

Bachelor Thesis (173312)
Angewandte Informatik (SPO1a)

Einsatz von Künstlicher Intelligenz zur Testfallgenerierung

Marvin Müller*

16. Januar 2025

Eingereicht bei Prof. Dr. rer. nat. Nicole Ondrusch

*212117, mmueller3@stud.hs-heilbronn.de

Inhaltsverzeichnis

Abkürzungsverzeichnis	IV
Abbildungsverzeichnis	V
Gender Hinweis	VI
Abstract	VII
Zusammenfassung	VIII
1 Einleitung	1
1.1 Motivation	1
1.2 Forschungsfrage	2
1.3 Ziel der Arbeit	3
1.4 Vorgehensweise	3
2 Grundlagen und Hintergrund	4
2.1 KI	4
2.1.1 Allgemein	4
2.1.2 Bereiche der KI	5
2.2 LLM	7
2.2.1 Definition LLM	7
2.2.2 Entwicklungen im Bereich LLM	7
2.3 Prompting	7
2.3.1 Definition Prompting	7
2.3.2 Prompt Engineering und Prompt Techniken	7
2.4 RAG	7
2.4.1 Definition RAG	7
2.4.2 Einsatzmöglichkeiten von RAG	9
2.5 Testfall	9
2.5.1 Definition Testfall	9
2.5.2 Aufbau eines Testfalls	9
2.6 Anforderungsspezifikation	10
2.6.1 Definition Anforderungsspezifikation	10
2.6.2 Bedeutung der Anforderungsspezifikation für die Testfallgenerierung	11
3 Related Work und aktuelle Forschung	13
4 Methodik TODO Vorläufig muss überarbeitet werden!!!!	15
4.1 Verwendetes LLM	16
4.2 Testdaten	17
4.3 Prompt und verwendete Methoden	17
4.4 Qualitätskriterien	17
5 Ergebnisse	18
5.1 Vorliegende Ergebnisse	18

5.2	Auswertung der Ergebnisse	18
5.3	Diskussion	18
6	Fazit und Ausblick	19
	Literatur	20
	Eidesstattliche Erklärung	23
	Anhang	24

Abkürzungsverzeichnis

AI: Artificial Intelligence

GUI: Graphical User Interface (Deutsch: Grafische Benutzeroberfläche)

LLM: Large Language Model (Deutsch: Großes Sprachmodell)

KI: Künstliche Intelligenz

RAG: Retrieval Augmented Generation

Abbildungsverzeichnis

2.1	Interaktion eines KI-Systems mit seiner Umwelt (Hecker et al., 2018) . . .	4
2.2	Die Beziehung zwischen KI, maschinellem Lernen, neuronalen Netzen und Deep Learning (Mocko und Bitton-Bailey, 2021)	5
2.3	Deep Learning Network Grafik (Dymatrix, 2018)	6
2.4	Wie funktioniert Retrieval Augmented Generation (RAG) im Detail (Honroth et al., 2024)	8
2.5	Testfall im SAP Solution Manager (Enderli, 2019)	10
2.6	Eigenschaften von funktionalen vs nichtfunktionalen Requirements (Jama Software, 2024)	11
4.1	Eine Anforderungsspezifikation mit Informationen und Anweisungsschritten	15
4.2	Aus der Anforderungsspezifikation erzeugten manuelle Testfälle	16

Gender Hinweis

In dieser Bachelorarbeit aus Gründen der besseren Lesbarkeit das generische Maskulinum verwendet. Dabei gilt die männliche Sprachform für alle Geschlechter.

Abstract

This bachelor thesis investigates the use of artificial intelligence in the creation of requirement test cases, taking into account the requirement specifications provided by artificial intelligence. The aim of the work and the evaluation of the results was to find out whether artificial intelligence is able to generate requirement test cases that achieve a higher test quality and test coverage than requirement test cases created by humans/developers. The same requirement specifications were used and the test cases were compared with each other. The intention was to contribute to the improvement of test case generation.

The results of this bachelor thesis show that

Keywords: AI, LLM, RAG, Prompting, Test Case Generation

Zusammenfassung

Diese hier vorliegende Bachelorarbeit erforscht den Einsatz von Künstlicher Intelligenz bei der Erstellung von Anforderungstestfällen unter Berücksichtigung der Künstlichen Intelligenz zur Verfügung gestellten Anforderungsspezifikationen. Ziel der Arbeit und der Auswertung der Ergebnisse war es herauszufinden, ob Künstliche Intelligenz dazu in der Lage ist, Anforderungstestfälle zu erzeugen, die eine höhere Testqualität und Testabdeckung erreichen als von Menschen/Entwicklern erstellte Anforderungstestfälle. Dabei wurden die gleichen Anforderungsspezifikationen verwendet und die Testfälle miteinander verglichen. Intention war es, einen Beitrag zur Verbesserung der Testfallgenerierung zu leisten.

Die Ergebnisse dieser Bachelorarbeit zeigen, dass

Stichwörter: KI, LLM, RAG, Prompting, Testfallgenerierung

1 Einleitung

Über die Motivation und das Ziel der Bachelorarbeit wird in Kapitel 1 Einleitung eingegangen.

