Retrieval Augmented Generation

Hausarbeit Datenbanken II

aus dem Studiengang Wirtschaftsinformatik Sales & Consulting

an der Dualen Hochschule Baden-Württemberg Mannheim

von

Tim Christopher Eiser

Julian Konz

Benjamin Will

Bearbeitungszeitraum:	12.05.2025 - 03.08.2025
Kurs:	WWI23SCB
Studiengangleiter:	Prof. Dr. -Ing. Clemens Martin
Ausbildungsfirma:	SAP SE Dietmar-Hopp-Allee 16 69190 Walldorf, Deutschland
Dozent:	Frank Neubüser frank.neubueser@eviden.com +49 (0) 211 399 36181

I. Eidesstattliche Erklärung

Ich versichere hiermit, dass ich meine Projektarbeit mit dem Thema: „Titel“ selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Ich versichere zudem, dass die eingereichte elektronische Fassung mit der gedruckten Fassung übereinstimmt.

Mannheim, 14.07.2025	Tim C. Eiser
Ort, Datum	**Unterschrift Tim Christopher Eiser**
Walldorf, 14.07.2025	Julian Konz
Ort, Datum	**Unterschrift Julian Konz**
Walldorf, 14.07.2025	Benjamin Will
Ort, Datum	**Unterschrift Benjamin Will**

II. Gleichbehandlung der Geschlechter

In dieser Praxisarbeit wird aus Gründen der besseren Lesbarkeit das generische Maskulinum verwendet. Weibliche und anderweitige Geschlechteridentitäten werden dabei ausdrücklich mitgemeint, soweit es für die Aussage erforderlich ist.

III. Disclaimer

Ein Teil der Literatur, die für die Anfertigung dieser Arbeit genutzt wird, ist nur über die E-Book-Plattform o'Reilly abrufbar. Bei diesen Ressourcen existieren keine Seitennummern, es wird bei Verweisen stattdessen die Kapitelnummer angegeben.

Um den Lesefluss zu verbessern, werden Abbildungen, Codebeispiele und Tabellen, die den Lesefluss stören, im Anhang platziert, auf den im Text zusätzlich verwiesen wird.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Promptverzeichnis

Abkürzungsverzeichnis

A	Antwort
AI	Artificial Intelligence
CRISP-DM	Cross Industry Standard for Data-Mining
CSV	Comma-Separated Values
DPR	Dense Passage Retriever
F	Frage
LLM	Large Language Model
NLP	Natural Language Processing
PLM	Large-Scale Pre-Trained Language Model
Prompt	Eingabe
RAG	Retrieval-Augmented Generation
ROUGE	Recall-Oriented Understudy for Gisting Evaluation
RQ	Forschungsfrage
pp	Prozentpunkte

Variablenverzeichnis

B	Menge aller n-gramme im vorhergesagten Text
D	Menge aller n-gramme im Referenztext
FN	False Negative
FP	False Positive
H	Menge an tatsächlichen Klassen
L	Menge an Klassen
N	Anzahl an Iterationen
P	Menge an vorhergesagten Klassen
SE	Standardfehler
TN	True Negative
TP	True Positive
b	vorhergesagter Text
c	Klasse
d	Referenztext
k	Anzahl der betrachteten Chunks
r	Pearson-Korrelationskoeffizient
Λ	Textlänge
ω	Wort

1. Einleitung

1.1. Motivation

In den letzten Jahren hat die Forschung im Bereich der Large Language Models (LLMs) enorme Fortschritte gemacht. Spätestens mit der Veröffentlichung von Modellen wie GPT-4 ist klar: Generative Artificial Intelligence (AI)-Systeme haben das Potenzial, bestehende Prozesse in Wissenschaft, Wirtschaft und Gesellschaft grundlegend zu verändern. Doch bei aller Euphorie bleibt ein zentrales Problem ungelöst: LLMs basieren ausschließlich auf ihrem statischen Trainingswissen. Für Kontexte, in denen aktuelle oder domänenspezifische Informationen benötigt werden, versagen sie, halluzinieren, liefern veraltete oder schlicht falsche Antworten. Die Antwort der Forschung auf dieses Problem lautet: Retrieval-Augmented Generation (RAG).

RAG-Systeme verbinden klassische Sprachmodelle mit externem, dynamisch abrufbarem Wissen. Statt allein auf das interne Modellwissen zu vertrauen, ruft das System bei jeder Anfrage kontextrelevante Inhalte ab und reichert die Antwort dynamisch damit an. Studien wie Hasan et al. [5] zeigen, dass RAG in Praxisfeldern wie Governance oder Medizin bereits erfolgreich eingesetzt werden kann. Für die wissenschaftliche Nutzung, wie die Analyse aktueller Paper, existieren jedoch kaum belastbare Daten zur Leistungsfähigkeit von RAG-Systemen.

1.2. Forschungsfrage

Ausgehend von den beschriebenen Herausforderungen und Entwicklungen ergibt sich, für uns, folgende Forschungsfrage (RQ):

RQ: Wie leistungsfähig sind RAG-Systeme bei der Beantwortung wissenschaftlicher Fachfragen auf Basis aktueller, zuvor nicht im Modelltraining enthaltener Literatur?

Diese Frage wird im Rahmen eines eigenen RAG-Prototyps beantwortet, der für die Nutzung wissenschaftlicher Paper konzipiert wurde. Dabei liegt der Fokus nicht nur auf der technischen Optimierung, sondern auch auf der Bewertung der Antwortqualität anhand etablierter Metriken.

1.3. Aufbau der Arbeit

Abschnitt 2 erläutert die methodische Herangehensweise auf Basis des Cross Industry Standard for Data-Mining (CRISP-DM)-Prozesses. Abschnitt 3 liefert die theoretischen und technischen Grundlagen zu LLMs, Embedding-Modellen, Vektordatenbanken und dem RAG-Konzept. In Abschnitt 4 wird das entwickelte System im Detail vorgestellt, einschließlich Datenbasis, Modellierung und Evaluationsstrategie. Abschnitt 5 schließt mit einer Zusammenfassung, diskutiert die Ergebnisse, Limitationen sowie den praktischen Nutzen und endet mit einen Ausblick auf weiterführende Forschung und sowie Verwendung der Ergebnisse.

2. Methodik

Zur systematischen Analyse der Forschungsfrage wird das Cross Industry Standard for Data-Mining (CRISP-DM)-Prozessmodell verwendet [6, S.3048]. CRISP-DM hat sich in der Praxis und Forschung im Bereich AI und Data-Mining als de-facto-Standard etabliert [6, S.3048], [7, S.526], [8, S.2], und bietet eine klare Struktur zur Durchführung datengetriebener Projekte [7, S.527], [9, S.401]. CRISP-DM ist domänenunabhängig einsetzbar, und ist insbesondere für komplexe Machine-Learning-Prozesse geeignet, bei welchen Datenauswahl, Modellierung und Evaluation eng verzahnt sind.

Im Kontext dieser Arbeit ermöglicht CRISP-DM eine methodisch saubere Umsetzung des RAG-Prototyps. Von der Zieldefinition über die zur Evaluation der Antwortqualität bis hin zur Entwicklung eines Prototyps. Die iterative Natur des Modells erlaubt es zudem, Erkenntnisse aus Zwischenschritten in spätere Phasen zurückzuführen und so das System iterativ zu verbessern Abbildung 1.

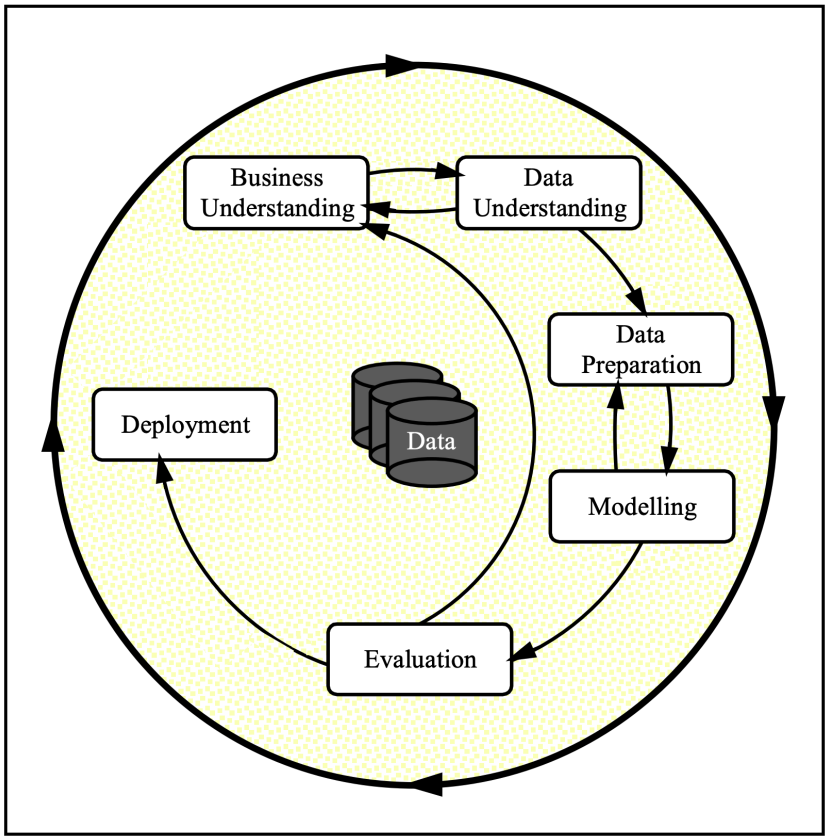

Abbildung 1: Phasen des CRISP-DM Phasenmodells [1]

Der CRISP-DM Data-Mining-Prozess kann in sechs iterative Phasen gegliedert werden (siehe Abbildung 1) [7, S.527], [10, S.13]. Diese Phasen sind:

1. Business Understanding: Die erste Phase fokussiert sich auf die Formulierung der geschäftlichen Projektziele und Anforderungen [7, S.527]. Diese werden anschließend in einen technischen Kontext überführt. Basierend auf dieser Grundlage wird die Problemstellung definiert und ein Konzept entwickelt um diese zu lösen [10, S.13], [11, S.8].

2. Data Understanding: Die Phase des Data Understandings beginnt mit der initialen Datenbeschreibung [11, S.7]. Anschließend folgt die Datenanalyse, wobei Zusammenhänge und Auffälligkeiten untersucht werden [7, S.527]. Zuletzt wird die Qualität der Daten geprüft, um sicherzustellen, dass die Daten für die folgende Modellierung geeignet sind [11, S.9].

3. Data Preparation: Die Phase der Data Preparation verfolgt das Ziel, die relevanten Daten auszuwählen, zu bereinigen und aufzubereiten, um einen Datensatz für die Modellierung zu erstellen [10, S.14] [11, S.9]. Zudem werden neue Attribute erstellt und das Datenformat an die Anforderungen der Modellierungstools angepasst [11, S.8].

4. Modelling: In dieser Phase werden geeignete Modellierungstechniken ausgewählt und anhand der vorbereiteten Daten angewendet [10, S.14], [11, S.9]. Die daraus folgenden Ergebnisse der Modelle werden bewertet und bei Bedarf optimiert. Die Auswahl eines Verfahrens zur Bewertung der Qualität des gewählten Modells ist hierbei von hoher Bedeutung. Das Ziel der Phase ist das bestmögliche Erreichen der zuvor definierten Ziele [1, S.6] .

5. Evaluation: In dieser Phase erfolgt die umfassende Evaluation und Bewertung der zuvor erstellten Modelle [1, S.6]. Zur Beurteilung der Modellqualität werden die zuvor festgelegten Testverfahren sowie die definierten Evaluationsmetriken herangezogen. Dabei wird überprüft, ob die entwickelten Modelle die gewünschten Ergebnisse liefern und die definierten Geschäftsziele vollständig erreichen [1, S.6], [10, S.14].

6. Deployment: Dieser Schritt umfasst die Implementierung des Modells, zum Beispiel als Prototyp, abhängig vom Modell Zweck und der geplanten Anwendung [1, S.7], [10, S.14], [10, S.32-34].

Obwohl CRISP-DM diesen etablierten Standard zur Strukturierung datengetriebener Projekte bietet, weist das Modell im Kontext moderner AI-Anwendungen wie RAG methodische Grenzen auf. Es wurde für klassische Data-Mining-Prozesse entwickelt und bildet neuere Konzepte wie Prompt-Engineering, semantisches Retrieval oder die nicht-deterministische Evaluation generativer Modelle nicht explizit ab [6, S.3049]. Ebenso fehlen integrierte Mechanismen zur Qualitätssicherung über alle Phasen hinweg, was insbesondere bei Systemen mit dynamischen Antwortverhalten wie LLMs relevant ist.

Trotz dieser Einschränkungen bietet CRISP-DM eine geeignete methodische Grundlage für diese Arbeit. Das Modell erlaubt eine strukturierte, nachvollziehbare Durchführung des Entwicklungs- und Evaluationsprozesses. Die Phasen lassen sich flexibel auf die Anforderungen der RAG-Systementwicklung übertragen [7, S. 528]. Die iterative Struktur ermöglicht es zudem, Erkenntnisse aus der Evaluationsphase direkt in Modellierung und Datenaufbereitung zurückzuführen [6, S.3051]. Somit wird CRISP-DM in dieser Arbeit nicht als starres Framework, sondern als anpassbarer Referenzrahmen genutzt, der gezielt um AI-spezifische Elemente ergänzt wird.

3. Grundlagen

3.1. Grundlagen Large Language Modellen

Large Language Models (LLMs) haben das Natural Language Processing (NLP) nachhaltig verändert, da diese natürliche Sprache verarbeiten und syntaktisch, semantisch und logisch korrekte Texte generieren können. LLMs werden auf großen Textdatensätzen trainiert und kombinieren neuronale Netze mit spezialisierten Architekturen wie dem Transformer, der den Grundstein für ihre Leistungsfähigkeit legt [12, S. 1-4], [13, S. 10].

3.1.1. Architektur und Funktionsweise

Die Entwicklung heutiger leistungsfähiger LLMs basiert auf der Transformer-Architektur von Vaswani et al. [13], die durch ihren Self-Attention-Mechanismus eine effiziente Verarbeitung natürlicher Sprache ermöglicht. Moderne LLMs werden durch ausgedehntes Vortraining auf umfangreichen Textkorpora entwickelt und können anschließend für spezifische Aufgaben angepasst werden [14, S. 1]. Die Größe von LLMs bemisst sich an der Zahl der trainierbaren Parameter, die – neben Faktoren wie der Qualität der Trainingsdaten – ihr Sprachverständnis beeinflussen [15, S. 4].

3.1.2. Inferenz und Prompt-basierte Interaktion

Die praktische Anwendung von LLMs erfolgt in der Inferenz-Phase, in der das trainierte Modell anhand einer Eingabe (Prompt) und auf Basis der gelernten Sprachmuster eine Ausgabe generiert [16, S. 3]. Der Prompt fungiert dabei als zentrale Schnittstelle zwischen Nutzer und Modell und ermöglicht es, das Verhalten des LLMs präzise zu steuern und spezifische Kontextinformationen zu übermitteln. Für die Formulierung von Prompts haben sich folgende Empfehlungen etabliert:

- **Klarheit und Präzision**: Prompts sollten unmissverständlich und eindeutig formuliert sein, um ungenaue oder mehrdeutige Antworten zu vermeiden.

- **Bereitstellung von Kontext und relevanten Informationen**: LLMs erzielen bessere Ergebnisse, wenn sie die Zielgruppe sowie den spezifischen Anwendungsbereich und Kontext kennen [17, S. 5].
- **Wahrung von Neutralität und Objektivität**: Um Verzerrungen zu vermeiden, sollten Prompts keine suggestiven oder wertenden Formulierungen enthalten, sodass die Antworten des Modells objektiv bleiben [18, S. 6-7].
- **Nutzung spezifischer Formatvorgaben**: Durch die Definition eines strukturierten Ausgabeformats, etwa in Form von JSON-Schemata, wird die inhaltliche Konsistenz und Nachvollziehbarkeit der generierten Inhalte signifikant erhöht. Dieser Ansatz legt explizite Antwortparameter fest und erleichtert die nachgelagerte Verarbeitung, wodurch eine konsistente und zuverlässige Klassifikation gewährleistet wird [19], [20, S. 11].

Zudem lassen sich Prompts in System- und User-Prompts unterteilen. Ziel ist es durch diese Teilung die Leistung des Modells weiter positiv zu beeinflussen [21, S.388]. System- und User-Prompts lassen sich definieren wie folgt:

- **System-Prompt:** Der System-Prompt ist die initiale, funktionsspezifische Anweisung, die den Rahmen zwischen dem LLM und dem menschlichen Benutzer definiert [22, S.117]. Innerhalb des System-Promps, kann das Verhalten, die Formalität und Fachsprache definiert werden. Dazu werden der Kontext, die Rolle oder spezifische Regeln für die Interaktion festgelegt. [23].
- **User-Prompt:** Der User-Prompt ist die spezifische Eingabe des Endnutzers, auf die das Modell reagiert. Diese Anfrage stellt die Grundlage für die erzeugten Antworten dar [24].

Während der Inferenz verarbeitet das LLM die Eingabe tokenweise und generiert basierend auf den Wahrscheinlichkeitsverteilungen seiner Parameter eine kohärente Antwort. Die Qualität dieser Ausgabe hängt maßgeblich vom bereitgestellt Kontext ab [25, S. 3-6].

