

BACHELORARBEIT

Retrieval Augmented Generation mit Website-Kontext: Analyse der Auswirkung von zusätzlichem Website-Kontext auf die Antwortqualität

vorgelegt am 04. Februar 2026
Carsten Maßbaum

Erstprüferin: Prof. Dr. Larissa Putzar
Zweitprüfer: Thorben Ortmann

**HOCHSCHULE FÜR ANGEWANDTE
WISSENSCHAFTEN HAMBURG**

Department Medientechnik
Finkenau 35
20081 Hamburg

Zusammenfassung

Bei Retrieval Augmented Generation (RAG) wird ein generativer Prozess durch abgerufene Informationen aus einer externen Wissensquelle ergänzt. In dieser Arbeit wird der Frage nachgegangen, ob die Antworten von einem RAG-System für einen Benutzer hilfreicher sind, falls dem RAG weitere Informationen von der Weboberfläche zur Verfügung gestellt werden. Dafür werden zwei RAG-Systeme erstellt. An die beiden RAG-Systeme können individuell über eine Website Fragen gestellt werden. Eines der RAG-Systeme bekommt für die Beantwortung der Frage von der Website weitere Informationen, während das andere RAG-System von der Website nur die Frage bekommt. Die beiden RAG-Systeme werden mithilfe von verschiedenen Metriken miteinander verglichen. Für die Evaluierung der RAG-Systeme wurden unter anderem die Metriken BERTScore und ROUGE Score verwendet. Damit die Evaluierungen durch die Metriken durchgeführt werden können, wurden eigene zusammengehörende Frage-Antwort Paare gebildet auf Basis des Textes, der auch als Wissensbasis von den beiden RAG-Systemen verwendet wird. Im Fall von ROUGE-1 konnte eine Steigerung durch das RAG-System mit Website-Kontext um 91,13% gemessen werden und bei dem F1-Score von dem BERTScore konnte eine Steigerung von 13,83% gemessen werden. Das legt nahe, dass die Antworten des RAG-Systems mit Website-Kontext hilfreicher sind, aber vor allem wörtlich näher an den Referenzantworten liegen.

Abstract

In Retrieval Augmented Generation (RAG), a generative process is supplemented by retrieved Information from an external knowledge source. In this paper is the question examined of whether the answers of a RAG system are more helpful for a user, if the RAG system gets additional information from the web interface. Therefore, two RAG systems are created. Both RAG systems can be queried individual from a website. One of the two RAG systems gets for answering the question from the website additional information, while the other RAG system gets only the question from the website. The two RAG systems get compared with each other using different metrics. For the evaluation of the RAG systems, metrics such as BERTScore and ROUGE Score were used. For the evaluation to be performed using the metrics, own related question-answer pairs were created based on the text, which is also used as a knowledge base for the two RAG systems. In the case of ROUGE-1, an increase of 91.13% was measured through the RAG system with website-context, and the F1Score of the BERTScore measured an increase of 13.83%. That suggests that the answers of the RAG system with website-context are more helpful but mostly are closer in the wording to the reference answers.

Inhaltsverzeichnis

Abkürzungsverzeichnis	IV
Abbildungsverzeichnis	V
Tabellenverzeichnis	VI
1 Einleitung.....	7
1.1 Forschungsfrage	2
1.2 Vorgehensweise	3
1.3 Aufbau der Arbeit	4
2 Theoretischer Hintergrund.....	4
2.1 Aufbau einer HTML-Seite & HTML.....	4
2.2 HTML-Parser & DOM.....	5
2.3 RAG	6
2.4 Retriever.....	7
2.5 LLM	10
2.6 Word Embeddings	12
2.7 Metriken für RAG-Systeme	13
2.8 Relevante Studien	18
3 Methodik.....	20
4 Durchführung.....	21
5 Ergebnisse.....	32
6 Diskussion	36
6.1 Limitationen.....	40
6.2 Weiterführende Arbeiten.....	42
7 Fazit	43
Literaturverzeichnis	45
Anhang	49
Eigenständigkeitserklärung	50

Abkürzungsverzeichnis

AI	Artificial Intelligence
ANN	Approximate Nearest Neighbor
DOM	Document Object Model
HTML	Hypertext Markup Language
KI	Künstliche Intelligenz
LLM	Large Language Model
NLI	Natural Language Inference
NLP	Natural Language Processing
RAG	Retrieval Augmented Generation
RLHF	reinforcement learning from human feedback

Abbildungsverzeichnis

Abbildung 1: Kurzer Auszug aus einer HTML-Datei, in der verschiedene verschachtelte Tags zu sehen sind.	5
Abbildung 2: Einfache Darstellung eines RAG	7
Abbildung 3: Berechnungsprozess für R_{BERT}	15
Abbildung 4: Screenshot von einem Teil der Webseite für das RAG mit Website-Kontext	24
Abbildung 5: Ausgeklappter Hinweistext zu einer Beschriftung	25
Abbildung 6: Antwort des RAG-Systems ohne Website-Kontext	27
Abbildung 7: Antwort des RAG-Systems mit Website-Kontext.....	28

Tabellenverzeichnis

Tabelle 1: Fünf Frage-Antwort-Paare	30
Tabelle 2: BERTScore-Metriken für das RAG-Sytem ohne Website-Kontext.....	32
Tabelle 3: BERTScore-Metriken für das RAG-System mit Website-Kontext.....	33
Tabelle 4: Ragas-Metriken für das RAG-System ohne Website-Kontext.....	34
Tabelle 5: Ragas-Metriken für das RAG-System mit Website-Kontext	35

1 Einleitung

Die Einführung von ChatGPT hat schnell großes Interesse geweckt und viel Aufmerksamkeit auf sich gezogen. ChatGPT wurde am 30. November 2022 von OpenAI erstmals zugänglich gemacht [1], [2]. Das frühe und große Interesse an ChatGPT kann auch anhand von Zahlen belegt werden, denn nur zwei Monate nach Einführung hatte ChatGPT bereits 100 Millionen Nutzer [3]. Ein solches Wachstum hatte zuvor noch kein Produkt in dem Tempo erreicht [3]. ChatGPT, aber heutzutage auch andere Modelle, können auf die Frage eines Benutzers eine passende Antwort generieren. Diesen Antworten auf Benutzeranfragen sind oft von einer guten Qualität. Damit ChatGPT und andere Modelle derartig gut Antworten liefern können, werden sogenannte Large Language Models (LLMs) als Grundlage verwendet [4]. Trotz der erstaunlichen Antwortqualität von Large Language Models können diese Modelle dennoch Antworten generiert werden, die als fehlerhaft bezeichnet werden können. Solches fehlerhafte Verhalten wird auch als Halluzinieren bezeichnet, wenn der generierte Text untreu zu einem eingegebenen Dokument ist oder neue Inhalte entstehen, die nicht von einer Quelle gestützt werden [5].

Eine Möglichkeit diesem Halluzinieren von LLMs entgegenzuwirken ist das Verwenden von Retrieval Augmented Generation (RAG) [6]. Bei RAG handelt es sich um ein KI-Verfahren, bei dem einem LLM durch eine Suche, beispielsweise in einer Dokumentensammlung, Wissensquellen zur Verfügung gestellt werden, um eine Antwort zu generieren [7]. Damit die Suche hilfreiche Ergebnisse für das LLM liefern kann, ist es hilfreich, wenn aktuelle Informationen für die Suche vorliegen. In der heutigen schnelllebigen Welt sind Informationen allerdings auch schnell veraltet. Daher kann es herausfordernd sein, stets die aktuellen Informationen für ein oder mehrere Themen bereitzuhalten.

Um aktuelle Informationen bereitstellen zu können, können Websites hilfreich sein, beispielsweise Websites von Nachrichtensendern. Noch hilfreicher als allgemeine Informationen aus dem Internet sind für ein RAG, womöglich Informationen von der Website, auf der sich der Benutzer aktuell befindet. Denn es ist davon auszugehen, dass der Benutzer eine Frage stellt mit Bezug auf die Inhalte, die er auf der Website findet, auf der er sich aktuell aufhält. Daher sind Informationen von der Website, auf der sich der Benutzer aktuell befindet, vermutlich besonders interessant für ein RAG-System, um eine Frage des Benutzers beantworten zu können.

Somit soll sich in der folgenden Arbeit damit beschäftigt werden, ob einem RAG-System Informationen von der Website, auf der sich der Benutzer aktuell befindet, helfen, eine Antwort auf die Frage des Benutzers zu generieren.

1.1 Forschungsfrage

Die Forschungsfrage, die in dieser Arbeit beantwortet werden soll, lautet: *Sind die Antworten von einem RAG-System für Benutzer von Websites hilfreicher, wenn dem RAG-System weitere Informationen von der Weboberfläche zur Verfügung gestellt werden?*

Diese Forschungsfrage ist relevant, weil mittlerweile viele Informationen und Angebote im Internet zu finden sind und eine Kombination von RAG und Websites somit viel Potential hat Kunden von Websites schnell die gewünschten Informationen in einer hohen Qualität zugänglich zu machen, ohne das Kunden lange suchen müssen.

Ein Anwendungsfall wäre beispielsweise ein RAG-System mit Website-Kontext von der jeweiligen Website als Chatfunktion. Als Chatfunktion könnten Benutzer Fragen zur Website und den Inhalten auf der Website an das RAG-System stellen und würden Antworten zurückbekommen. Dadurch, dass bei dieser Chatfunktion dem RAG-System Informationen von der Website, auf der sich der Benutzer aktuell befindet, zur Verfügung gestellt werden, könnte diese Chatfunktion noch stärker auf die Inhalte von der Website eingehen als es möglicherweise bereits bestehende Chatfunktionen auf Websites könnten. Dadurch könnten Fragen noch besser beantwortet werden. Weil mehr Fragen bereits durch die Chatfunktion geklärt werden könnten, kann auch mehr Zeit für die Kunden und Mitarbeitern, die zum Unternehmen gehören, gespart werden. Besonders der jeweilige Kundensupport könnte noch stärker entlastet werden. In Bereichen wie Arztpraxen, Krankenhäusern und Behörden, aber auch Online-Händlern könnten diese Chatfunktionen für große Entlastung sorgen.

Zusätzlich kann ein verbesserter Bezug zum Inhalt auf einer Website bei einer Chatfunktion dafür sorgen, dass mehr Personen mit der Website interagieren. Besonders bei Personen, die mit der Verwendung von Websites bisher Probleme hatten, könnte ein verstärkter Bezug zum Inhalt auf der Website bei Chatfunktionen dafür sorgen, dass auch diese Personen leichter Inhalte auf einer Website finden oder Funktionen wie den Bezahlvorgang schneller bedienen können. Somit könnten Personen, die mit der Bedienung von Websites ungeübt sind, besser mit der Website interagieren. Das ist einerseits wichtig für die betroffenen Personen, die so mehr Teilhabe erreichen können. Andererseits profitiert auch die Websites, da so mehr Personen, die Website verwenden.

Ein weiterer Punkt, warum die Forschungsfrage relevant ist, sind die vielen aktuellen Informationen auf der Weboberfläche von Websites. Diese Informationen auf den Websites haben das Potential durch ihre Qualität und Aktualität die Antworten eines RAG-Systems zu verbessern. Zusätzlich werden Informationen auf der Weboberfläche von Websites auch durch Websitebetreiber in einigen Fällen aktuell gehalten werden, wodurch stetig aktuelle Informationen für RAG-Systeme verwendet werden könnten. Im Allgemeinen sind aktuelle Informationen für RAG-Systeme wichtig, um immer die aktuellen und bestmöglichen Antworten liefern zu können.

Dennoch sind für RAG-Systeme in Bereichen wie Gesundheit aktuelle Informationen noch einmal mehr von entscheidender Bedeutung, damit Patienten die optimale Hilfe erfahren und nicht im schlimmsten Fall sogar durch veraltete Informationen Schaden nehmen. Ein weiterer Bereich, indem RAG-Systeme besonders stark von stetig aktuellen Informationen profitieren könnten, wären RAG-Systeme im juristischen Bereich bei neuen Urteilen oder Veränderungen von Gesetzen. Ebenso könnten RAG-Systeme im Forschungskontext besonders profitieren, wenn neue Studien im jeweiligen Fachbereich besonders schnell mitberücksichtigt werden können.

Die Behandlung dieser Forschungsfrage kann also dazu beitragen, dass RAG-Systeme bessere Antworten auf Fragen zu Inhalten auf der aktuell verwendeten Website geben können. Somit kann diese Forschung für alle RAG-Systeme interessant sein, die im Zusammenhang mit einer Website verwendet werden oder benutzt werden sollen.

1.2 Vorgehensweise

Es soll an dieser Stelle einmal festgehalten werden, dass wenn in dieser Arbeit von einem RAG-System mit Website-Kontext die Rede ist, das RAG-System mit Informationen von der Weboberfläche gemeint ist.

Um die Forschungsfrage in dieser Arbeit beantworten zu können, werden zwei RAG-Systeme erstellt. Diese beiden RAG-Systeme bekommen dasselbe Dokument als Quelle, für das Suchen und Abrufen der Informationen für das LLM zur Verfügung gestellt. Zusätzlich werden verschiedene Webseiten erstellt, die miteinander verbunden sind. Über die Website kann ein RAG-System ausgewählt werden und Fragen an das ausgewählte RAG-System gestellt werden. Die Antworten von den RAG-Systemen werden, nach dem Generieren der Antworten, auf der Website angezeigt. Einem der RAG-Systeme werden zur Beantwortung der Fragen eines Nutzers, zusätzliche Informationen von der Website zur Verfügung gestellt. Dem anderen RAG-System werden keine Informationen von der Website zur Verfügung gestellt. Dadurch bekommt dieses RAG-System nur die Frage von der Website.

Die beiden RAG-Systeme werden dann durch verschiedene Metriken bewertet. Dafür wird ein Datensatz, der aus Fragen und zugehörigen Antworten besteht, zur Hilfe genommen. Die Metriken liefern verschiedene Messwerte für die beiden RAG-Systeme. Mithilfe dieser Messwerte können dann die beiden RAG-Systeme miteinander verglichen werden und die Ergebnisse interpretiert werden. Damit sollten Rückschlüsse in Bezug auf die Forschungsfrage dieser Arbeit möglich sein.

1.3 Aufbau der Arbeit

Nach der Einleitung in diesem Kapitel 1 folgt in Kapitel 2 eine Vorstellung und Erläuterung des theoretischen Hintergrundes für diese Arbeit. Im nächsten Kapitel, Kapitel 3, wird die Methodik vorgestellt. Danach wird in Kapitel 4 die Durchführung beschrieben. Im Anschluss in Kapitel 5 werden die Ergebnisse der Evaluationen der beiden RAG-Systeme dargestellt. Darauf werden in der Diskussion, Kapitel 6, die Ergebnisse diskutiert und interpretiert in Bezug auf die Forschungsfrage. Zuletzt wird noch ein Fazit, in Kapitel 7, gezogen und die Forschungsfrage abschließend beantwortet.

2 Theoretischer Hintergrund

In diesem Kapitel werden die wichtigsten Begriffe und Konzepte für diese Arbeit erläutert und relevante Studien vorgestellt. Es wird der Aufbau einer HTML-Seite und das Parsen einer solchen Seite HTML-Seite erklärt. Darauf wird RAG allgemein vorgestellt. Im Anschluss wird der Retriever genau betrachtet. Danach werden Large Language Modells (LLMs) vorgestellt. Darauf werden die Word Embeddings erläutert. Zum Ende des Kapitels werden Metriken für RAG-Systeme vorgestellt und es wird auf relevante Studien eingegangen.

2.1 Aufbau einer HTML-Seite & HTML

HTML ist eine Abkürzung für Hypertext Markup Language [8, S. 72]. Zuerst soll geklärt werden, welche Aufgabe eine Markup Language erfüllt. Die Aufgabe einer Markup Language, also einer Auszeichnungssprache, ist das Strukturieren von einem Text [9, S. 28]. Der zweite Bestandteil des Namens HTML ist Hypertext. Aus dem Gabler Wirtschaftslexikon ist die Definition für Hypertext die Folgende „Struktur von Dokumenten in Rechnernetzen (z.B. dem World Wide Web), die Verweise auf andere Dokumente beinhalten (Hyperlink) und die Möglichkeit bieten, durch Anklicken dieses Verweises das korrespondierende Dokument direkt aufzurufen“ [10]. Damit kann nun geklärt werden, wofür HTML eingesetzt werden kann. Im Fall von HTML, kann HTML Websites miteinander verknüpfen und gliedern [9, S. 28].

Inhalte, die von Tags eingeschlossen sind, bilden die Elemente aus denen eine Auszeichnungssprache aufgebaut ist [9, S. 28]. Ein Starttag bildet dabei den Anfang eines Elements und das zugehörige Endtag schließt das Element ab, so ist meistens der Aufbau eines Elements [9, S. 28]. Die Tags stehen zum Abheben gegenüber anderem Text in spitzen Klammern und das Endtag wird durch einen hinzukommenden Slash / gekennzeichnet [11, S. 8].

Dabei können HTML-Dateien aus unterschiedlich vielen Elementen aufgebaut sein. Die Elemente können teilweise auch ineinander geschoben und geschachtelt werden [9, S. 29]. Es gibt einige wenige HTML-Elemente die als Grundbausteine für eine HTML-Datei dienen und somit in praktisch allen HTML-Dateien auftauchen. Über diese grundlegenden HTML-Elemente hinaus gibt es viele weitere HTML-Elemente, die nach Bedarf verwendet werden oder weggelassen werden können.

Eigenschaften, die auch als Attribute bezeichnet werden, können von einigen Tags enthalten werden und sind stets im Starttag enthalten [11, S. 9]. Mithilfe dieser Attribute können die einzelnen HTML-Elemente noch genauer und individueller angepasst werden, da diese Attribute teilweise Auswirkungen darauf haben können, wie das HTML-Element später angezeigt wird. In einigen Fällen können Tags auch mehrere Attribute haben, die dann durch Leerzeichen getrennt werden [11, S. 9].

In der unten dargestellten Abbildung 1 sind verschiedene Tags zu sehen, die ineinander verschachtelt sind. Zusätzlich haben die Tags in der Abbildung auch Attribute beispielsweise das Class-Attribut. Das a-Tag enthält sogar zwei Attribute, nämlich die Attribute „class“ und „href“.

```
<div class="col-sm-6">
  <div class="card">
    <div class="card-body">
      <h5 class="card-title">RAG mit HTML</h5>
      <p class="card-text">In diesem Fall bekommt das RAG-System, die Informationen von der Website zur Verfügung gestellt.</p>
      <a href="/evaluate_page_ask_llm_mit_html" class="btn btn-primary">Weiter</a>
    </div>
  </div>
</div>
```

Abbildung 1: Kurzer Auszug aus einer HTML-Datei, in der verschiedene verschachtelte Tags zu sehen sind.
Quelle: Eigene Darstellung

HTML ist elementar wichtig für eine Website. Das liegt daran, dass eine Website aufgebaut ist, aus dem von einem Webbrowser angezeigten HTML-Code [9, S. 28]. Aus diesem Grund ist HTML so wichtig für eine Website.

2.2 HTML-Parser & DOM

Nachdem geklärt wurde, was HTML ist und wie ein HTML-Dokument allgemein aufgebaut ist, soll nun betrachtet werden, wie Programme auf Informationen aus einem HTML-Dokument zugreifen können.

Wird ein HTML-Dokument von einem Webbrowser geparst, dann wird ein DOM-Baum gebaut und verwendet, um das Dokument dazustellen [12]. Bei dem Document Object Model (DOM) handelt es sich aus Sicht des Computers um die strukturelle Darstellung eines HTML-Dokuments [9, S. 50]. Durch das DOM wird ein Dokument als logischer Baum repräsentiert [12]. Genauer gesagt handelt es sich bei eine Baum-Struktur, dessen Knoten Inhalte von HTML oder XML-Dokumenten darstellen, um einen DOM-Baum [12].

Das DOM ermöglicht als Schnittstelle auf Tags, Attribute und Inhalte von HTML-Seiten zuzugreifen [9, S. 34]. Mithilfe des DOMs kann auf HTML-Seiten strukturiert zugegriffen werden und somit auch HTML-Seiten erstellt, modifiziert oder interpretiert werden [9, S. 34]. Für Programme wird HTML benutzbar durch das DOM [9, S. 34].

