungenau transkribierten Speech-to-Text-Ergebnissen. Die Verfahren sind dabei abhängig von der Eingabesprache, im System sind Methoden für Englisch und Deutsch hinterlegt.
Methoden	Soundex, Kölner Phonetik
Vorteile	+ Toleranz gegenüber ungenauer Spracherkennung
Nachteile	- Gefahr von falsch erkannten Entitäten, die ähnlich klingen
Nutzer	Nicht beschränkt

Je nach Nutzer, Einsatzdomäne und genutzten Clientgeräten können die Verfahren beliebig kombiniert werden. Durch die starke Formalisierung der OWL bietet sich ein Element-spezifischer Vergleich der Leistungsfähigkeit der einzelnen Retrievalmethoden an. So können feingranular verschiedene Schwellwerte für die semantische Mindestähnlichkeit von OWL-Entitäten (*Klassen* und *Individuen*), deren Attributen (*Datatype properties*) und Relationen (*Object properties*) festgelegt werden. Zudem wird die hierarchische ontologische Struktur berücksichtigt, die explizit weitere relevante Kandidaten im Retrievalprozess aufdeckt. Zur Evaluation wurden 1.380 Beispielfragen aus der Wartungsdomäne gebildet. Die Verfahren wurden hinsichtlich des F_1- und F_2-Scores bewertet (siehe Formel 1). Während F_1 den harmonischen Mittelwert aus Präzision und Recall bildet, ist die Gewichtung des Recall-Wertes in F_2 größer, sodass der Umgang der LLMs mit falschpositiven Werten hervorgehoben wird. Die besten Ergebnisse hat das explizite Retrievalverfahren für die Labels der OWL-Entitäten erbracht, dahinter das explizite Retrieval für Attribute und das Vektor-basierte Verfahren für die Entitäten.

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

Formel 1 F_β-Score

Im zweiten Schritt wurden die Elementtyp-spezifischen Retrieval-Verfahren zusammengefügt. Dafür wurden drei Kombinationsstrategien evaluiert (siehe Tabelle 5):

Tabelle 5 Strategien zur Kombination verschiedener Retrievalverfahren für OWL-basiertes RAG-Verfahren

First match	
Beschreibung	Alle Retrievalmethoden werden sequentiell ausgeführt. Sobald eine Methode mindestens einen Kandidaten findet, werden alle weiteren Retrievalmethoden nicht ausgeführt.
Intention	Priorisierung der am besten bewerteten Retrievalverfahren. Findet das beste Verfahren keine Ergebnisse greift das System auf das zweitbeste Verfahren zurück usw..

Union	
Beschreibung	Die als relevant eingestuften Kandidaten aller genutzten Retrievalverfahren werden addiert. Duplikate werden ignoriert.
Intention	Vereine die Vorteile der verschiedenen Retrievalverfahren.

Majority	
Beschreibung	Gleich zu Union, beschränkt Ergebnismenge relevanter Kandidaten auf die Kandidaten, die am häufigsten von verschiedenen Retrievalverfahren gefunden wurden.
Intention	Gleich Unsicherheit einzelner Methoden durch Übereinstimmung verschiedener Methoden aus.

Union- und *Majority*-Strategien führen zu besseren Ergebnissen als die Nutzung einzelner Retrieval-Methoden. Majority steigert den F_1-Score von 0,45 auf 0,61 und den F_2-Score von 0,51 auf 0,7.

Eine umfangreiche nutzergetriebene Evaluation zum Vergleich verschiedener LLMs im Generation-Schritt ist in Abschnitt 2.5 beschrieben. Das System wurde im Q1-Journal *Computers in Industry* im Jahr 2025 veröffentlicht (siehe Abschnitt 2.6).

Dokument-basiertes RAG-System

Wie bereits in Abschnitt 2.2 erwähnt, wurde neben dem OWL-basierten RAG-System ein zweites, Text-basiertes RAG-System mit MS Copilot Studio getestet. Das RAG-System nutzt Chat-GPT als LLM. Bei der Evaluation von Beispielfragen wurden dabei sowohl Abdeckungsfehler, als auch Halluzinationen festgestellt. Das bedeutet, dass einerseits in der Antwort Argumente fehlen, die in der Originalquelle vorkommen, andererseits Informationen hinzugefügt werden, die nicht im originalen Dokument aufgeführt sind. Die geschieht sowohl bei kurzen Anleitungen (20 Seiten), als auch bei umfangreicheren Dokumenten (182 Seiten). Zudem stellt das System bei Unklarheiten bei der Nutzereingabe keine Rückfragen, sodass z.T. Informationen zu falschen Bauteilen oder Wartungsprozessen ausgegeben werden. Zur Lösung der Problematik wurde hierfür ein Regel-basiertes System implementiert, das eine manuell geprüfte Initialisierungsphase besitzt, in der anhand des RAG-Verfahrens verschiedene Arbeits- und Behebungsanweisungen extrahiert werden, die durch den Expertennutzer händisch bestätigt, modifiziert oder gelöscht werden können (siehe Abb. 10). Dieses KI-

unterstützte *Human-in-the-Loop*-System wirkt fehlerhaften oder unvollständigen Halluzinationen entgegen, bedeutet aber Mehraufwand während der Initialisierung. Bei der Nutzung des Chatbots werden zunächst bekannte, validierte Arbeitsanweisungen durchsucht. Beantworten diese die Anfrage nicht, kann mittels RAG-Verfahren eine Lösung anhand des hinterlegten Dokuments gegeben werden, mit der Gefahr von Abdeckungsfehlern oder Halluzination. Similar dazu können Störungsursachen und Behebungsanweisungen ausgegeben werden.

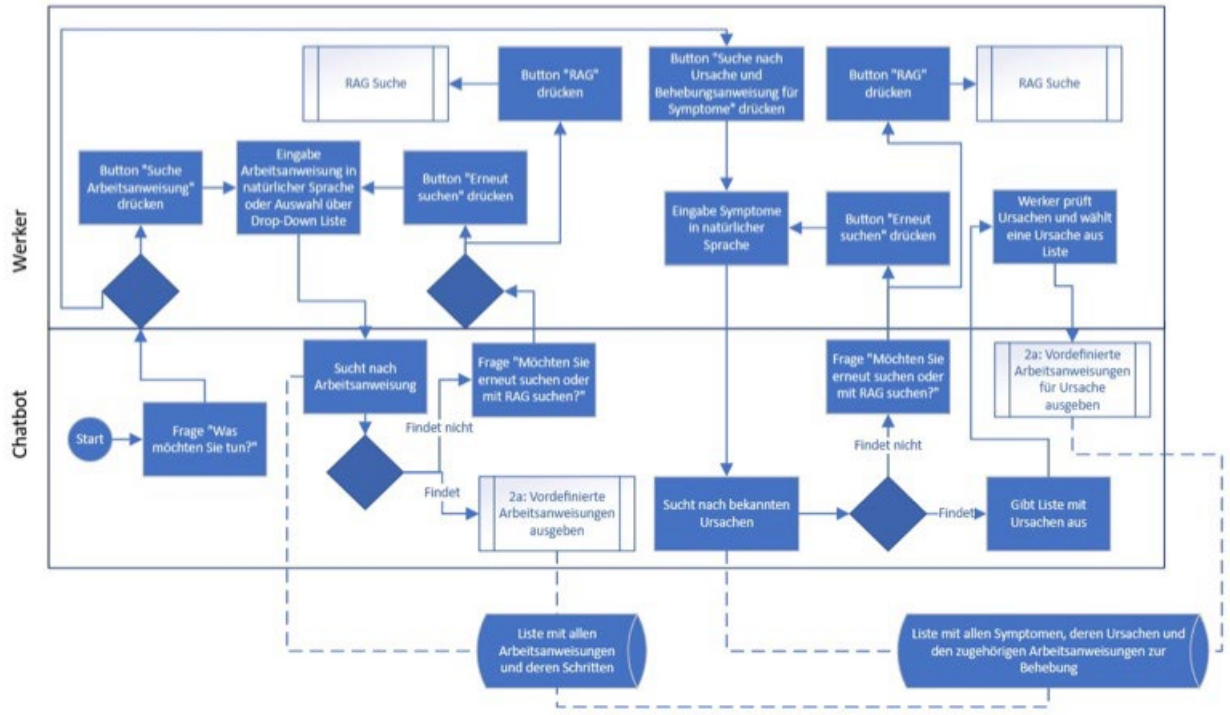

Abbildung 10 Aktivitätsdiagramm zur Funktionsweise des Dokument-basierten Chatbots

Aufsetzen der Wissensdatenbank

Hohe Transparenz, gute Erweiterbarkeit sowie gute Retrieval-Ergebnisse sind Vorteile bei der Nutzung von OWL-Strukturen. Dem gegenüber steht der vergleichsweise hohe Aufwand bei der händischen Erstellung der Graphen und das benötigte Modellierungswissen. Um dieses Problem zu lösen wurde ein LLM-gestützter Text-to-OWL-Ansatz gewählt. Ziel war es dabei, LLMs zu finetunen und zur OWL-Modellierung anhand von Textdaten zu befähigen. Das *Unsloth AI*-Framework hilft dabei, nur einen geringen Teil der Gewichte in den Modellen anzupassen und somit Trainingszeit und -hardwareaufwand zu reduzieren (unsloth 2025).