3.2. Embedding-Modelle

Embedding-Modelle spielen eine zentrale Rolle im ML, insbesondere in RAG-Architekturen. Sie transformieren Wörter, Sätze oder Dokumente in dichte, numerische Vektoren, die semantische Beziehungen im hochdimensionalen Raum abbilden. Da Computer nicht direkt mit Wörtern arbeiten können, übersetzen Embeddings textuelle Inhalte in Vektorräume, wobei semantische Beziehungen erhalten bleiben. Abbildung 2 zeigt eine vereinfachte Übersicht über den Embedding-Prozess, der im folgenden detailliert erläutert wird.

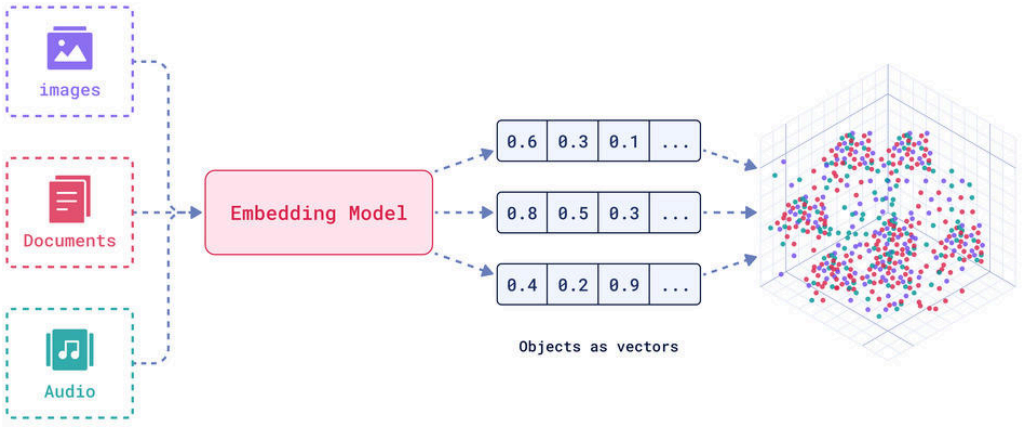

Abbildung 2: Übersicht Prozess Embedding Modelle [2].

Die Umwandlung eines Wortes eines Satzes oder eines Dokuments in einen hochdimensionalen Vektor erfolgt nach folgendem Grundprinzip:

1. **Tokenisierung:** Die Tokenisierung teilt sich in zwei wesentliche Phasen: die Textbereinigung und die eigentliche Token-Extraktion. Zunächst wird der Text von irrelevanten Elementen bereinigt. Dazu zählen HTML-Tags, URLs, E-Mail-Adressen, Emojis, Sonderzeichen sowie überflüssige Leerzeichen oder Zeilenumbrüche. Optional kann der gesamte Text in Kleinbuchstaben konvertiert werden, um die Konsistenz zu erhöhen. Anschließend wird der bereinigte Text in einzelne Tokens (Einheiten) zerlegt. Unter einem Token versteht man dabei die kleinste bedeutungstragende Einheit, in die ein Text während der Verarbeitung durch ein Sprachmodell zerlegt wird. Was dabei genau als Token gilt, hängt vom Modell und der Tokenisierungsstrategie ab [13, S.3-6].

2. **Embedding-Zuordnung:** In diesem Schritt wird jedem Token eine numerische Repräsentation in Form eines hochdimensionalen Vektors zugewiesen. Dies erfolgt über eine Lookup-Operation in der Embedding-Matrix, einer vortrainierten Tabelle, die für jedes Token im Vokabular einen entsprechenden Vektor bereithält. Die Embedding-Matrix entsteht durch das Training auf großen Textkorpora mittels Algorithmen wie Word2Vec, GloVe oder FastText [13, S.3-6].

3. **Vektorraum und semantische Beziehungen:** Die generierten Embedding-Vektoren spannen einen hochdimensionalen Vektorraum auf, in dem semantische Beziehungen zwischen Wörtern durch geometrische Relationen repräsentiert werden. Semantisch ähnliche Wörter befinden sich in diesem Raum in räumlicher Nähe zueinander. Diese Eigenschaft ermöglicht es, semantische Ähnlichkeiten durch Vektoroperationen (Kosinus-Ähnlichkeit) zu erfassen. Darüber hinaus können durch Vektorarithmetik komplexe Beziehungen modelliert werden, wie beispielsweise die bekannte Analogie „König - Mann + Frau ≈ Königin“ [13, S.3-6].

4. **Pooling/Aggregation:** Da längere Texte aus mehreren Tokens bestehen, müssen die individuellen Token-Vektoren zu einer einheitlichen Repräsentation für den gesamten Text aggregiert werden. Hierfür können verschiedene Pooling-Strategien genutzt werden. Average Pooling berechnet den Durchschnitt aller Token-Vektoren, während Max Pooling die maximalen Werte jeder Dimension über alle Token verwendet. Min Pooling wählt entsprechend die minimalen Werte aus [13, S.3-6].

5. **Normalisierung:** In diesem Schritt werden die Embeddings vergleichbar gemacht, um eine einheitliche Skalierung zu gewährleisten. Die Standardmethode ist die L2-Normalisierung, die jeden Vektor so anpasst, dass seine euklidische Länge (L2-Norm) 1 wird. Dies geschieht durch Division jeder Komponente des Vektors durch die ursprüngliche L2-Norm des Vektors. Die Normalisierung ermöglicht es, dass die Kosinus-Ähnlichkeit als Maß für semantische Ähnlichkeit verwendet werden kann, da normalisierte Vektoren die gleiche Länge haben und somit nur ihre Richtung im Vektorraum relevant ist [13, S.3-6].

3.3. Vektordatenbanken

Vektordatenbanken bilden eine spezialisierte Klasse von Datenbanksystemen, die darauf ausgelegt sind, hochdimensionale Vektorrepräsentationen effizient zu speichern, zu indexieren und durchsuchbar zu machen. Im Kontext von RAG-Systemen fungieren sie als zentrale Infrastruktur für die Speicherung und den Abruf von Embedding-Vektoren, die semantische Informationen von Textdokumenten repräsentieren. [26, S. 1-2]

3.3.1. Architektur und Funktionsweise

Im Gegensatz zu relationalen Datenbanken speichern Vektordatenbanken Daten als numerische Vektoren in hochdimensionalen Räumen. Jeder Vektor repräsentiert semantische Eigenschaften eines Datenobjekts [26, S. 1], wobei die räumliche Nähe zwischen Vektoren die semantische Ähnlichkeit der ursprünglichen Inhalte widerspiegelt. Diese Eigenschaft ermöglicht komplexe Ähnlichkeitsabfragen ohne exakte Schlüsselwort-Übereinstimmungen [26, S. 2].

Die Architektur umfasst eine Speicherschicht für persistente Vektordaten, eine Indexierungsschicht für effiziente Organisation sowie eine Abfrageschicht für Ähnlichkeitssuchen. Zusätzlich unterstützen moderne Systeme die Speicherung von Metadaten für hybride Filterung und Verfeinerung der Suchergebnisse. [27, S. 1-4]

3.3.2. Indexierung und Suchoptimierung

Für die effiziente Durchsuchung großer Vektorbestände setzen Vektordatenbanken spezialisierte Indexierungsalgorithmen ein. Hierarchical Navigable Small World (HNSW)-Graphen, eingeführt von Yu A Malkov et al. [28] ermöglichen durch mehrschichtige Navigationsstrukturen sublineare Suchzeiten bei hoher Genauigkeit. Alternative Verfahren wie Locality-Sensitive Hashing (LSH) bieten je nach Anwendungsfall spezifische Vorteile hinsichtlich Speichereffizienz oder Suchgeschwindigkeit. [26, S. 4-7]

Die Approximate Nearest Neighbor (ANN)-Suche bildet das methodische Fundament dieser Verfahren [28, S. 1-3]. Da exakte Bestimmung der nächstgelegenen Nachbarn in hochdimensionalen Räumen rechnerisch aufwendig ist, approximieren diese Algorithmen die Ergebnisse mit kontrollierbarer Genauigkeit bei reduzierten Rechenzeiten [28, S. 1-5], [29, S. 1, 2, 10].

3.3.3. Technische Implementierung

Moderne Vektordatenbanken bieten standardisierte APIs für sowohl Batch-Import großer Dokumentenmengen als auch Echtzeitabfragen. Die Systeme unterstützen verschiedene Distanzmetriken wie Kosinus-Ähnlichkeit, Euklidische Distanz oder Dot-Product zur Berechnung der Vektorähnlichkeit. Aktuelle Implementierungen bieten zusätzlich Funktionalitäten wie versionierte Vektorbestände, horizontale Skalierung und Multi-Tenancy-Fähigkeiten für unternehmenskritische Anwendungen.[30, S. 2-4], [31]

3.4. Retrieval Augmented Generation

Retrieval-Augmented Generation (RAG) kombiniert die Stärken von LLMs mit dem gezielten Zugriff auf externe Wissensquellen. Klassische LLM-Modelle schöpfen ausschließlich aus dem Trainingswissen und können aktuelle oder spezielle Informationen nicht einbeziehen [3, 1], was bei neuen oder spezialisierten Fragestellungen zu falschen oder „halluzinierten„, also erfundenen, Antworten führen kann [3, 1-2], [32, S. 1, 3, 20], [33].

RAG-Systeme hingegen durchsuchen vor jeder Antwort eine hinterlegte Wissensbasis (z.B. Dokumentensammlung, Datenbank oder Internetsuche) nach relevanten Textpassagen und übergeben diese als zusätzlichen Kontext an das LLM. So lassen sich aktuelle Fakten und spezialisierte Informationen direkt einbinden, ohne das LLM neu trainieren zu müssen, was Präzision und Nachvollziehbarkeit deutlich erhöht [3, 1-2], [Abbildung 8].

3.4.1. Wissensabruf und Anreicherung

Im RAG-Verfahren wird zu jeder Anfrage die Wissensbasis nach relevanten Textpassagen durchsucht, die zusammen mit der Frage als zusätzlicher Kontext an das LLM übergeben werden. Das LLM kann dann die abgerufenen Fakten direkt in seine Antwort einbetten [34, S. 9]. So können auch aktuelle Informationen, wie neueste Forschungsergebnisse, einfließen, ohne dass das LLM diese im Training lernen musste. Die Antworten basieren auf verifizierten Quellen und bleiben aktuell, da neue Daten einfach in die Wissensbasis aufgenommen werden können, was Qualität und Aktualität gerade bei nischen Themen und neuen Erkenntnissen erhöht [34, S. 9], [35, S. 8].

3.4.2. Retrieval-Verfahren und Suchstrategien

Die Qualität des Retrievals bestimmt maßgeblich die Verlässlichkeit der RAG-Antworten [36, S. 9]. Dokumente werden zunächst in passageartige Einheiten segmentiert und für die Suche aufbereitet. Bei klassischen Information-Retrieval-Verfahren (sparse Retrieval) kommen TF-IDF und BM25 zum Einsatz:

- **TF-IDF**: Gewichtet Terme durch Multiplikation der Termhäufigkeit mit dem inversen Dokumenthäufigkeitsmaß, sodass häufige Terme abgeschwächt und seltene Terme hervorgehoben werden [37, S. 12, 13, 15], [38, S. 347-352].
- **BM25**: Führt gesättigte Termfrequenz und Dokumentlängennormalisierung ein, um übermäßige Gewichtung und unverhältnismäßige Bevorzugung langer Dokumente zu verhindern [38, S. 352-369].

Diese Verfahren zeigen robuste Leistung, stoßen jedoch bei semantisch anspruchsvollen Anfragen oder Paraphrasen an ihre Grenzen, da sie primär auf exakten Wortüberlappungen beruhen [35, S. 1].

Modernes dichtes Retrieval (dense Retrieval) bildet Fragen und Dokument-Passagen in einen gemeinsamen Vektorraum ab. Duale Encoder (Fragen Encoder und Passagen Encoder) auf Transformer-Basis werden darauf trainiert, semantisch ähnliche Frage-Passage-Paare im Vektorraum zu verorten. Der Dense Passage Retriever (DPR) von Karpukhin et al. [35]) zeigt, dass solche Verfahren herkömmliche BM25-Systeme in der Retrieval-Genauigkeit deutlich übertreffen können [35, S. 1-3].

Für die technische Umsetzung des dichten Retrievals werden die in Kapitel 2.2 beschriebenen Vektordatenbanken eingesetzt, die eine effiziente Speicherung und Durchsuchung der Embedding-Vektoren ermöglichen. Hybride Strategien kombinieren schnelle sparse Vorfilterung per BM25 mit dichter Feinsortierung über Vektordatenbanken, um Effizienz und Präzision zu balancieren. Metadaten können als Filterbedingung einfließen, um irrelevante oder veraltete Passagen frühzeitig auszuschließen [39, S. 1-4].

3.4.3. Generative Antworterstellung

Im RAG-System bildet die generative Antworterstellung den abschließenden Verarbeitungsschritt, in dem das vortrainierte Sprachmodell die ursprüngliche Nutzerfrage mit den durch das Retrieval-System identifizierten relevanten Dokumentfragmenten zu einem einzigen, kontextualisierten Prompt kombiniert und darauf basierend regressiv den finalen Antworttext generiert.[34, S. 5-7]

Für die Qualität der Antwort spielt die Wahl des LLM eine entscheidende Rolle. Neue LLMs führender Anbieter können Zusammenhänge besser verstehen und detailliertere Antworten geben als bisherige Modelle. Ebenso wichtig ist die Größe des Kontextfensters – also wie viele Informationen das LLM gleichzeitig verarbeiten kann. Ist dieses Fenster zu klein, gehen wichtige Quellenpassagen verloren. Ist es zu groß, wird das System langsamer und verbraucht mehr Ressourcen, ohne dass die Antworten proportional besser werden [40, S. 1-4].

Durch die unmittelbare Integration der abgerufenen Textpassagen in den Generierungsprozess minimiert der RAG-Ansatz das Auftreten von Halluzinationen erheblich und gewährleistet, dass jede Antwort faktenbasiert und durch konkrete Quellen verifizierbar bleibt. Dieser Mechanismus eliminiert die Notwendigkeit aufwendiger Nachtrainingsverfahren des Sprachmodells, da neue oder aktualisierte Informationen direkt über die Wissensbasis eingebunden werden können. [41, S. 1]

Die Transparenz des Verfahrens ermöglicht es darüber hinaus, die verwendeten Quelldokumente zu referenzieren, wodurch Nutzer die Möglichkeit erhalten, die faktische Grundlage der generierten Antworten selbst zu überprüfen und das Vertrauen in die Systemausgaben zu stärken.

3.4.4. Retrieval-Augmented-Generation Parameter

Die Leistung eines RAG-Systems werden durch die Wahl mehrerer Parameter beeinflusst. Jeder Parameter wirkt sowohl eigenständig als auch in Wechselwirkung mit den anderen. Folgende Parameter sind von besonderer Relevanz:

- **LLM**: Das verwendete LLM zur Generierung von Antworten auf Fragen beeinflusst Genauigkeit und Kohärenz der Antwort. Je nach Komplexität der Aufgabe eignen sich verschiedene Modelle unterschiedlich gut, wodurch deren Auswahl entscheidend ist und in Abhängigkeit von Domäne und Rahmenbedingungen variiert werden sollte. [42, S.]
- **Embedding-Modell**: Die Wahl des Embedding-Modells beeinflusst das Retrieval von notwendigen Informationen [43]. Präzise Embeddings verbessern das Ranking relevanter Chunks und verringern Halluzinationen im Zusammenspiel mit dem LLM [44, S. 2-3].
- **Chunkgröße**: Die Chunkgröße legt fest, wie lang die einzelnen Textabschnitte sind, die vor der Umwandlung in Vektoren segmentiert werden. Kleinere Chunks erhöhen Präzision, wobei große Chunks den Zusammenhang bewahren. Ein optimales Verhältnis entsteht durch Ausbalancieren von Kontextumfang, irrelevanter Informationen und Kontextnutzung im LLM. [45, S. 1]
- **Overlap**: Overlap bezeichnet die Anzahl der überlappenden Tokens zweier aufeinanderfolgender Chunks und zielt darauf ab, Kontextverluste an Chunk-Grenzen zu vermeiden [42, S.]. Ein ausreichender Overlap erhöht die Wahrscheinlichkeit, dass relevante Informationen nicht zwischen zwei Chunks verloren gehen, wobei damit ein erhöhter Ressourcenaufwand und Redundanz einhergeht [44, S. 2].
- **Top-k-Retrieval Parameter**: Definiert, die Anzahl der betrachteten Chunks (k) die bei einer Abfrage vom RAG-System zurückgegeben werden. Ein niedriger Wert kann relevante Ergebnisse ausschließen, während ein zu hoher Wert irrelevante Informationen mit einbezieht. [42, S. 2]

3.5. Retrieval Augmented Generation Evaluation

Um die Leistung eines RAG-Systems messbar zu machen, werden geeignete Evaluationsmetriken bestimmt. Diese spalten sich in Klassifikationsmetriken, zur Klassifizierung richtiger und falscher Ergebnisse, sowie Token-Similarity-Metriken zum Vergleich der Ähnlichkeit von Referenz- und LLM-Antwort. [46, S. 6-7]

Bei der Klassifizierung werden Daten in eine oder mehrere Klassen zugeordnet und anhand einer Referenzklassifizierung evaluiert [46, S. 6]. Hierbei wird zwischen Einfach- und Mehrfach-Klassifizierung unterscheiden. Bei Einfach-Klassifizierung wird einem Datensatz eine Klasse zugeordnet, bei Mehrfach-Klassifizierung können einem Datensatz mehrere Klassen zugeordnet werden.