1.1 Motivation

In den vergangenen Jahren hat in dem Bereich Künstliche Intelligenz (KI) eine rasante Entwicklung stattgefunden. Diese hat dabei eine Auswirkung auf das private als auch auf das geschäftliche Leben. Der Artikel des Branchenverbandes der deutschen Informations- und Telekommunikationsbranche - Bitkom - führt die von ihnen durchgeführte Studie auf (Streim und Hecker, 2023). Nach dieser hat sich der Anteil der deutschen Unternehmen, welche KI nutzen, von 9% im Jahr 2022 auf 15% im Jahr 2023 nahezu verdoppelt. Dabei sehen 68% der Befragten KI als Chance.

In einer Pressemitteilung vom 25. November 2024 gibt das Statistische Bundesamt bekannt, dass bereits 20% der in Deutschland ansässigen Unternehmen KI verwenden (Statistisches Bundesamt, 2024). Dabei sind die drei häufigsten Einsatzbereiche für KI Textmining - Analyse geschriebener Sprache mit 48%, Spracherkennung mit 47% und die Erzeugung von natürlicher Sprache mit 34%. Dies zeigt eine rasante Steigerung der Nutzung von KI alleine in den letzten drei Jahren und es lässt sich erkennen, dass KI bereits vielfältig einsetzbar und nutzbar ist. Gerade in der Informatik und der Softwareentwicklung.

In dieser Arbeit stehen KI Sprachmodelle im Vordergrund. Diese sogenannten Large Language Models (LLM) bieten sich dafür an, redundante und sich ähnliche Aufgaben für den Anwender zu erledigen (Kerner, 2024). Gut geeignet ist dafür die Textgenerierung durch LLM. LLM werden deshalb bereits in verschiedensten Bereichen der Textgenerierung verwendet. Zu nennen sind dabei Übersetzung, Texterstellung, Textzusammenfassung, Kategorisierung und Klassifizierung. Eine Anwendungsmöglichkeit der Textgenerierung ist deshalb die Testfallgenerierung durch die KI. Das Erstellen von Testfällen mithilfe von KI kann erhebliche Zeit- und Kosteneinsparungen für Unternehmen und Tester ermöglichen.

Ein Testfall hat eine Struktur, welche definierten Regeln entspricht. In der Softwareentwicklung, aber auch allgemein in fast allen beruflichen und wissenschaftlichen Bereichen wird getestet. Somit ist die Testfallgenerierung durch KI nahezu universell einsetzbar. Es gibt bereits KI Tools und Arbeiten, welche sich mit dem Thema KI erzeugte Testfälle beschäftigt haben. Zu nennen sind dabei die Werke von (Bozic, 2022) und (Weingartz und Suleymanov, 2024). Vorteile von KI generierten Testfälle sind dabei eine gesteigerte Qualität und Kosten und Zeiteinsparungen.

In einer Arbeit wird die Wirksamkeit von LLM bei der Erstellung von Unit Tests untersucht (Ouédraogo et al., 2024). Dabei kommen die Autoren zu dem Schluss, dass die mit LLM erzeugten Junit Tests den von Menschen geschriebenen Tests in Bezug auf Codierungsstandards sehr ähneln.

Um das Thema des automatischen GUI Testings beschäftigt sich eine weitere Arbeit (Liu et al., 2024). Dabei sehen sie einen großen Fortschritt im Bereich des automatischen GUI Testing, kommen jedoch zu der Erkenntnis, dass von LLM erzeugte Tests noch eine geringe Aktivitätsabdeckung aufweisen.

Wie bei allen neuen Technologien ist es essenziell, sich auch der damit verbundenen Risiken bewusst zu sein. Ein Nachteil der KI ist zum Beispiel das Phänomen der Halluzination. Dieser Begriff beschreibt das Phänomen, dass eine KI Antwort, nicht mit den gegebenen Input übereinstimmt und faktisch nicht richtig ist (Siebert, 2024).

Ebenfalls ist der Umgang mit Daten nicht sicher. Ein mögliches Problem sind unternehmensinterne und auch persönliche Daten. Es gibt keine Transparenz darüber, wie diese behandelt werden, wenn sie einem LLM zur Verfügung gestellt werden (Möllers, 2024). Persönliche Daten könnten missbraucht werden und LLM könnten mit unternehmensinternen Daten antrainiert werden, welche die Konkurrenz dann nutzen könnte. Daher stellt sich die Frage, ob es sinnvoller ist, ein eigenes Modell anzutrainieren, um die Sicherheit und Kontrolle über die Daten zu gewährleisten. Eine empfohlene Maßnahme für dieses Risiko ist die Anonymisierung persönlicher und sensibler Daten, bevor man diese der LLM zur Verfügung stellt.

Da die im letzten Absatz beschriebene Lösung mit Zeit und Aufwand verbunden ist, bieten sich andere Methoden an und Modelle, welche Retrieval Augmented Generation (RAG) verwenden können, können sich gut für die Aufgabe der Testfallgenerierung eignen. Die Technik RAG beschreibt den Vorgang, dass das Wissen, welches das LLM benötigt, nicht aus dem Prompt - der Eingabe - kommen muss (Honroth et al., 2024). Informationen aus Dateien, die dem LLM zur Verfügung gestellt werden, können ebenfalls verwendet werden.

1.2 Forschungsfrage

Folgende Forschungsfrage soll untersucht werden und dient als Grundlage zur Durchführung der Arbeit. Sie ist dabei aus der Motivation entstanden.

1. Ist es möglich, dass Large Language Models dazu in der Lage sind, aus Anforderungsspezifikationen Anforderungstestfälle zu erzeugen, welche eine höhere Testqualität und Testabdeckung aufweisen, als manuell erstellte Testfälle, die aus den gleichen Anforderungsspezifikationen erstellt werden?