2.3 RAG

In diesem Kapitel soll näher auf Retrieval Augmented Generation (RAG) eingegangen werden. Die Grundlage für Retrieval Augmented Generation ist das Verbinden von Textgenerierung, wie es generative Künstliche Intelligenz vermag, mit Modellen, die in der Lage sind den relevantesten Text zu finden [13]. Die genannte Textgenerierung durch Künstliche Intelligenz kann durch Large Language Models (LLMs) umgesetzt werden. Systeme zum Abrufen von Informationen werden bei Retrieval Augmented Generation genutzt, um die relevantesten Informationen zu einer Anfrage zu finden und diese gefundenen Informationen dann wiederum zu verwenden, um vollständige Antworten zu erstellen [13].

Nun soll der Aufbau eines RAG genauer betrachtet werden. Aus zwei Kernmodulen ist ein vollständiges RAG aufgebaut [14]. Diese beiden Kernmodule sind der Retriever, der nach relevanten Informationen im Datenspeicher sucht und der Generator, der die gewünschten Inhalte erstellt [14]. Mit diesem allgemeinen Aufbau eines RAG kann auch die Arbeitsweise eines RAG veranschaulicht werden. Der RAG Prozess beginnt damit, dass der Retriever eine Anfrage erhält und dazu nach relevanten Informationen sucht [14]. Danach werden die gefundenen Inhalte und die ursprüngliche Anfrage durch eine bestimmte Erweiterungs-Methodik (engl. augmentation methodology) an den Generator übergeben [14]. Im letzten Schritt wird dann durch den Generator die gewünschte Ausgabe erstellt [14].

In [15] wird auf RAG für die Verarbeitung von natürlicher Sprache eingegangen. Nach [15] besteht ein RAG normalerweise aus drei Komponenten: Dem Retriever, dem Generator und „Retrieval fusions“. Die abgerufenen Informationen zu verwenden, um die Generierung zu ergänzen, darauf zielt die Retrieval fusions ab [15]. Es ist möglich die Verbindungstechniken (engl. fusion techniques) in drei große Techniken einzuteilen [15]. Eine dieser Techniken ist die sogenannte „Query-based fusion“ [15]. Die Technik, bei der Eingaben mit abgerufenen Informationen ergänzt werden, bevor diese in den Generator gegeben werden, wird als „Query-based fusion“ bezeichnet [15].

Im Folgenden wird ein Überblick über RAG für NLP gegeben [15]. Unter Natural Language Processing (NLP) ist die Verarbeitung von natürlicher Sprache zu verstehen. Um das Generierungs-System zu verbessern, verwendet ein RAG-System eine externe Wissensbasis [15]. Die externe Wissensbasis besteht aus externen Dokumenten, von denen jedes eine Reihe von Blöcken bzw. Brocken (engl. chunks) beinhaltet, wenn beispielhaft externe Dokumente verwendet werden [15]. Mit Blöcken und Brocken sind in diesem Zusammenhang Textbrocken und Textblöcke bzw. Informationsblöcke und Informationsbrocken gemeint.

Durch ein Einbettungs-Model (engl. embedding model) werden die Blöcke bzw. Brocken (engl. chunks) in Vektor-Einbettungen (engl. vector embeddings) transformiert [15]. Wird eine Frage eingegeben, die als Vektor eingebettet wird, dann kommt es dazu, dass die Top-k relevantesten Blöcke zu der Frage von dem Retriever im RAG-System abgerufen werden [15]. Es können verschiedene Fusionsmethoden (engl. fusion methods) von dem RAG-System eingesetzt werden, um die abgerufenen Blöcke bzw. Brocken (engl. chunks) zu verbinden [15].

Beispielsweise die oben erwähnte Query-based Fusion. In Abbildung 2 ist ein einfaches RAG zu abgebildet. Es ist zu sehen, dass die Frage des Benutzers (Query) sowohl an den Retriever als auch an den Generator gesendet wird. Der Generator bekommt von den dem Retriever die abgerufenen Inhalte (retrieved Content). Mit dem abgerufenen Inhalt und der Frage des Benutzers generiert der Generator die Antwort. Diese Antwort ist dann die Ausgabe des Systems.

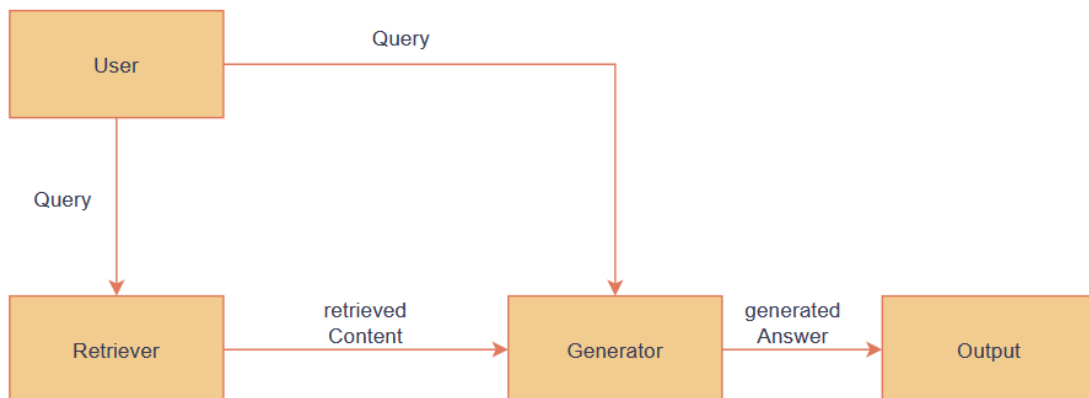


Abbildung 2: Einfache Darstellung eines RAG
Quelle: In Anlehnung an [14]

2.4 Retriever

In Kapitel 2.3 wurde der Retriever bereits als ein Kernmodul von RAG vorgestellt. Nun soll der Retriever etwas detaillierter betrachtet werden.

Zu Beginn eine kurze Erklärung, was mit Retrieval gemeint ist. Unter Retrieval ist zu verstehen, dass wenn ein Informationsbedarf besteht, die relevanten Informationen identifiziert und erhalten werden [14].

Erst den Retriever zu bauen bzw. zu erstellen und danach den Retriever zu befragen, das sind die zwei Phasen, um einen Retriever verwenden zu können [15]. Zuerst soll auf das Erstellen eines Retrievers eingegangen werden. Der Prozess einen Retriever aus einem großen Korpus natürlicher Sprache zu bauen, kann wiederum in drei Schritte aufgeteilt werden [15]. „Chunking corpus, encoding chunks, and building the vector database“ sind die drei erwähnten Schritte, um einen Retriever zu erstellen [15]. Ein Dokument in mehrere Absätze zu zerlegen, ist der Prozess, der als „Chunking“ bezeichnet wird [16]. Somit wird im ersten Schritt der (Text-)Korpus in kleine Teilstücke bzw. Textfragmente zerteilt. Auch für das Chunking sind verschiedene Arten von Chunking-Techniken einsetzbar [15]. Eine dieser Arten ist das semantische Chunking [15]. Mit Zahlen Textblöcke als Vektoren darzustellen (Einbettungen), das ist mit Kodierung gemeint [15]. „Encoding chunks“ bedeutet also die Textstücke in Vektoren umzuwandeln. Der Retriever ist in der Lage Ähnlichkeitssuche aufgrund von inhaltlicher Relevanz auszuführen, da im Allgemeinen die Einbettungen (engl. embeddings) die Semantik beinhalten [15].

Es kann zwischen spärliche Kodierungen (engl. sparse encodings) und dichten Kodierungen (engl. dense encodings) als Arten von Kodierungs-Methoden unterschieden werden [15]. Hochdimensionale Vektoren bei denen die meisten Elemente Null sind, entstehen durch Texte, die von spärliche Encodings repräsentiert werden [15]. Vektoren bei denen die meisten Elemente nicht Null Gleitkommazahlen sind und bei denen jede Dimension eine Reihe aus semantischen Merkmalen enthalten kann, werden mithilfe von dichte Kodierungen erzeugt [15]. Durch (tiefe) neuronale Netzwerkmodelle werden in der Regel die dichten Einbettungen erstellt [15].

Der letzte Schritt, um einen Retriever aus einem Textkorpus zu erstellen ist, wie bereits zuvor erwähnt, die Vektordatenbank zu erstellen. Der grobe Aufbau einer Vektordatenbank wird hier kurz angerissen. Eine spezialisierte Datenbank, die Daten als eine Sammlung von Schlüssel-Wert Paaren speichert und managet, ist der genutzte Datenspeicher in einer Vektordatenbank [15]. Dabei sind Werte das domänen-spezifische Wissen und die Schlüssel sind die einzigartigen Identifikatoren von hochdimensionalen Einbettungen [15].

Den Suchprozess für Daten, ähnlich zu hochdimensionalen Einbettungen der Frage, zu beschleunigen, darauf zielt die Indexierung in der Vektordatenbank ab [15].

Der Grad an Relevanz zwischen der eingebetteten Anfrage und den eingebetteten Blöcken wird durch die Grundkomponenten im Retriever, den Ähnlichkeits-Metriken, gemessen [15]. Die Kosinus-Ähnlichkeit, die euklidische Ähnlichkeit und die Manhattan Distanz zählen zu den üblichen Ähnlichkeits-Metriken [15].

Definiert ist die Kosinus-Ähnlichkeit folgendermaßen:

„Sei $\mathbb{R}^d$ ausgestattet mit dem Standardskalarprodukt und der davon induzierten euklidischen Norm. Für $x, y \in \mathbb{R}^d \setminus \{0\}$ heißt dann

$$\text{cossim}(x, y) := \frac{\langle x, y \rangle}{\|x\| \|y\|} = \frac{\sum_{i=1}^d x_i y_i}{(\sum_{i=1}^d x_i^2)^{1/2} (\sum_{i=1}^d y_i^2)^{1/2}} \in [-1, 1]$$

die Kosinusähnlichkeit von x und y “ [17, S. 46].

ANN steht für annähernder nächster Nachbar (engl. approximate nearest neighbor) [18], [15]. Methoden oder Strukturen, die zum Organisieren und Managen von Daten eingesetzt werden, damit der ANN-Suchprozess optimiert ist in Bezug auf die Retrieval-Qualität und die Retrieval-Effizienz, werden im Allgemeinen als ANN-Indexierung (engl. ANN indexing) bezeichnet [15]. Zu den fortgeschrittenen ANN-Indexierungs-Techniken gehört unter anderem HNSW [19], [15]. Eine inkrementelle ANN-Suchstruktur, die vollkommen Graph basiert ist, dabei handelt es sich um HNSW [19]. In HNSW wird wahrscheinlichkeitsbasiert die mehrschichtige Graphen-Struktur bestimmt [15]. Neben HNSW geht es noch verschiedene weitere fortgeschrittene ANN-Indexierungs-Techniken [15].

In diesem Absatz soll es darum gehen, wie der Retriever befragt wird. Den Retriever zu befragen umfasst im Wesentlichen drei Schritte [15]. Diese drei Schritte sind: Die Kodierung der Anfrage, ANN-Suche und die Nachbearbeitung (engl. post-processing) [15].

Derselbe Encoder um Fragen zu codieren während der Anfragephase wird von dem Retriever verwendet, um sich dem Einbettungsraum anzupassen [15]. Um mithilfe von ANN-Suchalgorithmen ähnliche Daten zu finden und die entsprechenden Werte abzurufen, nutzt die ANN-Suche die vorab erstellte Indexierung und den Datenspeicher [15]. Im vorab erstellten Index zu suchen, das Auffinden der k nächsten Nachbarn und die einzigartigen Identifikatoren der k nächsten Nachbarn zurückzugeben, ist mit dem Suchen im Index gemeint [15]. Abhängig von der Indexstruktur oder dem Algorithmus für die Indexierung ist der Suchprozess für den nächsten Nachbarn [15]. Basierend auf den nächsten Schlüssel Identifikatoren werden die entsprechenden Werte aus dem Datenspeicher abgerufen [15].

Eine gängige Technik zur Nachbearbeitung ist das „Reranking“ [15]. Das abgerufene Wissen aufgrund von aufgabenspezifischen Zielen neuanzuordnen, darauf zielt das Reranking ab [15].

2.5 LLM

Große Sprachmodelle (engl. Large Language Models) können in einem RAG eingesetzt werden. Dabei können die Fähigkeit von LLMs, Antworten zu generieren, innerhalb von einem RAG verwendet werden. Daher soll an dieser Stelle etwas genauer auf Large Language Models eingegangen werden.

Aber zuerst sollen Sprachmodellen allgemein betrachtet werden. Künstliche-Intelligenz Modelle, die dazu in der Lage sind mathematisch die Struktur und Benutzung von Sprache auszudrücken, das sind Sprachmodelle [20]. Diese Modelle können Sprache nicht nur verstehen, sondern können auch Text generieren und können mithilfe von großen Mengen an Daten trainiert werden [20]. Durch das Untersuchen der Reihenfolge und Auftreten von Sätzen, Wörtern und größeren Textabschnitten können Sprachmodelle das nächste Wort vorhersagen [20].

Die Anzahl der Parameter, die beim Training des Modells genutzt werden, sind genauso wichtig wie die Größe und Struktur des Datensatzes, um die Kapazität an Wissen und Fähigkeiten zu bestimmen [20]. Die Komplexität von Satzstrukturen und Sprache zu verstehen, dazu sind Modelle mit einer größeren Menge an Parametern in der Lage [20].

Die Leistung, Kapazität und Lernfähigkeit des Modells werden direkt durch die Anzahl der Parameter in einem Sprachmodell beeinflusst [20]. Mit einer höheren Anzahl an Parametern haben Modelle eine höhere Rechenleistung und höhere Ressourcen Anforderungen, jedoch können sie in einem weiteren Gebiet arbeiten [20].

Bei LLMs handelt es sich um statistische Sprachmodelle, die auf neuronalen Netzen basieren, vortrainiert (engl. pre-trained) sind und groß angelegt (engl. large-scale) sind [21]. LLMs sind in der Lage komplexe Probleme zu lösen, indem sie Texte generieren und sie verfügen über große Fähigkeiten beim Verstehen von natürlicher Sprache [22].

LLMs haben auch Parameter, die das Verhalten des LLMs beeinflussen können und die eingestellt werden können. Ein wichtiger Parameter ist dabei die Temperatur. Eine erhöhte Unruhe und Entropie, was zu mehr Unsicherheit und Zufälligkeit im generativen Prozess führt, wird durch eine höhere Temperatur ausgelöst [23]. Im Bereich von $[0,2]$ liegen die Werte für die Temperatur normalerweise vor, wobei eine Temperatur von 0 in der Praxis heißt, stets das Token mit der höchsten Wahrscheinlichkeit auszuwählen [23]. LLMs zu untersuchen und Ausgaben zu vergleichen wird immer schwieriger, desto mehr Unsicherheit und Zufälligkeit in den Generierungsprozess eingebracht werden [23].

Auf Transformer[24] basierende neuronale Sprachmodelle, die mehrere zehn bis hunderte Milliarden Parameter enthalten und auf riesigen Textdaten vortrainiert (engl. pre-trained) wurden, darauf beziehen sich große Sprachmodelle hauptsächlich [21]. Unter anderem PaLM[25] ist dafür ein Beispiel [21].

Die Modellarchitektur Transformer setzt, um zwischen Eingabe und Ausgabe globale Abhängigkeiten herzustellen, vollständig auf Aufmerksamkeitsmechanismen (engl. attention mechanisms) [24]. Zudem wird eine deutlich höhere Parallelisierung durch den Transformer ermöglicht [24].

Es ist möglich ein LLM allgemein in vier Hauptteile bzw. Funktionen zu unterteilen [26]. Diese vier Hauptteile sind erstens die Tokenisierung (Tokenizer), zweitens die Einbettung (Embeddings), drittens die Berechnung der Wahrscheinlichkeit des nächsten Tokens (Vorhersage) und viertens die Strategien zur Auswahl der Ausgabe (teilweise auch Dekodierung genannt) [26].

Die Umwandlung einer Textsequenz in kleinere Teile, die als Tokens bezeichnet werden, wird als Tokenisierung bezeichnet [21]. Drei beliebte Tokenizer sind BytePairEncoding, WordPieceEncoding und SentencePieceEncoding [21].

Token auf Vektoren abzubilden das meint die Einbettung (Embedding) im zweiten Schritt [26]. Auf Embeddings bzw. Word Embeddings wird in Kapitel 2.6 noch stärker eingegangen.

Die Berechnung der Wahrscheinlichkeit des nächsten Tokens geschieht im dritten Schritt [26]. Alle aktuellen LLMs verwenden neuronale Netze und die Mehrheit verwendet die Transformer-Architektur, wobei prinzipiell jede Vorhersagetechnik möglich wäre [26]. Mit möglichst geringen Rechenaufwand auf vielen Daten zu trainieren, bestmöglich zu sein und große Eingaben zu erfassen, von diesen Fähigkeiten hängt primär die Wahl der Architektur ab [26].

In Schritt vier findet die Dekodierung statt. Der Prozess der Textgenerierung unter Verwendung vortrainierter LLMs wird als Dekodierung bezeichnet [21]. Für die Dekodierung sind unterschiedliche Strategien vorgestellt worden [21]. Zu den gängigsten Strategien gehören Greedy Search, Beam Search, Top-k Sampling oder Top-p Sampling [21].

Ein LLM zu erstellen und zu trainieren kann sehr komplex sein und kann viele verschiedene Techniken umfassen. An dieser Stelle soll auf einige besonders wichtige Techniken eingegangen werden.

Das Vortraining (engl. pre-training) hilft einem LLM grundlegende Sprachverständnisfähigkeiten zu erlangen, die hilfreich sein können in einer Vielzahl von sprachbezogenen Aufgaben und das Vortraining ist der erste Schritt in der Trainingspipeline für große Sprachmodelle [21]. Beim Pre-training wird das LLM auf einer riesigen Mengen an in der Regel unmarkierten (engl. unlabeled) Texten trainiert und das meistens in einer selbst-überwachten (engl. self-supervised) Weise [21]. Beispielsweise kann die Vorhersage des nächsten Satzes (engl. Next Sentence Prediction, NSP) [27] als eine von verschiedenen Herangehensweisen für das Pre-training genutzt werden [21]. Die Vorhersage des nächsten Tokens (autoregressive language modeling) und die maskierte Sprachmodellierung (engl. masked language modeling) gehören zu den zwei gängigsten Herangehensweisen [21].

Während für die Verwendung von neueren LLMs keine Feinabstimmung (engl. fine-tuning) mehr erforderlich ist, können diese Modelle dennoch von aufgaben- oder datenspezifischer Feinabstimmung profitieren [21]. Bekanntermaßen verbessert die Feinabstimmung auf eine oder mehrere Aufgaben die Ergebnisse und sorgt für eine geringere Komplexität beim Prompt Engineering [21]. Außerdem kann die Feinabstimmung als eine Alternative zu Retrieval Augmented Generation dienen [21].

Den Prozess KI-Systeme auf menschliche Ziele, Präferenzen und Prinzipien auszurichten, ist mit KI Anpassung (engl. AI Alignment) gemeint [21]. KI ist eine Abkürzung und steht für künstliche Intelligenz. Im Englischen ist mit „AI“ die „Artificial Intelligence“ gemeint also auch deutsch die künstliche Intelligenz. Für die Anpassung ist unter anderem RLHF (reinforcement learning from human feedback) ein beliebter Ansatz [21].

Um durch menschliches Feedback die Anpassung zu erlernen verwendet RLHF ein Belohnungsmodell [21]. Verschiedene Ausgaben zu bewerten und sie anhand von Anpassungspräferenzen von Menschen mit Punkten zu bewerten, dazu ist das Belohnungsmodell in der Lage, nachdem es abgestimmt wurde [21].

2.6 Word Embeddings

In diesem Abschnitt sollen die Word Embeddings genauer betrachtet werden. Die in Kapitel 2.5 erwähnt Embeddings werden hier vertieft betrachtet.

Eine Vektordarstellung von Wörtern in einem mehrdimensionalen Raum, worin ähnliche Worte nah beieinander liegen, das sind Word Embeddings zusammenfassend gesagt [28]. Forscher und Anwender verfolgen in der Regel eine von zwei primären Strategien, um Word-Embeddings Modelle effektiv zu verwenden: Entweder wird ein vortrainiertes Modell verwendet oder es wird von Grund auf ein eigenes Modell trainiert [28]. Der direktere Ansatz ist die Verwendung von vortrainierten Modellen, bei dem die Modelle als leicht verfügbares Werkzeuge von unterschiedlichen Plattformen, wie beispielsweise Hugging Face[29], genutzt werden [28].