Verschiedene Voruntersuchungen zur Modellierung von OWL anhand von Text zeigten, dass sich der Detailgrad der Modellierung indirekt proportional zur Textmenge verhält, umfangreiche Eingabetexte führen zudem zu unvollständiger Datenverarbeitung („Lost-in-the-middle"-Phänomen (F. Liu, et al. 2023)). Um dem entgegenzuwirken und einen hohen Detailierungsgrad zu garantieren, wurde der zu verarbeitende Text (wie bspw. eine Wartungsanleitung) in einzelne überlappende Textbestandteile zerlegt, einzeln in OWL-Graphen umwandelt und abschließend alle so generierten OWL-Graphen zu einem Graphen zusammenfügt.

Im ersten Schritt mussten die Trainingsdaten erstellt werden (siehe Abb. 11). Hierbei sind Q&A-Datensätze üblich (*Instruction Tuning*), die über tausende Frage-Antwort-Paare verfügen. Da kein Datensatz zur textbasierten OWL-Modellierung existierte, war die Erstellung eines neuen Trainingsdatensatzes notwendig. Die

Grundidee ist die Ausformulierung von OWL-Triples in Text. Da diese in Subjekt-Predikat-Objekt-Notation vorliegen, reicht dafür eine einfache Aneinanderreihung der Bestandteile um (nahezu) natürliche Sätze zu erhalten. So ist die Bildung der Eingabe, wie „Bilde aus dem folgenden Text eine OWL-Ontologie: <ausformulierte OWL-Ontologie>“, möglich, die als entsprechende Ausgabe die formalisierte OWL-Ontologie beinhaltet. *Wikidata* bietet als frei zugänglicher, händisch modellierter OWL-Triplestore eine ideale Datenquelle.

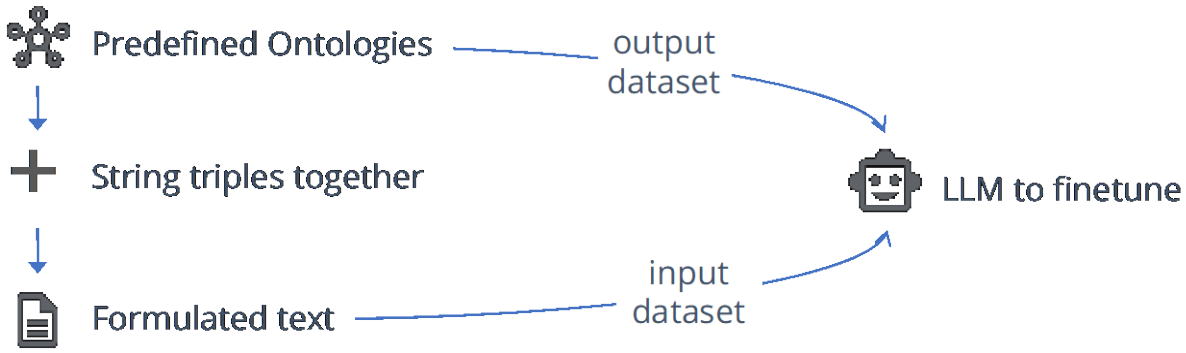

Abbildung 11 Schematische Darstellung des Datengenerierungsprozesses für das LLM Finetunting

Die Pipeline zur Trainingsdatengeneriung besteht aus 6 Schritten (siehe Abb. 12):

1 Datenextrahierung: Um Graphen aus Wikidata zu erstellen, wurden mittels *SPARQL*-Abfragen randomisierte Individuen, deren Typen, benachbarte Individuen (und deren Relationen) sowie Attribute extrahiert. Da die Wikidata-Query-Schnittstelle anfragenlimitiert ist, wurde für den Datensammlungsprozess ein Python-Programm implementiert, das die Abfrage übernimmt, Duplikate vermeidet und auf mehreren Rechnerinstanzen parallel ausgeführt werden kann. Diese Informationen wurden als einzelne Graphen abgespeichert und schlussendlich zusammengeführt.

2 OWL Formatanpassung: Während das OWL-basierte Retrievalsystem mit *OWL DL* arbeitet, ist Wikidata mit *OWL Full* modelliert. Dieses Format erlaubt in der Modellierung mehr Freiheiten, verliert aber die Möglichkeit zum Einsatz von Reasoning-Verfahren und schränkt die Schlussfolgerungen aus hierarchischen Strukturen ein. So wurde im zweiten Schritt der Trainingsdatengenerierung ein randomisierter Subgraph mit zusammenhängenden Elementen ausgewählt und algorithmisch in das OWL DL-Format überführt. Hierfür wurden u.a. Klassen und Individuen in einzelne OWL-Entitäten aufgeteilt, zusätzlich wurde Standard-Vokabular aus dem RDFS- und OWL-Namespace hinzugefügt (wie bspw. *rdfs:subclassof*, *rdfs:subpropertyof*, *owl:sameas*, *owl:differentFrom*, *owl:complementOf*) um gute Modellierungspraktiken zu trainieren. Zudem wurde abgesichert, dass sämtliche Elemente über Labels und kurze Beschreibungsannotationen verfügen.

3 Subgraph-Selektion: Durch verschiedene Konfigurationsparameter kann der Umfang und Inhalt der Trainings-OWL-Dateien angepasst werden. So können u.a. Anzahl Entitäten, Relationen und Attribute vorgegeben werden. In diesem Schritt wurde zudem überprüft, dass alle genutzten Elementtypen eine explizite Definition aufweisen (so dass bspw. die Individuen-Typen als explizite Klassen ausdefiniert sind).

4 Aufgabenformulierung: Jain et al. kommen zum Schluss, dass verschiedene Aufgaben in derselben Domäne zu besseren Trainingsergebnissen im FineTuning von LLMs führen (Jain, Maleki und Saade 2024). So wurden in diesem Schritt anhand des Subgraphs Fragen zur Extrahierung einzelner Elementtypen, bestimmter Attribute oder des kompletten OWL-Graphs generiert.

5 Output-Formatierung: OWL kann in verschiedenen Formaten dargestellt werden. Zwei vielgenutzte Formate sind OWL/XML-Format und Turtle-Format. Zweiteres hat den Vorteil der größeren Kompaktheit, sodass

die Kontextlängen bei der LLM-Datenverarbeitung reduziert werden können. Darüber hinaus wurde ein eigenes JSON- und YAML-Format getestet, dass beispielhaft im Prompt vorgegeben wird. Die Idee ist dabei das Ausgabeformat in seiner Komplexität zu reduzieren (um bspw. die Syntax zu vereinfachen) und programmatisch in einen OWL-Graphen zurück zu verwandeln.

6 Zusammenführung: Zuletzt wurden alle gebildeten Frage-Antwort-Paare in einer CSV-Datei kombiniert. Je nach zu trainierendem LLM-Modell werden Role Tags hinzugefügt um zwischen Nutzer und Modell zu unterscheiden.

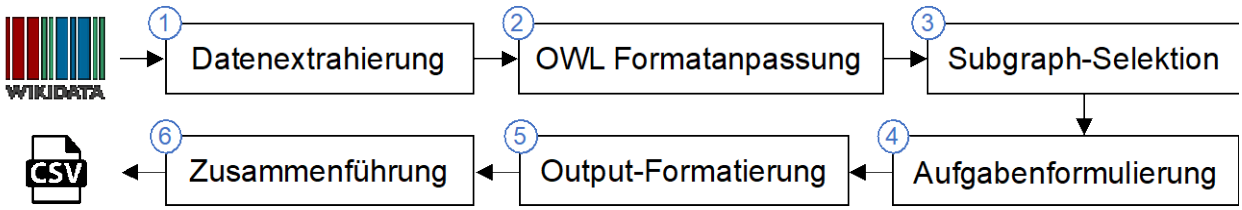

Abbildung 12 Einzelne Schritte der Trainingsdatensatzgenerierung

Nach erfolgreichem FineTuning sieht die Umwandlung von Texten (wie Wartungsanleitungen) zu OWL-Ontologien wie folgt aus (vgl. Abb. 13): Zunächst wird das Dokument eingelesen und dessen Text in überlappende Bestandteile untergliedert. Auf diese Wiese wird die Informationsmenge und damit die Komplexität der LLM-basierten Umwandlung reduziert. Im *Text-to-OWL*-Schritt werden die einzelnen Textbestandteile jeweils in OWL-Ontologien umgewandelt. Dieser Prozess ist parallelisierbar, indem mehrere LLMs ausgeführt werden, sodass auch umfangreichere Dokumente zügig umgewandelt werden können. Die Überlappung sorgt dafür, dass Zusammenhänge im Originaltext über Chunks hinweg bestehen bleiben. Im Merging-Schritt werden zuletzt alle erzeugten Ontologien zu einer Ontologie zusammengesetzt. Die Überlappung der Chunks sorgt hierbei für wiederkehrende Informationen über mehrere Chunks hinweg, sodass das Verknüpfen der einzelnen Ontologien sinnvoll möglich ist.