1.3 Ziel der Arbeit

Ziel der Arbeit ist es, die Grundlagen und Anwendungsmöglichkeiten und -verfahren zu erforschen, durchzuführen und zu evaluieren.

Diese Bachelorarbeit soll ein Konzept und eine Vorgehensweise entwickeln, die sich für den Einsatz von KI für die Generierung von Anforderungstestfällen aus der KI zur Verfügung gestellten Anforderungsspezifikationen eignet. Diese stützt sich dabei auf bereits bestehender Vorgehensweisen und Modelle. Durch die theoretische und praktische Aufarbeitung, Erprobung und Verbesserung bestehender Vorgehensweisen und Modellen soll sie dabei ein Beitrag sein, die Effizienz und Qualität der mit Hilfe von KI erstellten Testfälle zu verbessern.

1.4 Vorgehensweise

Der Plan der vorliegenden Arbeit gestaltet sich wie folgt: Zuerst wird eine Literaturrecherche durchgeführt. Dabei liegt der Fokus auf das Finden von bereits vorhandener Vorgehensweisen und Modellen im Bezug auf Textgenerierung im Allgemeinen und Testfallgenerierung im Besonderen. Es ist ebenfalls notwendig, sich über Begriffe und Techniken, welche während der Umsetzung der Arbeit notwendig sind, tiefer zu ergründen. Wichtig ist dabei zu ergründen, wie ein Testfall und eine Anforderungsspezifikation aufgebaut sind, welche Merkmale und welche Kriterien sie erfüllen müssen. Es muss erforscht werden, was eine KI ist, wie Modelle - LLM - aufgebaut sind und welche sich zur Testfallgenerierung eignen. Das Ermitteln geeigneter Prompting Methoden wie der Methode RAG ist ebenfalls erforderlich. Das Ziel ist dabei, die Untersuchung und Entdeckung bereits vorhandener Anwendungsmöglichkeiten, Vorgehensweisen und Modellen zur KI gesteuerten Testfallgenerierung.

Nachdem passende Modelle und Vorgehen ermittelt wurden und passende Testdaten, welche sich in Anforderungsspezifikationen und aus diesen manuell erstellte Anforderungstestfälle gliedern, beginnt die praktische Umsetzung der Arbeit. Dabei liegt die Durchführung und Dokumentation der von der KI erstellten Anforderungstestfälle im Vordergrund.

Zum Schluss werden die von der KI erstellten Anforderungstestfälle mit manuell erstellten Anforderungstestfälle verglichen und auf ihr Abschneiden im Bezug auf Testqualität und Testabdeckung ermittelt. Die Ergebnisse werden analysiert und hinsichtlich Aufwand, Durchführbarkeit und Richtigkeit ausgewertet. Dabei wird für die Auswertung Metriken verwendet, die in Kapitel 4 Methodik näher erläutert werden. Nachdem die Ergebnisse ausgewertet werden, gibt es eine Einordnung der Ergebnisse und eine Diskussion über mögliche zukünftige Entwicklungen in dem Bereich KI generierte Testfallerstellung.

2 Grundlagen und Hintergrund

Das Kapitel 2 Grundlagen und Hintergrund widmet sich den wissenschaftlichen Grundlagen und klärt über Begriffe und Methoden auf, die für das Verstehen und das Durchführen der Arbeit essenziell sind.

2.1 KI

Diese Arbeit behandelt das Thema KI, deshalb ist es essentiell hier einen allgemeinen Überblick über dieses Thema zu geben.

2.1.1 Allgemein

Der Begriff KI beziehungsweise im Englischen AI beschreibt ein Teilgebiet der Informatik. Dabei gibt es zwei Arten von KI. Erstens die schwache KI, welche auch als enge KI bezeichnet wird. Diese Art der KI kann nur einzelne und beschränkte Aufgaben lösen. Der Begriff KI wurde bereits im Jahr 1956 als Konzept vorgeschlagen. Seitdem entwickelte sie sich immer weiter. Im Jahr 1997 verliert der damalige Schachweltmeister Garry Kasparov gegen den Schachcomputer Deep Blue von IBM und im Jahr 2016 besiegt der Computer AlphaGo von Google Lee Se-dol, ein Go-Meister aus Südkorea. Diese Erfolge zeigen die Entwicklung der KI, sind jedoch auf die schwache KI beschränkt (Bhatt et al., 2021), (Hecker et al., 2018), (Mocko und Bitton-Bailey, 2021), (Roscher und Guderitz, 2025) und (Zhu et al., 2021). Als Zweites gibt es das Konzept der starken KI, auch als allgemeine KI bekannt. Folgende Grafik zeigt, wie diese ihre Umgebung wahrnimmt.