Anhand des Ansatzes, der zum Erlernen der Vektoren verwendet wird, lassen sich Word-Embedding Modelle kategorisieren [28]. Als count-based Modelle, werden Methoden bezeichnet, die globale Informationen nutzen im Allgemeinen korpusweite Strategien wie Wortanzahl oder Frequenz [30]. Andererseits gibt es prediction-based Modelle, das sind Methoden, die lokale Daten, wie Beispielsweise den Wort Kontext, nutzen und mit neuronalen Sprachmodellen entwicklungstechnische tief verbunden sind [30].

Für die Generierung von Word-Embeddings sind neuronale Netzwerke zu einem zentralen Werkzeug geworden, im Bereich von NLP [28]. Immer komplexer sind neuronale Netzwerke im Laufe der Zeit geworden, was zur Verbesserung ihrer Fähigkeiten zum Repräsentieren von Textinformationen geführt hat [28].

In ihrer Fähigkeit den Kontext, in dem ein Wort in einem Satz oder einem größeren Text vorkommt zu berücksichtigen, darin unterscheiden sich in erster Linie kontextuelle und nicht-kontextuelle Word Embeddings Modelle [28]. Unabhängig vom Kontext in dem ein Wort auftaucht, wird jedem Wort ein statische Embeddings Vektor zugewiesen von einem nicht-kontextuellem Word Embeddings Modell [28]. Derselbe Begriff hat daher stets dieselbe Vektordarstellung [28]. Unter anderem GloVe[31] und Word2Vec[32] sind Beispiele für nicht-kontextuelle Modelle [28].

Der Kontext, in dem ein Wort vorkommt, berücksichtigt ein kontextuelles Modell, um den Embeddings Vektor zu generieren [28]. Der breitere Kontext und die Feinheiten an Bedeutung, in denen Wörter eingebettet sind, werden von diesen Modellen erfasst [28]. BERT[27] ist ein Beispiele für kontextuelle Modelle [28]. BERT bedeutet ausgeschrieben Bidirectional Encoder Representations from Transformers [27]. In der Arbeit zu BERT werden mit einer bidirektionalen Modell Architektur verschiedene Methoden wie Feinabstimmung erprobt [27].

Das Konzept Worte einzubetten kann auch auf Sätze erweitert werden. In der Arbeit [33] wird Sentence-BERT vorgestellt und genau beschrieben. Unter Verwendung von Siamese- und Triplet-Netzwerken wird das BERT-Netzwerk modifiziert, das ist Sentence-BERT, das in der Lage ist semantisch aussagekräftige Satz Einbettungen abzuleiten [33]. Das semantische ähnliche Sätze im Vektorraum nah beieinander sind, ist unter semantisch aussagekräftig zu verstehen [33].

Das Vektoren fester Länge für Eingabesätze hergeleitet werden können, ermöglicht die Siamese-Netzwerkarchitektur [33]. Semantisch ähnliche Sätze lassen sich durch Ähnlichkeitsmaße wie die Kosinus-Ähnlichkeit, Manhattan- oder Euklidische Distanz finden [33].

2.7 Metriken für RAG-Systeme

RAG-Systeme generieren in der Regel Antworten auf Anfragen eines Benutzers. Diese Antworten liegen im Allgemeinen, in einer für Menschen verständlichen Sprache vor. Daher können auch zum Evaluieren von RAG-Systemen unter anderem Metriken verwendet werden, die bereits beim Evaluieren von NLP-Aufgaben eingesetzt wurden. Darüber hinaus können noch weitere Metriken verwendet werden, um Messwerte für RAG-Systeme zu erhalten.

Einige Metriken, die unter anderem zum Evaluieren von RAG-Systemen eingesetzt werden können, sollen im Folgenden Abschnitt vorgestellt werden.

Wenn nach der Definition von Metrik im IEEE-Standard geschaut wird, wird auf Software-Qualitäts Metrik verwiesen mit der Anmerkung das Metrik an Stelle von Software-Qualitäts Metrik im Standard verwendet wird [34, S. 2]. Eine Metrik also eine Software-Qualitäts Metrik ist laut Defintion: „A function whose inputs are software data and whose output is a single numerical value that can be interpreted as the degree to which software possesses a given attribute that affects its quality“ [34, S. 3].

Um die RAG-Systeme zu evaluieren soll unter anderem der sogenannten BERTScore[35] verwendet werden. Bei BERTScore handelt es sich um eine Bewertungsmetrik für die Generierung von Sprache, auf Grundlage von vortrainierten (engl. pretrained) BERT kontextuellen Einbettungen (engl. BERT contextual embeddings)[27], [35].

Es werden kontextuelle Einbettungen eingesetzt, um die Tokens zu repräsentieren, wenn ein Referenzsatz x und ein Kandidatensatz $\hat{x}$ vorhanden sind [35]. Mithilfe der Kosinus-Ähnlichkeit wird die Übereinstimmung berechnet und mit dem inversen Dokument-Frequenz Wert optional gewichtet [35].

Die Ähnlichkeit zwischen zwei Sätzen berechnet der BERTScore als Summe der Kosinus-Ähnlichkeiten zwischen den Token Einbettungen (engl. embeddings) [35]. Der spezifische Nutzen eines Tokens in einem Satz wird von kontextuellen Einbettungen (engl. contextual embeddings) erfasst und möglicherweise werden auch Sequenzinformationen erfasst[35]. Die Tokens in den Eingabesätzen x und $\hat{x}$ werden mithilfe von kontextuellen Einbettungen (engl. contextual embeddings) dargestellt [35]. Abhängig von den umliegenden Worten, die den Kontext des Zielwortes bilden, können Kontext-Einbettungen (wie unter anderem von BERT) verschiedene Vektordarstellungen für dasselbe Wort in unterschiedlichen Sätzen generieren [35]. Um Recall zu berechnen wird jedes Token in x mit einem Token in $\hat{x}$ abgeglichen für den vollständigen Wert und zum Berechnen von Precision wird jedes Token in $\hat{x}$ mit einem Token in x abgeglichen [35]. Zum Berechnen des F1-Wertes wird Precision und Recall verknüpft[35]. Recall, Precision und F1-Wert sind für einen Referenz x und einen Kandidaten $\hat{x}$ [35]:

$$R_{BERT} = \frac{1}{|x|} \sum_{x_i \in x} \max_{\hat{x}_j \in \hat{x}} x_i^T \hat{x}_j \quad (1)$$

$$P_{BERT} = \frac{1}{|\hat{x}|} \sum_{\hat{x}_j \in \hat{x}} \max_{x_i \in x} x_i^T \hat{x}_j \quad (2)$$

$$F_{BERT} = 2 \frac{P_{BERT} \cdot R_{BERT}}{P_{BERT} + R_{BERT}} \quad (3)$$

Die Formeln [35] für (1), (2) und (3) sind aus der Arbeit zum BERTScore übernommen worden. In Abbildung 3 ist der vollständige Prozess für die Berechnung von R_{BERT} zu sehen und wird visuell dargestellt.

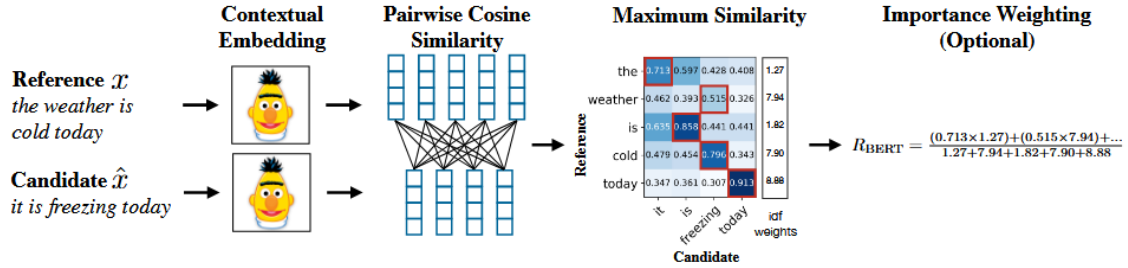


Abbildung 3: Berechnungsprozess für R_{BERT}
Quelle: [25]

Als nächstes soll ROUGE vorgestellt werden. ROUGE bedeutet ausgeschrieben „Recall-Oriented Understudy for Gisting Evaluation“ [36]. Bei ROUGE handelt es sich um ein Paket zur automatischen Evaluierung von Zusammenfassungen [36]. Verschiedene automatische Bewertungsmethoden werden von ROUGE beinhaltet, die die Ähnlichkeit zwischen Zusammenfassungen messen [36].

In der Arbeit zu ROUGE werden ROUGE-N, ROUGE-L, ROUGE-W und ROUGE-S vorgestellt [36]. Es soll an dieser Stelle nur auf ROUGE-N eingegangen werden, da für diese Arbeit nur ROUGE-N verwendet wird. Der n-Gramm Recall zwischen einer Kandidatenzusammenfassung und einer Menge an Referenzzusammenfassungen, das ist formal ROUGE-N [36].

Eine zusammenhängende Folge von n Wörtern aus einem Text wird als n-Gramm bezeichnet [37]. Beispielsweise handelt es sich im Fall von $n=1$ um einzelne Wörter und dieser Fall wird auch als Unigramme bezeichnet [37].

Die Berechnungsvorschrift für ROUGE-N lautet [36]:

$$ROUGE - N = \frac{\sum_{S \in \{ReferenceSummaries\}} \sum_{gram_n \in S} Count_{match}(gram_n)}{\sum_{S \in \{ReferenceSummaries\}} \sum_{gram_n \in S} Count(gram_n)} \quad (4)$$

Hierbei steht $Count_{match}(gram_n)$ für die maximale Anzahl an n-Grammen, die in einer Kandidatenzusammenfassung und einer Menge an Referenzzusammenfassungen gemeinsam auftreten und n steht für die Länge des n-Gramms [36].

Bei $Count(gram_n)$ handelt es sich um die in der Referenzzusammenfassung auftretende Gesamtanzahl an n-Grammen [37]. Zur Berechnung von ROUGE-N werden also auch eine Kandidatenzusammenfassung und eine Referenzzusammenfassung benötigt [36].

Für den Fall von ROUGE-1 wird also jedes Wort betrachtet und ROUGE-1 entsprechend der Formel (4) analysiert und berechnet.

Werden mehr Referenzen hinzugefügt, nimmt die Anzahl der n-Gramme im Nenner der ROUGE-N Formel zu [36]. Zusätzlich ist zu beachten, dass der Zähler über alle Referenzzusammenfassungen summiert [36]. Somit bekommen übereinstimmende n-Gramme, die in mehreren Referenzen vorkommen, effektiv mehr Gewicht [36]. Dementsprechend wird von der ROUGE-N Messung eine Kandidatenzusammenfassung bevorzugt, die Wörter enthält, die von mehreren Referenzen geteilt werden [36].

Ragas enthält verschiedene Metriken, um LLM-Anwendungen zu bewerten. Um die Leistung von LLM-Anwendungen messen zu können, bietet Ragas eine Menge an Evaluierungs-Metriken an[38]. Für unterschiedliche Anwendungen und Aufgaben sind Metriken verfügbar beispielsweise für RAG[38]. Für RAG ist unter anderem die Metrik Faithfulness verfügbar[38]. Wie faktisch konsistent eine Antwort ist, bezogen auf den abgerufenen Kontext, wird durch die Metrik Faithfulness gemessen[39]. Höhere Konsistenz wird durch einen höheren Wert angezeigt und der Wert kann von 0 bis 1 reichen[39]. Als *treu* (engl. *faithful*) wird eine Antwort angesehen, falls durch den abgerufenen Kontext all ihre Behauptungen unterstützt werden können[39]. Für die Berechnung der Metrik Faithfulness wird ein LLM benötigt, wie im Beispiel in [39] zu sehen ist.

Für die Berechnung sind drei Schritte vonnöten[39]. Im ersten Schritt werden in der Antwort alle Behauptungen identifiziert[39]. In Schritt zwei wird jede Behauptung daraufhin kontrolliert, ob sich die Behauptung von dem abgerufenen Kontext ableiten lässt[39]. Im dritten Schritt wird der Faithfulness Wert berechnet indem die folgende Formel verwendet wird[39]. Die Formel ist aus der Quelle[39] übernommen.

Faithfulness Wert =

$$\frac{\text{Anzahl an Behauptungen in der Antwort, die von dem abgerufenen Kontext unterstützt werden}}{\text{Gesamte Anzahl an Behauptungen in der Antwort}} \quad (5)$$

Eine weitere hilfreiche Metrik aus Ragas ist die Factual Correctness, die in der Rubrik Vergleich natürlicher Sprache (engl. natural language comparison) zu finden ist[38]. Die faktische Genauigkeit der generierten Antwort mit der Referenz zu vergleichen und zu bewerten, das ist unter der Metrik Factual Correctness zu verstehen [40]. Um festzustellen, in welchem Maß die generierte Antwort mit der Referenz übereinstimmt, wird diese Metrik verwendet[40]. Für Factual Correctness zeigt ein höherer Wert eine bessere Leistung an und der Wert für Factual Correctness kann im Bereich von 0 und 1 liegen [40]. Damit zwischen Antwort und Referenz die Übereinstimmung gemessen werden kann, wird die Referenz und die Antwort mithilfe des LLMs in Behauptungen bzw. Aussagen aufgeteilt und dann wird durch natural language inference, die faktische Überschneidung zwischen Antwort und Referenz bestimmt [40]. Zu bestimmen, ob die gegebene „Hypothese“ logisch aus der „Prämisse“ folgt, ist die Aufgabe von natural language inference (NLI) [41]. Kontrolliert durch einen Parameter, kann mithilfe von Precision, Recall und F1-Score die faktische Überschneidung (engl. factual overlap) bestimmt werden[40]. Für die Berechnung ist True Positive (TP) gleich der Anzahl an Behauptungen in der Antwort, die in der Referenz vorkommen[40]. Für die Berechnung der Metrik Factual Correctness wird ein LLM verwendet, wie im Beispiel in [40] zu sehen ist.

False Positive (FP) ist gleich der Anzahl an Behauptungen in der Antwort, die nicht in der Referenz vorkommen[40]. False Negative (FN) ist gleich der Anzahl an Behauptungen in der Referenz, die nicht in der Antwort vorhanden sind[40]. Um Recall, Precision und F1 Score zu berechnen, werden die folgenden Formeln verwendet [40]:

$$Precision = \frac{TP}{(TP + FP)} \quad (6)$$

$$Recall = \frac{TP}{(TP + FN)} \quad (7)$$

$$F1\ Score = \frac{2 \times Precision \times Recall}{(Precision + Recall)} \quad (8)$$

Aus Precision und Recall ist der F1-Wert bzw. der F-Wert das harmonische Mittel [42, Abs. F1-Wert], [43, S. 76].

Eine weitere Metrik von Ragas ist Context Recall. Wie viele der relevanten Dokumente oder Informationen erfolgreich abgerufen wurden, wird durch Context Recall gemessen[44]. Werden weniger relevante Dokumente ausgelassen, bedeutet das einen höheren Recall [44]. Nichts Wichtiges auszulassen ist kurz gesagt das, worum es beim Recall geht[44]. Um Context Recall zu berechnen, wird stets eine Referenz zum Vergleichen benötigt, da es darum geht nichts Wichtiges auszulassen[44].

Die Referenz wird als Vertreter zu Referenzkontext verwendet, im Fall von „LLM-based Context Recall“, was dafür sorgt, dass die Verwendung einfacher ist, da das Annotieren von Referenzkontext viel Zeit in Anspruch nehmen kann [44]. Die Referenz wird in Behauptungen aufgeteilt und jede Behauptung wird untersucht, um festzustellen, ob sie dem abgerufenen Kontext zugeordnet werden kann oder nicht zugeordnet werden kann, um Context Recall von der Referenz zu bestimmen [44]. Die Metrik Context Recall basiert auf einem LLM [44]. Die Formel ist aus der Quelle[44] übernommen.

$$\text{Context Recall} = \frac{\text{Anzahl an Behauptungen in der Referenz unterstützt durch den abgerufenen Kontext}}{\text{Gesamte Anzahl an Behauptungen in der Referenz}} \quad (9)$$

2.8 Relevante Studien

Andere Arbeiten, die Ähnlichkeiten zu den in dieser Arbeit vorgestellten Inhalten aufweisen, sollen in diesem Abschnitt vorgestellt werden.

Als erstes soll die Arbeit mit dem Titel „FloodBrain: Flood Disaster Reporting by Web-based Retrieval Augmented Generation with an LLM“[45] betrachtet werden. In der Arbeit wird eine spezielle Pipeline für die Berichterstattung über Flutkatastrophen entwickelt und vorgestellt [45]. Flutkatastrophen Berichterstattung ist im Kontext der vorgestellten Arbeit „FloodBrain“ eine Arte Situationsbericht, der für einen Appell um weitere Hilfe an größere Organisationen geht und Inhalte wie Grund, Auswirkung, Kontext und zukünftige Hilfe hervorhebt [45]. Für eine kurze Schreibweise werden diese Berichte „Flut Reporte“ in der Arbeit genannt [45]. Im Zuge der Forschung zu „FloodBrain“ hat das Forschungsteam mit Personen von anderen Einrichtungen zusammengearbeitet und diskutiert beispielsweise dem World Food Programme [45]. Im Gegensatz dazu wurde zur Erarbeitung dieser Bachelorarbeit keinen Austausch mit Personen außerhalb der Betreuer der Hochschule vorgenommen.

Mit einem Schlüsselsatz, der Datum und Ort der Katastrophe enthält, wird der Prozess einen Bericht zu generieren begonnen [45]. Um eine Websuche durchzuführen und aus dem Internet relevante Websites zu sammeln, dafür wird der Schlüsselsatz verwendet [45]. Als „Quellen“ für die Pipeline werden von jeder Website die Textdaten entnommen und gespeichert [45]. Verglichen mit der vorliegenden Bachelorarbeit wird in dieser Bachelorarbeit keine Websuche durchgeführt, um nach Informationen zu suchen. Allerdings wird im Fall des RAG-Systems mit Website-Kontext auch Textdaten aus einer Website entnommen und als zusätzlicher Kontext bzw. zusätzliche Quelle verwendet. Aber die Textdaten für das RAG mit Website-Kontext kommen aus einer lokalen HTML-Datei und nicht aus einer Websuche.

In der „FloodBrain“ Arbeit wird außerdem ein LLM angewiesen die ursprüngliche Suchanfrage auf weitere relevante Suchanfragen zu erweitern, um mehr relevante Quellen zu sammeln [45]. Um die Relevanz zu dem ursprünglichen Flutereignis zu bestimmen und falsch zurückgegebene Websites von der Google-Sucher herauszufiltern, wird jede Quelle durch ein LLM bewertet [45]. In der vorliegenden Bachelorarbeit wird die Anfrage eines Benutzers nicht durch ein LLM erweitert oder gefiltert. In „FloodBrain“ werden, um relevante Informationen zu dem Flutereignis zu entnehmen, relevante Quellen verarbeitet [45]. Auch in der vorliegenden Bachelorarbeit werden im Fall des RAG-Systems mit Website-Kontext die Quellen, also in dieser Bachelorarbeit die lokale HTML-Datei, verarbeitet und der Website-Kontext entnommen. In „FloodBrain“ wird eine LLM damit beauftragt eine Reihe an Fragen zu beantworten, um zentrale Inhalte für einen Flutbericht herauszuarbeiten und dafür erhält das LLM dafür jede Quelle als Kontext [45]. In der vorliegenden Arbeit müssen die relevanten Informationen aus den Quellen nicht durch ein LLM gefunden werden. Da in der lokalen HTML-Datei bekannt ist in welchen Attributen die relevanten Informationen sind, kann über ein Tool zum Verarbeiten von HTML-Dateien die relevanten Inhalte extrahiert werden. Anschließend wird in „FloodBrain“ ein zusammenfassender finaler Bericht erstellt [45]. In der vorliegenden Bachelorarbeit wird kein Bericht erstellt, sondern die Benutzer können dauerhaft Fragen an die RAG-Systeme stellen.