Das Training wurde auf dem Hochleistungsrechner der TU Dresden auf H100 NVIDIA GPUs durchgeführt. *Unsloth* erlaubt dabei den Aufbau einer generalisieren Trainingspipeline, sodass verschiedene LLMs für das FineTuning genutzt werden können (u.a. *Gemma*- und *Llama*-Modellfamilien) und für die lokale Ausführung mittels *Ollama* bereitgestellt werden (Ollama 2025). Erste Tests beim FineTuning kleiner LLMs (bis 12B Parameter) zeigten vielversprechende Ergebnisse und beinhalteten weniger Fehler, als deutlich aufwändigere Modelle ohne FineTuning. Zur Evaluation der Modellierungsqualität kamen verschiedene Metriken zum Einsatz. Dabei existieren KO-Kriterien, die zwingend einzuhalten sind, und Kriterien, die einen guten Modellierungsstil prüfen (siehe Tabelle 6). Gemessen wurden diese Kriterien an händisch gebauten OWL-Ontologien und deren Beschreibungstexten. Diese liegen in verschiedenen Komplexitätsstufen vor, die sich am Ontology Learning Layer-Cake orientieren.

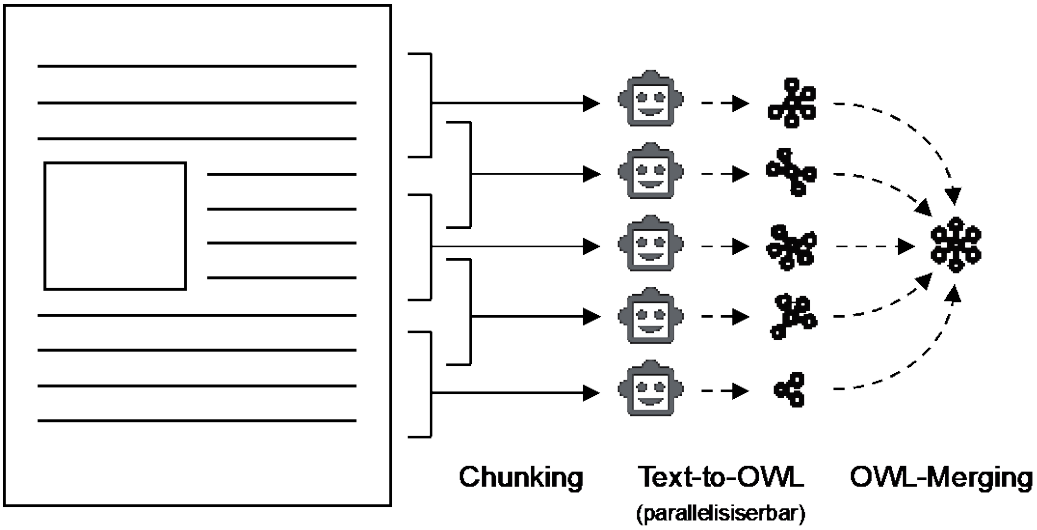

Abbildung 13 Ausführung Text-to-OWL-Umwandlung

Tabelle 6 Validierungskriterien für LLM-basierte OWL-Erzeugung

Kriterium	KO-Kriterium?
Das OWL-Format ist syntaktisch gültig	✓
OWL-Individuen besitzen mind. einen Typen	✓
OWL-Data Property Assertions besitzen eine explizite Definition	✓
OWL-Object-Data Property Assertions besitzen eine explizite Definition	✓
Alle definierten OWL-Data Properties werden mind. einmal genutzt	✓
Alle definierten OWL-Object Properties werden mind. einmal genutzt	✓
IRIs sind für jedes OWL-Element einzigartig	✓
Entitäten besitzen mind. ein Label	X
OWL-Properties verfügen über Domain und Range als Einschränkung	X
IRIs aller Entitäten besitzen konsistenten Schreibstil (bspw. camel case)	X
Definition aller genutzten Namespaces	X

2.4 Entwicklung Gesamtkonzept AVISSBA (AP 4)

Dashboard

Zur Visualisierung der Daten wurde ein Web-basiertes Dashboard implementiert, das Datenexploration und -suche ermöglicht. Für genaue Anfragen können so konkrete Suchstrings gebildet werden oder aber auf eine unscharfe Suche zurückgegriffen werden. Zudem verfügt das Dashboard über eine Chatbot-Anbindung. Das Dashboard greift auf die Schnittstellen des Backends zurück und wurde mit Hilfe des *Angular*-Frameworks realisiert. Durch die Nutzung von *WebGL* als Visualisierungstechnologie können auch Graphen mit hoher Knoten- und Kantenanzahl dargestellt werden. Informationen für alle Knoten und Kanten können durch Dialogfenster angezeigt werden (siehe Abb. 14). Zudem sind verschiedene Layout-Algorithmen zur Anordnung der Graph-Elemente auswählbar.

Sprachbasierte- und Chatoberfläche

Die Chatoberfläche zur dialoghaften Kommunikation mit dem digitalen Assistenten ist ebenfalls über eine Web-basierte Oberfläche realisiert. Während der sprachbasierte Modus für eine rein dialoghafte Interaktion ohne visuelle Interaktion auf dem Bildschirm ausgelegt ist, können in der textbasierten Chatoberfläche auch Fotos, Videos und Audioclips (mittels Nutzung der nativen HTML5-Elemente) ausgegeben werden. Mittels Prompting wird die Ausgabemenge und -art des LLMs je nach Interaktionsart und/oder Mitarbeiterqualifikation gesteuert. So sind sprachbasierte Ausgaben möglichst kurz zu halten, während die visuelle Ausgabe umfangreichere Informationsdarstellung zulässt. Zur Bestimmung der Qualifikation wurden drei Level aufgestellt, denen ebenfalls verschiedene Prompt-Textbausteine zugewiesen sind (siehe Tabelle 7).

Tabelle 7 Qualifikationslevel und deren Auswirkung auf LLM-basierte Informationsausgabe und -formatierung

Level	Beschreibung	Auszugebende Informationsmenge
Basis	Mitarbeiter ohne oder mit nur geringfügiger Erfahrung und/oder technischer Kenntnis. Benötigt umfangreiche Anweisung und Schritt-für-Schritt-Anleitung. Kennt ggf. exakte technische Terminologie nicht und umschreibt diese in Wissensfragen.	↑
Operator	Erfahrener Mitarbeiter mit allgemeinem technischem Grundverständnis und domänenspezifischen Hintergrundwissen. Benötigt kurze, strukturierte Anleitung.	→
Experte	Speziell ausgebildeter Mitarbeiter mit Fokus auf Wartungsarbeiten an speziellen Maschinen. Kennt vollständige Terminologie des Wartungsobjektes und der Wartungsprozesse und benötigt nur sehr kurze, präzise Anweisungen zum Vorgehen.	↓