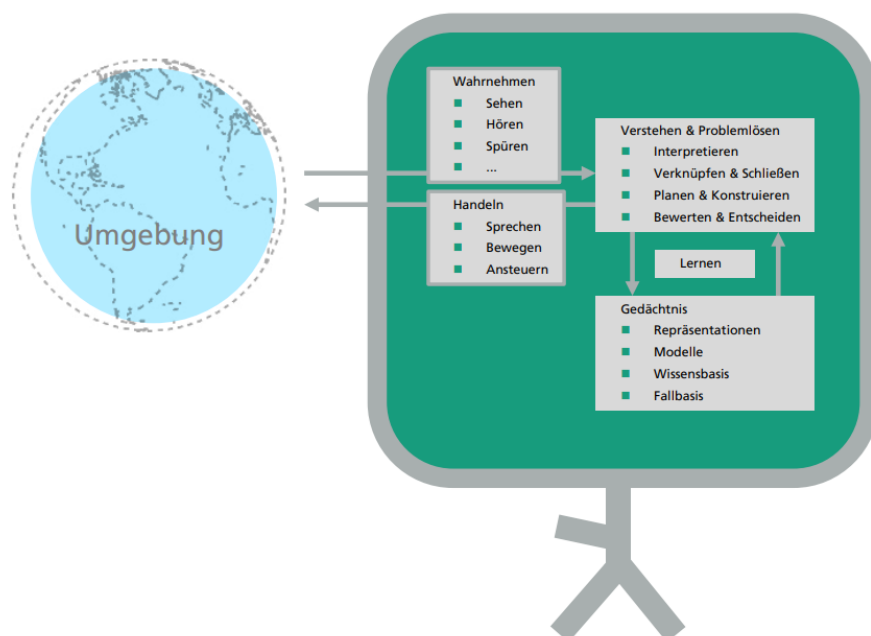


Abbildung 2.1: Interaktion eines KI-Systems mit seiner Umwelt (Hecker et al., 2018)

Das Konzept hinter dieser KI besteht darin, dass Maschinen Fähigkeiten verwenden beziehungsweise imitieren können, die menschlich sind.

Laut dem Fraunhofer Institut bilden sie kognitive Fähigkeiten ab. Es scheint dabei, dass die Maschinen sich selbst und ihre Umgebung wahrnehmen und interpretieren können. Ihre Wahrnehmung und ihre Interpretation speist sich dabei auf Daten, Fakten, konkreten Modellen und Regeln, welche ihr nächstes Handeln bestimmen. Während des Nutzens lernen die Maschinen dabei kontinuierlich weiter. Sie besitzen ein Gedächtnis. Es gibt noch eine dritte Form der KI, die sogenannte Superintelligenz, diese wird es jedoch in nächster Zeit nicht geben.

2.1.2 Bereiche der KI

Nachdem erklärt wurde um, was es sich bei KI handelt, ist es nun wichtig zu erklären, welche Bereiche und Teilgebiete es von KI gibt. Zu nennen sind dabei maschinelles Lernen, Neuronale Netze und Deep Learning (Bhatt et al., 2021), (Hecker et al., 2018), (Mocko und Bitton-Bailey, 2021), (Roscher und Guderitz, 2025) und (Zhu et al., 2021). Folgende Grafik zeigt anschaulich die einzelnen Teilgebiete der KI.

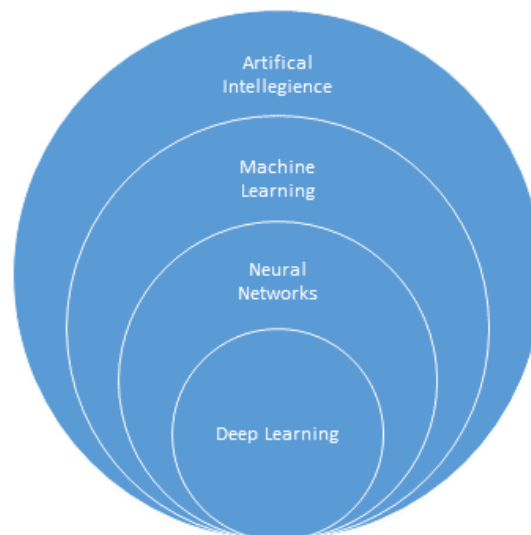


Abbildung 2.2: Die Beziehung zwischen KI, maschinellem Lernen, neuronalen Netzen und Deep Learning (Mocko und Bitton-Bailey, 2021)

KI beziehungsweise AI wird bereits in Kapitel 2.1.1 beschrieben und umfasst die anderen drei Bereiche. Die Grafik ist dabei so zu verstehen, dass der größere Kreis die jeweils kleineren Kreise umfasst, das heißt maschinelles Lernen ist ein Teil von KI, Neuronale Netze sind ein Teilgebiet von KI und von maschinellem Lernen und Deep Learning ist ein Teilgebiet von KI, von maschinellem Lernen und von Neuronalen Netzen. Im Folgenden werden nun die einzelnen Bereiche der KI erläutert.

Maschinelles Lernen:

Maschinelles Lernen auf Englisch Machine Learning ist ein Teilgebiet der KI. Es beschreibt das Verfahren, in welchem Maschinen durch das Wiederholen von Aufgaben lernen. Dabei gibt es Algorithmen, welche mit Daten gefüttert werden und der Lernprozess ist eine Schleife, in welcher dieser wiederholt wird. Dadurch lernt die Maschine die Aufgabe zu lösen. Dies geschieht, in dem es Feedback erhält und so lernt, wie es die Aufgabe richtig löst. Es wird dabei kein Lösungsweg vorgegeben wie bei herkömmlichen Algorithmen, sondern die Maschine lernt durch Versuch und Irrtum beziehungsweise auf Englisch durch Trial and Error.

Neuronale Netze:

Bei neuronalen Netzen handelt es sich um ein Teilgebiet des maschinellen Lernens. Idee der neuronalen Netze ist es, menschliche Neuronen nachzubilden und somit menschliches Denken nachzuahmen. Ein neuronales Netz besteht dabei aus Knoten die aneinander gereiht sind. Sie haben eine Eingabeschicht, ein Hidden Layer welches aus einer Neuronenschicht besteht und eine Ausgabeschicht die miteinander vernetzt sind. Sie trainieren ebenfalls in Schleifen und Durchgängen und werden durch das wiederholen besser und schneller.