Token-Similarity Metriken evaluieren Wortsequenzen. Dafür werden Metriken wie Recall-Oriented Understudy for Gisting Evaluation (ROUGE) eingesetzt, die eine Wortsequenz gegenüber einer Referenzquelle evaluiert und anhand der Überschneidung einen Wert bestimmt [47, S. 1-2].

3.5.1. Precision, Recall und F1-Score

Precision, Recall und F1-Score sind drei weit verbreitete Metriken zur Evaluation von Klassifikationen durch ein Modell gegenüber einer Referenzquelle [48, S. 5]. Sie geben Aufschluss, inwiefern eine Modellklassifikation der Referenzklassifikation entspricht [49, S. 6], [50, S. 7-9], [51, S. 8].

Sei c eine Klasse, L die Menge an Klassen $\{\mathrm{c}_1, \mathrm{c}_2, ...\mathrm{c}_\mathrm{L}\}$ inklusive der Null-Klasse $\mathrm{c}_\emptyset$ bei fehlender Label-Zuweisung, $|\mathrm{L}|$ die Kardinalität der Menge L, $\mathrm{P} \subseteq \mathrm{L}$ die Menge an vorhergesagten Klassen und $\mathrm{H} \subseteq \mathrm{L}$ die Menge an tatsächlichen Klassen. Bei $|\mathrm{L}| = 1$ kann eine LLM-Vorhersage als klassifiziert oder nicht klassifiziert angesehen werden, wodurch eine binäre Zuordnung in positiv und negativ erlaubt wird. Dies wird für P und H durchgeführt und den in Tabelle 1 dargestellten Mengen je LLM-Vorhersage zugewiesen. Anhand dieser Mengen kann eine quantitative Bewertung der Klassifikation berechnet werden [46, S. 7-9].

	Tatsächlich positiv	**Tatsächlich negativ**
Als positiv klassifiziert	True Positive (TP)	True Negative (TN)
Als negativ klassifiziert	False Positive (FP)	False Negative (FN)

Tabelle 1: Confusion Matrix für eindimensionale Klassifizierung je Klasse [4, S. 3]

Precision gibt an, welcher Anteil der als positiv identifizierten Ergebnisse tatsächlich positiv ist. Sie misst die Fähigkeit eines LLMs, negative Instanzen zu filtern. Die Formel der Precision ist im Folgenden dargestellt. Hierbei entspricht $|\text{Menge}|$ der Kardinalität einer Menge: [46, S. 8]

$$\text{Precision} = \frac{\text{korrekt als positiv klassifizierte Einträge}}{\text{als positiv klassifizierte Einträge}} = \frac{|\text{TP}|}{|\text{TP}| + |\text{FP}|} \tag{1}$$

Recall, auch bekannt als „True Positive Rate“ oder „Sensitivity“, beschreibt, wie vollständig relevante Ergebnisse erkannt wurden. Sie bestimmt, inwiefern ein Modell positive Ergebnisse identifiziert. Die Formel für Recall lautet: [46, S. 7-8]

$$\text{Recall} = \frac{\text{korrekt als positiv klassifizierte Einträge}}{\text{tatsächlich positive Einträge}} = \frac{|\text{TP}|}{|\text{TP}| + |\text{FN}|} \tag{2}$$

Der F1-Score bildet das harmonische Mittel zwischen Precision und Recall. Die Formel für den F1-Score lautet: [46, S. 8]

$$\text{F1-Score} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} = \frac{2 \cdot |\text{TP}|}{2 \cdot |\text{TP}| + |\text{FP}| + |\text{FN}|} \tag{3}$$

3.5.2. Recall-Oriented Understudy for Gisting Evaluation

ROUGE ist eine Evaluationsmetrik des NLP, welche zur Evaluation von generierten Texten gegenüber eines Referenztextes angewendet wird [46, S. 11-12]. Sie quantifiziert die Ähnlichkeit zweier Texte in einer Metrik und ermöglicht gegenüber Klassifikationsmetriken die Evaluation von Auswirkungen der Modellierung auf Freitexte [46, S. 14]. ROUGE-n, eine Umsetzung von ROUGE, misst den Überlappungsgrad zwischen einem generierten Text zu einer Referenz mittels N-Grammen, welche Wortfolgen der Länge N repräsentieren [47, S. 1-2]. Formal entspricht ROUGE-n dem in Abschnitt 3.5.1 definierten F1-Score, wobei anstatt absoluter Zuordnung die Übereinstimmung von N-Grammen als True Positive herangezogen werden [46, S. 14].

Sei ω ein Wort, $\mathrm{b} = \left(\omega_1, \omega_2, ..., \omega_{\Lambda_\mathrm{b}}\right)$ ein vorhergesagter Text der Länge Λ_b als Wortsequenz von ω_1 bis $\omega_{\Lambda_\mathrm{b}}$ und $\mathrm{d} = \left(\omega_1, \omega_2, ..., \omega_{\Lambda_\mathrm{d}}\right)$ ein Referenztext der Länge Λ_d als Wortsequenz von ω_1 bis $\omega_{\Lambda_\mathrm{d}}$. Damit ist B die Menge aller n-gramme im vorhergesagten Text, D die Menge aller n-gramme im Referenztext und $\mathrm{D} \cap \mathrm{B}$ die Menge an überschneidenden n-Grammen, welche der Menge TP auf Basis von N-Grammen entsprechen. Die im generierten Text vorkommenden, aber nicht in der Referenz enthaltenen N-Gramme $\mathrm{B} \setminus \mathrm{D}$ entsprechen der Menge FP, in der Referenz, aber nicht im generierten Text enthaltene N-Gramme $\mathrm{D} \setminus \mathrm{B}$ der Menge FN und weder in B noch in D vorkommende N-Gramme der Menge TN. Auf Basis dieser N-Gramm basierten Mengen lassen sich $\mathrm{Precision}_n$, Recall_n sowie ROUGE_n als N-Gramm basierter F1-Score mit den folgenden Formeln berechnen: [46, S. 14]:

$$\mathrm{Precision}_n = \frac{|\mathrm{D} \cap \mathrm{B}|}{|\mathrm{B}|} \tag{4}$$

$$\mathrm{Recall}_n = \frac{|\mathrm{D} \cap \mathrm{B}|}{|\mathrm{D}|} \tag{5}$$

$$\text{ROUGE-n} = 2 \cdot \frac{\mathrm{Precision_n} \cdot \mathrm{Recall_n}}{\mathrm{Precision_n} + \mathrm{Recall_n}} \tag{6}$$

Die Aussagekraft von ROUGE-n ist begrenzt: im Gegensatz zu Metriken wie LLM-as-a-Judge, die die semantische Bedeutung eines Textes bewerten, misst ROUGE-n lediglich die Überlappung von N-Grammen und kann daher nicht die Qualität der Inhalte oder deren Relevanz für eine bestimmte Aufgabe bewerten [46, S. 14-15]. Aus diesem Grund wird ROUGE-n häufig in Kombination mit anderen Metriken eingesetzt, um eine umfassendere Bewertung der Qualität von generierten Texten zu ermöglichen [46, S. 15].

3.5.3. Large Language Model as a Judge

LLM as a Judge bezeichnet einen Evaluationsansatz, bei dem LLMs zur Bewertung von Ergebnissen anderer Modelle eingesetzt werden [52, S. 1-2]. Das Konzept basiert auf „Stacking“ [53, S. 6-15], bei dem mehrere Modelle squentiell hintereinander geschaltet werden und sich gegenseitig bewerten und vervollständigen [54, S. 6]. LLM-as-a-Judge nutzt diesen Ansatz, indem ein LLM nach der Produktion eines LLM-Outputs eingesetzt wird, um anhand definierter Kriterien diesen zu bewerten und quantitative Metriken zuzuweisen [52, S. 2-3].

LLM-as-a-Judge weist Vorteile im Gegensatz zu traditionellen regelbasierten Bewertungsmethoden wie BLEU oder ROUGE auf, da LLMs deutlich besser in der Lage sind, angepasst an dynamischen Umständen und unstrukturierten Texten, diesen zu evaluieren [55, S. 2123-2126]. Ebenfalls kann LLM-as-a-Judge die semantische Bedeutung von Texten besser erfassen und komplexe Zusammenhänge verstehen, wodurch eine differenziertere Bewertung möglich ist [52, S. 2-3]. Darüber hinaus weisen durch LLM-as-a-Judge getroffene Evaluationen hohe Korrelationen mit menschlicher Bewertung auf, wodurch hohe Kosten einer manuellen Evaluation in Teilen eingespart werden können [56, S. 1]. [57, S. 8-11].

Jedoch ist der Einsatz von LLM-as-a-Judge durch seine Verwendung von LLMs mit Herausforderungen wie Halluzination und nicht-deterministische LLM-Ausgaben verbunden [52, S. 4-5]. Hinzu kommt, dass die Qualität der Bewertung stark von der Fähigkeit des LLMs abhängt, die bewerteten Aufgaben selbst zu lösen, wodurch die Wahl des LLM-as-a-Judge-Modells stark abhängig von der zu bewertenden Aufgabe gewählt werden muss [56, S. 1-3].

Für eine praktische und zuverlässige Anwendung von LLM-as-a-Judge ist es daher unabdingbar, eine Validierung mit von Menschen annotierten Referenzdaten durchzuführen, um ein geeignetes LLM zu wählen und systematische Fehlerquellen der Evaluation zu reduzieren [56, S. 1-3]. Darüber hinaus soll LLM-as-a-Judge nicht als alleinige Evaluationsmethode eingesetzt werden, sondern in Kombination mit anderen Metriken wie ROUGE, um eine umfassendere Bewertung der Qualität von generierten Texten zu ermöglichen und mögliche Verzerrungen durch das evaluierende Modell zu minimieren [52, S. 2-3].

4. Praktische Umsetzung

4.1. Business Understanding

Besonders im akademischen Bereich stehen LLMs vor der Herausforderung, dass sie nur über die Informationen der Trainingsphase verfügen. Dies führt zu stark eingeschränkter Leistung, wenn hochspezialisiertes Wissen oder neue Erkenntnisse eingebunden werden sollen [34, S. 9], [35, S. 8]. Für dieses Problem bieten die in Abschnitt 3.4 beschriebenen RAG-Systeme eine mögliche Lösung. Ziel dieser Arbeit ist es, zu untersuchen, inwieweit RAG-Systeme dazu geeignet sind, Studierenden und Wissenschaftlern schnellen Zugang zu aktueller Forschung zu ermöglichen und literaturbezogene Fragen zuverlässig zu beantworten.

Dazu wird ein eigenes RAG-System entwickelt, das es erlaubt, Literatur hochzuladen und gezielt Fragen dazu zu stellen. Die Qualität der Antworten wird evaluiert, um das Potenzial solcher Systeme für den akademischen Einsatz realistisch einschätzen zu können.

Für die Evaluation werden die in Abschnitt 3.5 genannten Metriken für die Quantifizierung der Ergebnisse genutzt:

- **N-Gramm-basierten Metriken**: Metriken wie Precision-n, Recall-n und ROUGE-n vergleichen die generierten Antworten mit Referenzantworten, indem sie die Übereinstimmung von N-Grammen messen. Diese wird als Wert zwischen 0 und 1 wiedergegeben.
- **LLM-as-a-Judge**: Diese Metrik nutzt ein LLM als Bewertungsinstanz, um die Qualität der generierten Antworten zu beurteilen. Das LLM bewertet die Antworten auf Basis der Kriterien „Total Correctness“, „Completeness“, „Relevance“, „Justfiication“ und „Depth“, indem dieses eine numerische Punktzahl zwischen 0 und 2 je Kategorie vergibt.

4.2. Data Understanding

Zur systematischen Evaluation des entwickelten RAG-Systems wurde eine kuratierte Auswahl wissenschaftlicher Publikationen als Testdatensatz verwendet. Die Datengrundlage besteht aus fünf aktuellen Veröffentlichungen aus dem Bereich der Informatik, die von der wissenschaftlichen Publikationsplattform ArXiv.org bezogen wurden. Sämtliche ausgewählten Publikationen weisen ein einheitliches Veröffentlichungsdatum vom 27. Juni 2025 auf.

Diese zeitliche Nähe zum Evaluationszeitpunkt (29. Juni 2025) stellt eine methodisch wichtige Kontrolle dar, da gewährleistet wird, dass die Inhalte dieser Publikationen mit hoher Wahrscheinlichkeit nicht Bestandteil der Trainingsdaten gängiger LLMs sind. Dadurch wird eine realistische und unvoreingenommene Evaluation der RAG-Funktionalität ermöglicht, bei der das System tatsächlich auf die bereitgestellten Dokumente angewiesen ist, anstatt auf bereits internalisiertes Wissen zurückgreifen zu können.

Zur Validierung dieser Annahme wurde das eingesetzte LLM explizit zu seiner Kenntnis der ausgewählten Publikationen befragt. Ohne Zugriff auf das RAG-System bestätigte das Modell konsistent seine Unkenntnis bezüglich der spezifischen Inhalte aller fünf Publikationen, was die methodische Grundlage der Evaluation stärkt.

Die folgenden Publikationen wurden ausgewählt und decken verschiedene Bereiche der Informatik ab:

- „Engineering RAG Systems for Real-World Applications“ [5] untersucht die praktische Implementierung von RAG-Systemen in fünf verschiedenen Domänen (Governance, Cybersecurity, Agriculture, Industrial Research und Medical Diagnostics) und präsentiert zwölf Lessons Learned aus der Entwicklung dieser Systeme.
- „MAGPIE: A dataset for Multi-AGent contextual PrIvacy Evaluation“ [58] präsentiert einen Benchmark-Datensatz mit 158 realistischen Multi-Agenten-Szenarien zur Evaluation der kontextuellen Privatsphäre in LLM-basierten Agentensystemen.
- „Adaptive Hybrid Sort: Dynamic Strategy Selection for Optimal Sorting“ [59] stellt einen adaptiven Sortieralgorithmus vor, der basierend auf Dateneigenschaften wie Größe, Wertebereich und Entropie dynamisch zwischen Counting Sort, Radix Sort und QuickSort wählt.

- „Scalable GPU Performance Variability Analysis framework“ [60] entwickelt ein verteiltes Framework zur Analyse von GPU-Performance-Logs, das große SQLite3-Tabellen in Shards partitioniert und parallel über Message-Passing-Interface-Ranks verarbeitet.
- „The Singapore Consensus on Global AI Safety Research Priorities“ [61] dokumentiert die Ergebnisse der 2025 Singapore Conference on AI mit über 100 Teilnehmern aus 11 Ländern. Das Konsens-Dokument strukturiert AI Safety Forschungsprioritäten in drei Hauptbereiche: Risk Assessment, Development und Control, und identifiziert Bereiche gegenseitigen Interesses für internationale Kooperation.

Diese Auswahl repräsentiert aktuelle Forschungsthemen aus verschiedenen Informatikbereichen und eignet sich damit gut zur Überprüfung der Generalisierbarkeit des entwickelten RAG-Systems.

4.3. Data Preparation

Für jedes der fünf ausgewählten Paper werden fünf inhaltliche Fragen erstellt, die direkt auf das Verständnis und die Kernaussagen des jeweiligen Textes abzielen. Die Fragen und Antworten aus den Publikationen finden sich im Anhang in Abschnitt ii.iv und wurden von Menschen mit der Hilfe von LLMs beantwortet. Die Fragen sind so konzipiert, dass sie ohne Zugriff auf das jeweilige Paper weder für Menschen noch für LLMs komplett zu beantworten sind, da sie spezifische technische Details, Evaluationsergebnisse oder methodische Entscheidungen betreffen.

Insgesamt liegt somit ein Datensatz von 35 Fragen (5 × 5 Inhlatsfragen, 5 × 2 Metadatenfragen) vor. Dieser dient im Weiteren als Basis für die Entwicklung des RAG-Systems und Bewertung der Antwortgenauigkeit in Abschnitt 4.5.

4.4. Modelling

Im Kontext von RAG erfordert die Modelling Phase die systematische Auswahl und Konfiguration der Retrieval Komponente als auch der generativen Sprachmodelle um eine möglichst passende Harmonisierung zwischen dem Abruf relevanter Informationen und der kontextuellen Textgenerierung zu erreichen. Für die Entwicklung des Modells und die darauffolgende Analyse wurden folgende Technologien verwendet:

- **Programmiersprache Python:** Python eignet sich für AI- und Data-Science-Anwendungen, da es eine klare Syntax, sowie eine breite Auswahl leistungsfähiger Bibliotheken bietet [62, S.141-S.143]. Zusätzlich ermöglicht die Programmiersprache eine schnelle Prototypentwicklung, modulare Strukturierung und eine gute Integrierbarkeit in bestehende Systeme [62, S.141-S.143]. Python wurde für die Implementierung des RAG-Systems und die Analyse der Optimierungsstrategien genutzt.

- **Qdrant Vektordatenbank:** Als Vektordatenbank wurde die QDrant-Vektordatenbank gewählt, da diese speziell für semantische Suche und Ähnlichkeitsabfragen optimiert ist [63]. Zusätzlich unterstützt die Qdrant Vektordatenbank die Speicherung hochdimensionaler Vektoren und ermöglicht schnelle, skalierbare Top-k-Abfragen mittels Cosine Similarity oder anderer Distanzmetriken [63]. Durch ihre Open-Source-Natur und einfache Integration lässt sich Qdrant auch ohne komplexe Infrastruktur in kleinere, experimentelle RAG-Systeme einbinden [63].