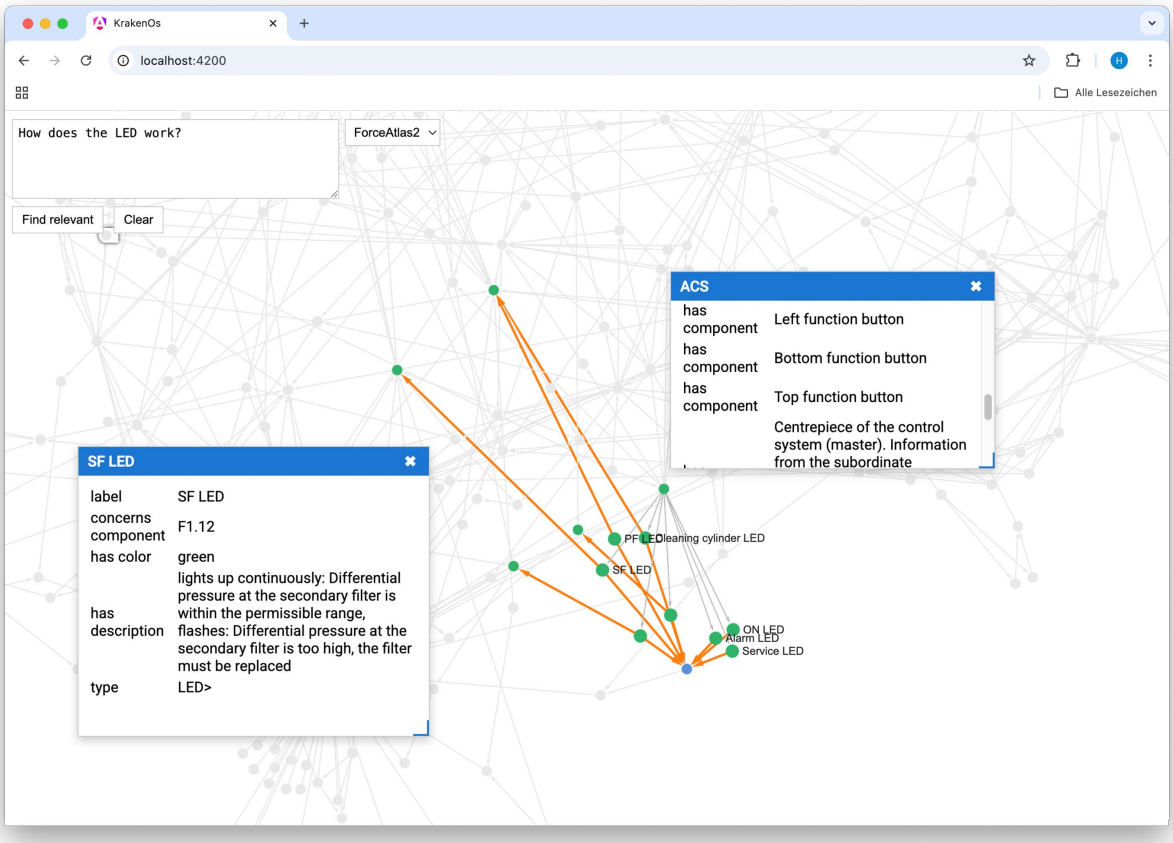

Abbildung 14 Dashboard zur Exploration des Wissensspeichers

Objekterkennung

Zum erweiterten Kontextverständnis des digitalen Assistenten wurde eine Objekterkennungskomponente hinzugefügt (Beispiel siehe Abb. 15). Hierfür wird mindestens eine Kamera vor dem Wartungsobjekt positioniert, sodass der Wartungsmitarbeiter nicht gezwungen ist, den kompletten Wartungskontext (bspw. den realen Aufbau oder Zustand von Maschinenbauteilen) verbal oder per Tastatur einzugeben, sondern eine automatische Auswertung stattfindet. Dafür findet eine Erweiterung des Wissensgraphen statt, sodass die OWL-basierte RAG-Komponente nicht angepasst werden muss, der KI-Assistent aber auf visuelle Informationen zugreifen kann.

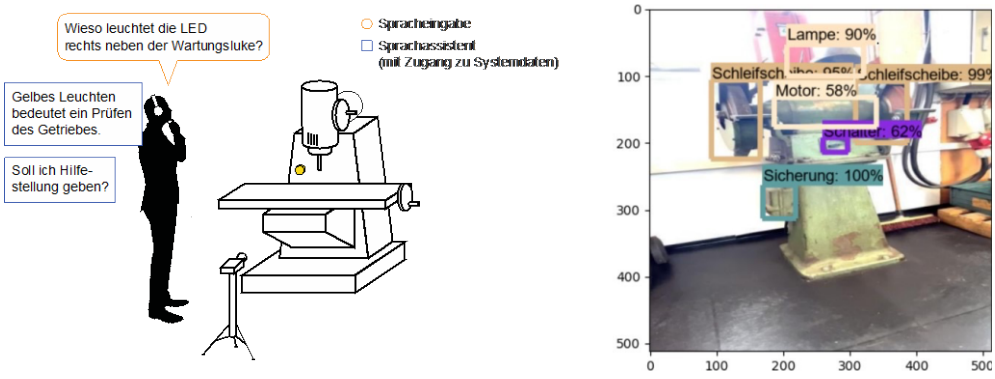

Abbildung 15 Vision und Feldtest zur Verknüpfung visueller Erkennung mit bestehendem Wissen

Dafür wurde ein FineTuning für ein Multi-Object-Detection-Modell durchgeführt, sodass einzelne Bauteile und deren Position an Maschinen erkannt werden können. Zum Einsatz kam eine Single Shot MultiBox Detector-Architektur (SSD) (Liu, et al. 2016). Als Trainingsdaten wurden Fotos und Videos verschiedener Produktionsmaschinen aufgenommen und händisch annotiert, wobei die Videoframes ebenfalls als einzelne Bilder dienen. Erweitert wurde der Datensatz durch verschiedene Verfahren der Data Augmentation, wie bspw. die Anwendung verschiedener Transformationsoperationen oder Farbwertänderungen. Die so erkannten Objekte und deren absolute und/oder relative Position werden automatisch in einem OWL-Graphen modelliert und anschließend mit dem OWL-Graphen verknüpft, der als zentraler Wissensspeicher dient. Ändern sich visuelle Gegebenheiten (bspw. durch Ändern der Kameraperspektive) wird der OWL-Graph entsprechend angepasst. Auf diese Weise werden dem Kontext im RAG-System weitere Informationen zur Verfügung gestellt. Zusätzlich wurde ein Multi-Kamera-Setup integriert (Beispiel siehe Abb. 16), bei dem mehrere Kameras um das Wartungsobjekt platziert werden und anhand von übereinstimmenden Positionen von Nachbarn gleiche Bauteile aus verschiedenen Perspektiven zusammengeführt werden (vgl. Formel 2).

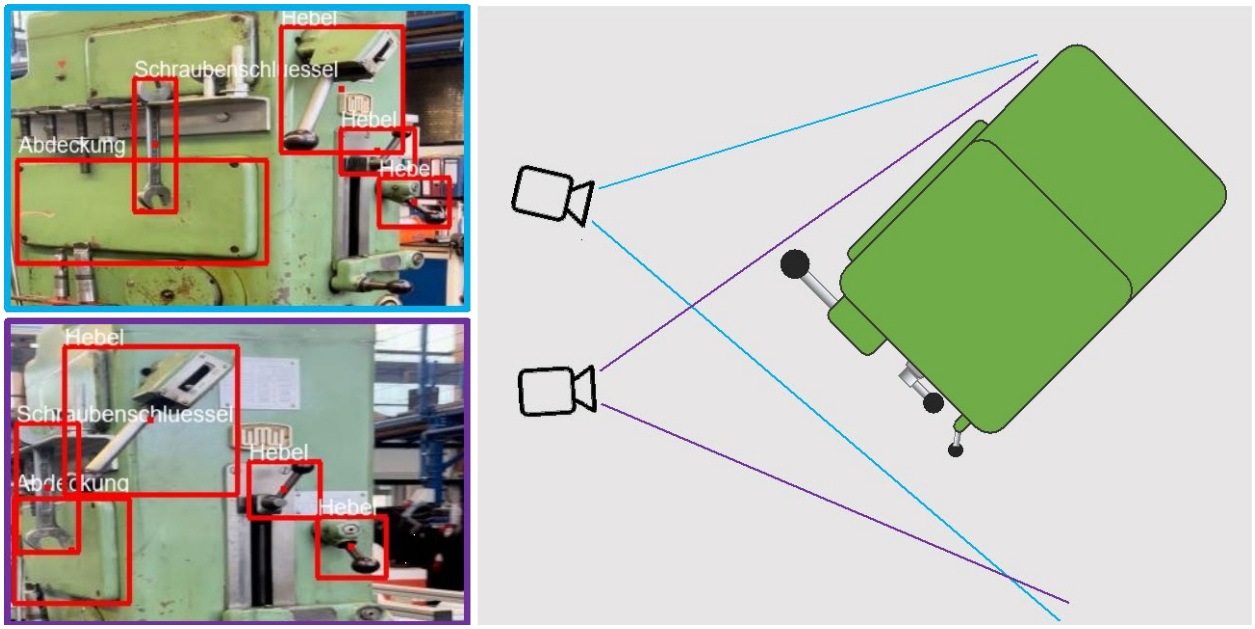

Abbildung 16 Beispielhaftes Multi-Kamera-Setup

Neben der Leistungsfähigkeit der Objekterkennung wurden auch verschiedene OWL-Modellierungsoptionen evaluiert. So wurden alle Kombinationen folgender Modellierungsformen gegenübergestellt:

Absolute Positionsbeschreibung: Allen Bauteilen werden die x- und y-Koordinaten („has xmin position“, „has ymin position“, „has xmax position“, „has ymax position“) der Bounding-Box als Attribute hinzugefügt. Die Auswertung hinsichtlich der Zusammenhänge zwischen verschiedenen Bauteilen muss auf diese Weise durch das LLM ausgewertet werden.

Relative Positionsbeschreibung: Alle Bauteile erhalten relative semantische Positionsbeschreibungen zu anderen erkannten Bauteilen („above“, „below“, „left of“, „right of“), die anhand der erkannten Bounding-Boxes abgeleitet werden. Diese Positionsbeschreibung ist weniger präzise als die absolute, ist aber semantisch besser auswertbar.