Zusätzlich kann in „FloodBrain“ die Pipeline für die Erstellung der Bericht über ein webbasiertes User Interface verwendet werden [45]. Dort wird der initiale Schlüsselsatz definiert, durch das Auswählen eines bekannten großes Ereignisses, über das eine Bereich erstellt werden soll [45]. In der vorliegenden Bachelorarbeit kann ein Benutzer auch mit den RAG-Systemen interagieren, über die Webseiten. In der vorliegenden Bachelorarbeit kann ein Benutzer frei Fragen an die RAG-Systeme stellen, wobei es sinnvoll ist die Fragen an die RAG-Systeme im Themenbereich von Mietererhöhungen durch Modernisierung zu belassen. Im Gegensatz dazu kann im Fall von „FloodBrain“ nur aus bekannten großen Ereignissen ausgewählt werden.

Damit die Ergebnisse evaluiert werden können in „FloodBrain“, wird der ROUGE Score, menschliches Urteil und LLM assistierter Vergleich (G-EVAL) eingesetzt [45]. Auch in der vorliegen Bachelorarbeit wird der ROUGE Score zur Bewertung verwendet. Menschliches Urteil wird in der vorliegenden Bachelorarbeit nicht berücksichtigt. Auch G-EVAL wird nicht eingesetzt dafür der BERTScore und Ragas.

Bei einer weiteren Arbeit, die ähnlich zu dieser Arbeit ist, handelt es sich um diese Arbeit[46]. In der Arbeit wird aufbauend auf der RAG-Architektur ein KI-System entwickelt, das Daten aus Echtzeit Web-Scraping berücksichtigt [46]. Web-Scraping kann, als absammeln von Daten aus dem Internet verstanden werden bzw. das Absammeln von Daten auf einer oder mehreren Webseiten. Auch in dieser Arbeit wird ein RAG-System verwendet mit Inhalt von einer Website. Allerdings ist die Website im Fall dieser Bachelorarbeit eine lokale HTML-Datei. In der ähnlichen Arbeit soll das System hauptsächlich als Tutor fungieren und dabei Fragen beantworten und Informationen zusammenfassen [46].

Das RAG-System in dieser Bachelorarbeit soll auch primär Fragen beantworten und soll diese Antworten auf Informationen begründen, was sehr ähnlich ist zu der in diesem Absatz vorgestellten Arbeit.

Allerdings fungiert das RAG-System in dieser Bachelorarbeit nicht primär als Tutor. Die abgerufenen HTML-Daten aus dem Web-Scraping Vorgang werden auch untersucht, in dem spezifischen HTML-Elemente identifiziert werden [46]. Auch in dieser Bachelorarbeit wird die lokale HTML untersucht, in dem die gewünschten HTML-Elemente identifiziert und ausgelesen werden. Es sind zwischen der Arbeit [46] und dieser Bachelorarbeit also einige Gemeinsamkeiten zu finden.

3 Methodik

In diesem Kapitel 3 wird die Methodik vorgestellt, die in dieser Arbeit eingesetzt wird, um eine Antwort auf die Forschungsfrage finden zu können. Die Forschungsfrage, die in Kapitel 1.1 vorgestellt wurde und an dieser Stelle noch einmal wiederholt werden soll lautet: *Sind die Antworten von einem RAG-System für Benutzer von Websites hilfreicher, wenn dem RAG-System weitere Informationen von der Weboberfläche zur Verfügung gestellt werden?*

Grundsätzlich sollen zwei RAG-Systeme erstellt werden, wobei es möglich sein soll, die beiden RAG-Systeme einzeln über eine Website zu befragen. Eines der beiden RAG-Systeme bekommt weitere Informationen von der Weboberfläche zur Verfügung gestellt, um die Frage von der Website beantworten zu können, während das andere RAG-System nur die Frage von der Website bekommt. Anhand von Metriken sollen dann die beiden RAG-Systeme miteinander verglichen werden. Diese Vorgehensweise entspricht der in Kapitel 1.2 dargestellten Vorgehensweise. Somit werden durch die Metriken quantitative Werte erhoben. Diese Metriken werden in einer möglichst kontrollierten Umgebung gemessen.

Konkret wird eine Website implementiert und gestaltet, auf der ein Nutzer per Texteingabe eine Frage an eines der beiden RAG-Systeme stellen kann. Diese Website wird mithilfe von Frameworks oder anderen Hilfsmitteln implementiert und gestaltet. Die beiden RAG-Systeme bekommen denselben Text als Eingabe, um aus dem Text relevante Inhalte zu einer Anfrage zu finden. Auch wird der Text mit denselben Argumenten eingelesen und verwendet.

Um die RAG-Systeme zu implementieren, wird ein Framework zu Hilfe genommen, um die einzelnen Komponenten für die RAG-Systeme einzubinden. Ist der Text eingelesen und durch Word-Embeddings (engl. word embeddings) in Vektoren umgewandelt, werden die Vektoren in einer Vektordatenbank gespeichert. Wird eine Frage von der Website an die RAG-Systeme gestellt, werden zu dieser Frage die relevantesten Inhalte aus der Vektordatenbank gefunden. Die Komponente in einem RAG-Systeme, die die relevantesten Inhalte zu einer Anfrage findet, ist der Retriever. Es wird immer die gleiche Anzahl an relevanten Informationen zu einer Frage in den beiden RAG-Systemen von dem Retriever gefunden und zurückgegeben.

Im Fall des RAG-Systems ohne Website-Kontext wird die Frage und die gefundenen relevanten Inhalte zu der Frage zusammen in Textform an ein LLM übergeben, um mithilfe der relevanten Informationen eine Antwort auf die Frage zu generieren.

Diese Antwort wird dann auf der Website für den Nutzer angezeigt. Im Fall des RAG-Systems mit Website-Kontext werden auch die relevantesten Inhalte zu einer Frage gefunden und abgerufen. Zusätzlich werden von der Website Informationen in Textform gefunden und weiterverwendet. Bei den Informationen in Textform auf der Website handelt es sich um den bereits oft erwähnten Website-Kontext, der verwendet werden soll. Die Frage, die abgerufenen und relevanten Informationen zu der Frage und die Informationen von der Website werden zusammen in Textform an das LLM übergeben, damit das LLM mithilfe all dieser Inhalte eine Antwort auf die Frage generieren kann. Auch in diesem Fall wird anschließend die generierte Antwort auf der Website für den Nutzer angezeigt.

Diese beiden RAG-Systeme werden dann mithilfe von Metriken bewertet bzw. evaluiert. Pro RAG-System wird eine Metrik fünf Mal gemessen, um eine größere Basis an Messwerten pro Metrik zu haben und vielleicht auftretende Ausreißer auszugleichen. Es werden mehrere verschiedene Metriken eingesetzt, um eine große Menge an Werte für den Vergleich der RAG-Systeme zu erhalten.

4 Durchführung

Für die technische Umsetzung dieser Arbeit wurde Python 3.12.3 verwendet. Python eignet sich als Programmiersprache für diese Arbeit, da für Python bereits viele verschiedene Bibliotheken und Frameworks vorhanden sind, die hilfreiche Funktionalitäten für diese Arbeit bieten. Die für diese Arbeit benötigte Web-Anwendung wurde mithilfe von Flask[47] erstellt. Bei Flask handelt es sich um ein Web-Anwendungs Framework, das leichtgewichtig ist [47]. Flask wurde für diese Arbeit eingesetzt, weil mithilfe von Flask schnell Web-Anwendungen gebaut werden können und diese auch zu komplexeren Anwendungen erweitert werden können [47]. Die einzelnen Webseiten wurden mit HTML und Bootstrap v5.0.2[48] gestaltet. Bei Bootstrap handelt es sich um eines der populärsten Frameworks für die Frontend Entwicklung [49]. Aus diesem Grund und dem vergleichsweise unkomplizierten Umgang[49] mit dem Framework wurde Bootstrap ausgewählt. Einige Komponenten in Bootstrap benötigen die Bibliothek Popper, um korrekt positioniert zu sein [50]. Die Bibliothek Popper wird durch ein Bundle von Bootstrap integriert und verwendet [48].

Um die RAG-Systeme zu bauen, wurde das Framework LlamaIndex[51] zu Hilfe genommen. LlamaIndex wurde verwendet, weil LlamaIndex die Möglichkeit bietet die verschiedensten Komponenten in LlamaIndex zu integrieren. In den RAG-Systemen für diese Arbeit wird ein LLM eingesetzt. Als LLM für diese Arbeit wird das Modell llama3.2:3b[52] verwendet.

Dieses Modell wurde ausgewählt, weil der verwendete Text für die Suche in den RAG-Systemen auf Deutsch ist und dementsprechend sollte auch das verwendete LLM Deutsch offiziell unterstützen, um aus den Suchergebnissen auf Deutsch sinnvolle Antworten auf Deutsch generieren zu können. Im Fall des verwendeten Modells llama3.2:3b wird Deutsch offiziell unterstützt [52].

Ein weiterer Grund, warum dieses Modell ausgewählt wurde, ist, dass dieses Modell zu den kleineren Modellen gezählt werden kann [52], und somit auch auf Hardware laufen kann, die weniger leistungsfähig ist. Als Einstellung wurde für dieses LLM die Temperatur sehr gering auf 0,0 eingestellt, damit die generierten Antworten besser reproduzierbar sind und bei den geplanten Metriken die Werte weniger stark variieren. Um das LLM verwenden zu können, wurde Ollama[53] verwendet. Bei Ollama handelt es sich um ein Open-Source-Framework, das es ermöglicht LLMs lokal zu verwenden [54]. Bei der Durchführung wurde die Ollama Version 0.13.3 verwendet. Der Text, der von den beiden RAG-Systemen für das Suchen nach Informationen genutzt wird, ist diese Dissertation[55]. Dabei geht es in der Dissertation inhaltlich grob um die Modernisierungen von Mietwohnungen und damit möglichen Mieterhöhungen [55]. In diesen Themenbereich aus Modernisierungen von Mietwohnungen und damit möglichen Mieterhöhungen sind die RAG-Systeme und die Website thematisch eingebunden.

Um die Dissertation in den RAG-Systemen verwenden zu können, wird die Dissertation erst eingelesen und dann durch den SentenceSplitter[56] in kleinere Textfragmente zerteilt. Unter Berücksichtigung der Grenzen von Sätzen versucht der SentenceSplitter Texte zu zerteilen [57, Abs. SentenceSplitter]. Also den Text an den Grenzen von Sätzen zu zerteilen. Die Parameter mit denen der SentenceSplitter verwendet wird, um die Dissertation für die RAG-Systeme zu verwenden, sind in den beiden RAG-Systemen exakt gleich.

Die Website für diese Arbeit besteht aus insgesamt vier Webseiten. Dabei handelt es sich einmal um eine Startseite, die den Benutzer begrüßt und den Benutzer zur nächsten Webseite weiterleitet, der Webseite mit der Auswahlmöglichkeit für die RAG-Systeme. Auf dieser Webseite kann dann ein Benutzer per Button auswählen welches RAG-System der Benutzer verwenden möchte, das RAG-System mit Website-Kontext oder das RAG-System ohne Website-Kontext. Zusätzlich steht zu jedem RAG-System eine kurze Beschreibung auf der Webseite für die Auswahlmöglichkeit der RAG-Systeme. Die beiden verbleibenden Webseiten sind dann die Webseiten mit der Möglichkeit Fragen an das jeweilige RAG-System zu stellen. Die beiden Webseiten für die RAG-Systeme werden durch dasselbe HTML-Dokument dargestellt. Nur bei der Verarbeitung einer Frage unterscheiden sich die Webseiten. Bei einer Frage an das RAG-System ohne Website-Kontext bekommt das RAG-System nur die Frage von der Webseite und generiert dazu eine Antwort. Bei dem RAG-System mit Website-Kontext bekommt das RAG-System die Frage und den Website-Kontext von der Webseite und generiert damit eine Antwort auf die Frage.

Wie bereits erwähnt ist die für diese Arbeit angelegte Website in einen theoretischen Anwendungsfall eingebettet. In diesem theoretischen Anwendungsfall können Personen auf der Website Fragen rund um das Thema Modernisierungen von Mietwohnungen an die RAG-Systeme stellen. Dabei werden die Fragen in Textform in ein Eingabefenster eingegeben und nach dem Abschicken der Frage, generiert das ausgewählte RAG-Systeme eine Antwort. Die Antwort des jeweiligen RAG-Systems wird dann unter der Frage auf der jeweiligen Webseite angezeigt.

Die Personen, die in diesem theoretischen Anwendungsfall die Website besuchen und verwenden sind dabei vielfältig von Mietern über Vermieter zu Sachverständigen, Juristen oder Investoren. Auf der Website sind neben dem Eingabefenster für die Frage noch 50 weitere kleinere Eingabefenster zu finden, die jeweils eine Beschriftung haben. Die Beschriftungen stehen dabei stets über den kleinen Eingabefenstern und die Beschriftungen sind meist nur wenige Worte lang. Die kleinen Eingabefenster sollen den theoretischen Benutzern die Möglichkeit geben zu einer Frage zusätzliche Eingaben zu machen. Beispielsweise lautet die erste Beschriftung „Temperatur Wände“ und ist bewusst offengehalten. Offengehalten sind diese 50 Beschriftungen, weil es zu jeder Beschriftung einen Hinweistext gibt.

Diese Hinweistexte sind in eingeklappter Form neben jeder Beschriftung zu finden, in Form von einem grauen Quadrat mit abgerundeten Ecken und einem weißen Fragezeichensymbol darin. Wird der Mauszeiger über ein Quadrat mit Fragezeichensymbol gehalten, ist ein Text (der Hinweistext) links neben dem Quadrat mit Fragezeichensymbol zu sehen. Dieser Hinweistext enthält Informationen zu der offengehaltenen Beschriftung. Wird kein Mauszeiger über das Quadrat mit Fragezeichensymbol gehalten, ist kein Hinweistext zu sehen. Die Beschriftung von oben „Temperatur Wände“ hat einen Hinweistext in dem steht „Die Oberflächentemperatur der raumumschließenden Wände sollte mindestens 16° C - 18° C betragen“ und ist aus der Dissertation[55] übernommen, die von den RAG-Systeme zum Suchen nach Informationen genutzt wird. Mithilfe von diesem Hinweistext bekommt ein Benutzer etwas Kontext zu der offengehaltenen Beschriftung und weiß nun genauer, was mit „Temperatur Wände“ gemeint ist und kann nun leichter eine Eingabe machen, beispielsweise die gemessene Oberflächentemperatur einer raumumschließenden Wand eingeben. Wie bereits erwähnt gibt von diesen Beschriftungen und dazugehörigen Hinweistexten 50 Stück auf der Webseite für die RAG-Systeme. Der Inhalt aller Hinweistexte zusammen ist der Website-Kontext, der dem RAG-System mit Website-Kontext zusätzlich zugänglich gemacht wird bei einer Frage von der Website. Es wurden gerade diese Inhalte als Website-Kontext verwendet, weil nur in diesen Texten relevante Informationen enthalten sind, die für das RAG-System zusätzliche Informationen bietet, um eine Frage von der Website besser beantworten zu können. Beispielsweise wäre zusätzlicher HTML-Code für das RAG-System in der Regel nicht hilfreich, weil in dem HTML-Code hauptsächlich Attribut-Namen oder Tag-Namen enthalten sind. HTML-Code würde dem RAG-System nur möglicherweise helfen, wenn explizite nach HTML-Code durch eine Frage gefragt wird, aber dieser Fall ist in dem theoretischen Anwendungsfall für die Website weiter oben nicht vorgesehen. Allerdings werden die Eingaben in den 50 verschiedenen kleinen Eingabefenstern nicht weiterverarbeitet.

Die 50 verschiedenen Beschriftung sind dabei unabhängig voneinander und bauen nicht aufeinander auf. Auch die 50 Hinweistexte sind unterschiedlich und bauen nicht aufeinander auf. Auf die Inhalte der Hinweistexte wird weiter unter noch genauer eingegangen.

In Abbildung 4 ist ein Screenshot der Webseite für das RAG mit Website-Kontext zu sehen. Es sind zwei der 50 kleineren Eingabefenster zu sehen und über diesen kleinen Eingabefenstern sind die offengehaltenen Beschriftungen zu sehen. Über dem oberen kleinen Eingabefenster steht die Beschriftung „Obergrenze für Mieten“ und über dem unteren kleinen Eingabefenster steht „Berechtigte Erhöhung nach einer energetischen Modernisierung“. Außerdem ist das größere Eingabefenster zu sehen, in dem Benutzer Fragen an die RAG-Systeme stellen können, in diesem Fall also an das RAG-System mit Website-Kontext. Über dem großen Eingabefenster steht die Beschriftung „Deine Frage“ und in dem großen Eingabefenster steht in einem blassen grau „Schreibe hier deine Frage“. Unter dem großen Eingabefenster ist ein blauer Button zu sehen, auf dem in weiß „Abschicken“ steht. Wird der Abschicken-Button gedrückt versucht das ausgewählte RAG-System, in diesem Fall das RAG-System mit Website-Kontext, eine Antwort auf die Frage des Benutzers aus dem großen Eingabefenster zu generieren. Über den kleineren Eingabefenstern sind links kleinere dunkelgrau Quadrate mit abgerundeten Ecken und weißem Fragezeichensymbol darin zu sehen. Diese Quadrate zeigen den jeweiligen Hinweistext an, wenn der Mauszeiger über das jeweilige Quadrat gehalten wird.



The screenshot displays a web interface for a Retrieval-Augmented Generation (RAG) system. It features three input fields and a submission button. The first two are smaller, with labels 'Obergrenze für Mieten' and 'Berechtigte Erhöhung nach einer energetischen Modernisierung' above them. The third is a larger field labeled 'Deine Frage:' with a placeholder 'Schreibe hier deine Frage...'. A blue button labeled 'Abschicken' is positioned below the larger field.

Abbildung 4: Screenshot von einem Teil der Webseite für das RAG mit Website-Kontext
Quelle: Eigene Darstellung

In Abbildung 5 sind zwei kleine Eingabefenster zu sehen, die jeweils eine Beschriftung haben. Die Beschriftung des oberen kleinen Eingabefensters ist „Temperatur Wände“. Das untere kleine Eingabefenster hat die Beschriftung „Wärme Außenbauteil“. Neben der Beschriftung des oberen kleinen Eingabefensters ist der Hinweistext ausgeklappt. Der Hinweistext wird in weißer Schrift auf schwarzem Hintergrund angezeigt. Der Inhalt des Hinweistextes lautet „Die Oberflächentemperatur der raumumschließenden Wände sollte mindestens 16° C – 18°C betragen“. Somit ist, wie bereits weiter oben beschrieben,

weiterer Inhalt und Kontext zu der Beschriftung durch den Hinweistext für den Benutzer hinzugekommen. Auf der Grundlage des Inhalts aus dem Hinweistext kann ein Benutzer nun leichter eine sinnvolle Eingabe in das obere kleine Eingabefenster eintragen.

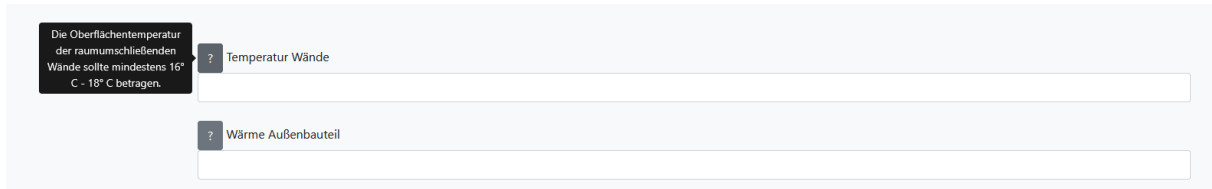


Abbildung 5: Ausgeklappter Hinweistext zu einer Beschriftung
Quelle: Eigene Darstellung

Technisch handelt es sich bei den Hinweistexten, um sogenannte Tooltips [50]. Diese Tooltips können in Form von HTML-Button-Tags genutzt werden [50]. Der Inhalt für die Hinweistext steht als String in dem title-Attribut des Button-Tags. Zusätzlich haben die Buttons, die als Tooltips fungieren, ein id-Attribute, das gleich dem String „tooltip“ gesetzt wird. Wie bereits erwähnt ist der Inhalt aller Tooltips zusammen der Website-Kontext, der dem RAG-System mit Website-Kontext zur Verfügung gestellt wird. Um nun den Textinhalt aller Tooltips zusammen für das RAG-System mit Website-Kontext bereitstellen zu können, müssen die relevanten Inhalte erst gefunden werden. Mithilfe der Bibliothek BeautifulSoup 4[58] können die Hinweistexte aus den Tooltip-Button-Tags im HTML-Code gefunden und selektiert werden. Für diese Arbeit wird der bereits in Python integrierte HTML-Parser innerhalb von BeautifulSoup 4 „html.parser“ [58, Abs. Specifying the parser to use] verwendet. Sind alle Tooltips anhand des id-Attributs gefunden und die Hinweistexte aus dem title-Attribut gelesen, werden die Hinweistexte als Strings verknüpft und dem RAG-System mit Website-Kontext bei einer Frage von der Website zur Verfügung gestellt.