Deep Learning:

Deep Learning ist ein Teilgebiet der Neuronalen Netze. Sie sind dadurch gekennzeichnet, sie ein Hidden Layer besitzen, welches aus mehr als einer Neuronenschicht besteht (Bhatt et al., 2021), (Dymatrix, 2018) und (Zhu et al., 2021). Folgende Grafik zeigt anschaulich ein Neuronales Netz und ein Deep Learning Netz.

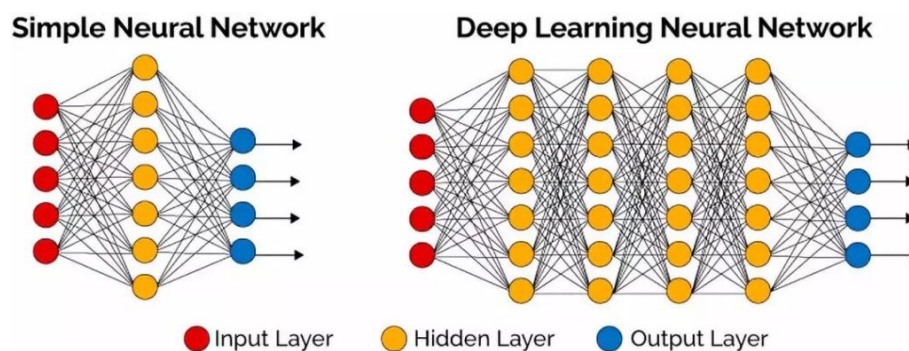


Abbildung 2.3: Deep Learning Network Grafik (Dymatrix, 2018)

Sie können dabei Millionen Neuronenschichten beinhalten und dadurch immer komplexere Aufgaben lösen.

Dies ist eine kurze Abhandlung über das Thema KI (Bhatt et al., 2021), (Hecker et al., 2018), (Mocko und Bitton-Bailey, 2021), (Roscher und Guderitz, 2025) und (Zhu et al., 2021). Weitere wichtige KI-Bereiche wie LLM, Prompting und RAG werden in den Kapiteln 2.2 LLM, 2.3 Prompting und im Kapitel 2.4 erläutert, da diese eine wichtige Rolle in dieser Arbeit spielen.

2.2 LLM

In dieser Arbeit wird/werden LLM(s) verwendet, über ihre Eigenschaften und Entwicklungen klärt dieses Kapitel auf

2.2.1 Definition LLM

Unter dem Begriff LLM versteht man folgendes: TODO QUELLEN finden ausschreiben

2.2.2 Entwicklungen im Bereich LLM

Im Bereich LLM gibt es schnelllebige Entwicklungen.

Viele LLM

Hugging Face

Eigene LLM entwickeln antrainieren

TODO QUELLEN finden ausschreiben

2.3 Prompting

Die Eingabe bzw. der Prompt ist ein wesentlicher Bestandteil, zum Nutzen von KI. Deshalb ist es sehr wichtig diesen Begriff und die Vorgehensweisen, welche in dieser Arbeit getätigt werden zu erklären.

2.3.1 Definition Prompting

Unter dem Begriff Prompting versteht man folgendes: TODO QUELLEN finden ausschreiben

2.3.2 Prompt Engineering und Prompt Techniken

Prompting ist wichtig im Umgang mit KI.

Prompt Engineering

Es gibt viele Prompt Techniken dabei bieten sich in dieser Arbeit folgende an um Testfälle zu generieren. TODO QUELLEN finden ausschreiben

2.4 RAG

Bereits in Kapitel 1 Motivation wird RAG erwähnt. RAG ist für diese Arbeit von besonderer Wichtigkeit.

2.4.1 Definition RAG

RAG ist eine Technik, in welcher einem LLM Dokumente zur Verfügung gestellt werden. Dabei werden die in den Dokumenten enthaltenen Informationen für das LLM nutzbar gemacht (Gao et al., 2024), (Honroth et al., 2024), (Lewis et al., 2021), (Miesle, 2023), (Salemi und Zamani, 2024), (Schmid und Sommer, 2024) und (Wu et al., 2024). Die Dokumente und Anfragen werden dabei in kleine Abschnitte, welche als Chunks bezeichnet werden, aufgeteilt. RAG lässt sich dabei in sehr vielen LLM anwenden und optimiert

diese, ohne dass diese mit viel Aufwand trainiert werden müssen. Folgende Grafik zeigt anschaulich, wie RAG funktioniert:

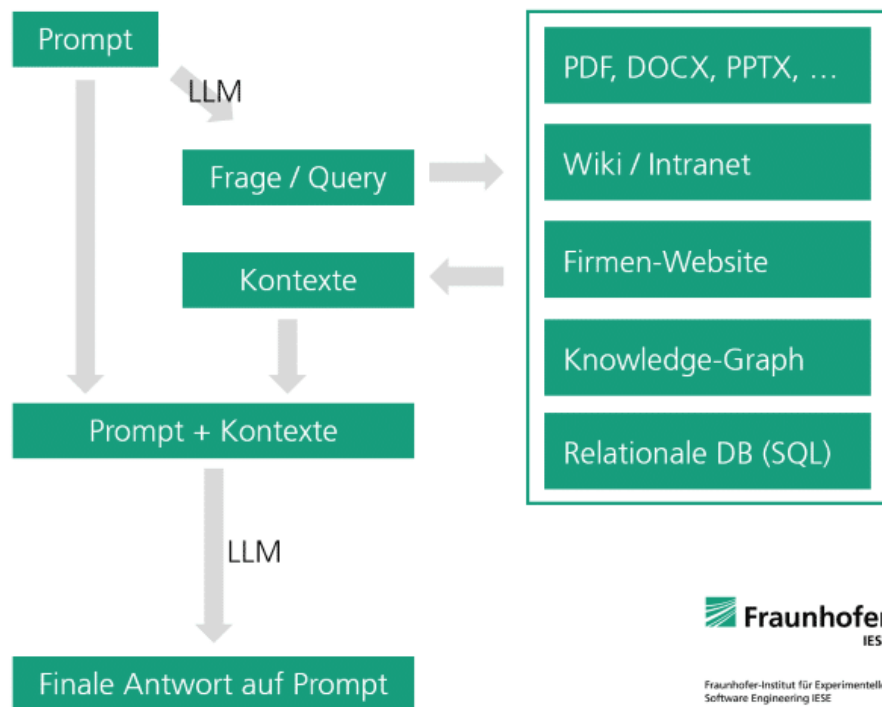


Abbildung 2.4: Wie funktioniert Retrieval Augmented Generation (RAG) im Detail (Honroth et al., 2024)

Nun werden die einzelnen Schritte von RAG erklärt (Gao et al., 2024), (Honroth et al., 2024), (Lewis et al., 2021), (Miesle, 2023), (Salemi und Zamani, 2024), (Schmid und Sommer, 2024) und (Wu et al., 2024):

Indexierung:

Die Indexierung beschreibt den Prozess, wie die Daten aus einem Dokument in kürzere Abschnitte den Chunks aufgeteilt werden, um die Daten für die LLM verwertbar zu machen. Die Chunks werden maschinenlesbar gemacht und in einer Vektordatenbank gespeichert, aus welcher sie nun für die LLM zugreifbar sind. Sie werden vektorisiert.

Benutzereingabe:

Die Benutzereingabe erfolgt als Prompt in natürlicher Sprache, welche sich auf die im Dokument befindlichen Daten beziehen.

Retrieval:

Nachdem die Maschine die Eingabe erhalten hat und diese ebenfalls vektorisiert hat, wird mit der Methode des nächsten Nachbarn, wird die höchste Ähnlichkeit zwischen der Eingabe und der in der Vektordatenbank gespeicherten Chunks priorisiert.

Augmentation:

Der LLM wird diese Kombination aus Prompt des Benutzers und der Kontext zu Verfügung gestellt.

Generation:

Es wird nun eine Antwort generiert, welches die erhaltenen Informationen und das Training des jeweiligen LLM kombiniert.

Wissenschaftliche Werke kommen zu dem Schluss, dass durch RAG LLM bessere Ergebnisse liefern können (Gao et al., 2024), (Lewis et al., 2021), (Salemi und Zamani, 2024) und (Wu et al., 2024). Ergebnisse, welche RAG benutzen, zeichnen sich bei ihrer Performance aus und sind spezifischer und faktenreicher als Ergebnisse die nur von der LLM, ohne RAG erzeugt wurden.

2.4.2 Einsatzmöglichkeiten von RAG

RAG spielt in dieser Arbeit eine entscheidende Rolle. Mithilfe von RAG kann der LLM eine Anforderungsspezifikation zur Verfügung gestellt werden, aus welcher bereits manuell Anforderungstestfälle generiert wurden. Durch RAG kann die LLM diese Anforderungsspezifikation, welche eine PDF-Datei ist, gut verarbeiten und daraus Informationen ziehen. Dabei kann sie Informationen in Form von Text und von Bildern verarbeiten und berücksichtigen. Dies kann getrennt von einander geschehen, also PDF-Dateien sein, welche entweder nur Text oder Bilder enthalten, oder PDF-Dateien, welche Text und Bilder beinhalten.

2.5 Testfall

In dieser Arbeit geht es um die Generierung von Testfällen durch KI. Es ist wichtig zu erklären was ein Testfall ist und welche Merkmale und Kriterien ein Testfall hat.

2.5.1 Definition Testfall

Ein Testfall beschreibt folgendes.

Allgemein und auch Anforderungstestfall spezifisch TODO QUELLEN finden ausschreiben

2.5.2 Aufbau eines Testfalls

Folgende Graphik zeigt ein Testfalldesign für eine SAP Applikation:

Test Case Titel		Upload external catalog to Ariba Network			
Preconditions		Ariba network users with the required permissions Catalog file in the desired format			
Testobject		Ariba Catalog			
Goals for this Test Case		1. Functional goal: Successfully upload of a catalog file to Ariba Network. 2. Nonfunctional goal: The file can be uploaded within 10 seconds.			


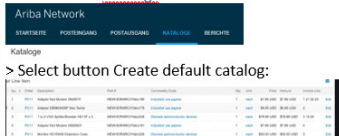

Step	Description (Perform actions and screenshots)	Executable (Transaction, Fiori, URL, etc.)	Possible test data	Expected result (Is required to detect defects)
01	Log in Log in to the system as an external supplier with <username> and <password>.	http://supplier.ariba.com		You will see "cockpit" home site. 
02	Go to Upload external catalog view Select menu <catalogs>  > Select button Create default catalog:			View «details» appears: 
03	Create a new catalog Enter <catalog name> Formulate a <description> Click <save>			Catalog has been successfully uploaded and saved. The status of the catalog is "published".