Implizite & explizite Positionsmodellierung: Die relative Positionsbeschreibung wird entweder nur zu den direkt umliegenden Bauteilen (implizite Positionsmodellierung) oder zu allen Bauteilen (explizite

Positionsmodellierung) hinzugefügt. Je nach Anzahl erkannter Bauteile läuft die explizite Modellierung Gefahr sehr große Ontologien zu erstellen, da die Anzahl an Relationen mit der Menge an Bauteilen quadratisch ansteigt.

- A, B: Zu vergleichende Individuen
- R: Menge aller zu betrachteten Relationen (z. B. *above*, *below*, *left_to*, ...)
- $r(A)$: Menge aller Ziel-Individuen, mit denen A über die Relation r verknüpft ist
- $|M|$: Anzahl der Elemente in der Menge M

$$\text{similarity}(A, B) = \frac{\sum_{r \in R} |r(A) \cap r(B)|}{\sum_{r \in R} |r(A)|}$$

$$\text{threshold} = \begin{cases} \text{threshold}_{\text{equal}}, & \text{wenn name}(A) = \text{name}(B) \\ \text{threshold}_{\text{diff}}, & \text{wenn name}(A) \neq \text{name}(B) \end{cases}$$

$$\text{mit} \quad \begin{cases} \text{threshold}_{\text{equal}} = \text{threshold} - \text{tolerance} \\ \text{threshold}_{\text{diff}} = \text{threshold} \end{cases}$$

$$\text{Dann gilt:} \quad \text{similarity}(A, B) > \text{threshold} \quad \Rightarrow \quad A \equiv B$$

Formel 2 Erkennung identischer Objekte anhand Objekterkennung aus verschiedenen Perspektiven

Für die Bewertung der Modellierungsarten wurde ein Fragenkatalog aufgebaut, der von verschieden leistungsstarken LLMs mittels zur Verfügung gestellter Ontologie beantwortet werden sollte. Dabei teilen sich die Fragen in drei Kategorien unterschiedlicher Schwierigkeit auf. Als „Einfache Wissensabfragen“ sind Positionsfragen klassifiziert, die ausgehend von einem Bauteil umliegende Bauteile anhand ihrer relativen Position abfragen (bspw. „Welche Komponente befindet sich über dem Motor?“). „Multihop-Wissensabfragen“ beinhalten komplexere Positionsabfragen zu mehreren Bauteilen und benötigen zur Antwort somit das Verständnis über die Positionierung aller Triples (bspw. „Wo befindet sich die linke Schleifscheibe?“). „Multihop-Wissensabfragen mit Allgemeinwissen“ bezeichnet Fragen mit der höchsten Komplexität und fügt dem komplexen Positionierungsverständnis inhaltliche Details hinzu (bspw. „Wenn der Motor überhitzt, welche Teile wären davon betroffen?“). Die Tests wurden mit den Modellen *DeepSeek-R1*, *DeepSeek-V3*, *Llama-3.1-8b-instruct* und *Qwen-2.5-3B* ausgeführt. Auffällig ist, dass Modelle mit geringer Parameteranzahl bei steigender Graphkomplexität (z. B. Multi-Image-Graphen) deutlich an Leistung verlieren (vgl. Abb. 17). Besonders sichtbar wird dies beim Qwen-2.5-3B-Modell, dessen Leistung stark einbricht, sobald zusätzlich zu den Multi-Image-Graphen auch die Koordinaten der Individuen einbezogen werden. Beim Llama-3.1-8B-instruct zeigt sich zudem, dass die implizite Modellierung der expliziten klar überlegen ist. Beide Modelle profitieren in der Single-Image-Modellierung durch Koordinaten, in der Multi-Image-Modellierung führte dies zu ungenaueren Ergebnissen. Die Ergebnisse deuten darauf hin, dass zusätzliche Koordinateninformationen nur bis zu einer bestimmten Graphkomplexität hilfreich sind. Für Modelle dieser Größe sollten Koordinaten daher nur integriert werden, wenn der Wissensgraph weniger als etwa zwölf Individuen umfasst, ein Schwellenwert, ab dem in der Evaluation erstmals Leistungsabfälle beobachtet wurden.

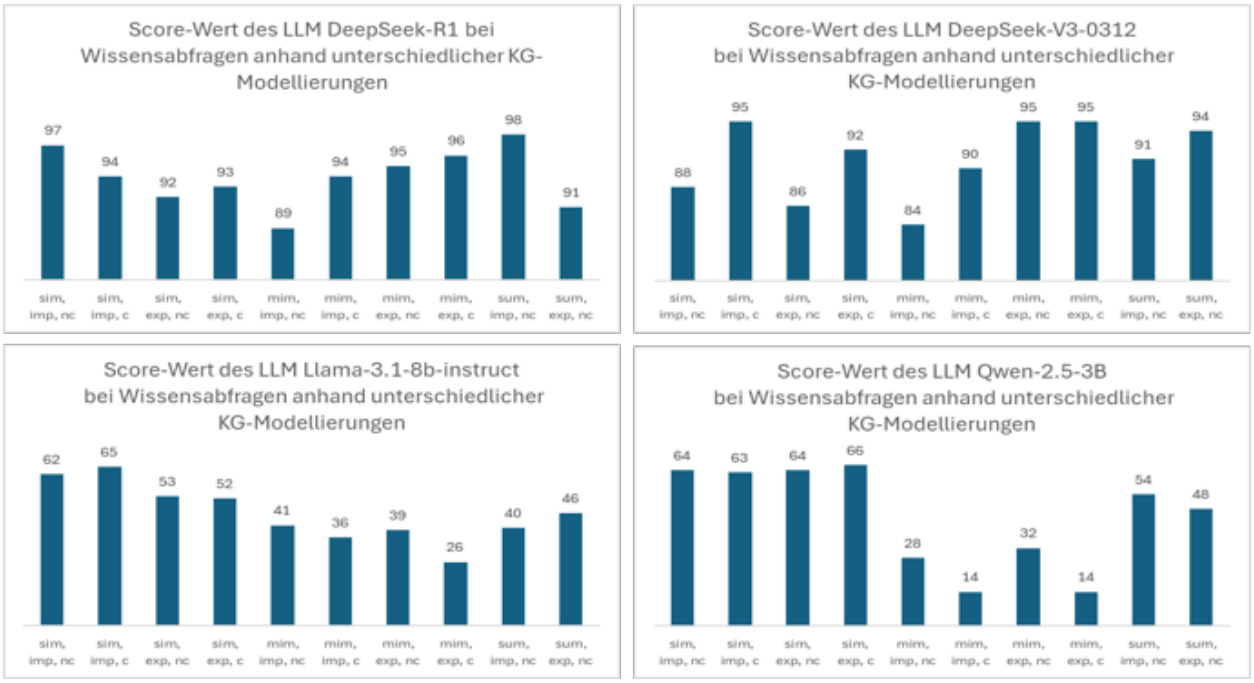

Abbildung 17 Evaluationsergebnisse für LLM-basierte Verarbeitung der Informationen der Objekterkennung

In der praktischen Anwendung sind Reasoning-LLMs auszuschließen, da die Antwortzeiten für sprachbasierte Interaktion zu lang sind.

2.5 Evaluation anhand von Fallbespielen in der Praxis (AP 5)

Zur Überprüfung der Leistungsfähigkeit und allgemeinen Tauglichkeit wurden verschiedene Fallbeispiele anhand von Praxisanwendungen getestet. So wurden unter anderem praxisnahe Umgebungen in der Versuchshalle der TU Dresden durchgeführt, Informationsqualität in anonymen Vergleichen von Nutzern händisch bewertet und verschiedene Informationsquellen aus der Praxis in einen digitalen Assistenten überführt und evaluiert.

Als Referenzen wurde u.a. eine umfangreiche Ontologie für eine Filteranlage eines Projektpartners anhand von Betriebsanleitungen aufgebaut. Diese beinhaltet Wartungsprozesse, Verbrauchsmittel, Werkzeuge, typische Störungen und Bauteile sowie eine Beschreibung des Softwaresystems und dessen Bedienung. Für das OWL-basierte RAG-System wurden daran sowohl Retrieval- als auch Generation-Schritt evaluiert (vgl. Abschnitt „OWL-basiertes RAG-System“). Während die Qualität des Datenretrievals durch einen Experten bewertet wurde, erfolgte die Prüfung der Leistungsfähigkeit verschiedener LLMs durch einen Community-driven-Ansatz. Für die drei unterschiedlich leistungsstarken LLMs CahtGPT-4o, Llama-3 8B und Gemma-2B wurden zu 33 Beispielfragen identische Retrieval-Ergebnisse zur Verfügung gestellt und anschließend die jeweiligen Antworten anonymisiert und in Duellen miteinander verglichen. Neben den LLMs wurde auch die Kontextgröße variiert, indem Nachbarn relevanter Knoten in der hinterlegten OWL-Ontologie mit einbezogen wurden ($H_{depth} \in \{0,1,2\}$). Die Aufgabe der Probanden war es, die Antworten zu bewerten und sich entweder für eine bessere Antwort zu entscheiden, beide Antworten als Unentschieden oder als ungenügend zu bewerten (siehe Abb. 18). Anschließend wurde für jedes LLM ein Elo-Score berechnet und visualisiert (siehe Abb. 19). Das Bewertungssystem wurde durch die LLM-Arena-Plattform inspiriert (LMArena 2025).