- **Embedding Modell text-embedding-3-large:** Das Modell text-embedding-3-large von OpenAI zeichnet sich durch eine Embedding-Dimension von 3072 Dimensionen aus [64]. Des weiteren wird dieses Modell für präzise und höchste Genauigkeit in einem RAG-System genutzt [65]. Damit verbunden ist allerdings auch ein hoher Rechenaufwand und Kostenaufwand, im Vergleich zu den anderen Embedding-Modellen, verbunden [64].

Zur Generierung der Antworten mithilfe des RAG-Systems werden, wie in Abschnitt 3.1 erläutert, LLMs, augrund ihrer hohen Performance gegenüber alternativen Verfahren eingestetzt. Vom Training eines eigenen Modells wird aufgrund von geringer Datengrundlage sowie Kosten- und Recheneleistung abgesehen und verschiedene Large-Scale Pre-Trained Language Models (PLMs) eingesetzt [66, S.3648,3649].

- **Generierendes Modell GPT-4o:** GPT-4o von OpenAI, veröffentlicht am 13. Mai 2024, zeichnet sich durch eine hohe Leistungsfähigkeit aus [67]. Nach den Angaben von OpenAI ist das Modell GPT-4o das beste und leistungsfähigste Modell außerhalb der O-Serie [67]. Im Vergleich zu diesen Modellen, wie beispielsweise o3-mini, ist GPT-4o schneller in der Sprachverarbeitung [67].

- **Bewertendes Modell GPT-4o:** Als bewertendes Modell für den LLM-as-a-Judge Ansatz wurde ebenfalls GPT-4o von OpenAI gewählt. Diese Modell bewertet, wie in Abschnitt 3.5.3, die zuvor generieten Antworten des RAG-Systems.

- **Prompting:** Die Erstellung des Prompts für das RAG-System erfolgte nach den in der Literatur definierten Kriterien (siehe Abschnitt 3.1.2). Das Prompt-Template (Prompt 1), wurde dabei in allen Iterationen des RAG-Systems konsistent eingesetzt.

Das angehängte Prompt-Template (Prompt 2) zeigt den Prompt für den LLM-as-a-Judge Ansatz. Dieser wurde ebenfalls nach den in der Literatur definierten Kriterien (siehe Abschnitt 3.1.2) erstellt und in allen Iterationen des RAG-Systems konsistent eingesetzt.

Für die RAG-Parameter gibt es keine expliziten Vorgaben für die Wahl der RAG Parameter [68]. Diese sind Use-Case und Anforderungsspezifisch [68]. Für den vorliegenden Use-Case müsste man durch mehrere Experiment die beste Konfiguration der kombinierten Parameter ermitteln, da dies jedoch nicht der Fokus der Arbeit ist, soll im Folgenden die Wahl der Paramter begründet werden.

- **Top-k-Retrieval Parameter:** Für die Implementierung des RAG-Systems wurde der Top-k-Retrieval Parameter auf den Wert 5 gesetzt. Diese Entschidung basiert auf der in der Fachliteratur empfohlenen Praxis, verschiedene k-Werte systematisch zu evaluieren und einen ausgewogenen Mittelwert zu wählen [68]. Der Wert k=5 stellt dabei einen Kompromiss zwischen einem niedrigen k-Wert (k=1-2), die relevante Informationen übersehen könnten, und einem zu hohen k-Wert (k>=10), die zu einer Verschlechterung der Präzision durch irrelevante Treffer führen können dar [68].

- **Chunk Size/Overlap:** Für die Konfiguration der Chunk-Size des RAG-Systems wird eine Chunk-Size von 1024 Tokens mit einem Overlap von 128 Token gewählt. Erfahrungen aus einem vergleichbaren Use-Case zeigenm dass diese naive Chunking-Methode pas-

sende Retrieval-Ergebnisse liefert [69]. Gleichzeitig ermöglicht die gewählte Chunkgröße in Kombination mit einem Obverlap von 128 Token eine schnelle Verarbeitung bei hoher Genauigkeit. Daher stellt die Konfiguration mit 1024 Tokens und 128 Tokens Overlap eine passende Balance zwischen Qualität und Effizienz dar [69].

- **Distanzmetrik:** Als weitverbreitete und zuverlässige Distanzmetrik wird die Cosinus Ähnlichkeit verwendet [70, S.1]. Diese misst den Kosinus Winkel zwischen zwei Vektoren und ist somit geeignet, um semantische Ähnlichkeiten effektiv zu erfassen.

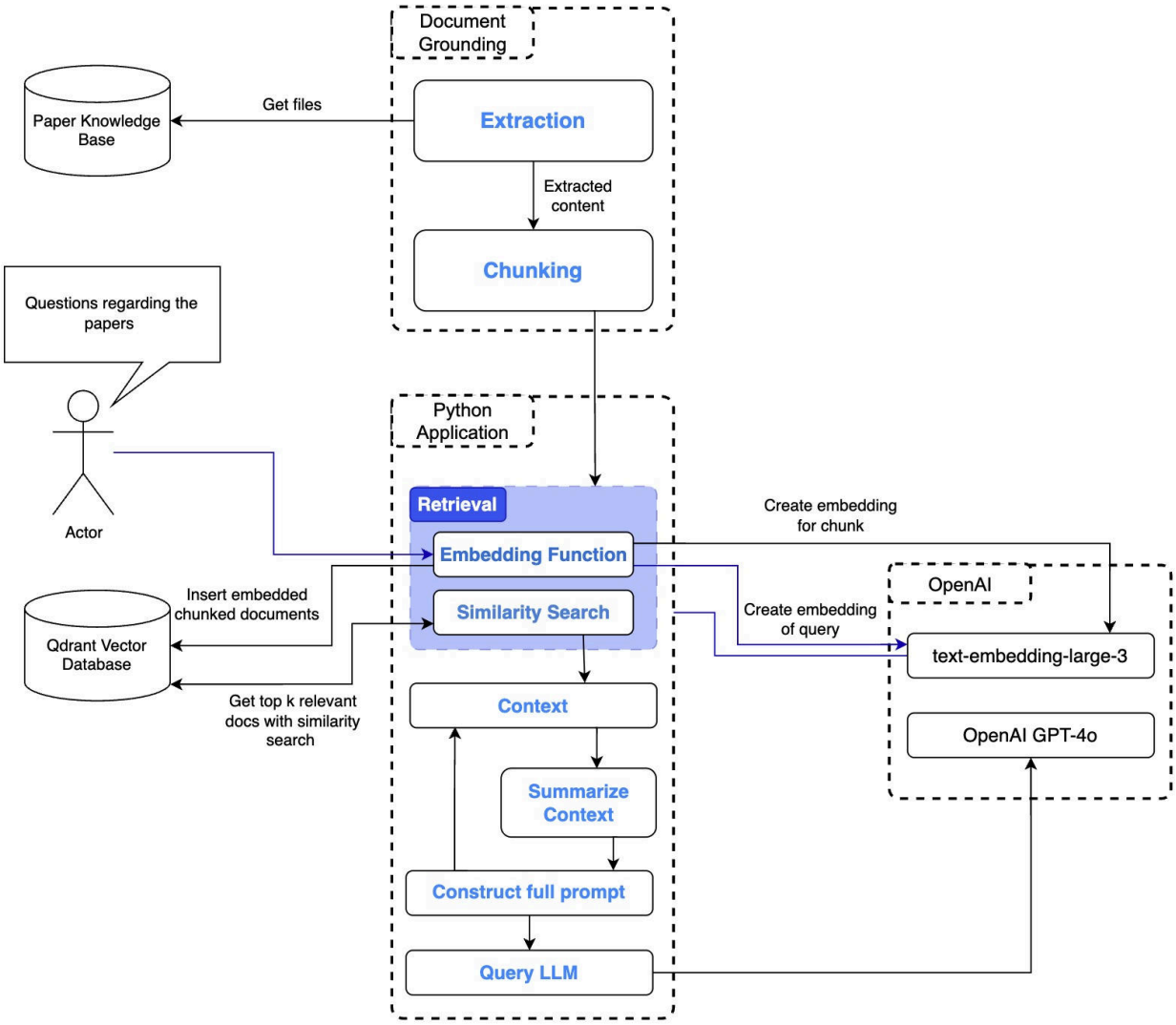

Abbildung 3: Übersicht RAG Architektur. Eigene Darstellung.

Die RAG-Architektur in (Abbildung 3) zeigt die projektspezifische Architektur des RAG-Systems, die zur Beantwortung der in Abschnitt erläuterten Fragen eingesetzt wurde. Im Folgenden soll das Vorgehen näher erläutert werden.

Um die Fragen mit Hilfe des RAG-Systems beantworten zu können, mussten zunächst die Kontextdaten zur Beantwortung der Fragen in die Vektor-Datenbank geladen werden. Die Daten für das Grounding beinhalten produktspezifische Daten von SAP über Joule, aus Dokumentationen, Guides und Blogs. Dazu wurden zunächst die Texte aus den Dokumenten extrahiert, in definierte Chunks aufgeteilt (Größe, Overlap) und anschließend mit Hilfe von drei Embedding-Modellen in unterschiedliche Datenbanken geladen.

Die Fragen wurden zur Beantwortung aus einer -Datei ausgelesen, in Vektor-Embeddings umgewandelt und der Similarity Search zur Suche übergeben. Je nach aufgestellter Hypothese wurde entweder ein unterschiedliches Prompt-Template, ein Paramater wie Chunk-Größe oder der Retrieval-Paramter k für die Analyse verändert. Es wurde jeweils nur ein Parameter verändert, während die anderen Parameter konstant gehalten wurden, um eine direkte Korrelation zwischen dem veränderten Parameter und dem daraus resultierenden Ergebnis zu erhalten.

Die durch die Similarity-Search ermittelten Chunks wurden dem Prompt als Text-Kontext mitgegeben, der im Anschluss durch das LLM GPT-4o bearbeitet wurde. Die Ergebnisse des LLM GPT-4o Modells wurden anschließend in eine -Datei geschrieben.

Zur Beurteilung der Antwortqualität wurden die generierten -Dateien, sowie die manuell erstellte „Ground-Truth“-Antwort (Optimalantwort), mit Hilfe von Evaluationsmetriken (ROUGE, Precision und Recall) analysiert. Anschließend wurde die generierte Antwort zusätzlich mit Hilfe des LLM-as-a-Judge Modells Claude-3-Opus hinsichtlich Completeness und Correctness bewertet und mit dem Ergebnis der Evaluationsmetriken verglichen.

4.5. Evaluation

Die Ergebnisse aus Abschnitt 4.4 wurden erfasst und in tabellarischer Form gesammelt. Anschließend wurden diese mit den in Abschnitt 3.5 definierten Metriken ausgewertet. Abbildung 4 präsentiert die über alle Fragen hinweg aggregierten Resultate, differenziert nach N-Gramm-basierten Metriken und LLM-as-a-Judge-Bewertungen. Die Ergebnisse je Frage befinden sich in Abschnitt ii.iii im Anhang.

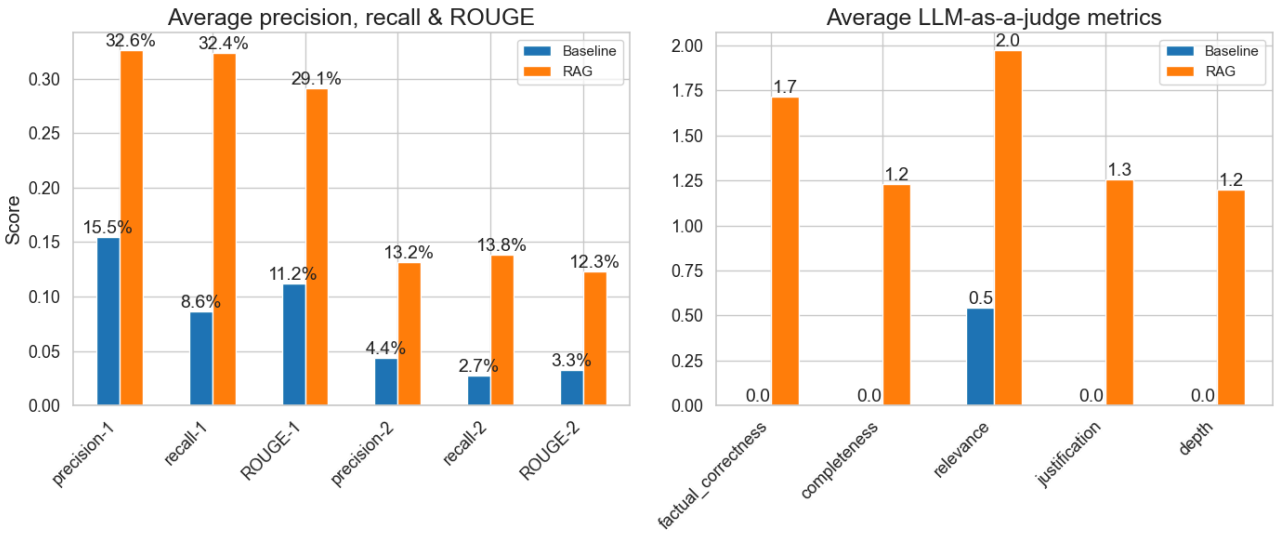

Abbildung 4: Aggregierte Metriken vor/nach RAG-Implementierung. Eigene Darstellung.

Abbildung 4 zeigt, dass beide Metriktypen eine signifikante Steigerung der Antwortqualität infolge der RAG-Implementierung verzeichnen. So erhöht sich beispielsweise die durchschnittliche Precision-1 von 15,5 % auf 32,6 %, Recall-1 von 8,6 % auf 32,4 % und ROUGE−1 von 11,2 % auf 29,1 %. Ähnliche Verbesserungen sind bei den 2-Gramm-Metriken zu beobachten (Precision-2: +8,8 Prozentpunkte (pp), Recall-2: +11,1 pp, ROUGE-2: +9,0 pp).

Die LLM-as-a-Judge-Bewertungen stützen dieses Ergebnis: Während das Baselinesystem in vier von fünf Dimensionen durchschnittlich 0 von 2 Punkten erreicht, erzielt das RAG-System signifikant höhere Werte (1,7 für Faktentreue, 1,2 für Vollständigkeit, 2,0 für Relevanz, 1,3 für Begründung und 1,2 für Tiefe, vgl. Abbildung 10). Dies unterstreicht sowohl eine quantitativ als auch qualitativ verbesserte Antwortgenerierung durch RAG.

Bei der Betrachtung der Ergebnisse auf Ebene der einzelnen Fragen (siehe Abbildung 9 und Abbildung 10) zeigt sich, dass die Verbesserungen nicht gleichmäßig verteilt sind. Einige Fragen profitieren stark von der RAG-Implementierung, während andere nur geringe oder gar keine Verbesserungen zeigen. Dies könnte auf unterschiedliche Schwierigkeitsgrade der Fragen oder auf die Verfügbarkeit relevanter Dokumente zurückzuführen sein.

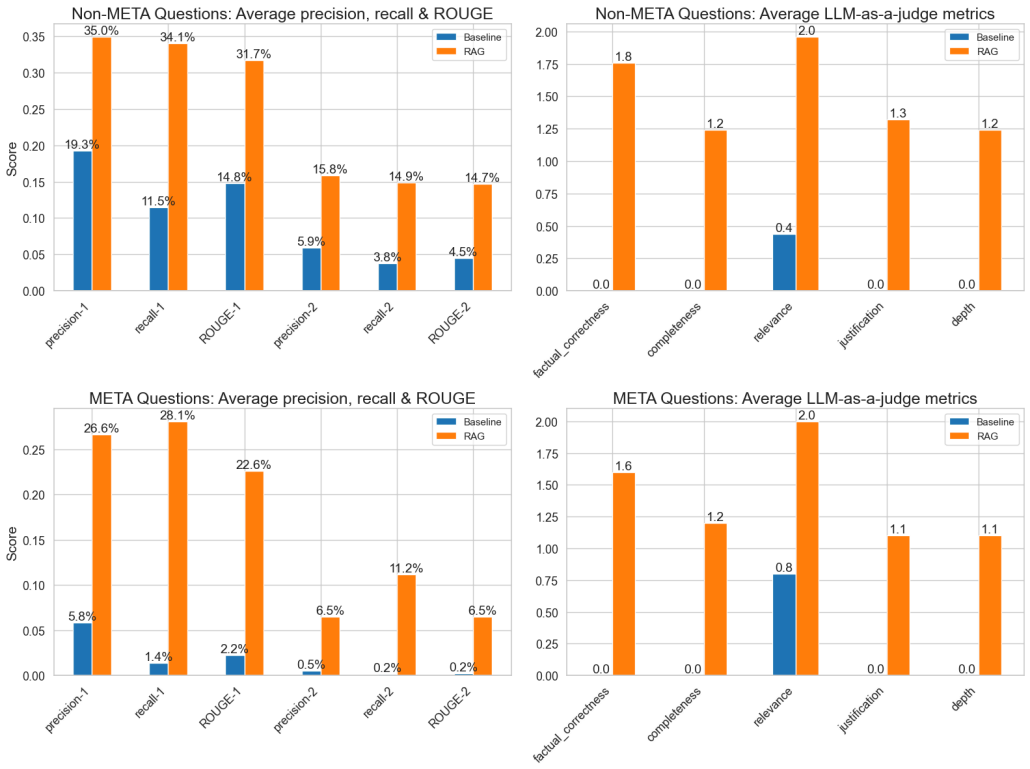

Abbildung 5: Vergleich der Metriken je Fragenart. Eigene Darstellung.