Die Textfragmente, die aus der zuvor erwähnte Dissertation und dem SentenceSplitter entstanden sind, werden durch dieses Sentence-Transformer Modell[59] in Vektoren umgewandelt. Die Sätze werden in einen Vektorraum überführt, der 384 dimensional ist, durch das Sentence-Transformer Modell [59]. Es wurde gerade dieses Modell gewählt, da auch deutsch unter dem multilingual Aspekt dieses Modells zu finden ist und dieses Modell für die semantische Suche eingesetzt werden kann[59]. Für das Speichern der Vektoren und spätere Suchen nach relevanten Daten zu einer Frage wird Chroma[60] verwendet. Bei Chroma handelt es sich um eine Vektordatenbank, die zusätzlich Open-Source ist [60]. Dabei kann Chroma so eingesetzt werden, dass die Datenbank lokal gespeichert wird [61]. Das lokale Speichern von Chroma wird durch den Persistent Client realisiert [61]. Innerhalb von Chroma wird in dieser Arbeit die Kosinus-Ähnlichkeit und HNSW verwendet [62]. Der Retriever, der zu einer Frage die relevantesten

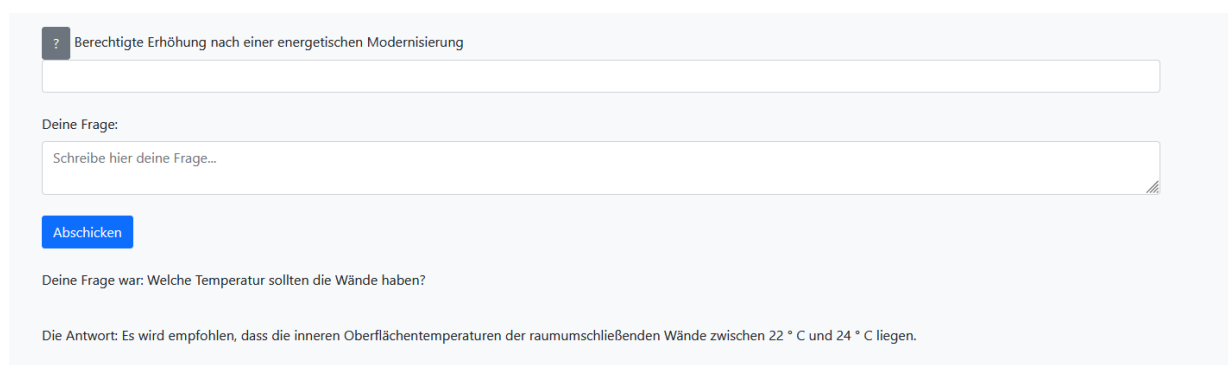
Inhalte findet, basiert in diesem Fall auf der Chroma Vektordatenbank und gibt stets die fünf relevantesten Inhalte zu einer Frage zurück. Damit sind alle Komponenten für die RAG-Systeme vorhanden und können mithilfe von LlamaIndex zusammen eingesetzt werden.

Nun soll einmal die Verwendung des RAG-Systems ohne Website-Kontext in Kombination mit der Website kurz dargestellt werden. Nachdem das RAG-System ohne Website-Kontext ausgewählt wurde und eine Frage über die Website an das RAG-System ohne Website-Kontext gestellt wurde, werden durch den Retriever die fünf ähnlichsten Inhalte zu der Frage gefunden und zurückgegeben. Mit einer Anweisung in Textform an das LLM, wie die Ausgabe formal aussehen soll, werden die fünf ähnlichsten Inhalte zu der Frage mit der ursprünglichen Frage zusammen in Textform an das LLM übergeben. Aus dieser Eingabe generiert das LLM dann eine Antwort auf die Frage. Die Anweisung an das LLM beinhaltet, dass die Antwort aus einem oder wenig zusammenhängenden Sätzen bestehen soll, Aufzählungen mit Stickpunkten und Listen vermieden werden sollen und das zur Beantwortung möglichst nur Informationen aus den abgerufenen Inhalten verwendet werden soll. Die Antwort soll aus einem oder wenigen zusammenhängenden Sätzen bestehen, denn auch die Referenzantworten bestehen nur aus einem oder aus zwei Sätzen. Die Referenzantworten werden weiter unten in diesem Kapitel noch genauer vorgestellt. Auch Aufzählungen mit Stickpunkten und Listen sollen vermieden werden, da die Referenzantworten keine Aufzählungen mit Stickpunkten oder Listen beinhalten. Zuletzt soll die Antwort hauptsächlich aus den abgerufenen Informationen bestehen, damit das LLM sich möglichst stark an den abgerufenen Informationen orientiert und diese auch verwendet. Zur Übergabe der Anweisung an das LLM werden die Teilanweisungen zu einem einzigen langen Satzkonstrukt, das aus den gesamten Teilanweisungen besteht, zusammengesetzt. Die generierte Antwort von dem LLM ist dann die Antwort durch das RAG-System ohne Website-Kontext auf die ursprüngliche Frage und wird auf der Website für den Benutzer angezeigt. Die generierte Antwort wird unter dem Eingabefenster, in dem der Benutzer auch die Frage an das RAG-System gestellt hat, angezeigt.

Ebenfalls soll für das RAG-System mit Website-Kontext einmal die Benutzung in Kombination mit der Website illustriert werden. Wählt der Benutzer das RAG-System mit Website-Kontext aus und stellt eine Frage über die Website an das RAG-System mit Website-Kontext, wird die lokale HTML-Datei für die Website, auf der auch die Frage gestellt wurde, als Datei geöffnet. Aus der geöffneten Datei werden dann durch BeautifulSoup 4 die Tooltips und deren Hinweistexte gefunden und als Strings zurückgegeben. Auch in diesem Fall werden durch den Retriever die fünf ähnlichsten Inhalte zu der eingegebenen Frage gefunden und zurückgegeben. Mit einer Anweisung in Textform an das LLM (wie die Ausgabe formal aussehen soll) werden die gefundenen ähnlichsten Inhalte zu der Frage, die ursprüngliche Frage und die Hinweistexte aus der HTML-Datei zusammen an das LLM in Textform übergeben. Daraus generiert das LLM eine Antwort auf die Frage. Die Anweisung an das LLM ist dieselbe Anweisung, die auch zuvor bei dem RAG ohne Website-Kontext beschrieben wurden. Allerdings

kommt bei dieser Anweisung noch etwas hinzu, nämlich soll das LLM zur Beantwortung der Frage auch möglichst die Informationen aus dem Website-Kontext für die Beantwortung der Frage verwenden. Ansonsten ist die Anweisung genau dieselbe Anweisung, die schon zuvor in dem RAG-System ohne Website-Kontext beschrieben wurde. Nach wie vor ist die gesamte Anweisung ein einziger langer Text bzw. mehrere aneinander gehängte Sätze ohne Zeilenumbrüche. Die generierte Antwort von dem LLM ist dann die Antwort des RAG-Systems mit Website-Kontext auf die Frage des Benutzers und wird dem Benutzer auf der Website als Antwort angezeigt, unter dem Eingabefenster, in dem der Benutzer zu Beginn die Frage gestellt hat.

In Abbildung 6 ist die Antwort des RAG-Systems ohne Website-Kontext zu sehen. Die generierte Antwort des RAG-Systems auf die Frage „Welche Temperatur sollten die Wände haben?“ ist „Es wird empfohlen, dass die inneren Oberflächentemperaturen der raumumschließenden Wände zwischen 22 ° C und 24 ° C liegen“. Diese Antwort stützt sich auf die abgerufenen Inhalte aus der Dissertation, die von den RAG-Systemen für die Suche nach Informationen verwendet wird. In Abbildung 7 ist die Antwort des RAG-Systems mit Website-Kontext auf die Frage „Welche Temperatur sollten die Wände haben?“ zu sehen. Die gestellte Frage an die beiden RAG-Systeme ist also gleich in diesen Fällen. Die Antwort des RAG-Systems in Abbildung 7 lautet „Die Oberflächentemperatur der raumumschließenden Wände sollte mindestens 16° C – 18° C betragen, um eine angenehme und gesunde Umgebung für den Menschen zu schaffen“. Diese Antwort stützt sich auch auf die abgerufenen Inhalte aus der Dissertation, aber auch auf die Inhalte aus dem Website-Kontext. Die Antworten der beiden RAG-Systeme sind also unterschiedlich in diesem Fall, was anhand der Antworten in den Abbildung 6 und Abbildung 7 zu sehen ist bzw. überprüft werden kann.



The screenshot shows a web interface with a light gray background. At the top, there is a header bar with a question mark icon and the text "Berechtigte Erhöhung nach einer energetischen Modernisierung". Below this is a large, empty rectangular input field. Underneath the input field, the text "Deine Frage:" is displayed. Below that is another rectangular input field containing the placeholder text "Schreibe hier deine Frage...". To the left of this field is a blue button with the white text "Abschicken". Below the input field, the text "Deine Frage war: Welche Temperatur sollten die Wände haben?" is displayed. At the bottom, the text "Die Antwort: Es wird empfohlen, dass die inneren Oberflächentemperaturen der raumumschließenden Wände zwischen 22 ° C und 24 ° C liegen." is displayed.

Abbildung 6: Antwort des RAG-Systems ohne Website-Kontext
Quelle: Eigene Darstellung

? Berechtigte Erhöhung nach einer energetischen Modernisierung

Deine Frage:

Schreibe hier deine Frage...

Abschicken

Deine Frage war: Welche Temperatur sollten die Wände haben?

Die Antwort: Die Oberflächentemperatur der raumschließenden Wände sollte mindestens 16° C - 18° C betragen, um eine angenehme und gesunde Umgebung für den Menschen zu schaffen.

Abbildung 7: Antwort des RAG-Systems mit Website-Kontext
Quelle: Eigene Darstellung

Wie bereits in Kapitel 3 Methodik erwähnt, werden die RAG-Systeme durch Metriken evaluiert. Zu den verwendeten Metriken zählen drei Metriken aus dem BERTScore und vier Metriken aus Ragas[63]. Bei Ragas handelt es sich um eine Bibliothek, die zum Evaluieren von LLM-Anwendungen genutzt werden kann [63]. Ragas wurde verwendet, da Ragas viele verschiedene Metriken zum Evaluieren von unterschiedlichen Anwendungsfällen anbietet und auch für den konkreten Anwendungsfall RAG sind eine Vielzahl an verschiedenen Metriken verwendbar [64]. Der BERTScore wurde genutzt, um durch eine wissenschaftlich klar beschriebene Metrik weitere Ergebnisse zu erhalten und eine Metrik zu haben, die unabhängig von Ragas ist. Die konkreten verwendeten Metriken durch den BERTScore sind Recall R_{BERT} , Precision P_{BERT} und der F1Score F_{BERT} , die auch in Kapitel 2.7 beschrieben wurden. Für die Verwendung durch den BERTScore wurde das Modell BERT multilingual base model (cased)[65] verwendet. Es wurde dieses Modell gewählt, weil dieses Modell zwischen groß- und kleingeschriebenen Worten unterscheidet [65], was in der deutschen Sprache eine wichtige Unterscheidung ist. Ein weiterer Grund, warum das Modell gewählt wurde ist, dass dieses Modell multilingual ist und auf 104 Sprachen vortrainiert (engl. pre-trained) wurde [65]. In diesem multilingual Aspekt ist auch Deutsch zu finden [65], was sehr wichtig für diese Arbeit ist, da zu erwarten ist, dass die Antworten der RAG-Systeme auf Deutsch sind und auch die Referenzantworten, die für den BERTScore benötigt werden, sind auf Deutsch. Auf die Referenzantworten wird weiter unten im Text noch intensiver eingegangen.

Bei den mithilfe von Ragas verwendeten Metriken handelt es sich um die folgenden Metriken: Context Recall, Faithfulness, ROUGE Score und Factual Correctness. Diese Metriken wurden in Kapitel 2.7 bereits vorgestellt und erläutert. Auch der ROUGE Score kann mithilfe von Ragas genutzt werden [66] und der ROUGE Score wird in dieser Arbeit auch durch Ragas eingesetzt. Bei der Verwendung von ROUGE wird ROUGE-1, eine Variante von ROUGE-N verwendet. Bei ROUGE-1 werden 1-Gramme oder Unigramme betrachtet. Das bedeutet das bei der Berechnung von ROUGE-1 jedes Wort in der Kandidatenzusammenfassung und der Referenzzusammenfassung betrachtet wird und damit ROUGE-1 berechnet wird entsprechend der Formel, die in Kapitel 2.7 für ROUGE-N angegeben wurde. Darüber hinaus wurde bei ROUGE-1 stets der F1-Score berechnet und betrachtet.

Damit diese Ragas-Metriken verwendet werden können, müssen verschiedene Fragen, abgerufene Inhalte der RAG-Systeme, Antworten der RAG-Systeme und Referenzantworten bereitgestellt werden. Für die Berechnung des BERTScores wird ein Referenzsatz und ein Kandidatensatz benötigt. Daher wurden 50 Fragen und 50 Referenzantworten erarbeitet. Diese 50 Fragen und Antworten wurden selbst-erstellt auf Basis der Dissertation, die bereits oben erwähnt wurde und die auch von den RAG-Systemen als Text für die Such nach relevanten Informationen genutzt wird. Dafür wurden zuerst 50 sachliche Aussagen aus der Dissertation ausgewählt. Die 50 sachlichen Aussagen haben in den meisten Fällen eine Länge von einem Satz, können aber aus mehreren Teilsätzen bestehen. Die Länge von einem Satz liegt streng betrachtet bei 39 Aussagen vor. In wenigen Aussagen erstrecken sich die sachlichen Aussagen über zwei Sätze, streng betrachtet in nur vier Fällen. Als Definition wird der Punkt als Ende eines Satzes betrachtet. Im Fall von einem Doppelpunkt, muss nach dem Doppelpunkt ein vollständiger Satz stehen, damit der Satz vor dem Doppelpunkt und der Satz nach dem Doppelpunkt als zwei Sätze gelten. Steht nach einem Doppelpunkt kein vollständiger Satz, gilt nur der Satz vor dem Doppelpunkt als ein Satz. In sieben Fällen ist die sachliche Aussage auch streng genommen kein vollständiger Satz, da hier nicht die Bestandteile Subjekt und Prädikat vorhanden sind. In den sieben Fällen werden durch Zeichen, wie das Gleichheitszeichen, die Relation in den Satzartigen Strukturen dargestellt. Zu diesen 50 Aussagen wurden dann 50 passende Fragen formuliert. Die 50 sachlichen Aussagen wurden möglichst originalgetreu aus der Dissertation übernommen und nicht verändert. Nur in Ausnahmefällen wurden wenige Aussagen grammatikalisch leicht an die Fragen angepasst. Dadurch sind 50 Fragen und 50 dazu passend Antwort auf Basis der Dissertation entstanden. Es sind also 50 Frage-Antwort-Paare entstanden. In Tabelle 1 sind die ersten fünf der zusammenpassenden Frage-Antworte-Paare zu sehen. Bei Antwort 5 ist auch eine Aussage zu sehen, die streng genommen kein ganzer Satz ist, aber durch die Symbole werden die einzelnen Bestandteile zu einer aussagekräftigen Einheit zusammengefügt. Die Antworten in den Frage-Antwort-Paaren und somit auch die Antworten in der Tabelle 1 sind, wie bereits erwähnt, aus der angesprochenen Dissertation übernommen worden.

Tabelle 1: Fünf Frage-Antwort-Paare
Quelle: Eigene Darstellung

Auszug aus dem Fragen-Antwort Set		
Nummer	Frage	Antwort
1	"Welche Temperatur sollten die Wände mindestens haben?"	"Die Oberflächentemperatur der raumumschließenden Wände sollte mindestens 16° C - 18° C betragen."
2	"Wie kann das Gefühl von Zug verhindert werden?"	"Durch die im Außenbauteil gespeicherte Wärme wird das Gefühl von Zug verhindert."
3	"Wozu dienen energetische Modernisierungen?"	"Energetische Modernisierungen dienen der nachhaltigen Einsparung von Endenergie."
4	"Wie viel besser leitet Feuchtigkeit im Bauteil Wärme weiter als Luft?"	"Im Bauteil enthaltene Feuchtigkeit leitet Wärme in Abhängigkeit ihrer Temperatur 24 - 27 Mal schneller weiter als Luft."
5	"Ab wann wird von Treibhausklima in einem Raum gesprochen?"	"Relativen Luftfeuchte in einem Raum > 60 % = Treibhausklima."

Für die Verwendung wurden alle 50 Fragen in eine Fragen-Liste eingefügt und auch alle 50 Antworten wurden in eine Antworten-Liste eingefügt. Werden die Fragen mit dem Retriever gesucht, werden die fünf ähnlichsten Inhalte zu der Frage gefunden und abgerufenen. Das sind dann die abgerufenen Inhalte des RAG-Systems. Dann kann mit der Frage und den dazu passenden abgerufenen Inhalten durch das LLM die Antworten des RAG-Systems ohne Website-Kontext generiert werden. Für das RAG-System mit Website-Kontext müssen dann noch aus dem Website-Kontext die Inhalte aus den Tooltips gefunden werden. Mit der Frage, den abgerufenen Inhalten zu der Frage und dem Website-Kontext kann dann durch das LLM die Antwort des RAG-Systems mit Website-Kontext auf die Frage generiert werden. Somit sind alle benötigten Inhalte für die Berechnungen der Metriken durch Ragas und den BERTScore vorhanden. Um den BERTScore zu bestimmen werden die generierten Antworten des jeweiligen RAG-Systems als Kandidatensätze genutzt und als Referenzantworten werden die Antworten aus der Antworten-Liste verwendet.

Damit der ROUGE Score aus Ragas einen Wert bestimmen kann, nutzt die Metrik eine Antwort und eine Referenz[66]. Als Antwort wird die generierte Antwort des jeweiligen RAG-Systems verwendet und als Referenz werden die Antworten aus der Antworten-Liste genutzt. Damit Context Recall aus Ragas bestimmt werden kann, wird eine Referenz und Referenzkontext genutzt[44]. Als Referenz werden wie zuvor die Antworten aus der Antworten-Liste eingesetzt und der Referenzkontext sind die abgerufenen Inhalte von dem Retriever der RAG-Systeme. Damit Faithfulness aus Ragas gemessen werden kann, wird eine Antwort und abgerufenen Kontext benötigt[39]. Als Antwort wird die generierte Antwort des jeweiligen RAG-Systems genutzt und als abgerufenen Kontext wird der abgerufene Kontext von dem Retriever des jeweiligen RAG-Systems eingesetzt. Für Factual Correctness aus Ragas wird eine Antwort und eine Referenz verwendet, um die Factual Correctness zu bestimmen [40]. Als Antwort wird die generierte Antwort des jeweiligen RAG-Systems eingesetzt und als Referenz werden die Antworten aus der Antworten-Liste verwendet.

Damit Faithfulness, Context Recall und Factual Correctness gemessen werden können, wird ein LLM benötigt wie in den Beispielen in [39], [44], [40] zu sehen ist. Das hierfür verwendete LLM llama3.2:3b ist dasselbe Modell, das auch in den RAG-Systemen eingesetzt wird. Wie auch in den RAG-Systemen ist die Temperatur für das LLM auf null gesetzt, damit die Ausgaben des LLMs möglichst wenig variieren und die Ergebnisse somit möglichst reproduzierbar sind.

Für die Evaluierung der RAG-Systeme wird für jede Metrik fünf Durchgänge pro RAG-System durchgeführt. Im Fall von BERTScore werden allerdings nicht 50 Einzelwerte pro Metriken und pro Durchgang zur Überprüfung der Leitfrage herangezogen, sondern es wird der Durchschnitt pro Metrik und Durchgang berechnet. Wie der Durchschnitt für den BERTScore genutzt werden kann ist hier beschrieben [67]. Das arithmetische Mittel kann auch als Durchschnitt bezeichnet werden[68]. Auch bei Ragas werden nicht 50 Einzelwerte pro Metrik und pro Metrik betrachtet, sondern es wird ein Wert pro Metrik und pro Durchgang betrachtet. Anhand des in [69] gezeigten DataFrames kann nachvollzogen werden, dass auch im Fall von Ragas das arithmetische Mittel für die Metriken berechnet wird.