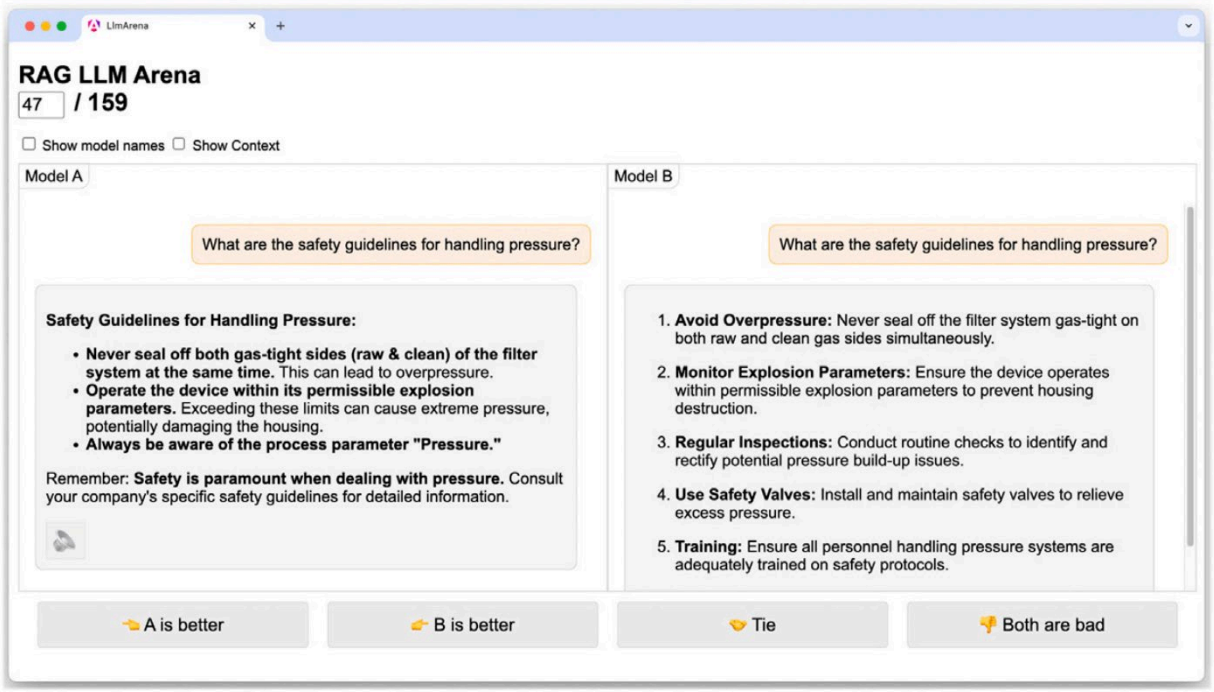

Abbildung 18 Webbasierte RAG LLM Arena für den anonymen, paarweisen Vergleich der Generierungsqualität verschiedener LLMs

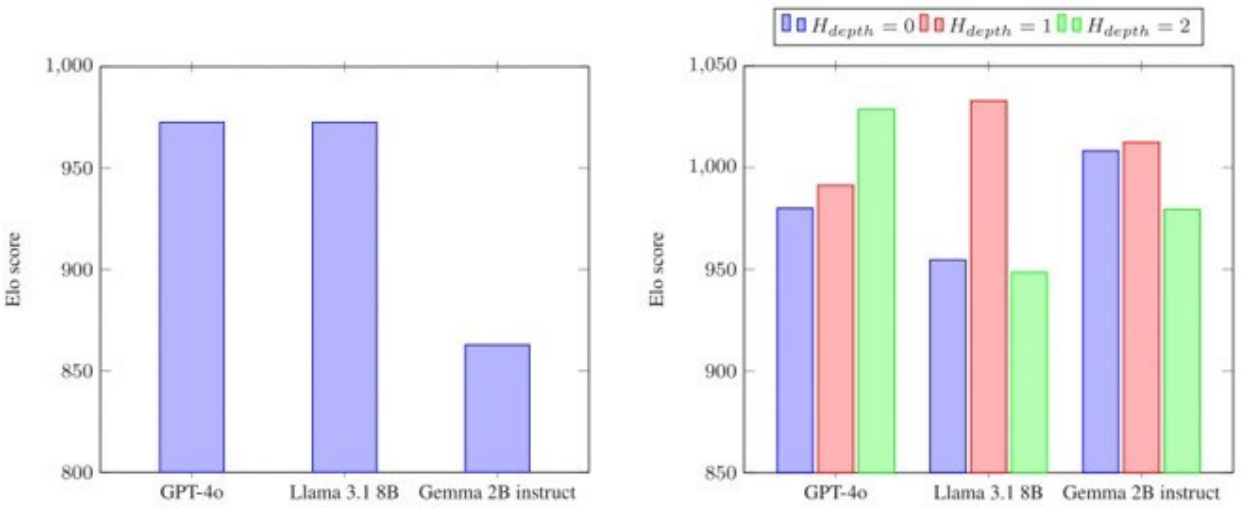

Abbildung 19 Auswertung Elo-Scores für modellübergreifende Vergleiche (links) und Intra-Modellvergleich verschiedener Kontextgrößen (rechts)

In den Vergleichen zeigte sich, dass auch leistungsschwächere LLMs sinnvolle Informationen anhand der ausformulierten OWL-Triples ausgeben können. Zudem wurde deutlich, dass benachbarte Knoten in der OWL-Struktur den Kontext sinnvoll um naheliegende Informationen erweitern, was sich positiv auf die Ergebnisgüte auswirkt. Wie zu erwarten kann ChatGPT-4o aufgrund seiner allgemein höheren Leistungsfähigkeit größere Kontexte besser als das Llama und das Gemma-Modell verarbeiten und profitiert somit vom Einbezug weiterer benachbarter Knoten. Eine umfangreichere Auswertung ist in der Veröffentlichung „An ontology-based retrieval augmented generation procedure for a voice-controlled maintenance assistant“ einsehbar (Ludwig, Schmidt und Kühn 2025, siehe Abschnitt 2.6).

Von einem weiteren Praxispartner wurden verschiedene Fehlermeldungen, betroffene Arbeitsmittel, Fehler-Ort, Ursache und Behebungsmöglichkeiten tabellarisch in einer Excel-Datei algorithmisch in einen OWL-Wissensgraphen überführt. Da die Daten bereits strukturiert vorliegen, wird die Umwandlung vereinfacht. Die Spaltenüberschriften stellen dabei die ontologischen Oberklassen für die einzelnen Individuen dar, die mittels Relationen („hat Ursache“, „betrifft Arbeitsmittel“) miteinander verknüpft sind. Da das RAG-Verfahren universell für OWL-Ontologien einsetzbar ist, sind keine weiteren Schritte zu tun. Eine zweite, ähnlich strukturierte Tabelle wurde ebenfalls in eine Ontologie umgewandelt und mit der ersten zusammengeführt, sodass ein geeinter Wissensspeicher geschaffen wurde.

Eine weitere Expertenbegutachtung fand für das Dokument-basierte RAG-System statt. Hierbei hat ein Fertigungsexperte aus der Industrie 20 Datensets (10 für Störungsursachsensuche; 10 für Ausgabe der Anweisungsschritte) begutachtet. Dabei wurden Musterlösungen aus drei verschiedenen Handbüchern und der jeweilige LLM-Prompt vorgegeben und die Antwort durch den Experten bewertet. Auf einer Skala von 1 (keine Relevanz/Vollständigkeit/ Korrektheit) bis 5 (hohe Relevanz/Vollständigkeit/ Korrektheit) wurden die RAG-Antworten bewertet (siehe Tabelle 8).

Tabelle 8 Evaluationsergebnisse Dokument-basiertes RAG-System

	Relevanz	Vollständigkeit	Korrektheit
Ursachensuche	3,7	4,2	3,9
Anweisungsschritte	4,1	4,3	4,2

Praxisnahe Tests zur Objekterkennung wurden in der Werkstatt der Versuchshalle der TU Dresden durchgeführt. Für verschiedene Bearbeitungsmaschinen wurden einzelne Komponenten für einen Trainingsdatensatz aufgenommen. In der Versuchshalle kam es dabei zu verschiedenen Licht- und Staubbedingungen, was im aufgenommenen Trainingsdatensatz zu einer höheren Robustheit des Objekterkennungsmodells führte. Zudem wurde auch die Wandlung von Sprache-zu-Text mit einem industriellen Noice-Cancelling-Headset unter industriellem Umgebungslärm getestet. Dabei wurde die allgemeine Tauglichkeit von Sprachassistenz produktionslogistischen Umfeld bestätigt, *Whisper* hat sich als geeignetes Framework herausgestellt (siehe Abschnitt 2.1).