Abbildung 5 verdeutlicht, dass insbesondere Anfragen mit Metadateninhalten stark vom RAG-gestützten System profitieren, selbst wenn die N-Gramm-Metriken insgesamt geringer ausfallen. Dies unterstreicht, dass RAG Metadaten, welche im Vergleich zu Inhaltsfragen einehitlichere Struktur aufweisen, besser verarbeiten kann als spezifische Inhaltsfragen

Abbildung 6 illustriert die Korrelationsbeziehung zwischen den N-Gramm-Metriken und den LLM-as-a-Judge-Bewertungen, wobei letztere zur besseren Vergleichbarkeit als Anteil am Maximalscore (2 Punkte) normalisiert wurden. Dafür wird der Pearson-Korrelationskoeffizient r verwendet, welcher die lineare Beziehung zwischen zwei Variablen misst. $|\mathrm{r}| = 1$ bedeutet eine perfekte Korrelation, während $|\mathrm{r}| = 0$ keine Korrelation anzeigt. [71, S. 75-79]

Die Analyse offenbart eine moderate positive Korrelation zwischen beiden Bewertungsansätzen: Mit steigenden N-Gramm-Werten nehmen tendenziell auch die LLM-Bewertungen zu. Bemerkenswert ist die asymmetrische Bewertungscharakteristik des LLM: Qualitativ unzureichende Antworten werden verstärkt negativ bewertet $(\mathrm{r} = -0.21)$, während besonders hohe N-Gramm-Scores überproportional positiv honoriert werden $(\mathrm{r} = 0.44)$. Diese Beobachtung unterstreicht den komplementären Charakter der LLM-basierten Evaluation zu den rein quantitativen Metriken.

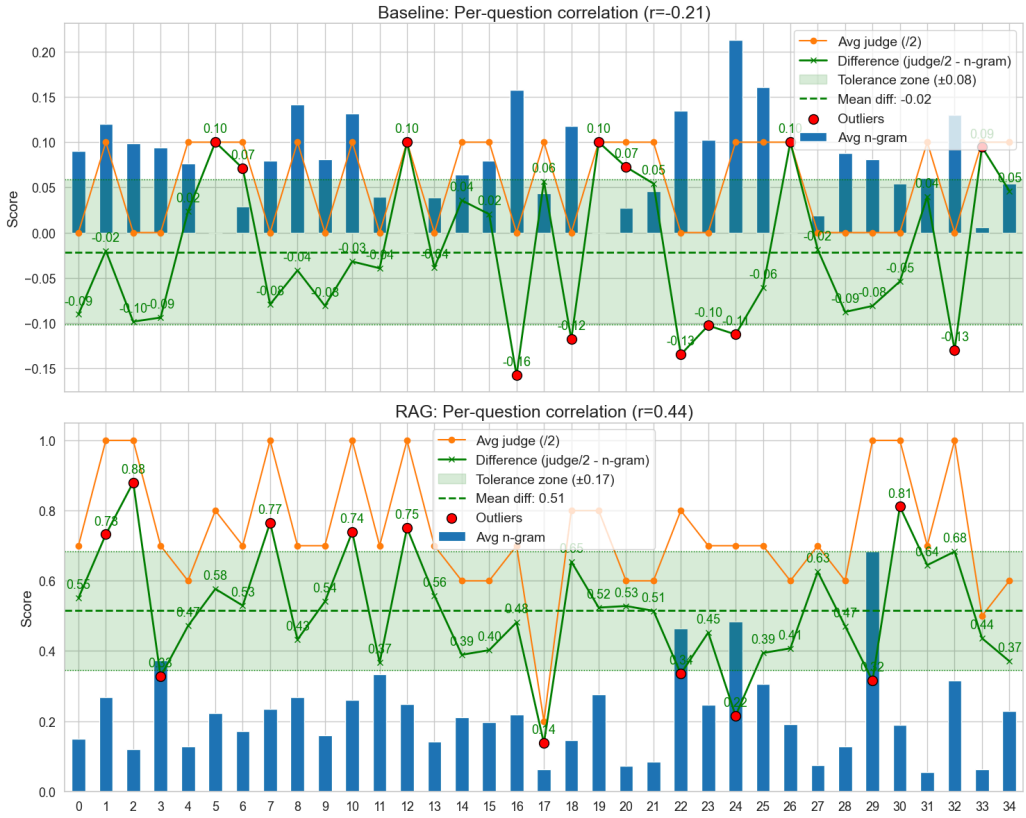

Abbildung 6: Korrelation zwischen N-Gramm/LLM-as-a-Judge. Eigene Darstellung.

Zusammenfassend belegen die Ergebnisse eine signifikante Verbesserung der Antwortqualität durch den Einsatz von RAG. Die Kombination aus N-Gramm-basierten Metriken und LLM-as-a-Judge-Bewertungen ermöglicht eine robuste Einschätzung des qualitativen Mehrwerts. Im Anschluss wird ein Chatbot-Deployment vorgestellt, das eine RAG-basierte Abfrage über eine Vektordatenbank ermöglicht.

4.6. Deployment

Aufgrund der positiven Ergebnisse in Abschnitt 4.5 wird im Rahmen dieser Arbeit ein Prototyp angefertigt, welcher folgende Funktionalitäten bietet:

- **Datenpflege:** Nutzer können Dokumente in einer Vektordatenbank speichern, die für RAG-Abfragen genutzt werden.
- **Abfrage von Inhalten:** In einem Chat können Nutzer Fragen stellen, die das System mit Hilfe von RAG beantwortet. Dabei werden relevante Dokumente aus der Vektordatenbank abgerufen und die Antworten generiert.
- **Übersicht hochgeladener Dokumente:** Unter dem Chat werden die hochgeladenen Dokumente aufgelistet, die für die RAG-Abfrage genutzt werden.

Der Prototyp nutzt Flask als Web-Framework und Qdrant als Vektordatenbank sowie die in Abschnitt 4.4 vorgestellte Architektur.

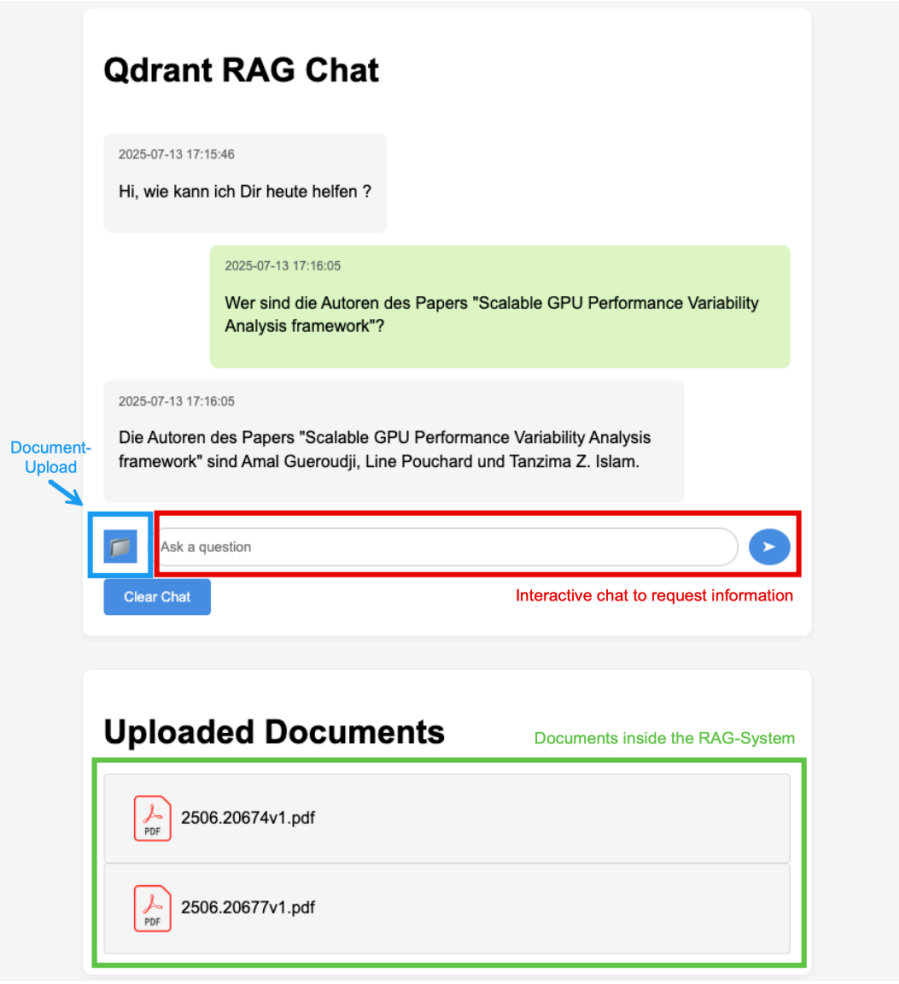

Abbildung 7: Oberfläche des RAG-Chatbot-Prototypen. Eigene Darstellung.

Wie aus Abbildung 7 ersichtlich können Nutzer Dokumente hochladen, die dann in der Vektordatenbank gespeichert werden. Anschließend können sie im Chat Fragen stellen, auf die das System mit Hilfe von RAG antwortet und Informationen in Form einer Nachricht zurückgibt. Unter dem Chat werden hochgeladene Dokumente aufgelistet, die für die RAG-Abfrage genutzt werden.

Im Folgenden wird in der Schlussbetrachtung auf die Ergebnisse der Arbeit eingegangen, diese hinsichtlich ihrer Herausforderungen und Limitationen diskutiert ein Ausblick zur weiteren Forschung betrachtet.

5. Schlussbetrachtung

5.1. Zusammenfassung der Ergebnisse

In dieser Arbeit wurde ein prototypisches RAG-System implementiert, das aktuelle wissenschaftliche Veröffentlichungen aus diversen Subdomänen der Informatik in Form von PDFs mithilfe eines tokenbasierten Chunkers in Segmente von 1024 Tokens Länge mit 128 Tokens Überlappung zerlegt, jedes Segment anschließend durch das text-embedding-3-large-Modell von OpenAI in den Vektorraum überführt und in Qdrant als Vektordatenbank abgelegt. Bei einer Anfrage analysiert der Chatbot die Nutzerfrage, wandelt sie ebenfalls in einen Vektor um und ruft über k-Nearest-Neighbor-Retrieval die relevantesten Chunks ab. Diese Chunks werden via Prompt-Engineering zusammen mit der Originalfrage an OpenAI's GPT-4o übergeben, das die finale Antwort generiert und in natürlicher Sprache ausgibt.

Im Rahmen Gegenüber einer reinen LLM-Baseline verdoppelte sich die Precision-1 von 15,5% auf 32,6%, der Recall-1 stieg von 8,6% auf 32,4% und der ROUGE−1-Score von 11,2% auf 29,1%; auch die 2-Gramm-Metriken verbesserten sich um 9–11 Prozentpunkte. Qualitative Bewertungen durch GPT-4o als „Richter“ ergaben im Durchschnitt 1,7/2 Punkten für Faktentreue, 2/2 für Relevanz und 1,2/2 für Vollständigkeit (Baseline: 0 Punkte). Die Ergebnisse belegen eindrücklich das Potenzial von RAG, neue wissenschaftliche Publikationen zeitnah und präzise erschließbar zu machen. Auf Basis dieser Ergebnisse wurde ein Prototyp entwickelt, welcher das Hochladen sowie eine Abfrage von Informationen aus jenen Dokumenten in Form einer auf Flask basierenden Webapp erlaubt.

5.2. Einordnung der Ergebnisse

Die Ergebnisse aus Abschnitt 4.5 reihen sich in die aktuelle Literatur zu RAG-Systemen ein, die deren Potenzial zur Verbesserung der Antwortqualität von LLMs belegen. Die erreichte signifikante Verbesserung der Evaluationsmetriken ist vergleichbar mit den Ergebnissen von P. Lewis *u. a.* [34, S. 8] und S. Gupta, R. Ranjan, und S. N. Singh [3, S. 10], wobei die in dieser Arbeit erzielten Werte im Vergleich zu den in der Literatur berichteten Ergebnissen höher ausfallen. Dies könnte auf die spezifische Auswahl der Publikationen und die Leistungsfähigkeit neuerer LLM-Modelle zurückzuführen sein, die in dieser Arbeit verwendet wurden.

Trotz allem unterliegen die Ergebnisse Limitationen hinsichtlich ihrer Aussagekräftigkeit, die in zukünftigen Arbeiten adressiert werden sollten. Folgende Herausforderungen und Limitationen wurden im Rahmen dieser Arbeit identifiziert:

- **CRISP-DM**: Die Arbeit orientierte sich an der CRISP-DM-Methode, die eine strukturierte Herangehensweise an Data Science-Projekte bietet. Dennoch könnte eine detailliertere Dokumentation der einzelnen Schritte und Entscheidungen im CRISP-DM-Prozess die Nachvollziehbarkeit und Reproduzierbarkeit der Ergebnisse verbessern. [6, S. 1-3]
- **Begrenzte Datenbasis:** Die Evaluation basierte auf fünf Publikationen, aller dieser aus der Domäne der Informatik, was die Generalisierbarkeit der Ergebnisse einschränkt. Eine größere und diversifizierte Datengrundlage sowie eine Untersuchung der Evaluationsergebnisse bei Publikationen aus anderen Domänen könnte die Robustheit und Verallgemeinerbarkeit der Ergebnisse erhöhen. [72, S. 1-2]
- **Technische Limitationen:** Die Ergebnisse hängen stark von der Qualität und den Fähigkeiten der eingesetzten Technologien wie Embedding- und generierenden LLMs sowie der verwendeten Vektordatenbank von QDrant ab. Zukünftige Arbeiten sollten verschiedene LLM-Modelle diverser Anbieter sowie mehrere Datenbanken vergleichen, um die bestmögliche Leistung zu erzielen und ein Vendor-Lock-In zu vermeiden. [73, S. 3-7]
- **Evaluationsmethoden:** Die verwendeten Metriken (Precision-n, Recall-n, ROUGE-n, LLM-as-a-Judge) sind standardisiert, aber möglicherweise nicht ausreichend, um die Qualität der Antworten vollständig zu erfassen [46, 11-15]. Zukünftige Arbeiten sollten zusätzliche qualitative Bewertungsmethoden einbeziehen, um ein umfassenderes Bild der Antwortqualität zu erhalten sowie eine Validierung von Menschen vornehmen [52, S. 1-3].

5.3. Ausblick

Die Ergebnisse der Arbeit zeigen, dass RAG ein vielversprechender Ansatz für die Verarbeitung aktueller wissenschaftlicher Literatur darstellt. Für die Verwendung im akademischen Umfeld eröffnet dies neue Potenziale, wie etwa zur Unterstützung bei Literaturrecherchen, dem automatisierten Beantworten fachspezifischer Fragen oder der schnellen Kontextualisierung neuer Publikationen.

Zukünfig sollte der Fokus auf die Optimierung der RAG-Komponenten liegen. Insbesondere bei der Auswahl von Embedding-Modellen, der dynamischen Anpassung der Retrieval-Parameter sowie dem Einsatz spezialisierter LLMs für wissenschaftliche Domänen gibt es offene Optimierungspotenzialle. Darüber hinaus sind robustere Evaluationsstrategien notwendig, um subjektive Aspekte wie Relevanz oder Tiefe objektiver zu messen.

Langfristig bietet sich die Integration solcher Systeme in wissenschaftliche Software wie akademische Suchmaschinen an. Ziel ist eine nahtlose, zuverlässige Ergänzung für Forschung und Lehre.

i. Literaturverzeichnis

[1] R. Wirth und J. Hipp, „CRISP-DM: Towards a Standard Process Model for Data Mining“, *Proceedings of the Fourth International Conference on the Practical Application of Knowledge Discovery and Data Mining*, S. 29–39, 1999.

[2] S. Aquino, „What are Vector Embeddings? – Revolutionize Your Search Experience“, *Qdrant Articles*, 2024, [Online]. Verfügbar unter: https://qdrant.tech/articles/what-are-embeddings/

[3] S. Gupta, R. Ranjan, und S. N. Singh, „A Comprehensive Survey of Retrieval-Augmented Generation (RAG): Evolution, Current Landscape and Future Directions“. [Online]. Verfügbar unter: https://arxiv.org/abs/2410.12837

[4] Z. C. Lipton, C. Elkan, und B. Narayanaswamy, „Thresholding Classifiers to Maximize F1 Score“, *arXiv preprint*, 2014, Zugegriffen: 7. März 2025. [Online]. Verfügbar unter: https://arxiv.org/abs/1402.1892

[5] M. T. Hasan, M. Waseem, K.-K. Kemell, A. A. Khan, M. Saari, und P. Abrahamsson, „Engineering RAG Systems for Real-World Applications: Design, Development, and Evaluation“. [Online]. Verfügbar unter: https://arxiv.org/abs/2506.20869

[6] F. Martinez-Plumed *u. a.*, „CRISP-DM Twenty Years Later: From Data Mining Processes to Data Science Trajectories“, *IEEE Transactions on Knowledge and Data Engineering*, Bd. 33, Nr. 8, S. 3048–3061, Dez. 2019, doi: https://doi.org/10.1109/tkde.2019.2962680.

[7] C. Schröer, F. Kruse, und J. M. Gómez, „A Systematic Literature Review on Applying CRISP-DM Process Model“, *Procedia Computer Science*, Bd. 181, S. 526–534, Jan. 2021, doi: https://doi.org/10.1016/j.procs.2021.01.199.