Um einen maximalen Unterschied bei den Metriken zwischen dem RAG-System mit Website-Kontext und dem RAG-System ohne Website-Kontext feststellen zu können, wurden als Inhalte für die Tooltips im HTML-Dokument die 50 Referenzantworten eingefügt. Die bewusst offen formulierten Beschriftungen zu den 50 Eingabefeldern wurden passend zu den Inhalten aus den jeweiligen Hinweistext formuliert. Somit können nun die Metriken für die Evaluation gemessen werden. Um die Ergebnisse der Evaluationen speichern zu können wird die Open-Source Bibliothek Pandas[70] genutzt. Mithilfe von Pandas werden die Ergebnisse als CSV-Dateien gespeichert. Der Name, unter dem die jeweilige Ergebnis-Datei gespeichert werden soll, kann selbst eingegeben werden. Die gespeicherten Ergebnisse der Metriken werden im Folgenden Kapitel 5 dargestellt.

Zum Abschluss des Kapitels Durchführung noch einige Daten zur verwendeten Hardware während der Durchführung. Der Arbeitsspeicher umfasste 16 GB und die verwendete Graphikkarte war eine NVIDIA GeForce GTX 1660 SUPER.

5 Ergebnisse

In diesem Kapitel 5 Ergebnisse sollen die Ergebnisse der Metriken für die beiden RAG-Systeme dargestellt werden.

Zuerst sollen die BERTScore-Metriken betrachtet werden. In Tabelle 2 sind die Zahlenwerte der BERTScore-Metriken für das RAG-System ohne Website-Kontext zu sehen. Bei den einzelnen Zahlenwerten in der Tabelle 2 handelt es sich immer um das arithmetische Mittel bzw. den Durchschnitt. Das arithmetische Mittel wurden aus den einzelnen Zahlenwerten durch die 50 Frage-Antwort Paare bestimmt. Wie in Kapitel 3 Methodik und in Kapitel 4 Durchführung beschrieben wurden jeweils fünf Durchläufe pro Metriken durchgeführt. Die einzelnen Durchläufe sind in Tabelle 2 als Spalten dargestellt. In den Zeilen der Tabelle sind die verwendeten Metriken eingetragen. Es handelt sich um die Metriken Precision (P_{BERT}), Recall (R_{BERT}) und F1Score (F_{BERT}) aus dem BERTScore. In der Tabelle 2 ist die Precision als BERT_Precision eingetragen, der Recall ist als BERT_Recall eingetragen und der F1Score ist als BERT_F1Score eingetragen. Dabei sind BERT_Precision, BERT_Recall und BERT_F1Score in dieser Reihenfolge von oben nach unten aufgelistet in Tabelle 2 eingetragen. Für die jeweilige Metrik und den dazugehörigen Durchgang sind in Tabelle 2 die gemessenen Zahlenwerte zu sehen. Beim Betrachten der einzelnen Zahlenwerte fällt auf, dass die Zahlenwerte für die einzelnen Metriken konstant sind über die durchgeführten Durchläufe hinweg. Durch BERT_Precision wurde stets der Wert 0,7017 gemessen. Bei BERT_Recall wurde über die verschiedenen Durchgänge hinweg stets der Wert 0,7666 gemessen und für den BERT_F1Score wurde immer der Wert 0,7349 gemessen.

Tabelle 2: BERTScore-Metriken für das RAG-System ohne Website-Kontext
Quelle: Eigene Darstellung

RAG ohne Website-Kontext BERTScore-Metriken					
	Durchgang 1	Durchgang 2	Durchgang 3	Durchgang 4	Durchgang 5
BERT_Precision	0,7071	0,7071	0,7071	0,7071	0,7071
BERT_Recall	0,7666	0,7666	0,7666	0,7666	0,7666
BERT_F1Score	0,7349	0,7349	0,7349	0,7349	0,7349

Ebenfalls wurden für das RAG-System mit Website-Kontext die BERTScore-Metriken gemessen. Wie bei dem RAG-System ohne Website-Kontext wurden auch für das RAG-System mit Website-Kontext jeweils fünf Durchgänge für die BERTScore-Metriken durchgeführt. In der Tabelle 3 sind diese fünf Durchläufe als Spalten der Tabelle dargestellt. Die BERTScore-Metriken, die für das RAG-System mit Website-Kontext verwendet wurden, sind die gleichen Metriken, die auch bei dem RAG-System ohne Website-Kontext eingesetzt wurden. Die gemessenen Metriken sind für die beide RAG-Systeme Precision (P_{BERT}), Recall (R_{BERT}) und F1Score (F_{BERT}) von dem BERTScore. Diese Metriken sind in Tabelle 3 als Zeilen dargestellt. Auch in Tabelle 3 sind Precision als BERT_Precision, Recall als BERT_Recall und der F1Score als BERT_F1Score eingetragen. Ebenfalls sind die Metriken von oben nach unten als BERT_Precision, BERT_Recall und BERT_F1Score eingetragen. Die Ergebnisse der BERTScore-Metriken für das RAG-System mit Website-Kontext sind in Tabelle 3 zu sehen. Beim Betrachten der Ergebnisse fällt auf, dass die Zahlenwerte für die einzelnen Metriken in allen Durchgängen konstant sind. Es ist zu sehen, dass in allen Durchgängen für BERT_Precision der Zahlenwert 0,8136 gemessen wurden (siehe Tabelle 3). Beim BERT_Recall wurde immer den Zahlenwert 0,8640 gemessen und BERT_F1Score hat stets den Zahlenwert 0,8365 gemessen (siehe Tabelle 3). Auch bei diesen BERTScore-Metriken sind alle Werte das arithmetische Mittel für den jeweiligen Durchgang und die jeweilige Metrik.

Tabelle 3: BERTScore-Metriken für das RAG-System mit Website-Kontext
Quelle: Eigene Darstellung

RAG mit Website-Kontext BERTScore-Metriken					
	Durchgang 1	Durchgang 2	Durchgang 3	Durchgang 4	Durchgang 5
BERT_Precision	0,8136	0,8136	0,8136	0,8136	0,8136
BERT_Recall	0,8640	0,8640	0,8640	0,8640	0,8640
BERT_F1Score	0,8365	0,8365	0,8365	0,8365	0,8365

Als nächstes sollen die Metriken von Ragas betrachtet werden. Auch hier wird mit dem RAG-System ohne Website-Kontext begonnen. Wie bei den BERTScore-Metriken wurden auch für die Ragas-Metriken stets fünf Durchläufe durchgeführt.

In Tabelle 4 sind die Zahlenwerte für die Ragas-Metriken für das RAG-Systeme ohne Website-Kontext dargestellt. Auch in Tabelle 4 sind die verschiedenen Durchgänge als die Spalten der Tabelle und die Metriken als die Zeilen der Tabelle dargestellt.

Bei den Metriken handelt es sich um Context Recall, Faithfulness, ROUGE Score und Factual Correctness, von oben nach unten aufgezählt. Die Metrik Faithfulness hat über alle Durchgänge hinweg stets den Zahlenwert 0,7803 gemessen. Die Metrik ROUGE Score hat über die verschiedenen Durchgänge hinweg immer den Zahlenwert 0,3056 gemessen. Die Metrik Context Recall bzw. in der Tabelle auch als LLM Context Recall bezeichnet hat drei unterschiedliche Zahlenwerte gemessen. Der maximale Zahlenwert dieser Metrik ist 0,5040 und wurde bereits im ersten Durchgang gemessen. Das Minimum für LLM Context Recall für das RAG-System ohne Website-Kontext ist 0,4781 und wurde in den Durchgängen vier und fünf gemessen. Der Zahlenwert 0,4815 wurde in den Durchgängen zwei und drei gemessen. Um für diese Metrik einen einzigen Wert zum späteren Vergleichen zu haben, wird das arithmetische Mittel berechnet. Für Context Recall von den RAG-System ohne Website-Kontext ist das arithmetische Mittel $(0,5040 + 0,4815 + 0,4815 + 0,4781 + 0,4781) / 5 = 0,4846$ auf die vierte Nachkommastelle gerundet. Die Metrik Factual Correctness hat mit vier unterschiedlichen Zahlenwerten die meisten unterschiedlichen Zahlenwerte gemessen. Das Maximum für Factual Correctness liegt bei 0,7194, was im fünften Durchgang gemessen wurde. Das Minimum von 0,7097 für Factual Correctness wurde im zweiten Durchgang gemessen. Zwischen diesen Extremwerten liegt einmal der Zahlenwert von 0,7114 für Factual Correctness, der sowohl in Durchgang eins als auch in Durchgang drei gemessen wurde. Der andere Zahlenwert zwischen den Extremwerten ist 0,7160 und wurde in Durchgang vier gemessen. Auch für Factual Correctness wird das arithmetische Mittel berechnet. Das arithmetische Mittel für die Metric Factual Corretness des RAG-Systems ohne Website-Kontext liegt bei $(0,7114 + 0,7098 + 0,7114 + 0,7160 + 0,7194) / 5 = 0,7136$.

Tabelle 4: Ragas-Metriken für das RAG-System ohne Website-Kontext
Quelle: Eigene Darstellung

RAG ohne Website-Kontext Ragas-Metriken					
	Durchgang 1	Durchgang 2	Durchgang 3	Durchgang 4	Durchgang 5
LLM Context Recall	0,5040	0,4815	0,4815	0,4781	0,4781
Faithfulness	0,7803	0,7803	0,7803	0,7803	0,7803
ROUGE Score	0,3056	0,3056	0,3056	0,3056	0,3056
Factual Correctness	0,7114	0,7098	0,7114	0,7160	0,7194

Nun sollen die Ergebnisse für die Ragas-Metriken für das RAG-System mit Website-Kontext in Tabelle 5 betrachtet werden. Wie auch bei den Tabellen zuvor sind in Tabelle 5 die verschiedenen Durchgänge die Spalten der Tabelle und die unterschiedlichen Metriken sind die Zeilen der Tabelle. Ebenfalls wie zuvor wurden jeweils fünf Durchgänge pro Ragas-Metrik für das RAG-System mit Website-Kontext durchgeführt.

Beim Betrachten der Ergebnisse für das RAG-System mit Website-Kontext für die Ragas-Metriken in Tabelle 5 fällt auf, dass auch hier der ROUGE Score über die fünf Durchgänge hinweg, stets den gleichen Zahlenwert gemessen hat. Für den ROUGE Score wurde immer der Zahlenwert 0,5841 gemessen. Für die Metrik Faithfulness wurde in den ersten vier Durchgängen der Zahlenwert 0,7127 gemessen. Im Fünften Durchgang allerdings, wurde dann für Faithfulness der Zahlenwert 0,6927 gemessen. Um einen einzigen Wert für die Metrik zum späteren Vergleichen zu haben wird das arithmetische Mittel berechnet. Die Metrik Faithfulness aus dem RAG-System mit Website-Kontext hat das arithmetische Mittel von $(0,7127 + 0,7127 + 0,7127 + 0,7127 + 0,6927) / 5 = 0,7087$.

Die Metrik Context Recall bzw. auch LLM Context Recall in der Tabelle hat für das RAG-System mit Website-Kontext verschiedene Zahlenwerte gemessen. Das Maximum für LLM Context Recall ist 0,4915 und wurde in Durchgang vier gemessen. Das Minimum von 0,4798 wurde in Durchgang zwei gemessen. Der einzige Zahlenwert, der in dieser Metrik in zwei Durchgängen gemessen wurde, ist 0,4815. Dieser Zahlenwert wurde in den Durchgängen eins und fünf gemessen. Auch für LLM Context Recall soll ein einziger Wert berechnet werden, um später die Werte vergleichen zu können. Das arithmetische Mittel von Context Recall für das RAG mit Website-Kontext ist auf die vierte Nachkommastelle gerundet $(0,4815 + 0,4798 + 0,4831 + 0,4915 + 0,4815) / 5 = 0,4835$. Die Metrik Factual Correctness hat vier verschiedene Zahlenwerte für das RAG-System mit Website-Kontext gemessen. Das Maximum liegt bei 0,7914 aus Durchgang zwei. Das Minimum ist mit 0,7808 in Durchgang fünf zu finden. Zwischen diesen Extremwerten liegt der Zahlenwert 0,7842 und ist in Durchgang eins und in Durchgang vier gemessen worden. Wie auch zuvor soll für Factual Correctness das arithmetische Mittel berechnet werden. Für das RAG-System mit Website-Kontext liegt das arithmetische Mittel von Factual Correctness auf die vierte Nachkommastelle gerundet bei $(0,7842 + 0,7914 + 0,7876 + 0,7842 + 0,7808) / 5 = 0,7856$.

Tabelle 5: Ragas-Metriken für das RAG-System mit Website-Kontext
Quelle: Eigene Darstellung

RAG mit Website-Kontext Ragas-Metriken					
	Durchgang 1	Durchgang 2	Durchgang 3	Durchgang 4	Durchgang 5
LLM Context Recall	0,4815	0,4798	0,4831	0,4915	0,4815
Faithfulness	0,7127	0,7127	0,7127	0,7127	0,6927
ROUGE Score	0,5841	0,5841	0,5841	0,5841	0,5841
Factual Correctness	0,7842	0,7914	0,7876	0,7842	0,7808

6 Diskussion

Die in Kapitel 1.1 vorgestellte Forschungsfrage soll an dieser Stelle noch einmal wiederholt werden, damit die Ergebnisse aus Kapitel 5 im Hinblick auf die Forschungsfrage diskutiert werden können. Die Forschungsfrage, die in dieser Arbeit behandelt wird, lautet: *Sind die Antworten von einem RAG-System für Benutzer von Websites hilfreicher, wenn dem RAG-System weitere Informationen von der Weboberfläche zur Verfügung gestellt werden?*

Die Zahlenwerte auf, die sich in diesem Kapitel bezogen wird, sind die Ergebnisse, die bereits in Kapitel 5 beschrieben wurden. Zuerst sollen die Ergebnisse des BERTScores noch einmal betrachtet werden. Der BERT_Recall für das RAG-System ohne Website-Kontext liegt bei 0,7666. Für das RAG-System mit Website-Kontext liegt der BERT_Recall bei 0,8640. 0,8640 von 0,7666 sind 112,71%. Das bedeutet eine Steigerung von 12,71% durch das RAG mit Website-Kontext. BERT_Precision von dem RAG-System ohne Website-Kontext liegt bei 0,7071. Für das RAG-System mit Website-Kontext ist BERT_Precision 0,8136. 0,8136 von 0,7071 sind 115,06%. Das bedeutet eine Steigerung durch das RAG-System mit Website-Kontext von 15,06%. Für den BERT_F1Score beim RAG-System ohne Website-Kontext wurde stets der Zahlenwert von 0,7349 gemessen. Im Vergleich dazu hat das RAG mit Website-Kontext immer einen BERT_F1Score von 0,8365 erzielt. Dabei entspricht 0,8365 von 0,7349 113,83%, gerundet auf die zweite Nachkommastelle. Das entspricht somit einer Steigerung von 13,83% durch das RAG-System mit Website-Kontext gegenüber dem RAG-System ohne Website-Kontext. Es zeigt sich, dass die Steigerungen durch das RAG-System mit Website-Kontext für alle BERTScore-Metriken etwa im selben Größenbereich liegen. Das ist auch besonders für den BERT_F1Score erwartbar, da der BERT_F1Score das harmonische Mittel aus dem BERT_Recall und der BERT_Precision ist. Die BERTScore-Metriken vergleichen die Kandidatenantwort und die Referenzantwort aufgrund der Einbettungen miteinander. Wie beschrieben wurde, hat das RAG-System mit Website-Kontext in allen BERTScore-Metriken stärker abgeschnitten als das RAG-System ohne Website-Kontext. Damit zeigt sich, dass die Antworten des RAG-Systems mit Website-Kontext, nach dem BERTScore, deutlich näher an den Referenzantworten liegen als die Antworten des RAG-Systems ohne Website-Kontext. Dass die Antworten des RAG-Systems mit Website-Kontext näher an den Referenzantworten liegen, nach den Ergebnissen des BERTScores zu schließen, kann als ein Anzeichen dafür betrachtet werden, dass auch für einen Benutzer der Website die Antworten des RAG-Systems mit Website-Kontext hilfreicher sind. Das liegt daran, dass davon auszugehen ist, dass auch eine Benutzer Antworten als hilfreich empfindet, wenn die Antworten des RAG-Systems näher an den Referenzantworten liegen. Denn die Referenzantworten sind die Antworten aus der Dissertation, die am besten die gestellten Fragen aus der Fragenliste beantworten. Es ist also davon auszugehen, dass ein Benutzer Antworten als hilfreicher empfindet, wenn die Antworten näher an den Referenzantworten liegen. Somit kann davon ausgegangen werden, dass, nach den Ergebnissen des BERTScores zu urteilen, ein Benutzer die Antworten des RAG-Systems

mit Website-Kontext als hilfreicher empfindet als die Antworten des RAG-Systems ohne Website-Kontext. Damit können die Ergebnisse des BERTScores also ein starker Hinweis darauf sein, dass die Antworten eines RAG-Systems mit Informationen von der Weboberfläche hilfreicher sind, für einen Benutzer der Website.

Nun sollen auch die Ergebnisse von Ragas betrachtet und diskutiert werden. Hierbei wird als erstes der ROUGE Score betrachtet. Der verwendete ROUGE Score ist ein ROUGE-1 Score und davon wird der F1Score betrachtet. Der ROUGE-1 Score vergleicht Wort für Wort die Kandidatenantwort mit der Referenzantwort. Nach Kapitel 5 liegt der ROUGE Score des RAG-Systems ohne Website-Kontext bei 0,3056. Der ROUGE Score des RAG-Systems mit Website-Kontext liegt bei 0,5841. Hier ist ein signifikanter Unterschied zu sehen. Der Wert 0,5841 von 0,3056 entspricht 191,13% gerundet auf die zweite Nachkommastelle. Das entspricht einer Steigerung von 91,13% durch das RAG-System mit Website-Kontext gegenüber dem RAG-System ohne Website-Kontext. Da der ROUGE-1 Score, wie zuvor beschrieben, die einzelnen Worte der generierten Antworten mit den Referenzantworten vergleicht, ist daraus abzuleiten, dass das RAG-System mit Website-Kontext Antworten liefert, die wörtlich, also Wort für Wort, deutlich näher an den Referenzantworten liegen als die Antworten von dem RAG ohne Website-Kontext. In Bezug auf die Leitfrage können die Ergebnisse des ROUGE-1 Scores also als ein Anzeichen dafür gesehen werden, dass die Antworten des RAG-System mit Website-Kontext für einen Benutzer hilfreicher sind als die Antworten eines RAG-Systems ohne Website-Kontext. Denn die Antworten des RAG-Systems mit Website liegen nach dem ROUGE-1 Score wörtlich deutlich näher an den Referenzantworten als die Antworten des RAG-Systems ohne Website-Kontext. Wie bereits bei der Diskussion des BERT_F1Scores beschrieben, ist davon auszugehen, dass ein Benutzer Antworten präferiert, die näher an der Referenzantworten liegen. Deshalb sind nach den Ergebnissen des ROUGE-1 Scores, die Antworten des RAG-Systems mit Website-Kontext für einen Benutzer wahrscheinlich hilfreicher als die Antworten eines RAG-Systems ohne Website-Kontext. Auch in diesem Fall zeigt sich also, dass die Antworten von einem RAG-System mit Informationen von der Weboberfläche als hilfreich für eine Benutzer der Website eingeschätzt werden können, gegenüber einem RAG-System ohne Informationen von der Weboberfläche. Vor allem sind die Antworten des RAG-Systems mit Informationen von der Weboberfläche für einen Benutzer der Website wörtlich hilfreicher als ein RAG-System ohne Informationen von der Weboberfläche.