2.6 Dokumentation und Veröffentlichungen (AP 6)

Während des Projektes wurden Zwischenergebnisse publiziert und auf Konferenzen vorgestellt. Eine konkrete Liste ist im Abschnitt „Publikation und Lehre“ eingefügt.

3 Verwendung der Zuwendung

Tabelle 9 Tatsächlicher Personaleinsatz der Prof. Technische Logistik (TL) und Prof. Wirtschaftsinformatik – Business Engineering (BE)

Arbeitspaket (AP)	Beschreibung	Verwendung der Zuwendung	
		TL:	BE:
1	Analyse	2 PM	2 PM
2	Identifikation Einsatzszenarien	2,5 PM	6 PM
3	Modellierung Wissensdatenstruktur	3,5 PM	7 PM
4	Entwicklung Gesamtkonzept AVISSBA	1 PM	0 PM
5	Evaluation	2 PM	0 PM
6	Dokumentation und Veröffentlichung	0,825 PM	0 PM
Summe		11,825 PM	15 PM

4 Notwendigkeit und Angemessenheit der geleisteten Arbeit

Die durchgeführten Forschungsarbeiten waren notwendig und angemessen, da sie den Punkten im Arbeitsplan des Projektantrages entsprachen. Dadurch wurden alle im Arbeitsplan formulierten Aufgaben erfolgreich bearbeitet.

5 Darstellung des wissenschaftlich-technischen und wirtschaftlichen Nutzens der erzielten Ergebnisse insbesondere für KMU sowie ihres innovativen Beitrags und ihrer industriellen Anwendungsmöglichkeiten

Im Rahmen des Forschungsprojektes wurde in Zusammenspiel mit den Projektpartnern ein Demonstrator geschaffen, der die textuelle und/oder sprachbasierte Abfrage von Wartungsinformationen ermöglicht. Dabei ist das System flexibel genug, dass in verschiedensten Domänen eingesetzt werden kann.

Durch Workshops wurden zudem verschiedene Strategien zur Einführung von KI-Assistenzsystemen beleuchtet und deren Vor- und Nachteile in konkreten Anwendungsfelder gegenübergestellt.

Sowohl die Untersuchung der industriellen Nutzung des intelligenten digitalen Assistenzsystems, als auch die Graph-basierte Datenhaltung haben einen klaren wirtschaftlichen Nutzen und sind aus wissenschaftlich-technischer Sicht in dieser Domäne als neuartig anzusehen.

6 Plan zum Ergebnistransfer in die Wirtschaft

6.1 Durchgeführte Transfermaßnahmen

Maßnahme	Ziel	Ort/Rahmen	Zeitraum
Ansprache weiterer interessierter Unternehmen	Direktansprache weiterer Unternehmen, Gewinnung für PA	Telefonische Akquise	Ab Ende 2023
PA-Meeting	Kick-Off Meeting	TU Dresden	27. November 2023
Projektbegleitender Ausschuss (PA)	Unternehmensanalyse, Analyse von Wartungsprozessen	Treffen vor Ort bei Alumina Systems	29. Januar 2024
Projektbegleitender Ausschuss (PA)	Unternehmensanalyse, Wartungsprozesse in der Lebensmittelproduktion	Treffen vor Ort bei Friends not Food GmbH, weitere Absprachen online	05. Februar bis 09. Februar 2024
Projektbegleitender Ausschuss (PA)	Unternehmensanalyse, Analyse von Wartungsprozessen	Treffen Deutsche Werkstätten Beteiligungs GmbH	04. März 2024
Projektbegleitender Ausschuss (PA)	Beratung Vorgehen Softwareentwicklung und Produktmanagement	Treffen vor Ort, Feedback-Schleifen als Online-Meetings	04. März bis 12 März 2024
Projektbegleitender Ausschuss (PA)	Allgemeine Beratung Vorgehensweisen Softwareentwicklung, insb. robuste Webanwendungen	Auftakt-Treffen vor Ort bei SaxMS GmbH, weitere Treffen als Online-Meetings	16. April bis 19. April 2024
Ansprache weiterer interessierter Unternehmen	Direktansprache weiterer Unternehmen, Gewinnung für PA	Hannover Messe	22. April 2024
Projektbegleitender Ausschuss (PA)	Unternehmensanalyse, Analyse von Wartungsprozessen in der Aviation-Branche	Treffen vor Ort bei EFW GmbH	24. Mai 2024
Projektbegleitender Ausschuss (PA)	Wissensmanagement in der Halbleiterbranche, Workshop systematische Datenauswertung mit Praxisdaten	FlowLogiX GmbH an der TU Dresden	16. Oktober 2024 bis 22. Oktober 2024
Projektbegleitender Ausschuss (PA)	Analyse und Status quo von Wartungsprozessen & KI-Einsatz (u.a. Infield) bei der Bundeswehr	Treffen vor Ort bei BAAINBw	24. Oktober 2024
Vortrag	Vorstellung von (technischen) Zwischenergebnissen und Erkenntnissen	SAP Predictive TownHall Meeting	14. Mai 2024

PA-Meeting	Zwischenpräsentation 2024	TU Dresden	04. Dezember 2024
Projektbegleitender Ausschuss (PA)	Workshop Konzeptionierung Assistenzsysteme	Treffen vor Ort bei FlowLogiX GmbH	21. Januar bis 23. Januar 2025
Projektbegleitender Ausschuss (PA)	Analyse von Wartungsprozessen im Bahnbau, Weitere Hinweise zur Nutzung digitaler KI-Assistenzsysteme	Treffen vor Ort bei SaxMS GmbH	05. März 2025
Vortrag	Digitaler Wartungsassistent als Use Case für KI-Anwendung in der Produktionslogistik	32. Deutscher Materialfluss-Kongress: future.meets.logistics - Was bringt die Zukunft?	20. – 21. März 2025
Projektbegleitender Ausschuss (PA)	Software-Beratung + Workshop zum Aufbau mobiler Assistenzsysteme	Treffen vor Ort bei LOGSOL GmbH	08. April bis 10. April 2025
Vortrag	Lessons Learned zur Einführung von KI-Methoden, Demonstratorpräsentation, Werbung neuer PA-Partner	Treffen des Vereins Deutscher Maschinenbau-Anstalten (VDMA) Ost - Arbeitskreis Produktion: KI in der Produktion & Losgröße 1	08. Mai 2025
Projektbegleitender Ausschuss (PA)	Feedback Demonstrator, Workshop zur Einführung digitaler Assistenzsysteme	Treffen vor Ort bei Friends not Food GmbH, weitere Absprachen online	19. Mai bis 21. Mai 2025
Projektbegleitender Ausschuss (PA)	Aufzeigen von Wartungsarbeiten in der Halbleiterindustrie, Austausch über Methoden im Wissensmanagement	Treffen vor Ort bei FlowLogiX GmbH	15. Juli 2025
Projektbegleitender Ausschuss (PA)		Treffen vor Ort bei FlowLogiX GmbH	11. September 2025
PA-Meeting	Abschlusspräsentation 2025	TU Dresden	22. September 2025

6.2 Publikationen und Lehre

Maßname	Beschreibung	Datum
Publikation	Ludwig, H., Schmidt, T., Kühn, M. „An ontology-based retrieval augmented generation procedure for a voice-controlled maintenance assistant“, Computers in Industry	März 2025
Publikation	Schmidt, T., Ludwig, H., Kühn, M. 2025. „Store-by-voice: A Voice-controlled Putaway and Stocktaking System.“ 17th IM-HRC Proceedings	Juli 2025