Abbildung 2.5: Testfall im SAP Solution Manager (Enderli, 2019)

Ein Testfall besitzt einen Namen, Vorbedingungen welche erfüllt sein müssen, damit der Testfall durchgeführt werden kann, Ziele die erreicht werden sollen. Der Testfall gliedert sich in Schritten, welche eine Beschreibung und ein erwartetes Ergebnis besitzen. In den Schritten wird somit überprüft, ob die Anwendung die Anforderungen erfüllt und somit das erwartete Ergebnis eintrifft. Testfälle können dabei positiv, als auch negativ sein. Bei negativen Testfällen wird überprüft ob das erwartete Ergebnis, wie erwartet nicht eintrifft, da die Voraussetzungen nicht erfüllt sind. In dieser Arbeit wird von der KI erwartet, dass es wenn es ein Testfall schreibt, diesen Aufbau einhält. Wenn sie es nicht macht, wird dies als negativ bewertet.

2.6 Anforderungsspezifikation

Eine Anforderungsspezifikation ist das Fundament, aus welcher Testfälle erstellt werden. Ein Testfall überprüft ob die Anforderungen aus dieser Spezifikation erfüllt werden.

2.6.1 Definition Anforderungsspezifikation

Eine Anforderungsspezifikation ist ein Dokument, welches alle Anforderungen beziehungsweise Requirements enthält, welche das System erfüllen muss und dessen Erfüllbarkeit und nicht Erfüllbarkeit mit Testfällen abgedeckt und getestet werden. Da sie der Grund für die Testfälle sind, müssen Anforderungsspezifikationen vollständig, präzise und in sich konsistent sein. Anforderungsspezifikationen enthalten System- und Benutzeranforderungen. Systemanforderungen beschreiben dabei die Benutzeranforderungen genau, wobei die Benutzeranforderungen funktionale und nicht funktionale Requirements besitzen. Folgende Grafik veranschaulicht anhand eines Beispiels, den Unterschied zwischen funktionalen und nicht funktionalen Requirements:

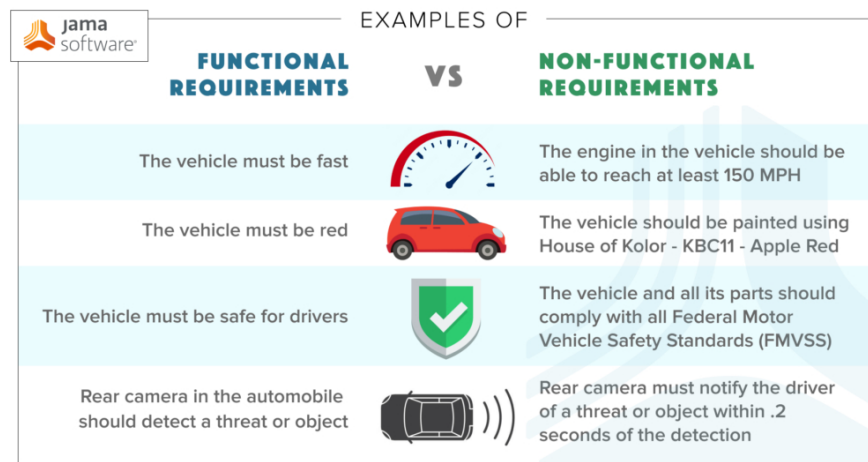


Abbildung 2.6: Eigenschaften von funktionalen vs nichtfunktionalen Requirements (Jama Software, 2024)

Nicht funktionale Requirements beschreiben zum Beispiel Sicherheit, Wartung und Systemanforderungen. In dieser Arbeit geht es um funktionale Requirements, diese legen fest, was wie das System funktioniert und was passiert, wenn etwas im Positiven als auch wenn etwas im Negativen passiert. Den Unterschied erkennt man gut an der Grafik. Während die funktionellen Requirements festlegen, dass das Auto schnell sein und eine rote Farbe besitzen muss, legt die nicht funktionalen Requirements exakt fest, welche Geschwindigkeit das Auto erreichen und welchen genauen Rotton dieses besitzen muss. Funktionale Requirements dienen als Grundlage für die Testfallgenerierung durch die LLM in dieser Arbeit (Aysolmaz et al., 2018), (Barmi et al., 2011), (Jama Software, 2024), (Mustafa et al., 2021) und (Visure, 2024).

2.6.2 Bedeutung der Anforderungsspezifikation für die Testfallgenerierung

In vielen Softwareprojekten werden Testfälle durch Requirements erzeugt (Aysolmaz et al., 2018), (Barmi et al., 2011), (Jama Software, 2024), (Mustafa et al., 2021) und (Visure, 2024). Anforderungsspezifikationen dienen als Grundlage für die Testfallgenerierung durch die LLM in dieser Arbeit. Sie werden der LLM per RAG übermittelt und per Prompt dazu aufgerufen, Testfälle zu generieren, welche diese Requirements in Positiv- und Negativfällen abdecken. Dies ist essenziell für diese Arbeit, da es ohne Requirements nicht möglich ist, die von der LLM generierten Testfälle, auszuwerten. Mit ihrer Hilfe kann nun jedoch präzise festgestellt werden, wie diese im Bereich Vollständigkeit, Qualität und Fehlerhäufigkeit vorkommen.

Vollständigkeit:

Werden alle Requirements, welche dem LLM übergeben werden, im positiven wie im negativen vollständig abgedeckt? Wenn Nein, wie viele werden ausgelassen?