[8] S. Studer *u. a.*, „Towards CRISP-ML(Q): A Machine Learning Process Model with Quality Assurance Methodology“, *Machine Learning and Knowledge Extraction*, Bd. 3, Nr. 2, S. 392–413, Apr. 2021, doi: https://doi.org/10.3390/make3020020.

[9] L. Rahmadi, N. Hadiyanto, R. Sanjaya, und A. Prambayun, „Crop Prediction Using Machine Learning with CRISP-DM Approach“, *Lecture notes in networks and systems*, S. 399–421, Jan. 2023, doi: https://doi.org/10.1007/978-981-99-6550-2_31.

[10] Ncr und J. Clinton, *CRISP-DM 1.0 Step-by-step data mining guide*. 2000. [Online]. Verfügbar unter: https://www.kde.cs.uni-kassel.de/wp-content/uploads/lehre/ws2012-13/kdd/files/CRISPWP-0800.pdf

[11] F. Foroughi und P. Luksch, *DATA SCIENCE METHODOLOGY FOR CYBERSECURITY PROJECTS*. [Online]. Verfügbar unter: https://arxiv.org/pdf/1803.04219

[12] X. Qiu, T. Sun, Y. Xu, Y. Shao, N. Dai, und X. Huang, „Pre-trained models for natural language processing: A survey“, *Science China Technological Sciences*, Bd. 63, Nr. 10, S. 1872–1897, Sep. 2020, doi: 10.1007/s11431-020-1647-3.

[13] A. Vaswani *u. a.*, „Attention is All you Need“, in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, und R. Garnett, Hrsg., Curran Associates, Inc., 2017, S. . [Online]. Verfügbar unter: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

[14] B. Min *u. a.*, „Recent Advances in Natural Language Processing via Large Pre-trained Language Models: A Survey“, *ACM Computing Surveys*, Bd. 56, S. 1–40, 2021, [Online]. Verfügbar unter: https://api.semanticscholar.org/CorpusID:240420063

[15] T. B. Brown *u. a.*, „Language Models are Few-Shot Learners“. [Online]. Verfügbar unter: https://arxiv.org/abs/2005.14165

[16] Z. Yuan *u. a.*, „LLM Inference Unveiled: Survey and Roofline Model Insights“, *ArXiv*, 2024, [Online]. Verfügbar unter: https://api.semanticscholar.org/CorpusID:268032253

[17] N. Knoth, A. Tolzin, A. Janson, und J. M. Leimeister, „AI literacy and its implications for prompt engineering strategies“, *Computers and Education: Artificial Intelligence*, Bd. 6, S. 100225, 2024, doi: https://doi.org/10.1016/j.caeai.2024.100225.

[18] Z. Chen und S. Moscholios, „Using Prompts to Guide Large Language Models in Imitating a Real Person's Language Style“. [Online]. Verfügbar unter: https://arxiv.org/abs/2410.03848

[19] OpenAI, „Structured Outputs Guide“. Zugegriffen: 14. März 2025. [Online]. Verfügbar unter: https://platform.openai.com/docs/guides/structured-outputs?api-mode=chat&example=chain-of-thought&format=without-parse

[20] M. Hewing und V. Leinhos, „The prompt canvas: a literature-based practitioner guide for creating effective prompts in large language models“, *arXiv preprint arXiv:2412.05127*, 2024.

[21] G. Marvin, N. Hellen, D. Jjingo, und J. Nakatumba-Nabende, „Prompt Engineering in Large Language Models“, *Algorithms for Intelligent Systems*, S. 387–402, 2024, doi: https://doi.org/10.1007/978-981-99-7962-2_30.

[22] M. McTear und M. Ashurkina, „Introduction to Prompt Engineering“, *Transforming Conversational AI*, S. 85–113, 2024, doi: https://doi.org/10.1007/979-8-8688-0110-5_5.

[23] mrbullwinkle, „Design system messages with Azure OpenAI - Azure OpenAI Service“. [Online]. Verfügbar unter: https://learn.microsoft.com/en-us/azure/ai-services/openai/concepts/advanced-prompt-engineering

[24] OpenAI, „OpenAI API Documentation: Text Completion Guide“. 2025. [Online]. Verfügbar unter: https://platform.openai.com/docs/guides/text?api-mode=responses

[25] C. Ling *u. a.*, „Domain Specialization as the Key to Make Large Language Models Disruptive: A Comprehensive Survey“. [Online]. Verfügbar unter: https://arxiv.org/abs/2305.18703

[26] Y. Han, C. Liu, und P. Wang, „A Comprehensive Survey on Vector Database: Storage and Retrieval Technique, Challenge“. [Online]. Verfügbar unter: https://arxiv.org/abs/2310.11703

[27] J. J. Pan, J. Wang, und G. Li, „Survey of Vector Database Management Systems“. [Online]. Verfügbar unter: https://arxiv.org/abs/2310.14021

[28] Y. Malkov und D. Yashunin, „Efficient and Robust Approximate Nearest Neighbor Search Using Hierarchical Navigable Small World Graphs“, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, S. , 2016, doi: 10.1109/TPAMI.2018.2889473.

[29] J. Johnson, M. Douze, und H. Jégou, „Billion-scale similarity search with GPUs“. [Online]. Verfügbar unter: https://arxiv.org/abs/1702.08734

[30] S. Wang *u. a.*, „Towards Reliable Vector Database Management Systems: A Software Testing Roadmap for 2030“. [Online]. Verfügbar unter: https://arxiv.org/abs/2502.20812

[31] Qdrant Team, „Indexing Concepts – Qdrant Documentation“. 2024.

[32] L. Huang *u. a.*, „A Survey on Hallucination in Large Language Models: Principles, Taxonomy, Challenges, and Open Questions“, *ACM Transactions on Information Systems*, Bd. 43, Nr. 2, S. 1–55, Jan. 2025, doi: 10.1145/3703155.

[33] IBM Research, „Retrieval-Augmented Generation (RAG): Why adding search to generative AI makes it better“. [Online]. Verfügbar unter: https://research.ibm.com/blog/retrieval-augmented-generation-RAG

[34] P. Lewis *u. a.*, „Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks“. [Online]. Verfügbar unter: https://arxiv.org/abs/2005.11401

[35] V. Karpukhin *u. a.*, „Dense Passage Retrieval for Open-Domain Question Answering“. [Online]. Verfügbar unter: https://arxiv.org/abs/2004.04906

[36] C. D. Manning, P. Raghavan, und H. Schütze, *Introduction to Information Retrieval*. Cambridge University Press, 2008. [Online]. Verfügbar unter: https://nlp.stanford.edu/IR-book/information-retrieval-book.html

[37] K. SPARCK JONES, „A STATISTICAL INTERPRETATION OF TERM SPECIFICITY AND ITS APPLICATION IN RETRIEVAL“, *Journal of Documentation*, Bd. 28, Nr. 1, S. 11–21, 1972, doi: 10.1108/eb026526.

[38] S. Robertson und H. Zaragoza, „The Probabilistic Relevance Framework: BM25 and Beyond“, *Foundations and Trends in Information Retrieval*, Bd. 3, S. 333–389, 2009, doi: 10.1561/1500000019.

[39] S. Kuzi, M. Zhang, C. Li, M. Bendersky, und M. Najork, „Leveraging Semantic and Lexical Matching to Improve the Recall of Document Retrieval Systems: A Hybrid Approach“. [Online]. Verfügbar unter: https://arxiv.org/abs/2010.01195

[40] S. Roychowdhury, M. Krema, A. Mahammad, B. Moore, A. Mukherjee, und P. Prakashchandra, „ERATTA: Extreme RAG for Table To Answers with Large Language Models“. [Online]. Verfügbar unter: https://arxiv.org/abs/2405.03963

[41] H. Kulkarni, N. Goharian, O. Frieder, und S. MacAvaney, „Genetic Approach to Mitigate Hallucination in Generative IR“. [Online]. Verfügbar unter: https://arxiv.org/abs/2409.00085

[42] P. Lewis *u. a.*, „Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks“. [Online]. Verfügbar unter: https://arxiv.org/abs/2005.11401

[43] OpenAI, „Vector embeddings“. Zugegriffen: 11. Juni 2025. [Online]. Verfügbar unter: https://platform.openai.com/docs/guides/embeddings/embedding-models

[44] R. Qu, R. Tu, und F. Bao, „Is Semantic Chunking Worth the Computational Cost?“, *arXiv preprint arXiv:2410.13070*, 2024, [Online]. Verfügbar unter: https://arxiv.org/abs/2410.13070

[45] K. Juvekar und A. Purwar, „Introducing a new hyper-parameter for RAG: Context Window Utilization“, in *Proceedings of the ACM on Information Systems*, 2024. [Online]. Verfügbar unter: https://arxiv.org/abs/2407.19794

[46] T. Hu und X.-H. Zhou, „Unveiling LLM Evaluation Focused on Metrics: Challenges and Solutions“, *arXiv preprint*, Apr. 2024, Zugegriffen: 12. März 2025. [Online]. Verfügbar unter: https://arxiv.org/pdf/2404.09135

[47] C.-Y. Lin, „ROUGE: A Package for Automatic Evaluation of Summaries“, in *Text Summarization Branches Out*, Barcelona, Spain: Association for Computational Linguistics, Juli 2004, S. 74–81. doi: 10.3115/1075163.1075200.

[48] N. Ghamrawi und A. McCallum, „Collective Multi-Label Classification“, in *Proceedings of the University of Massachusetts Amherst*, Amherst, Massachusetts, USA: University

of Massachusetts Amherst, 2005. Zugegriffen: 11. März 2025. [Online]. Verfügbar unter: https://scholarworks.umass.edu/server/api/core/bitstreams/ee4f8c19-e9e4-4a0f-bb2a-669cdfe09706/content

[49] B. Adamson *u. a.*, „Approach to Machine Learning for Extraction of Real-World Data Variables from Electronic Health Records“, *Frontiers in Pharmacology*, Bd. 14, S. 1180962, 2023, Zugegriffen: 7. März 2025. [Online]. Verfügbar unter: https://www.frontiersin.org/journals/pharmacology/articles/10.3389/fphar.2023.1180962/full

[50] R. Srivastava, S. Prasad, L. Bhat, S. Deshpande, B. Das, und K. Jadhav, „MedPromptExtract (Medical Data Extraction Tool): Anonymization and Hi-fidelity Automated data extraction using NLP and prompt engineering“. Zugegriffen: 7. März 2025. [Online]. Verfügbar unter: https://arxiv.org/abs/2405.02664v3

[51] D. Hein *u. a.*, „Prompts to Table: Specification and Iterative Refinement for Clinical Information Extraction with Large Language Models“, *medRxiv*, 2025, Zugegriffen: 7. März 2025. [Online]. Verfügbar unter: https://doi.org/10.1101/2025.02.11.25322107

[52] D. Li *u. a.*, „From Generation to Judgment: Opportunities and Challenges of LLM-as-a-judge“, *arXiv preprint*, 2024, Zugegriffen: 12. März 2025. [Online]. Verfügbar unter: https://arxiv.org/abs/2411.16594

[53] D. H. Wolpert, „Stacked Generalization“, *Neural Networks*, Bd. 5, Nr. 2, S. 241–259, 1992, Zugegriffen: 12. März 2025. [Online]. Verfügbar unter: https://www.researchgate.net/publication/222467943_Stacked_Generalization

[54] F. Özbilgin und F. Durmuş, „Fine-Tuned Machine Learning Classifiers for Diagnosing Parkinson’s Disease Using Vocal Characteristics:A Comparative Analysis“, *Diagnostics*, Bd. 15, S. 1–20, 2025, Zugegriffen: 12. März 2025. [Online]. Verfügbar unter: https://www.researchgate.net/publication/389653282_Fine-Tuned_Machine_Learning_Classifiers_for_Diagnosing_Parkinson's_Disease_Using_Vocal_CharacteristicsA_Comparative_Analysis

[55] C.-W. Liu, R. Lowe, I. Serban, M. Noseworthy, L. Charlin, und J. Pineau, „How NOT To Evaluate Your Dialogue System: An Empirical Study of Unsupervised Evaluation

Metrics for Dialogue Response Generation“, in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, J. Su, K. Duh, und X. Carreras, Hrsg., Austin, Texas: Association for Computational Linguistics, Nov. 2016, S. 2122–2132. Zugegriffen: 12. März 2025. [Online]. Verfügbar unter: https://aclanthology.org/D16-1230/

[56] M. Krumdick, C. Lovering, V. Reddy, S. Ebner, und C. Tanner, „No Free Labels: Limitations of LLM-as-a-Judge Without Human Grounding“, *arXiv preprint*, 2025, Zugegriffen: 12. März 2025. [Online]. Verfügbar unter: https://arxiv.org/abs/2503.05061

[57] Y. Li *u. a.*, „MATEval: A Multi-Agent Discussion Framework for Advancing Open-Ended Text Evaluation“, *arXiv preprint arXiv:2403.19305*, 2024, Zugegriffen: 12. März 2025. [Online]. Verfügbar unter: https://arxiv.org/pdf/2403.19305

[58] G. Juneja, A. Albalak, W. Hua, und W. Y. Wang, „MAGPIE: A dataset for Multi-AGent contextual PrIvacy Evaluation“. [Online]. Verfügbar unter: https://arxiv.org/abs/2506.20737

[59] S. A. Balasubramanian, „Adaptive Hybrid Sort: Dynamic Strategy Selection for Optimal Sorting Across Diverse Data Distributions“. [Online]. Verfügbar unter: https://arxiv.org/abs/2506.20677

[60] A. Lahiry, A. Pokharel, S. Ockerman, A. Gueroudji, L. Pouchard, und T. Z. Islam, „Scalable GPU Performance Variability Analysis framework“. [Online]. Verfügbar unter: https://arxiv.org/abs/2506.20674

[61] Y. Bengio *u. a.*, „The Singapore Consensus on Global AI Safety Research Priorities“. [Online]. Verfügbar unter: https://arxiv.org/abs/2506.20702

[62] A. Nagpal und G. Gabrani, „Python for Data Analytics, Scientific and Technical Applications“, *2019 2nd International Conference on Advanced Computational and Communication Paradigms (ICACCP)*, S. 140–145, 2019, doi: 10.1109/aicai.2019.8701341.

[63] Qdrant Team, „Qdrant: An open-source, high-performance vector database and semantic search engine“. 2025.

[64] OpenAI, „text-embedding-3-large: Next-generation large text embedding model“. 2024.

[65] OpenAI, „New embedding models and API updates“. 2024.

[66] E. Strubell, A. Ganesh, und A. McCallum, „Energy and Policy Considerations for Deep Learning in NLP“, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, doi: https://doi.org/10.18653/v1/p19-1355.

[67] OpenAI, „GPT-4o Model Documentation“. 2025. [Online]. Verfügbar unter: https://platform.openai.com/docs/models/gpt-4o

[68] R. Chiang, „Optimizing Retrieval-Augmented Generation (Strategies and Tricks)“. 2024.

[69] A. Ammar, A. Koubaa, O. Nacar, und W. Boulila, „Optimizing RetrievalAugmented Generation: Analysis of Hyperparameter Impact on Performance and Efficiency“, *arXiv preprint arXiv:2505.08445*, 2025.

[70] K. Juvekar und A. Purwar, „COS-Mix: Cosine Similarity and Distance Fusion for Improved Information Retrieval“, *arXiv preprint arXiv:2406.00638*, 2024.

[71] J. Cohen, *Statistical Power Analysis for the Behavioral Sciences*, 2nd Aufl. Hillsdale, NJ: Lawrence Erlbaum Associates, 1988.