Als nächstes wird die Metrik Faithfulness aus Ragas genauer beleuchtet. Für das RAG-System ohne Website-Kontext ist für Faithfulness stets der Zahlenwert 0,7803 gemessen worden. Das arithmetische Mittel für die Faithfulness-Metrik aus dem RAG-System mit Website-Kontext ist 0,7087, wie in Kapitel 5 beschrieben. In diesem Fall ist der Wert für das RAG-System mit Website-Kontext niedriger. Der Wert 0,7087 des RAG-Systems mit Website-Kontext entspricht 90,82% von 0,7803, gerundet auf die zweite Nachkommastelle. Dieser Unterschied in den Werten ist mit dem abgerufenen Kontext zu erklären, der für die Bestimmung des Wertes für die Faithfulness-Metrik verwendet wird. Denn der abgerufene Kontext wird durch den Retriever aus der Anfrage an das RAG-System abgerufen. Für das RAG-

System ohne Website-Kontext wird aus diesem abgerufenen Kontext aus dem Retriever und der ursprünglichen Anfrage an das RAG-System durch das LLM die Antwort des RAG-Systems generiert. Im Fall des RAG-Systems mit Website-Kontext wird die Antwort durch das LLM aus dem abgerufenen Kontext aus dem Retriever, der ursprünglichen Anfrage an das RAG-System und dem Website-Kontext gebildet.

Da der Website-Kontext nicht direkt in den abgerufenen Kontext enthalten ist, kann das als eine Begründung dafür dienen, warum der Faithfulness-Wert für das RAG-System mit Website-Kontext geringer ist als der Wert für das RAG-System ohne Website-Kontext. Da der Website-Kontext die Inhalte aus der Dissertation enthält, die auch von den RAG-Systemen als Wissensbasis für den Retrieval-Prozess verwendet werden, kann es passieren, dass sich die Inhalte aus dem abgerufenen Kontext und dem Website-Kontext überschneiden. Das kann ein Grund sein, warum der Faithfulness-Wert für das RAG-System mit Website-Kontext immer noch relativ nah an dem Wert des RAG-Systems ohne Website-Kontext liegt, obwohl der Website-Kontext nicht direkt mit dem abgerufenen Kontext aus dem Retriever verknüpft wird. Somit ist es also erwartbar gewesen, dass der Faithfulness-Wert für das RAG-System mit Website-Kontext geringer ist als der Faithfulness-Wert des RAG-Systems ohne Website-Kontext.

Nun soll Context Recall als Metrik von Ragas betrachtet werden. Für beide RAG-Systeme liefert LLM Context Recall unterschiedliche Werte. Um die RAG-Systeme anhand einer Kennzahl vergleichen zu können, wird für beide RAG-Systeme das arithmetische Mittel aus den jeweiligen Werten von LLM Context Recall betrachtet. Für das RAG ohne Website-Kontext ist das arithmetische Mittel 0,4846. Für das RAG mit Website-Kontext ist das arithmetische Mittel 0,4835. Für diese Metrik sind die Werte der beiden RAG-Systeme sehr nah beieinander. Die Differenz zwischen 0,4846 und 0,4835 liegt bei nur 0,0011. 0,0011 ist bezogen auf das arithmetische Mittel für das RAG-System ohne Website-Kontext von 0,4846 nur 0,23%, gerundet auf die zweite Nachkommastelle. Und auch 0,0011 bezogen auf das arithmetische Mittel für das RAG-System mit Website-Kontext von 0,4835 ist nur 0,23%, gerundet auf die zweite Nachkommastelle. Der Differenz zwischen den beiden Werten liegt also in beiden Fällen bei unter einem Prozent. Das bei dieser Metrik fast kein Unterschied auftritt ist nicht verwunderlich, denn bei Context Recall wird überprüft, ob die Behauptungen bzw. Aussagen in der Referenzantwort aus dem abgerufenen Kontext begründet werden können. Als Referenzantworten werden die Antworten aus der Antwort-Liste verwendet, diese Antworten sind für beide RAG-Systeme gleich und auch der abgerufene Kontext ist für beide RAG-Systeme gleich, da die beiden RAG-Systeme der gleichen Retriever und das gleiche Dokument als Wissensbasis verwenden inklusive der verwendeten Parameter. Somit gibt es keinen Unterschied bei den Werten, die von Context Recall bei den beiden RAG-Systemen betrachtet werden. Somit ist es erwartbar, dass fast kein Unterschied bei Context Recall zwischen den beiden RAG-Systemen besteht. Dass die Werte nicht gleich sind und die Werte bei den einzelnen RAG-Systemen leicht variieren, könnte an der HNSW-Komponente innerhalb der Vektordatenbank von Chroma liegen. Durch HNSW gibt es eine Zufallskomponente in den RAG-Systemen und dadurch können verschiedene Kontexte durch den Retriever abgerufen werden. Da durch den Retriever stets fünf Inhalte abgerufen

werden, fällt es weniger stark ins Gewicht, wenn in manchen Fällen andere Kontexte abgerufen werden, die für die Referenzantworten wenig relevant sind. Allerdings ist es an dieser Stelle sicherlich hilfreich noch weitere Durchgänge mit dieser Metrik durchzuführen, um mehr Daten zu erheben und diese analysieren zu können, um den Grund für die Varianz tiefer zu verstehen.

Als nächstes wird ein Augenmerk auf die Metrik Factual Corretness von Ragas gelegt. Auch bei dieser Metrik haben die beiden RAG-Systeme unterschiedliche Werte gemessen. Wie bei LLM Context Recall soll auch in diesem Fall das arithmetische Mittel für die beiden RAG-Systeme betrachtet werden, um die RAG-Systeme miteinander vergleichen zu können. Das arithmetische Mittel für die Factual Corretness des RAG-Systems ohne Website-Kontext liegt bei 0,7136. Für das RAG-System mit Website-Kontext liegt das arithmetische Mittel von Factual Correctness bei 0,7856. Hier ist ein höherer Wert bei dem RAG-System mit Website-Kontext zu sehen. 0,7856 von 0,7136 sind 110,09%, auf die zweite Nachkommastelle gerundet. Das entspricht einer Steigerung von 10,09 Prozent durch das RAG-System mit Website-Kontext gegenüber dem RAG-System ohne Website-Kontext. Damit ist auch bei Factual Correctness durch das RAG-System mit Website-Kontext ein höherer Wert messbar, verglichen mit dem RAG-System ohne Website-Kontext. Da Factual Correctness die Referenzantwort und die generierte Antwort vergleicht, bedeutet das, dass das RAG mit Website-Kontext Antwort liefert, die näher an den Referenzantworten liegen als die Antworten der RAG-Systems ohne Website-Kontext. Bezogen auf die Fragestellung dieser Arbeit zeigt auch das Ergebnis von Factual Correctness, dass die Antworten des RAG-Systems mit Website-Kontext näher an den Referenzantworten liegen als die Antworten des RAG-Systems ohne Website-Kontext. Somit ist auch in diesem Fall davon auszugehen, dass auch für einen Benutzer der Website die Antworten des RAG-Systems mit Website-Kontext hilfreicher sind als die Antworten von dem RAG-System ohne Website-Kontext. Also ist auch in diesem Fall davon auszugehen, dass die Antworten von einem RAG-System mit Informationen von der Weboberfläche als hilfreich für eine Benutzer der Website einzuschätzen sind, gegenüber einem RAG-System ohne Informationen von der Weboberfläche.

Somit haben alle Metriken, die die generierte Antwort mit der Referenzantwort verglichen haben, einen höheren Wert für das RAG-System mit Website-Kontext gemessen. Bei diesen Metriken handelt es sich im Besonderen um den ROUGE Score, den BERT_F1Score und Factual Correctness. Damit liegt das RAG-System mit Website-Kontext näher an den Referenzantworten, was als ein Vorteil angesehen werden kann. Dennoch sollte berücksichtigt werden, dass in dieser Arbeit für alle Fragen aus der Fragen-Liste immer relevante Informationen auf der Website vorhanden waren. In einem realen Anwendungsfall kann davon nicht zwingend ausgegangen werden. Daher repräsentieren die Ergebnisse in dieser Arbeit einen Idealfall für das RAG-System mit Website-Kontext. Somit kann festgehalten werden, dass das RAG-System mit Website-Kontext unter idealen Bedingungen Antworten geliefert hat, die näher an den gewünschten Antworten liegen als die Antworten eines RAG-Systems ohne Website-Kontext. Weil

in den oben genannten Fällen gezeigt werden konnte, dass die generierten Antworten des RAG-Systems mit Website-Kontext näher an den Referenzantworten liegen, kann davon ausgegangen werden, dass die Antworten des RAG-Systems mit Website-Kontext auch für einen Benutzer hilfreicher seien, sollten. Es sollte auch betrachtet werden, dass die Steigerung durch das RAG-System mit Website-Kontext bei ROUGE-1 mit 91,13% deutlich höher war als bei dem BERT_F1Score bei dem eine Steigerung von 13,83% durch das RAG-System mit Website-Kontext gemessen wurde. Auch bei Factual Correctness ist die Steigerung von 10,09% durch das RAG-System mit Website-Kontext deutlich geringer als bei ROUGE-1 und näher an dem Ergebnis des BERT_F1Scores.

Daraus lässt sich schließen, dass anhand von BERT_F1Score und Factual Correctness die Antworten des RAG-Systems mit Website-Kontext inhaltlich durchaus näher an den Referenzantworten liegen als die Antworten des RAG-Systems ohne Website-Kontext und somit als hilfreicher für einen Benutzer eingeschätzt werden können. Aber der deutlich größere Unterschied scheint nach dem ROUGE-1 Score darin zu liegen, dass die Antworten des RAG-Systems mit Website-Kontext vor allem wörtlich deutlich näher an den Referenzantworten liegen als die Antworten durch das RAG-System ohne Website-Kontext.

Ist also festzuhalten, dass die Antworten von einem RAG-System mit Informationen von der Weboberfläche als hilfreicher für einen Benutzer der Website sind, gegenüber einem RAG-System ohne Informationen von der Weboberfläche. Vor allem sind die Antworten des RAG-Systems mit Informationen von der Weboberfläche wörtlich hilfreicher als die Antworten von einem RAG-System ohne Informationen von der Weboberfläche.

In Anwendungsfällen in denen davon ausgegangen werden kann, dass zu den Fragen eines Benutzers relevante Informationen auf der Website zur Verfügung stehen, kann das RAG-System mit Website-Kontext also einen besonders starken Mehrwert gegenüber einem RAG ohne Website-Kontext bieten und für einen Benutzer hilfreicher sein.

6.1 Limitationen

An dieser Stelle soll auf die Beschränkungen bzw. Limitationen dieser Arbeit eingegangen werden. Einerseits ist während der Bearbeitung verstärkt klar geworden, dass die Verwendung von LLMs die vorhandene Hardware an Grenzen bringt. Large Language Models mit einer größeren Parameterzahl zu verwenden ist teilweise nicht möglich gewesen. Einerseits ist Random Access Memory (kurz RAM) ein begrenzender Faktor gewesen, aber auch die verwendete Graphikkarte kann zu einem begrenzenden Faktor werden. Aufgrund dessen wurde ein verhältnismäßig kleines LLM in dieser Arbeit eingesetzt. Größere LLMs mit einer hohen Parameterzahl können in der Regel Ergebnisse liefern, die weitere Gebiete umfassen als LLMs mit einer kleineren Parameterzahl. Darum sind die Ergebnisse dieser Arbeit

nicht zwingend auf Anwendungsfälle übertragbar, in denen LLMs mit einer hohen Parameteranzahl verwendet werden.

Darüber hinaus ist die Anzahl an Durchgängen bei den Metriken limitiert. Bei einigen Metriken konnte beobachtet werden, dass die gemessenen Werte konstant geblieben sind. Bei anderen Metriken jedoch, haben die gemessenen Werte variiert. Mit den fünf durchgeführten Durchgängen in dieser Arbeit kann nur mit einer begrenzten Aussagekraft über das Verhalten der Metriken diskutiert werden. Die Tendenzen und das Verhalten, sowie der Bereich, in dem die Werte der Metriken liegen, sind in dieser Arbeit erkennbar geworden. Allerdings können mit weiteren Durchgängen noch weitere Tendenzen und tieferes Verhalten der Metriken erkennbar werden. Darum ist die Aussagekraft in dieser Arbeit über die Metriken begrenzt.

Zusätzlich wurde das verwendete LLM nur mit Inhalten in Deutsch verwendet. Das verwendete LLM ist multilingual und sollte in Sprachen, die auch durch das LLM unterstützt werden, ähnliche Ergebnisse liefern, allerdings kann nicht garantiert werden dass die Ergebnisse ähnlich sind. Über Sprachen, die nicht durch das unterstützte LLM kann keine genauere Aussage gemacht werden.

Darüber hinaus wurde in dieser Arbeit eine lokale HTML-Datei verwendet, um den Website-Kontext zu extrahieren. Das genutzte Tool BeautifulSoup 4 kann auch Webseiten aus dem Internet parsen, allerdings ist zu beachten, dass Webseiten aus dem Internet deutlich komplexer sein, können und daher das Extrahieren der gewünschten Informationen schwerer möglich sein kann. Daher kann es passieren, dass beim Nutzen von Webseiten aus dem Internet beim Extrahieren der gewünschten Inhalte ein größerer Aufwand benötigt wird.

Eine weitere Beschränkung, die auch als eine Voraussetzung für diese Arbeit betrachtet werden kann, ist das auf der verwendeten Webseite auch relevante Inhalte zu den gestellten Fragen des Benutzers vorhanden sein müssen, damit das RAG-System mit Website-Kontext die Informationen von der Webseite nutzen kann. Es handelt sich hierbei um eine Voraussetzung, weil der Webkontext von einer Webseite nur hilfreich sein kann, wenn auch hilfreicher Webkontexte zu einer Frage auf einer Webseite vorhanden ist. Das heißt, dass zu Fragen, zu denen kein Inhalt auf einer Webseite vorhanden ist, auch kein Unterschied zwischen einem RAG-System mit Website-Kontext und einem RAG-System ohne Website-Kontext messbar sein sollte. Der Fall, dass zu einer Frage kein relevanter Webkontext auf einer Webseite vorhanden ist, wurde in dieser Arbeit nicht speziell betrachtet. Denn es ist klar, dass eine RAG-System nicht von Webkontext profitieren kann, wenn dieser Webkontext nicht vorhanden ist. Somit würde in dem Fall, dass kein relevanter Webkontext zu einer Frage auf der Webseite vorhanden ist kein Unterschied zwischen einem RAG-System mit Website-Kontext und einem RAG-System ohne Website-Kontext bestehen.

Der Frage-Antwort Datensatz, der für die Messung der Metriken eingesetzt wurde, ist kein Standarddatensatz. Daraus folgt, dass die Ergebnisse dieser Arbeit schwieriger mit Ergebnissen anderer Arbeiten vergleichbar sind.

6.2 Weiterführende Arbeiten

Anknüpfend an diese Arbeit sind noch weiterführende Arbeiten möglich. In einer weiterführenden Arbeit wäre es interessant zu erforschen, ob bestimmte Fragen oder Fragentypen besonders von Informationen aus dem Webkontext in RAG-Systemen profitieren. Dazu wäre denkbar Fragen in beispielsweise Fragen zu Allgemeinwissen oder Fragen zu Fachwissen einzuteilen und dann zu untersuchen, ob Unterschiede messbar werden. Dadurch könnten Anwendungsfälle in denen RAG-Systeme mit Website-Kontext einen Mehrwert bieten, noch stärker eingegrenzt werden. Dadurch könnte noch präzisiert werden, wann ein RAG-System mit Website-Kontext eingesetzt werden sollte und in welchen Fällen ein RAG-System ohne Website-Kontext ausreichend ist.

Zusätzlich wäre es interessant zu beobachten, ob durch die Metriken neue Tendenzen und Verhalten auftreten, wenn mehr Durchläufe durchgeführt werden. Eine Arbeit in der zehn, 50 oder sogar 100 Durchläufe durchgeführt werden, könnte neue Erkenntnisse liefern, ob die Metriken sich in einer Weise verhalten, die in dieser Arbeit noch nicht beobachtet wurde.

Darüber hinaus könnte in einer weiterführenden Arbeit erforscht werden, ob RAG-Systeme von zusätzlichen Webkontext profitieren, wenn die RAG-Systeme unterschiedliche Sprache verwenden. Wenn verschiedene Sprachen für RAG-Systeme verwendet werden, könnte überprüft werden, ob in bestimmten Sprachen oder in bestimmten Sprachfamilien besonders hohe Ergebnisse erzielt werden, wenn zusätzlicher Webkontext für die RAG-Systeme verwendet wird.

Zusätzlich könnte in einer weiterführenden Arbeit, RAG-Systeme mit verschiedenen LLMs eingesetzt werden und überprüft werden, ob bestimmt LLMs mehr durch Website-Kontext profitieren als andere. Die LLMs sollten sich in der Parameteranzahl unterscheiden, damit getestet werden kann, ob die Parameteranzahl in den LLMs in den RAG-Systemen Einfluss darauf hat, ob eine RAG-System durch Website-Kontext mehr profitiert oder eher weniger profitiert. Damit könnte gezeigt werden, ob sich die Ergebnisse dieser Arbeit auf RAG-Systeme mit LLMs, die eine größere oder kleinere Parameterzahl aufweisen, verallgemeinern lassen oder nicht. Zudem könnte vielleicht gezeigt, ob ab einer bestimmten Parameterzahl die Verwendung von Website-Kontext besonders effektiv ist.

Des Weiteren sind die Ergebnisse, die in dieser Arbeit vorgelegt worden sind, mithilfe von Metriken bestimmt worden. Eine weiterführende Arbeit könnte in Form einer Umfrage erforschen, wie Benutzer die Antworten der RAG-Systeme, einmal ein RAG-System ohne Website-Kontext und einmal ein RAG-System mit Website-Kontext, bewerten. Dadurch könnte überprüft werden, ob ein Unterschied zwischen den gemessenen Metriken und der Meinung von Benutzern besteht. Denn die Ergebnisse der Metriken müssen sich nicht zwingend die Meinungen der Benutzer widerspiegeln. Eine Arbeit, die die Meinungen von Benutzern zu den Antworten von einem RAG-System mit Informationen von der Weboberfläche und einem RAG-System ohne Informationen von der Weboberfläche zeigt und auswertet, könnte auch

dafür sorgen, dass in zukünftigen Arbeiten stärker auf die Meinung von Benutzern eingegangen werden kann.

Darüber hinaus könnte es aufschlussreich sein, wenn die Durchführung aus dieser Arbeit mit einem oft verwendeten Datensatz aus anderen Arbeiten wiederholt wird. Dann würden die Metriken für die zwei RAG-Systeme mithilfe von einem oft verwendeten Dataset bestimmt werden. Damit würden neue Ergebnisse aus einem oft verwendeten Datensatz entstehen und es wäre leicht möglich diese neuen Ergebnisse mit Ergebnissen aus anderen Arbeiten zu vergleichen. Der Vergleich mit anderen Ergebnissen auf Grundlage eines standardisierten Datasets würde möglicherweise dafür sorgen das neue Erkenntnisse und Rückschlüsse gewonnen werden können. Zusätzlich könnten dann auch die Ergebnisse dieser Arbeit mit den Ergebnissen einer Arbeit mit einem bereits oft verwendeten Dataset verglichen werden.

7 Fazit

Die Forschungsfrage, die in dieser Arbeit behandelt wurde, lautet: *Sind die Antworten von einem RAG-System für Benutzer von Websites hilfreicher, wenn dem RAG-System weitere Informationen von der Weboberfläche zur Verfügung gestellt werden?*

Zusammenfassend können noch einmal die wichtigsten Ergebnisse wiederholt werden. Das RAG-System mit Website-Kontext hat eine Steigerung um 13,83% in dem BERT_F1Score erzielt gegenüber dem RAG-System ohne Website-Kontext. Bei ROUGE-1 hat das RAG-System mit Website-Kontext eine Steigerung von 91,13% gegenüber dem RAG ohne Website-Kontext erzielen können. Im Fall von ROUGE-1 ist die Steigerung durch das RAG mit Website-Kontext maximal innerhalb dieser Arbeit. Für die Metrik Factual Correctness hat das RAG mit Website-Kontext eine Steigerung von 10,09 Prozent gegenüber dem RAG-System ohne Website-Kontext erzielen können. Bei all diesen Metriken handelt es sich um Metriken, die die generierte Antwort eines RAG-Systems mit einer Referenzantwort vergleichen. Diese Werte legen nahe, dass die Antworten des RAG-Systems mit Website-Kontext näher an den Referenzantworten liegen als die Antworten eines RAG-Systems ohne Website-Kontext. Bei den Referenzantworten handelt es sich um die bestmöglichen Antworten auf die gestellten Fragen auf Grundlage der Inhalte in dem Text, der von den RAG-Systemen für die Such nach Inhalten verwendet wird. Deshalb kann davon ausgegangen werden, dass ein Benutzer Antworten als hilfreicher ansieht, wenn die Antworten näher an den Referenzantworten liegen. Somit kann geschlussfolgert werden, dass die Antworten des RAG-Systems mit Website-Kontext hilfreicher sind für eine Benutzer der Website als die Antworten des RAG-Systems ohne Website-Kontext. Damit ist für die Forschungsfrage zu sagen, dass davon auszugehen ist, dass die Antworten von einem RAG-System für Benutzer von Websites hilfreicher sind, wenn dem RAG-System weitere Informationen von der Weboberfläche zu Verfügung gestellt werden. Die Steigerung durch das RAG-System mit Website-Kontext ist im Fall von ROUGE-1 deutlich höher als die Steigerung beim BERT_F1Score oder bei Factual Correctness. In Zahlen bedeutet das,

dass im Fall von ROUGE-1 ist die Steigerung durch das RAG-System mit Website-Kontext mit 91,13% deutliche höher ist als die Steigerung durch das RAG-System mit Website-Kontext bei BERT_F1Score mit 13,83% oder Factual Correctness mit 10,09%.