Publikation	Scharfe, P., Horn, F. “ Retrieval-Augmented Generation zur Wissensextraktion und Strukturierung: Entwicklung eines Assistenztools zur Erstellung von Instruktions-Chatbots ” (in Überarbeitung)	Januar 2026
Publikation	Ludwig, H., Schmidt, T.,Kühn, M. „Text-to-OWL: Fine-Tuning Large Language Models for Ontological Knowledge Engineering“ (in Erstellung)	Vorr. Feburar 2026
Publikation	Ludwig, H., Schmidt, T.,Kühn, M. „ Semantic Scene Understanding through Object Detection and Knowledge Graphs for Retrieval-Augmented Assistants“ (in Erstellung)	Vorr. März 2026
Vortrag	Ludwig, H. „Bridging Minds: The Transformative Friendship of Large Language Models and Knowledge Graphs“, SAP Research	Mai 2024
Vortrag, Konferenzband	Ludwig, H. „Hey ChatGPT, wie sieht der Materialfluss von morgen aus? Potenzialanalyse von Sprachassistenten in der Logistik”, 32. Deutscher Materialfluss-Kongress: future.meets.logistics - Was bringt die Zukunft?	März 2025
Vortrag	Ludwig, H. „Das erledige ich mit ChatGPT“ – Interaktion von Mensch und KI in der Produktion: Status Quo, Praxisbeispiele und Perspektiven”, Treffen des Vereins Deutscher Maschinenbau-Anstalten (VDMA) Ost - Arbeitskreis Produktion: KI in der Produktion & Losgröße 1	Mai 2025
Studien- und Abschlussarbeiten	Wiegand, J. „Potentialanalyse zum Einsatz generativer künstlicher Intelligenz im Produktionsumfeld“, Forschungsseminar	Mai 2024
Studien- und Abschlussarbeiten	Horn, F. „Designing Chatbot Solutions for SMEs: Leveraging Microsoft's Copilot Studio for Instruction Assistance in Manufacturing Environments“, Belegarbeit	August 2024
Studien- und Abschlussarbeiten	Wiegand, J. „Leitfaden zum Einsatz von Generative-AI in produzierenden Großunternehmen“, Diplomarbeit	Februar 2025
Studien- und Abschlussarbeiten	Morgenstern, R. „Automatisierte Wissensgraph-Erstellung per Multi-Objekterkennung für einen KI-Wartungsassistenten“, Bachelorarbeit	September 2025
Internet	News-Post auf Lehrstuhl-Webseite	ab 2023
Internet	Linkedin-Posts auf Lehrstuhl-Account	ab 2025
Lehre	Ausbildung von Absolventen	fortlaufend
Lehre	Ergänzung des Themenkomplexes in Vorlesungen (bspw. „Produktions- und Logistiksystemgestaltung“, „Produktions- und Logistiksystembetrieb“)	fortlaufend
Lehre	Ergänzung der Lean-Lernwerkstatt um eine sprachbasierte Assistenzkomponente	fortlaufend

6.3 Geplante Transfermaßnahmen

Maßnahme	Beschreibung	Zeitraum
Wissenschaftliche Qualifikation	Weitere Aspekte und Varianten der im Projekt identifizierten Problemstellung sollen im Rahmen einer Dissertation untersucht werden: Promotionsvorhaben Ludwig: Voice Operator 4.0 (Arbeitstitel)	Ab 2026
Demonstratornutzung	Projektpartner nutzen den im Projekt konzipierten und erstellten Demonstrator zu Wartungszwecken. Die Weiterentwicklung zu einem Produkt/produktnahen Prototypen ist denkbar.	Ab 2026
Lehre	Aufnahme von wissenschaftlich gewonnenen Erkenntnissen zum Projektthema in die Lehrveranstaltungen	fortlaufend

6.4 Einschätzung zur Realisierbarkeit des vorgeschlagenen und aktualisierten Transferkonzepts

In Gesprächen mit Mitgliedern des PA hat sich neben der Wartung und dem Betrieb von technischen Anlagen das Wissensmanagement im Allgemeinen als Kernherausforderung der kommenden Jahre herausgestellt. Pensionierung, Mitarbeitermangel und Fluktuation sorgen dafür, dass ohne die Absicherung des impliziten Wissens insbesondere spezialisierte KMU Gefahr laufen, spezialisierte Kompetenzen und Erfahrungswerte zu verlieren. Dies wurde auch auf dem VDMA-Tag durch mehrere Unternehmen bestätigt. Während die Wissensabfrage mittels OWL-basiertem Retrieval während des Projektes einsatzbereit war, konnten die finalen Evaluationen zur LLM-basierten OWL-Erzeugung nicht mehr innerhalb der Projektlaufzeit durchgeführt werden. Stellt sich jedoch heraus, dass die bereits vielversprechenden Ergebnisse auch im Detail den Anforderungen genügen, im Idealfall unter geringem Hardwareaufwand, ist die Überführung des Gesamtsystems in die Praxis hochlukrativ. So entstehen kaum Personalkosten durch Schulungsaufwand oder hohe Ausgaben für Hardwarebeschaffung, da das System selbstlernend ist und per natürlicher Sprache auch unter Einsatz von weniger komplexen LLMs abgefragt werden kann. Zudem ist das System hinsichtlich des Datenschutzes unbedenklich, da es vollständig lokal betrieben werden kann und so keinerlei Informationen das Unternehmen verlassen.

7 Quellen

Cai, Weilin, Juyong Jiang, Fan Wang, Jing Tang, Sunghun Kim, und Jiayi Huang. 2025. „A Survey on Mixture of Experts in Large Language Models.“ IEEE Transactions on Knowledge and Data Engineering, 3896-3915.

Cutting-Decelle, A. F., R.I.M. Young, J.J. Michel, R. Grangel, J. Le Cardinal, und J.P. Bourey. 2007. „ISO 15531 MANDATE: A Product-process-resource based Approach for Managing Modularity in Production Management.“ Concurrent Engineering 15, 217-235.

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, und Kristina Toutanova. 2019. „BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.“ arXiv.

F. Liu, Nelson, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, und Percy Liang. 2023. „Lost in the Middle: How Language Models Use Long Contexts.“ arXiv.

F. Noy, Natalya, und Deborah L. McGuinness. 2001. „Ontology Development 101: A Guide to Creating Your First Ontology.“

Fernández, Mariano, Asunción Gómez-Pérez, und Natalia Juristo. 1997. „Methontology: From Ontological Art Towards Ontological Engineering.“ Ontological Engineering, Papers from the 1997 AAAI Spring Symposium.

FreeTTS. 2025. FreeTTS. Zugriff am 1. Oktober 2025. https://freetts.com/.

Gao, Yunfan, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang, und Haofen Wang. 2024. „Retrieval-Augmented Generation for Large Language Models: A Survey.“ arXiv.

Google. 2025. Angular. Zugriff am 1. Oktober 2025. https://angular.dev/.

Jain, Aditya, Amir Maleki, und Nathalie Saade. 2024. How to fine-tune: Focus on effective datasets. Zugriff am 1. 10 2015. https://ai.meta.com/blog/how-to-fine-tune-llms-peft-dataset-curation/.

Khadir, Ahlem Chérifa, Hassina Aliane, und Ahmed Guessoum. 2021. „Ontology learning: Grand tour and challenges.“ Computer Science Review, Volume 39.

Liu, Wei, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, und Alexander C. Berg. 2016. „SSD: Single Shot MultiBox Detector.“ arXiv.

LMArena. 2025. Zugriff am 1. Oktober 2025. https://lmarena.ai/.

Ludwig, Heiner, Thorsten Schmidt, und Mathias Kühn. 2025. „An ontology-based retrieval augmented generation procedure for a voice-controlled maintenance assistant.“ Computers in Industry, Volume 169.

Mazzola, Luca, Patrick Kapahnke, Marko Vujic, und Matthias Klusch. 2016. „CDM-Core: A Manufacturing Domain Ontology in OWL2 for Production and Maintenance.“ Proceedings of the International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management, 136–143.

Mozilla Foundation. 2025. Web Speech API . Zugriff am 1. Oktober 2025. https://developer.mozilla.org/en-US/docs/Web/API/Web_Speech_API.

National Institute of General Medical Sciences. 2025. Protégé. Zugriff am 1. Oktober 2025. https://protege.stanford.edu/.

Ollama. 2025. Ollama: Chat & build with open models. Zugriff am 1. Oktober 2025. https://ollama.com/.

Pan, Shirui, Linhao Luo, Yufei Wang, Chen Chen, Jiapu Wang, und Xindong Wu. 2024. „Unifying Large Language Models and Knowledge Graphs: A Roadmap.“ IEEE Transactions on Knowledge and Data Engineering, 3580-3599.

Parsia, Bijan, Nicolas Matentzoglu, Gonçalves Rafael S., Birte Glimm, und Andreas Steigmiller. 2017. „The OWL Reasoner Evaluation (ORE) 2015 Competition Report.“ Journal of Automated Reasoning.

Radford, Alec, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, und Ilya Sutskever. 2022. „Robust Speech Recognition via Large-Scale Weak Supervision.“ arXiv.

RDFLib Team. 2025. RDFLib. Zugriff am 1. Oktober 2025. https://rdflib.readthedocs.io/en/stable/.

Reimers, Nils, und Iryna Gurevych. 2019. „Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks.“ arXiv.

Schmidt, Thorsten, Heiner Ludwig, und Mathias Kühn. 2025. „Store-by-voice: A Voice-controlled Putaway and Stocktaking System.“ 17th IMHRC Proceedings (Trondheim, Norway-2025).

2025. spaCy: Industrial-Strength Natural Language Processing. Zugriff am 1. Oktober 2025. https://spacy.io/.

The Apache Software Foundation. 2025. Apache Jena. Zugriff am 1. Oktober 2025. https://jena.apache.org/.

University of Manchester. 2025. OWL API main repository . Zugriff am 1. Oktober 2025. https://github.com/owlcs/owlapi.

unsloth. 2025. Unsloth AI. Zugriff am 1. Oktober 2025. https://unsloth.ai/.

W3 Semantic Web Standards. 2012. Web Ontology Language (OWL). Zugriff am 1. Oktober 2025. https://www.w3.org/OWL/.