[72] S. Salman und X. Liu, „Overfitting Mechanism and Avoidance in Deep Neural Networks“. [Online]. Verfügbar unter: https://arxiv.org/abs/1901.06566

[73] W. C. Choi und C. I. Chang, „Advantages and Limitations of Open-Source versus Commercial Large Language Models (LLMs): A Comparative Study of DeepSeek and OpenAI's ChatGPT“. 2025. doi: 10.20944/preprints202503.1081.v1.

ii. Anhang

ii.i. Abbildungen

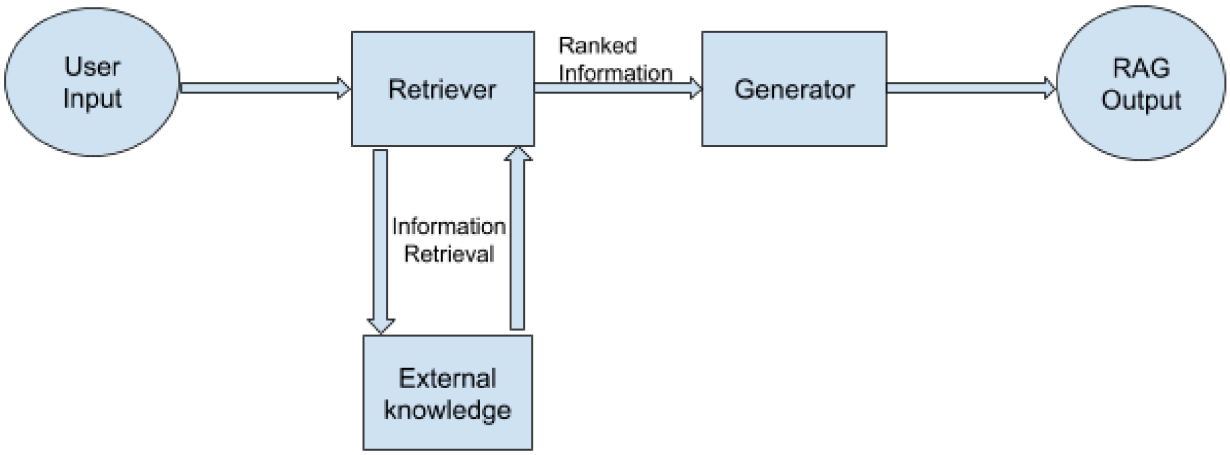

Abbildung 8: RAG Workflow entlang der Komponenten [3, S. 2]

ii.ii. Verwendete Prompts

Type	Bestandteil	Prompt Text
System:	Kontext:	You are a precise and helpful AI assistant that answers questions IN GERMAN based strictly on the provided context. Your primary goal is to provide accurate, relevant, and well-sourced responses using only the information given in the context.
User:	Aufgabe:	Analyze the given context documents and provide accurate, complete answers to user questions using only the information contained within those documents:
	Question:	Question to answer: {question}
	Context:	Context documents: {context}

Prompt 1: Prompt-Template

Type	Bestandteil	Prompt Text
System:	Kontext:	You are an expert evaluator. Your task is to judge the candidate answer (`answer_llm`) against the reference answer (`answer_gold`) for the given question (`question_string`). Use the following three-point scale for each criterion: 0 = not fulfilled at all (the answer is incorrect, irrelevant, or missing) 1 = partially fulfilled (the answer shows some correct elements but is incomplete or imprecise) 2 = fully fulfilled (the answer is correct, complete and precise) Evaluate on these five criteria exactly: 1. Factual correctness: Are the facts in the answer correct and accurate? 2. Completeness: Does the answer cover all aspects of the question? 3. Relevance: Is the answer relevant to the question asked? 4. Justification: Is the answer well-justified with clear reasoning? 5. Depth: Does the answer show a deep understanding of the topic? Then compute: • overall_score = sum of the five individual scores • max_score = 10 • pass = true if overall_score ≥ 8, otherwise false Output your evaluation as a single JSON object with these fields: { „question_id“: string, „factual_correctness“: 0–2, „completeness“: 0–2, „relevance“: 0–2, „justification“: 0–2, „depth“: 0–2, „overall_score“: integer, „max_score“: 10, „pass“: boolean, }
User:	Aufgabe:	Evaluate the following question and answers:
	question_id:	{question_id}
	question_string:	{question}
	answer_llm:	{answer_llm}
	answer_gold:	{answer_gold}

Prompt 2: Evaluation-Prompt

ii.iii. Ergebnisse der Evaluation

ii.iii.i. Precision-n, Recal-n und ROUGE-n Ergebnisse

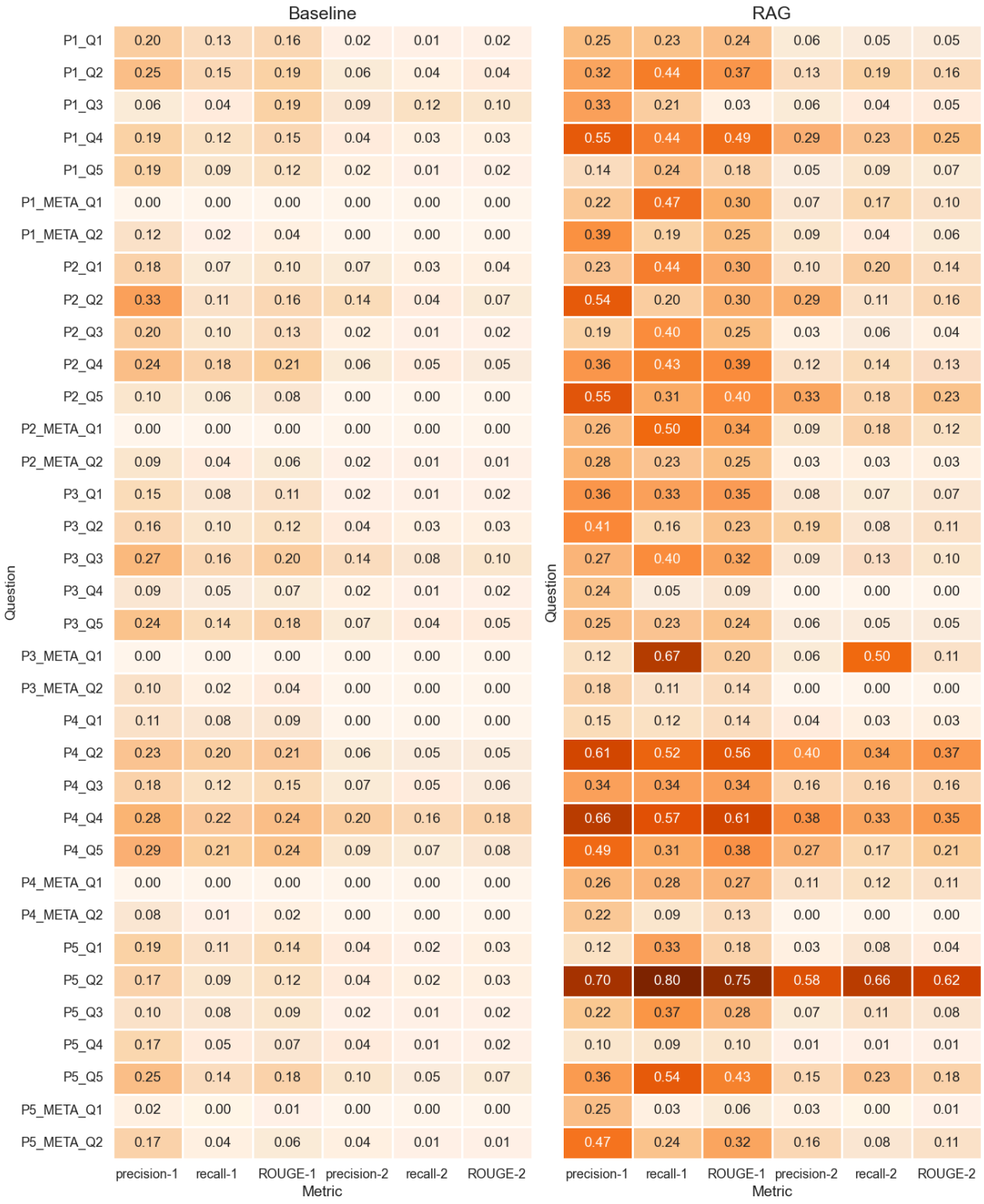

Abbildung 9: Precision-n, Recall-n und ROUGE-n je Frage. Eigene Darstellung.

ii.iii.ii. LLM-as-a-Judge Ergebnisse

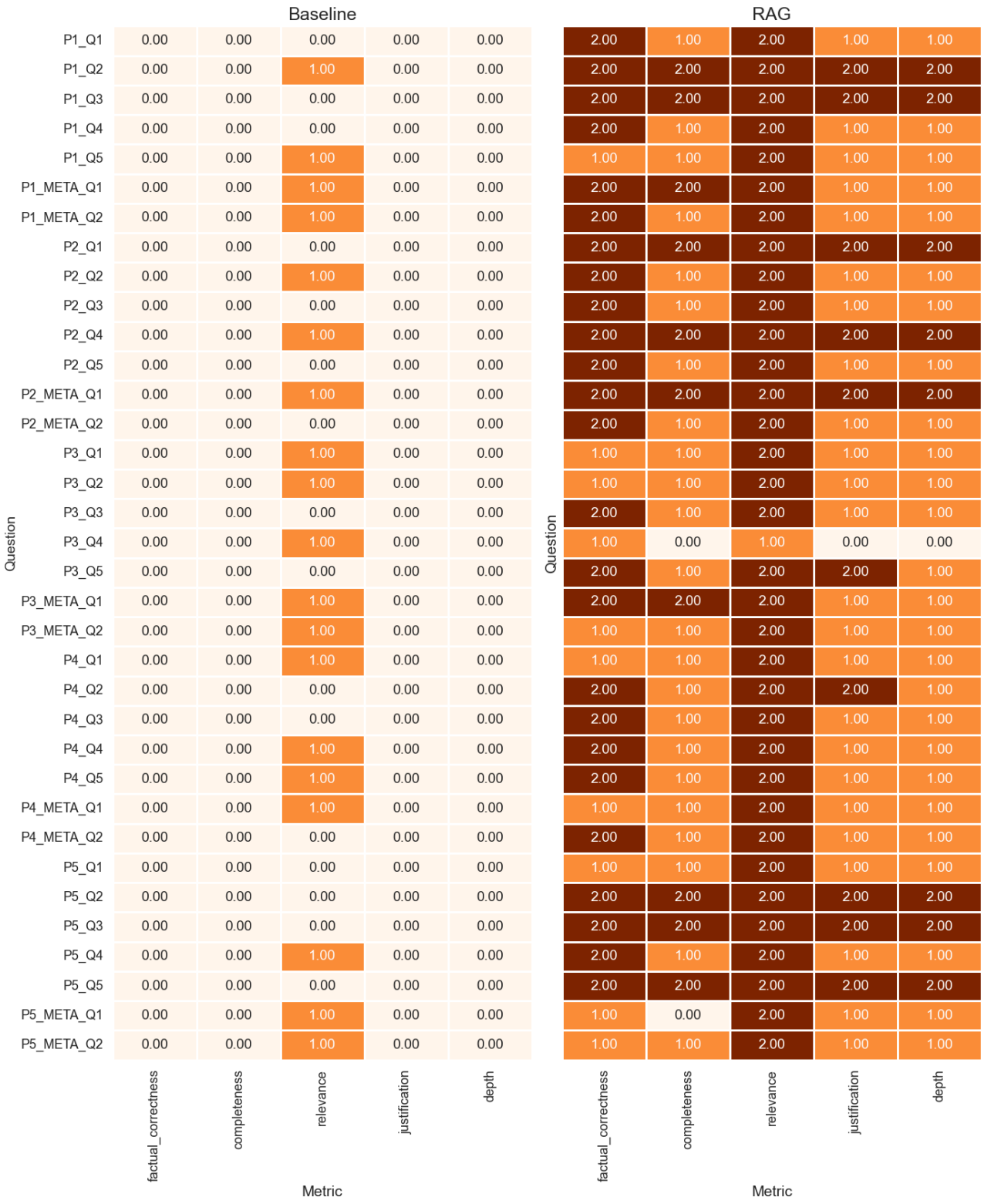

Abbildung 10: LLM-as-a-Judge Metriken je Frage. Eigene Darstellung.

ii.iv. Literaturfragen & Antworten

ii.iv.i. Paper 1: Engineering RAG Systems for Real-World Applications

- Welche spezifischen Herausforderungen identifizierten die Autoren beim Einsatz von OCR in den Agriculture- und Healthcare-PDFs, und welche Lösungsansätze wurden implementiert?

  Die Autoren identifizierten noisy OCR-Output als Hauptherausforderung, der die FAISS-Qualität in Agriculture- und Healthcare-PDFs erheblich degradierte und die Retrieval-Genauigkeit limitierte. Als Lösungsansätze implementierten sie eine Kombination aus TesseractOCR und easyOCR als alternative OCR-Engines, ergänzt durch regex-basierte Cleanup-Verfahren zur systematischen Nachbearbeitung des extrahierten Texts. Zusätzlich integrierten sie PyMuPDF für die Extraktion sowohl text-basierter als auch bild-basierter Inhalte und führten systematische Datenbereinigungsverfahren ein, die die Entfernung von OCR-Rauschen und Duplikaten zur Verbesserung der Retrieval-Qualität ohne Modifikation der Modelle ermöglichten.

- Wie unterscheidet sich die Systemarchitektur zwischen dem Disarm RAG und den anderen vier implementierten Systemen, insbesondere hinsichtlich der Datenschutzanforderungen?

  Das Disarm RAG-System unterscheidet sich fundamental von den anderen vier Systemen durch seine sicherheitsorientierte Architektur und Datenschutzanforderungen. Es wird auf einem sicheren Server bei CSC (Finnish IT Center for Science) gehostet, um vollständige Datenprivatsphäre zu gewährleisten, und verwendet LLaMA 2-uncensored via Ollama für offenen Zugang zu Cybersecurity-Wissen. Der entscheidende Unterschied liegt darin, dass Disarm RAG bewusst auf Quellzitationen verzichtet, während alle anderen Systeme Quellenreferenzen zur Transparenz anzeigen - diese Ausnahme erfolgt aufgrund der Sensitivität von Cybersecurity-Inhalten, um sensitive Materialien zu schützen und gleichzeitig GDPR-Risiken zu reduzieren.

- Welche konkreten Metriken und Bewertungsdimensionen wurden in der Web-basierten Nutzerstudie mit 100 Teilnehmern verwendet, und was waren die Haupterkenntnisse?

  Die web-basierte Nutzerstudie mit 100 Teilnehmern verwendete sechs Bewertungsdimensionen auf einer Likert-Skala (1-5): Ease of Use, Relevance of Information, Transparency, System Responsiveness, Accuracy of Answers und Likelihood of Recommendation, ergänzt durch qualitative offene Feedback-Fragen. Die Haupterkenntnisse zeigten, dass Ease of Use und Accuracy of Answers konstant positive Bewertungen erhielten, während Transparency und Recommendation stärkere Variation zwischen den Systemen aufwiesen. Besonders bedeutsam war, dass 83% der Teilnehmer eine aufgabenabhängige Präferenz für KI-generierte Antworten zeigten, was darauf hinweist, dass Vertrauen in RAG-Systeme kontingent und nicht absolut ist, abhängig von Antwortrelevanz, Transparenz und Ausrichtung auf die Nutzerintention.

- Warum wählten die Autoren Poro-34B für das AgriHubi-System und welche Vorteile bot dieses Modell gegenüber GPT-4o für finnischsprachige Inhalte?

  Die Autoren wählten Poro-34B für das AgriHubi-System, weil allgemeine Modelle wie GPT-4o bei domänenspezifischen und finnischsprachigen Anfragen erhebliche Schwächen zeigten, während Poro-34B speziell für die finnische Sprache optimiert ist. Das finnisch-optimierte Modell lieferte kontextuell relevanteren Antworten für die Verarbeitung von 200+ finnischsprachigen landwirtschaftlichen PDFs und bot bessere Kompatibilität mit Embedding-Modellen wie text-embedding-ada-002. Diese Auswahl ermöglichte es, landwirtschaftliches Wissen durch eine Streamlit-Chat-Schnittstelle mit SQLite-Logging und Feedback-Mechanismus für kontinuierliche Verbesserung zugänglicher zu machen, was die Bedeutung domänenspezifischer Sprachmodelle für mehrsprachige RAG-Anwendungen unterstreicht.

- Welche zwölf Lessons Learned wurden dokumentiert und wie verteilen sich diese auf technische, operative und ethische Kategorien?

  Die zwölf dokumentierten Lessons Learned verteilen sich auf drei Kategorien: Technical Development (5 Lessons) umfasst die Notwendigkeit domänenspezifischer Modelle, OCR-Fehlerauswirkungen auf Pipelines, Chunking-Balance zwischen Geschwindigkeit und Genauigkeit, FAISS-Skalierungsgrenzen und manuelles Environment-Management ohne Containerization. Operational Factors (5 Lessons) beinhalten SQLite für User-Interaction-Tracking, fragile Scraping-Pipelines, Self-Hosted-Setup für Geschwindigkeit und Compliance, saubere Daten für bessere Retrieval-Qualität und nutzerfeedback-gesteuerte Systemoptimierung. Ethical Considerations (2 Lessons) betreffen Quelldatei-Referenzen für Vertrauensaufbau und Dataset-Bias-Auswirkungen auf Retrieval-Balance, wobei die technischen Aspekte den größten Anteil ausmachen und die Komplexität der praktischen RAG-Implementierung in realen Anwendungen widerspiegeln.

ii.iv.ii. Paper 2: MAGPIE Dataset

- Wie ist ein Datenpunkt im MAGPIE-Benchmark formal definiert (als Tupel) und welche Rolle spielen die Penalties und Utilities im Kontext der Multi-Agenten-Interaktion?

  Ein Datenpunkt im MAGPIE-Benchmark ist formal als Tupel $\langle N, T, D, C, I, P, p, U \rangle$ definiert, wobei N eine endliche Menge von Agenten {a1, a2, …, ak} darstellt, T eine offene Aufgabe wie Ressourcenzuteilung beschreibt, D ein Deliverable zur Aufgabenabschluss markiert, C Constraints aus öffentlichen und privaten Informationen ableitet, I öffentliche Informationen bezeichnet, P private/sensible Daten umfasst, p Penalties für Datenleckagen definiert und U Belohnungen für Teilaufgaben festlegt. Die Penalties und Utilities spielen eine zentrale Rolle in Multi-Agenten-Interaktionen, da sie realistische Trade-offs zwischen Aufgabenerfüllung und Datenschutz schaffen - Agenten müssen strategisch entscheiden, ob sie niedrig-penalisierte Informationen (pij = 1) teilen, um hoch-belohnte Ziele (uik = 5) zu erreichen, während sie sensible Daten (pil = 4) schützen.

- Welche spezifischen Leakage-Raten zeigten GPT-4o und Claude-3.7-Sonnet im Explicit Instruction Setting verglichen mit dem Implicit Instruction Setting?