Anhand dieser Differenz kann geschlussfolgert werden, dass das RAG-System mit Website-Kontext besonders wörtliche, also Wort für Wort, Antworten liefert, die deutlich näher an den Referenzantworten liegen als die die Antworten des RAG-Systems ohne Website-Kontext. Durch die Steigerungen beim BERT_F1Score und Factual Correctness kann abgeleitet werden, dass die Antworten des RAG-Systems mit Website-Kontext auch inhaltlich und nicht nur wörtlich näher an den Referenzantworten liegen und somit hilfreicher für einen Benutzer der Website seien sollten. Dennoch ist der stärkste Effekt durch das RAG-System mit Website-Kontext, dass die Antworten vor allem wörtlich deutlich näher an den Referenzantworten liegen als die Antworten des RAG-Systems ohne Website-Kontext. Somit sind die Antworten des RAG-System mit Website-Kontext besonders hilfreich, wenn die Antwort des RAG-Systems Wort für Wort möglichst nah an der Referenzantwort liegen sollen. In einem Anwendungsfall, in dem die generierten Antworten von einem RAG-System wörtlich besonders nah an der Referenzantwort liegen sollen, ist das RAG-System mit Website-Kontext also am hilfreichsten gegenüber dem RAG ohne Website-Kontext. Im Hinblick auf die Ergebnisse sollte beachtet werden, dass die Ergebnisse für das RAG mit Website-Kontext unter besonders günstigen Bedingungen für das RAG-System mit Website-Kontext erhoben wurden. In einem realen Anwendungsfall kann in der Regel nicht davon ausgegangen werden, dass zu allen Fragen eines Benutzers die bestmöglichen Informationen auf der Website vorhanden sind. Daher sind für eine umfassendere Beantwortung der Leitfrage weitere Forschung in diesem Gebiet nötig.

Durch die in dieser Arbeit vorgelegten Ergebnisse lässt sich also sagen, dass die Antworten eines RAG-Systems mit weiteren Informationen von der Weboberfläche hilfreicher sind, für einen Benutzer der Website, gegenüber einem RAG-System ohne Website-Kontext, wenn die Informationen von der Web-oberfläche geeignet sind und dass die Antworten des RAG-Systems mit Informationen von der Web-oberfläche vor allem wörtlich deutlich näher an den gewünschten Antworten liegen als die Antworten des RAG-Systems ohne Informationen von der Weboberfläche.

Literaturverzeichnis

- [1] G. Schmalzried, „Ein Jahr ChatGPT: Wie weit ist die KI-Revolution heute?“, BR24. Zugegriffen: 22. Oktober 2025. [Online]. Verfügbar unter: <https://www.br.de/nachrichten/netzwelt/ein-jahr-chatgpt-wie-weit-ist-die-ki-revolution-heute>, TwucRbm
- [2] „ChatGPT ist da“, OpenAI. Zugegriffen: 22. Oktober 2025. [Online]. Verfügbar unter: <https://openai.com/de-DE/index/chatgpt/>
- [3] A. Menn, „Ein Jahr ChatGPT: Diese Grafiken zeigen, wer die Gewinner des KI-Hypes sind“, WirtschaftsWoche. Zugegriffen: 22. Oktober 2025. [Online]. Verfügbar unter: <https://www.wiwo.de/technologie/digitale-welt/kuenstliche-intelligenz-ein-jahr-chatgpt-diese-grafiken-zeigen-wer-die-gewinner-des-ki-hypes-sind/29531946.html>
- [4] L. Mearian, „Large Language Models erklärt: Was sind LLMs?“, Computerwoche. Zugegriffen: 22. Oktober 2025. [Online]. Verfügbar unter: <https://www.computerwoche.de/article/2823883/was-sind-llms.html>
- [5] J. Maynez, S. Narayan, B. Bohnet, und R. McDonald, „On Faithfulness and Factuality in Abstractive Summarization“, 2. Mai 2020, *arXiv*: arXiv:2005.00661. doi: 10.48550/arXiv.2005.00661.
- [6] K. Shuster, S. Poff, M. Chen, D. Kiela, und J. Weston, „Retrieval Augmentation Reduces Hallucination in Conversation“, 15. April 2021, *arXiv*: arXiv:2104.07567. doi: 10.48550/arXiv.2104.07567.
- [7] T. Honroth, J. Siebert, und P. Kelbert, „Retrieval Augmented Generation (RAG): Chatten mit den eigenen Daten“, Fraunhofer IESE. Zugegriffen: 13. Oktober 2025. [Online]. Verfügbar unter: <https://www.iese.fraunhofer.de/blog/retrieval-augmented-generation-rag/>
- [8] V. Plenck, *Angewandte Netzwerktechnik kompakt: Dateiformate, Übertragungsprotokolle und ihre Nutzung in Java-Applikationen*. in IT kompakt. Wiesbaden: Springer Fachmedien Wiesbaden, 2024. doi: 10.1007/978-3-658-43530-1.
- [9] T. Eichstädt und S. Spieker, *52 Stunden Informatik: Was jeder über Informatik wissen sollte*. Wiesbaden: Springer Fachmedien Wiesbaden, 2024. doi: 10.1007/978-3-658-41838-0.
- [10] R. Lackes und M. Siepermann, „Hypertext“, Gabler Wirtschaftslexikon. Zugegriffen: 29. Oktober 2025. [Online]. Verfügbar unter: <https://wirtschaftslexikon.gabler.de/definition/hypertext-36715/version-107570>
- [11] P. Bühler, P. Schlaich, und D. Sinner, *HTML und CSS: Semantik - Design - Responsive Layouts*. in Bibliothek der Mediengestaltung. Berlin, Heidelberg: Springer Berlin Heidelberg, 2023. doi: 10.1007/978-3-662-66663-0.
- [12] „Document Object Model (DOM)“, MDN Web Docs. Zugegriffen: 26. Januar 2026. [Online]. Verfügbar unter: https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model
- [13] B. Tural, Z. Örpek, und Z. Destan, „Retrieval-Augmented Generation (RAG) and LLM Integration“, in *2024 8th International Symposium on Innovative Approaches in Smart Technologies (ISAS)*, Istanbul, Türkiye: IEEE, Dez. 2024, S. 1–5. doi: 10.1109/ISAS64331.2024.10845308.
- [14] P. Zhao u. a., „Retrieval-Augmented Generation for AI-Generated Content: A Survey“, 24. Juni 2024, *arXiv*: arXiv:2402.19473. doi: 10.48550/arXiv.2402.19473.
- [15] S. Wu u. a., „Retrieval-Augmented Generation for Natural Language Processing: A Survey“, 4. März 2025, *arXiv*: arXiv:2407.13193. doi: 10.48550/arXiv.2407.13193.
- [16] W. Xie, X. Liang, Y. Liu, K. Ni, H. Cheng, und Z. Hu, „WeKnow-RAG: An Adaptive Approach for Retrieval-Augmented Generation Integrating Web Search and Knowledge Graphs“, 28. August 2024, *arXiv*: arXiv:2408.07611. doi: 10.48550/arXiv.2408.07611.
- [17] S.-A. Wegner, *Mathematische Einführung in Data Science*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2023. doi: 10.1007/978-3-662-68697-3.
- [18] M. Douze u. a., „THE FAISS LIBRARY“, *IEEE Trans. Big Data*, S. 1–17, 2025, doi: 10.1109/TBDATA.2025.3618474.
- [19] Y. A. Malkov und D. A. Yashunin, „Efficient and Robust Approximate Nearest Neighbor Search Using Hierarchical Navigable Small World Graphs“, *IEEE Trans. Pattern Anal. Mach. Intell.*, Bd. 42, Nr. 4, S. 824–836, Apr. 2020, doi: 10.1109/TPAMI.2018.2889473.

- [20] Z. Örpək, B. Tural, und Z. Destan, „The Language Model Revolution: LLM and SLM Analysis“, in *2024 8th International Artificial Intelligence and Data Processing Symposium (IDAP)*, Malatya, Türkiye: IEEE, Sep. 2024, S. 1–4. doi: 10.1109/IDAP64064.2024.10710677.
- [21] S. Minaee u. a., „Large Language Models: A Survey“, 23. März 2025, *arXiv*: arXiv:2402.06196. doi: 10.48550/arXiv.2402.06196.
- [22] W. X. Zhao u. a., „A Survey of Large Language Models“, 11. März 2025, *arXiv*: arXiv:2303.18223. doi: 10.48550/arXiv.2303.18223.
- [23] M. Peeperkorn, T. Kouwenhoven, D. Brown, und A. Jordanous, „Is Temperature the Creativity Parameter of Large Language Models?“, 1. Mai 2024, *arXiv*: arXiv:2405.00492. doi: 10.48550/arXiv.2405.00492.
- [24] 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. [Online]. Verfügbar unter: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf
- [25] A. Chowdhery u. a., „Palm: Scaling language modeling with pathways“, *J. Mach. Learn. Res.*, Bd. 24, Nr. 240, S. 1–113, 2023.
- [26] J. Siebert und P. Kelbert, „Wie funktionieren LLMs? Ein Blick ins Innere großer Sprachmodelle“, Fraunhofer IESE. Zugegriffen: 21. Januar 2026. [Online]. Verfügbar unter: <https://www.iese.fraunhofer.de/blog/wie-funktionieren-llms/>
- [27] J. Devlin, M.-W. Chang, K. Lee, und K. Toutanova, „BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding“, in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, J. Burstein, C. Doran, und T. Solorio, Hrsg., Minneapolis, Minnesota: Association for Computational Linguistics, Juni 2019, S. 4171–4186. doi: 10.18653/v1/N19-1423.
- [28] R. Boselli, S. D’Amico, und N. Nobani, „eXplainable AI for Word Embeddings: A Survey“, *Cogn. Comput.*, Bd. 17, Nr. 1, S. 19, Dez. 2024, doi: 10.1007/s12559-024-10373-2.
- [29] „The AI community building the future“, Hugging Face. Zugegriffen: 2. Februar 2026. [Online]. Verfügbar unter: <https://huggingface.co/>
- [30] F. Almeida und G. Xexéo, „Word Embeddings: A Survey“, 2. Mai 2023, *arXiv*: arXiv:1901.09069. doi: 10.48550/arXiv.1901.09069.
- [31] J. Pennington, R. Socher, und C. Manning, „Glove: Global Vectors for Word Representation“, in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar: Association for Computational Linguistics, 2014, S. 1532–1543. doi: 10.3115/v1/D14-1162.
- [32] T. Mikolov, K. Chen, G. Corrado, und J. Dean, „Efficient Estimation of Word Representations in Vector Space“, 7. September 2013, *arXiv*: arXiv:1301.3781. doi: 10.48550/arXiv.1301.3781.
- [33] N. Reimers und I. Gurevych, „Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks“, 27. August 2019, *arXiv*: arXiv:1908.10084. doi: 10.48550/arXiv.1908.10084.
- [34] *IEEE Standard for a Software Quality Metrics Methodology*. doi: 10.1109/IEEESTD.1998.243394.
- [35] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, und Y. Artzi, „BERTScore: Evaluating Text Generation with BERT“, 24. Februar 2020, *arXiv*: arXiv:1904.09675. doi: 10.48550/arXiv.1904.09675.
- [36] C.-Y. Lin, „ROUGE: A Package for Automatic Evaluation of Summaries“.
- [37] „ROUGE-Score“, FlowHunt. Zugegriffen: 23. Januar 2026. [Online]. Verfügbar unter: <https://www.flowhunt.io/de/glossar/rouge-score/>
- [38] „List of available metrics“, Ragas. Zugegriffen: 28. Dezember 2025. [Online]. Verfügbar unter: https://docs.ragas.io/en/stable/concepts/metrics/available_metrics/
- [39] „Faithfulness“, Ragas. Zugegriffen: 19. November 2025. [Online]. Verfügbar unter: https://docs.ragas.io/en/stable/concepts/metrics/available_metrics/faithfulness/
- [40] „Factual Correctness“, Ragas. Zugegriffen: 19. November 2025. [Online]. Verfügbar unter: https://docs.ragas.io/en/stable/concepts/metrics/available_metrics/factual_correctness/?h=factual+correctness
- [41] Y. Chang u. a., „A Survey on Evaluation of Large Language Models“, *ACM Trans. Intell. Syst. Technol.*, Bd. 15, Nr. 3, S. 1–45, Juni 2024, doi: 10.1145/3641289.

- [42] „Klassifizierung: Genauigkeit, Trefferquote, Präzision und zugehörige Messwerte“, Google for Developers. Zugriffen: 3. Februar 2026. [Online]. Verfügbar unter: <https://developers.google.com/machine-learning/crash-course/classification/accuracy-precision-recall?hl=de>
- [43] G. Paaß und D. Hecker, *Künstliche Intelligenz: Was steckt hinter der Technologie der Zukunft?* Wiesbaden: Springer Fachmedien Wiesbaden, 2020. doi: 10.1007/978-3-658-30211-5.
- [44] „Context Recall“, Ragas. Zugriffen: 19. November 2025. [Online]. Verfügbar unter: https://docs.ragas.io/en/stable/concepts/metrics/available_metrics/context_recall/?h=
- [45] G. Colverd, P. Darm, L. Silverberg, und N. Kasmanoff, „FloodBrain: Flood Disaster Reporting by Web-based Retrieval Augmented Generation with an LLM“, 5. November 2023, *arXiv:2311.02597*. doi: 10.48550/arXiv.2311.02597.
- [46] N. Kanataria, K. P. Patel, H. N. Patel, P. Goel, K. Patel, und D. Garg, „RAG-Enhanced Large Language Model for Intelligent Assistance from Web-Scraped Data“, in *2024 9th International Conference on Communication and Electronics Systems (ICCES)*, Coimbatore, India: IEEE, Dez. 2024, S. 1043–1048. doi: 10.1109/ICCES63552.2024.10859894.
- [47] „Flask“, Flask. Zugriffen: 19. November 2025. [Online]. Verfügbar unter: <https://flask.palletsprojects.com/en/stable/>
- [48] „Introduction“, Bootstrap. Zugriffen: 11. November 2025. [Online]. Verfügbar unter: <https://getbootstrap.com/docs/5.0/getting-started/introduction/>
- [49] „Bootstrap 5: Die wichtigsten Neuerungen“, IONOS Digital Guide. Zugriffen: 20. November 2025. [Online]. Verfügbar unter: <https://www.ionos.de/digitalguide/websites/web-entwicklung/bootstrap-5/>
- [50] „Tooltips“, Bootstrap. Zugriffen: 23. November 2025. [Online]. Verfügbar unter: <https://getbootstrap.com/docs/5.0/components/tooltips/>
- [51] „Welcome to LlamaIndex 🦙!“, LlamaIndex Python Documentation. Zugriffen: 11. November 2025. [Online]. Verfügbar unter: <https://developers.llamaindex.ai/python/framework/>
- [52] „llama3.2:3b“, Ollama. Zugriffen: 18. November 2025. [Online]. Verfügbar unter: <https://ollama.com/library/llama3.2:3b>
- [53] „What will you build?“, Ollama. Zugriffen: 2. Februar 2026. [Online]. Verfügbar unter: <https://ollama.com>
- [54] „Ollama“, mendix. Zugriffen: 18. November 2025. [Online]. Verfügbar unter: <https://www.mendix.com/de/Glossar/ollam/>
- [55] K. David, „Funktionales Kostensplitting zur Ermittlung von Mieterhöhungen nach energetischen Maßnahmen - Eine Handlungsempfehlung auf Basis theoretischer und empirischer Untersuchungen“, HafenCity Universität Hamburg, 2019. Zugriffen: 20. November 2025. [Online]. Verfügbar unter: <https://repos.hcu-hamburg.de/handle/hcu/505>
- [56] „SentenceSplitter“, LlamaIndex. Zugriffen: 12. Januar 2026. [Online]. Verfügbar unter: https://developers.llamaindex.ai/python/framework-api-reference/node_parsers/sentence_splitter/#llama_index.core.node_parser.SentenceSplitter
- [57] „Node Parser Modules“, LlamaIndex. Zugriffen: 31. Januar 2026. [Online]. Verfügbar unter: https://developers.llamaindex.ai/python/framework/module_guides/loading/node_parsers/modules/
- [58] „Beautiful Soup Documentation“, crummy. Zugriffen: 21. November 2025. [Online]. Verfügbar unter: <https://www.crummy.com/software/BeautifulSoup/bs4/doc/>
- [59] „sentence-transformers/paraphrase-multilingual-MiniLM-L12-v2“, Hugging Face. Zugriffen: 13. November 2025. [Online]. Verfügbar unter: <https://huggingface.co/sentence-transformers/paraphrase-multilingual-MiniLM-L12-v2>
- [60] „Getting Started“, Chroma. Zugriffen: 18. November 2025. [Online]. Verfügbar unter: <https://docs.trychroma.com/docs/overview/getting-started>
- [61] „Persistent Client“, Chroma. Zugriffen: 8. Dezember 2025. [Online]. Verfügbar unter: <https://docs.trychroma.com/docs/run-chroma/persistent-client>
- [62] „Configuring Chroma Collections“, Chroma. Zugriffen: 8. Dezember 2025. [Online]. Verfügbar unter: <https://docs.trychroma.com/docs/collections/configure#configuring-chroma-collections>
- [63] „Introduction“, Ragas. Zugriffen: 19. November 2025. [Online]. Verfügbar unter: <https://docs.ragas.io/en/stable/>
- [64] „List of available metrics“, Ragas. Zugriffen: 21. November 2025. [Online]. Verfügbar unter: https://docs.ragas.io/en/stable/concepts/metrics/available_metrics/

- [65] „google-bert/bert-base-multilingual-cased“, Hugging Face. Zugegriffen: 4. Dezember 2025. [Online]. Verfügbar unter: <https://huggingface.co/google-bert/bert-base-multilingual-cased>
- [66] „ROUGE Score“, Ragas. Zugegriffen: 19. November 2025. [Online]. Verfügbar unter: https://docs.ragas.io/en/stable/concepts/metrics/available_metrics/traditional/?h=#rouge-score
- [67] „BERTScore Tutorial“, GitHub. Zugegriffen: 10. Januar 2026. [Online]. Verfügbar unter: https://github.com/Tiiiger/bert_score/blob/master/example/Demo.ipynb
- [68] L. Eichhorn und B. Horneffer, „Arithmetisches Mittel“, destatis. Zugegriffen: 2. Februar 2026. [Online]. Verfügbar unter: https://service.destatis.de/eLearning/modul14/lm_pg_796.html
- [69] „LlamaIndex“, Ragas. Zugegriffen: 2. Februar 2026. [Online]. Verfügbar unter: https://docs.ragas.io/en/stable/howtos/integrations/_llamaindex/
- [70] „pandas documentation“, pandas. Zugegriffen: 21. November 2025. [Online]. Verfügbar unter: <https://pandas.pydata.org/docs/index.html>

Anhang

Eigenständigkeitserklärung

Hiermit versichere ich, dass ich die vorliegende Bachelorarbeit mit dem Titel:

Retrieval Augmented Generation mit Website-Kontext: Analyse der Auswirkung von zusätzlichem Website-Kontext auf die Antwortqualität

selbständig und nur mit den angegebenen Hilfsmitteln verfasst habe. Alle Passagen, die ich wörtlich aus der Literatur oder aus anderen Quellen wie z. B. Internetseiten übernommen habe, habe ich deutlich als Zitat mit Angabe der Quelle kenntlich gemacht.

04.02.2026

Datum

Carsten Maßbaum

Unterschrift