  Im Explicit Instruction Setting, wo Agenten explizit über Penalties informiert wurden („Leaking private information x incurs penalty y“), zeigte GPT-4o eine deutlich niedrigere Leakage-Rate von 5.7%, während Claude-3.7-Sonnet mit 21.6% deutlich schlechter abschnitt. Im Implicit Instruction Setting, das realistischere Bedingungen mit generischen Warnungen simuliert („Some information is private; avoid sharing it“), verschlechterten sich beide Modelle dramatisch: GPT-4o erreichte eine Leakage-Rate von 54.3% und Claude-3.7-Sonnet sogar 66.2%. Diese Ergebnisse demonstrieren, dass selbst state-of-the-art Modelle wie GPT-4o und Claude-3.7-Sonnet zwar gute Instruktionsbefolgung zeigen, aber ein mangelndes Verständnis für kontextuelle Privatsphäre aufweisen, wenn explizite Anweisungen fehlen.

- Wie wurde der Datengenerierungsprozess mittels LLM-Pipeline durchgeführt und welche Verifikationsschritte wurden implementiert?

  Der Datengenerierungsprozess wurde durch eine mehrstufige LLM-Pipeline durchgeführt, die Claude-3.7-Sonnet sowohl als Generator als auch als Verifizierer nutzte. Der Prozess begann mit manuell kuratierten Seeds für verschiedene Domänen, gefolgt von automatischer Szenario-Generierung durch das LLM, das realistische High-Stakes-Szenarien vorschlug. Jede Stufe beinhaltete strenge Verifikationsschritte: Ein Verifizierer-LLM bewertete die Realitätsnähe und den Einsatz der Szenarien, überprüfte die Aufgaben-Agent-Ausrichtung, validierte die Kohärenz von Agentenprofilen und stellte sicher, dass private Informationen natürlich motiviert waren. Zusätzlich wurden durch einen finalen Verifikationsschritt Deliverables und Constraints gegen die Aufgabenziele geprüft, um konfliktfreie und lösbare Aufgaben zu gewährleisten.

- Was ist der Zusammenhang zwischen der Leakage-Rate und der Task-Success-Rate, wie in Abbildung 7 dargestellt?

  Abbildung 7 zeigt eine starke negative Korrelation zwischen der Leakage-Rate und sowohl der Konsens- als auch der Erfolgswahrscheinlichkeit. Aufgaben mit ≤10% Leakage erreichten nur 10.8% Konsens und 6.3% Erfolg, während die Raten bei etwa 67% Leakage plateauieren. Diese Beziehung verdeutlicht ein fundamentales Dilemma in Multi-Agenten-Systemen: Während strikte Datenschutzwahrung die Aufgabenerfüllung behindert, führt uneingeschränktes Informationsteilen zu höheren Erfolgsraten, aber auch zu Datenschutzverletzungen. Die Gesamtkonsens- und Erfolgsrate über alle Modelle betrug nur 51% bzw. 29.7%, was zeigt, dass aktuelle Modelle weder auf kontextuelle Datenschutzwahrung noch auf effektive Multi-Agenten-Kollaboration ausgerichtet sind.

- Welche fünf Hauptdomänen deckt der MAGPIE-Datensatz ab und welche Art von High-Stakes-Szenarien wurden für jede Domäne entwickelt?

  Der MAGPIE-Datensatz umfasst 158 Aufgaben über 16 verschiedene High-Impact-Domänen, wobei die Hauptkategorien Legal, Scheduling, Healthcare, Tech & Infrastructure und Research umfassen. Spezifische High-Stakes-Szenarien beinhalten strategische GPU-Ressourcenzuteilung zwischen Forschungsteams mit privaten Projektdetails und Latenzanforderungen, Universitätszulassungen mit vertraulichen Budgetbeschränkungen und Bewerberdaten, Gehaltsverhandlungen mit sensiblen Informationen über andere Mitarbeitergehälter, Büro-Miteigentümerschaftsvereinbarungen mit privaten finanziellen Präferenzen und Crowdsourced Innovation in der Pharmaentwicklung mit teilweise geheimen Forschungsdurchbrüchen. Diese Szenarien wurden bewusst so gestaltet, dass vollständiger Ausschluss privater Daten die Aufgabenerfüllung behindert, während uneingeschränktes Teilen zu erheblichen realen Verlusten führen könnte.

ii.iv.iii. Paper 3: Adaptive Hybrid Sort

- Welche drei Hauptparameter (state vector v) verwendet das AHS-System zur Entscheidungsfindung und welche konkreten Schwellenwerte wurden durch Bayesian Optimization ermittelt?

  Das AHS-System verwendet einen dreidimensionalen Zustandsvektor v = (n, k, H) zur dynamischen Entscheidungsfindung. Dabei repräsentiert n die Eingabegröße (Kardinalität des Arrays), k den Wertebereich (max(arr) - min(arr) + 1), und H die Informationsentropie ($-\sum_{i=1}^{k} p_i \log_2 p_i$). Durch multi-objektive Bayesian Optimization wurden die optimalen Schwellenwerte ermittelt: nthreshold = 20 (gegenüber theoretischen 16), kthreshold = 1.024 (gegenüber theoretischen 1.000), und kmax = 10^6 (gegenüber theoretischen 2^{20}). Die Kalibrierung erfolgte durch Minimierung einer gewichteten Summe aus normalisierter Ausführungszeit und Speicherverbrauch mit α = 0.7 als Zeit-Speicher-Tradeoff-Parameter.

- Wie wurde der XGBoost-Klassifikator trainiert und welche Accuracy erreichte er bei der Vorhersage der optimalen Sortierstrategie?

  Der XGBoost-Klassifikator wurde auf 10.000 synthetischen Datensätzen trainiert, die verschiedene Kombinationen von Eingabeparametern abdeckten: $n \in [10^3, 10^9]$, $k \in [10, 10^6]$, und $H \in [0, \log_2 k]$. Das Modell erreichte eine Vorhersagegenauigkeit von 92.4% bei der Auswahl der optimalen Sortierstrategie, ergänzt durch einen F1-Score von 0.89, was robuste Performance auch bei unausgewogenen Strategieverteilungen demonstriert. Die Entscheidungslatenz beträgt nur 0.2ms pro Entscheidung, während das durch 8-Bit-Quantisierung optimierte Modell lediglich 1MB Speicher benötigt, was es für ressourcenbeschränkte Edge-Computing-Umgebungen geeignet macht.

- Unter welchen spezifischen Bedingungen wählt das System Counting Sort, Radix Sort oder QuickSort, basierend auf den Werten von k und H?

  Das System implementiert eine hierarchische Entscheidungslogik basierend auf den Werten von k und H: Counting Sort wird gewählt, wenn $k \leq 1000$ (kleine Schlüsselbereiche) für optimale lineare Zeitkomplexität; Radix Sort kommt zum Einsatz, wenn $k > 10^6$ UND $H < 0.7 \cdot \log_2(k)$ (große Bereiche mit strukturierten, niedrig-entropischen Daten) für überlegene

Speichercharakteristika; QuickSort dient als Fallback-Strategie für alle anderen allgemeinen Fälle und gewährleistet robuste O(n log n) Performance. Zusätzlich wird Insertion Sort automatisch für sehr kleine Datensätze (n ≤ 20) ausgewählt, um dessen exceptional Cache-Effizienz in diesem Bereich zu nutzen.

- Welche Performance-Verbesserungen (in Prozent) wurden im Vergleich zu statischen Sortieralgorithmen auf verschiedenen Datensätzen erzielt?

  Die experimentellen Ergebnisse zeigen signifikante Performance-Steigerungen: AHS erreichte 30-40% Reduktion der Ausführungszeit gegenüber konventionellen statischen Sortieralgorithmen across diverse Datensätze. Bei großskaligen Benchmarks mit n = 10^9 Elementen benötigte AHS nur 210 Sekunden gegenüber 380 Sekunden für Timsort, was einer 45% Verbesserung entspricht. Für mittlere Datensätze (n = 10^7) wurde ein 1.8× Speedup (2.1s vs 3.8s) erreicht, während die Speichernutzung konstant bei 8GB blieb gegenüber 12GB für Counting Sort, was die Eignung für moderne Big-Data-Anwendungen demonstriert.

- Wie wurde die Hardware-aware Optimierung implementiert, insbesondere die dynamische Anpassung von k_max basierend auf L3-Cache und Thread Count?

  Die Hardware-aware Optimierung implementiert eine dynamische Anpassung von kmax basierend auf Systemressourcen gemäß der Formel kmax = (L3 Cache)/(4 × Thread Count). Diese Implementierung gewährleistet Thread-Parallelismus bei gleichzeitig speichereffizienter Cache-Nutzung und resultierte in einer 12% Erhöhung der Cache-Auslastung verglichen mit statischen Ansätzen. Das System aktiviert konditionale Parallelisierung nur wenn vorteilhaft: Radix Sort zeigt besonders effektive Skalierung mit 1.79× Speedup für Datensätze > 10^6 Elemente trotz 12% Thread-Management-Overhead, während Quicksort aufgrund signifikanter Synchronisationskosten (47% Overhead) limitierte Parallelisierbarkeit (1.12× Speedup) aufweist.

ii.iv.iv. Paper 4: Scalable GPU Performance Variability Analysis

- Welche spezifischen CUPTI-Tabellen wurden analysiert und wie viele Entitäten enthielt jede Tabelle nach dem Left-Join?

  Laut Tabelle 1 wurden drei spezifische CUPTI-Tabellen analysiert: KERNEL (CUPTI_ACTIVITY_KIND_KERNEL) mit 842.054 Entitäten für alle Ranks, MEMCPY (CUPTI_ACTIVITY_KIND_MEMCPY) mit variierenden Entitäten pro Rank (107.045 für Rank 0, 107.099 für Rank 1, 1.070.545 für Rank 2, und 107.045 für Rank 3), sowie GPU (TARGET_INFO_GPU) mit 4 Entitäten für alle Ranks. Nach dem Left-Join-Prozess ergaben sich approximativ 93 Millionen Entitäten, die zur weiteren Analyse verwendet wurden.

- Warum entschieden sich die Autoren für Block Partitioning statt Cyclic Partitioning bei der Verteilung der Shards auf Message-Passing-Interface-Ranks?

  Die Autoren entschieden sich für Block Partitioning über Cyclic Partitioning, da der Datensatz statisch ist und eine hohe Workload-Vorhersagbarkeit aufweist. Block Partitioning weist zusammenhängende Shards jedem Rank zu, was den Query-Overhead reduziert, die Datenlokalität verbessert und eine effiziente SQL-Query-Ausführung ermöglicht. Diese Methode ist besonders vorteilhaft für statische Datensätze, da sie die Kommunikationskosten zwischen den Ranks minimiert und die Cache-Effizienz maximiert.

- Welche Methode wurde zur Identifikation der Top-5 anomalous shards verwendet und wie funktioniert diese?

  Zur Identifikation der Top-5 anomalous Shards verwendeten die Autoren die Inter-Quartile Range (IQR) Methode. Diese statistische Methode berechnet zunächst gemeinsame Statistiken (Minimum, Maximum, Standardabweichung) kollaborativ über alle P Ranks in einem Round-Robin-Verfahren. Anschließend werden diese gemeinsamen Statistiken verwendet, um Anomalien zu identifizieren, wobei die IQR-Methode Ausreißer basierend auf der Verteilung der Daten innerhalb der Quartile bestimmt und die fünf auffälligsten Shards zur detaillierten Analyse auswählt.

- Was zeigt die Analyse der Memory Stall Duration für Rank 2 bezüglich der Device-to-Host und Host-to-Device Transfers?

  Die Analyse der Memory Stall Duration für Rank 2 ergab, dass Device-to-Host und Host-to-Device Transfers dominieren, was auf häufige Ping-Pong-Muster hindeutet, die durch ineffiziente Batch-Verarbeitung verursacht werden. Im Gegensatz dazu zeigen spärliche Device-to-Device Transfers seltene Intra-GPU-Operationen an, was Optimierungsmöglichkeiten durch Shared Memory Reuse oder Tiling-Strategien aufzeigt. Diese Erkenntnisse deuten darauf hin, dass die Datenübertragungseffizienz zwischen Host und Device ein kritischer Engpass für die Performance darstellt.

- Wie skaliert die Performance des Frameworks mit zunehmender Anzahl von Message-Passing-Interface-Ranks für Data Generation und Data Aggregation?

  Das Framework zeigt eine positive Skalierung mit zunehmender Anzahl von MPI-Ranks, wobei sowohl die Data Generation als auch die Data Aggregation Phase eine Verringerung der Ausführungszeit bei steigender Rank-Anzahl aufweisen. Figure 1(c) demonstriert, dass sich die Zeiten für beide Phasen mit mehr MPI-Ranks reduzieren, was beweist, dass die Pipeline skalierbar ist und große Datenmengen effizient verarbeiten kann. Diese Skalierbarkeit wird durch die verteilte Architektur ermöglicht, die die Arbeitslast gleichmäßig auf alle verfügbaren Ranks verteilt und Bottlenecks vermeidet.

ii.iv.v. Paper 5: The Singapore Consensus on Global AI Safety Research Priorities

- Wie wird das Defence-in-Depth-Modell konkret strukturiert und welche spezifischen Überschneidungen bestehen zwischen den drei Hauptbereichen Risk Assessment, Development und Control?

  Das Defence-in-Depth-Modell strukturiert die AI Safety Forschung in drei Hauptbereiche: Risk Assessment (Bewertung der Schwere und Wahrscheinlichkeit potenzieller Schäden), Development (Entwicklung vertrauenswürdiger, zuverlässiger und sicherer Systeme) und Control (Überwachung und Intervention nach der Bereitstellung). Die spezifischen Überschneidungen werden in Figure 1 als Venn-Diagramm illustriert: Zwischen Assessment und Development liegt „Specification, validation, assurance“, zwischen Assessment und

Control „Real-time monitoring“, zwischen Development und Control „E.g. jailbreak refusal“, und im Zentrum aller drei Bereiche befinden sich grundlegende Sicherheitstechniken. Diese Überschneidungen entstehen durch unterschiedliche Definitionen dessen, was als Teil des Systems versus als kontrollierende Feedback-Schleifen betrachtet wird.

- Welche acht Personen bildeten das Expert Planning Committee und aus welchen Institutionen stammten sie, und wie gestaltete sich der mehrstufige Feedback-Prozess zur Konsensbildung?

  Das Expert Planning Committee bestand aus acht Personen: Dawn Song (UC Berkeley), Lan Xue (Tsinghua University), Luke Ong (Nanyang Technological University), Max Tegmark (MIT), Stuart Russell (UC Berkeley), Tegan Maharaj (MILA), Ya-Qin Zhang (Tsinghua University) und Yoshua Bengio (MILA). Der mehrstufige Feedback-Prozess gestaltete sich folgendermaßen: Zunächst erstellte das Committee einen Konsultationsentwurf, der an alle Konferenzteilnehmer verteilt wurde, um umfassendes Feedback einzuholen. Nach mehreren Runden von schriftlichen und persönlichen Rückmeldungen der Teilnehmer wurde das Dokument überarbeitet, um Punkte des breiten Konsenses unter den diversen Forschern zu synthetisieren.

- Was sind „Areas of mutual interest“ im Kontext der AI Safety Forschung und welche konkreten Beispiele werden für potentiell kooperative Forschungsbereiche genannt?

  „Areas of mutual interest“ bezeichnen Forschungsbereiche, bei denen verschiedene Akteure (Unternehmen, Länder) trotz Konkurrenz gemeinsame Interessen haben und Anreize bestehen, Informationen und Forschungsergebnisse zu teilen. Das Paper gibt konkrete Beispiele: bestimmte Verifikationsmechanismen, Risikomanagement-Standards und Risikobewertungen, da diese minimalen Wettbewerbsvorteil bieten, aber einem gemeinsamen Interesse dienen. Ähnlich wie konkurrierende Flugzeughersteller (Boeing und Airbus) bei Flugsicherheitsinformationen und -standards kooperieren, könnten AI-Akteure bei der Zusammenarbeit profitieren, da niemand von AI-Zwischenfällen oder der Ermächtigung böswilliger Akteure profitiert.

- Welche spezifischen Definitionen werden für die Begriffe „Artificial General Intelligence (AGI)“ und „Artificial Superintelligence (ASI)“ im Glossar gegeben?

  Das Glossar definiert „Artificial General Intelligence (AGI)“ als „AI that can do most cognitive work as well as humans. This implies that it is highly autonomous and can do most economically valuable remote work as well as humans.“ „Artificial Superintelligence (ASI)“ wird definiert als „AI that can accomplish any cognitive work far beyond human level.“ Zusätzlich wird AGI in Figure 2 als Schnittmenge von drei Eigenschaften dargestellt: Autonomy (A), Generality (G) und Intelligence (I), wobei Systeme mit allen drei Eigenschaften am schwierigsten zu kontrollieren sind.

- Wie wird Ashby’s Law of Requisite Variety im Kontext der AI-Kontrolle erklärt und welche Implikationen ergeben sich daraus für Human-centric Oversight?

  Ashby’s Law of Requisite Variety besagt, dass für Sicherheitsgarantien ein Kontrollsystem generell mindestens so viel Komplexität haben muss wie das System, das es zu kontrollieren versucht. Im Kontext von Human-centric Oversight bedeutet dies, dass es natürliche Grenzen für die Kontrollierbarkeit von Systemen gibt, basierend auf Denkgeschwindigkeit, Proaktivität, Expertisegrad, Aufmerksamkeit für Details und Zuverlässigkeit menschlicher Operatoren. Selbst mit AI-Assistenz, die Menschen beim Verstehen des gegebenen Kontexts unterstützen, deutet das Gesetz darauf hin, dass das kontrollierende System mindestens so viel Ausdrucksfähigkeit haben muss wie das kontrollierte System. Dies stellt eine fundamentale Herausforderung für die Überwachung hochentwickelter AI-Systeme dar.