Qualität:

Werden alle Requirements präzise wiedergegeben und vollständig getestet, oder gibt es Rechtschreibfehler oder eine falsche Anordnung der Reihenfolge, oder werden Requirements aufgeteilt und falsch dargestellt?

Fehlerhäufigkeit:

Wie viele Testschritte enthalten Informationen, welche sich nicht in der mitgegebenen Anforderungsspezifikation befinden und welche sind komplett frei erfunden und weisen keinen Zusammenhang mit dem System auf? So kann die Anzahl der Halluzinationen bestimmt und gezählt werden.

3 Related Work und aktuelle Forschung

Zum Thema KI im Allgemeinen und besonders im Hinblick auf Sprachmodelle zur Textgenerierung gibt es bereits viele wissenschaftliche Arbeiten (Iqbal und Qureshi, 2022), (Yuan et al., 2021), (Khan et al., 2024), (Bhandari et al., 2024), (Gu et al., 2024), (Chen et al., 2024) und (Wang und Zhu, 2024). Auf diese wird in diesem Kapitel näher eingegangen.

Für den Einstieg zu dem Thema eignet sich ein wissenschaftliches Paper, welches einen Überblick bietet (Iqbal und Qureshi, 2022). Dieses Paper stellt viele Deep Learning Modelle vor, die für die Generierung von Text verwendet wurden. Dabei werden die verschiedenen Modelle zusammengefasst und das Paper ermöglicht einen detaillierten Überblick über die Vergangenheit, Gegenwart und Zukunft von Textgenerierungsmodellen im Deep Learning. Die Autoren gehen davon aus, dass in Zukunft immer bessere Modelle entwickelt werden und dass die Forschung im Bereich der Texterzeugung weiter vorangetrieben wird. Das Paper macht jedoch auch deutlich, dass Modelle bei längeren Texten die natürliche Sprache noch nicht vollständig begriffen haben.

Nicht nur die Textgenerierung durch KI ist wichtig, sondern auch die Auswertung des generierten Textes (Yuan et al., 2021). Die Autoren stellten sich die Frage, wie man die generierten Texte nach Flüssigkeit, Genauigkeit und Effektivität beurteilen kann. Dafür entwickelten sie eine Metrik: Der sogenannte „BARTSCORE“ bewertet flexibel und unüberwacht aus unterschiedlichen Blickwinkeln. Unter anderem betrachtet er Informativität, Geläufigkeit oder Faktizität der generierten Texte. Die Autoren kommen zum Schluss, dass ihr Modell in 16 von 22 Settings besser abgeschnitten hat als andere gut bewertete Metriken.

Ziel einer dieser Arbeiten ist es, einen Überblick über die Rolle von KI bei der Automatisierung von Softwaretests zu geben (Khan et al., 2024). Die Autoren stellen die These auf, dass das Ziel der automatisierten Softwaretesterstellung mit KI möglich sein könnte. Im Werk werden zum Schluss entscheidende Schwierigkeiten, Probleme und Voraussetzungen vorgestellt. Am Ende bekräftigen die Autoren jedoch ihre These, dass das Testen von Software mit einer Reihe von Automatisierungsgadgets möglich sein könnte.

Eine weitere Arbeit beschäftigt sich mit dem Potenzial der Anpassung von LLMs für den Bereich des Chip Testens (Bhandari et al., 2024). Dabei sind funktionale Tests, die sich auf Testbenches stützen, wichtig für das Chipdesign. Es wird Feedback in die LLM eingegeben, um so die Testbench Generierung zu verbessern. Dies findet iterativ statt, um die Testabdeckung zu verbessern. Die Autoren nennen als Herausforderungen die Wiederholung der Antworten und die Feinabstimmung des LLM mit dem Umfang der Arbeit. Die Wertung der Testbenches in Bezug auf die Abdeckung gestaltet sich als nicht einfach. Trotzdem kommen sie zum Schluss, dass durch das Feedback die Testbenches besser verstanden werden. Ein entscheidender Faktor für die Qualität der Ausgabe ist dabei der Prompt.

Weitere Werke beschäftigen sich mit dem Einsatz von KI bei der Erstellung von Unit Tests (Gu et al., 2024). Die Autoren nennen drei mögliche Probleme bei der Erzeugung

von Testfällen. Ein unzureichender Kontext, fehlende Test- und Abdeckungsinformationen und das Problem, dass LLM sich in Wiederholungsschleifen befinden können. Als Lösung bieten sie das Tool TestART an, welches eine Erfolgsquote von 78,55% aufweist. In einer weiteren Arbeit wird ein Tool namens ChatUniTest vorgestellt, welches ebenfalls die Qualität von durch KI erzeugte Unit Tests verbessern soll (Chen et al., 2024).

Der aus Sprachmodellen generierte Code wirft Fragen im Bereich der Qualität und Korrektheit auf (Wang und Zhu, 2024). Die Autoren gehen davon aus, dass sie Fehler im Code aufspüren können, indem sie Inkonsistenzen finden. Sie kommen zu dem Ergebnis, dass sie in der Lage sind, 75% der von GPT-4 generierten fehlerhaften Programme zu erkennen, wobei sie eine Falsch Positiv Rate von 8,6% haben. Die Autoren kommen zu der Erkenntnis, dass Entwickler möglicherweise nur eine sehr kleine Teilmenge des LLM generierten Codes untersuchen müssen, um die meisten Fehler zu finden. Die in der Arbeit vorgestellte Methodik dient als Inspiration, wie diese Arbeit aufgebaut ist.

4 Methodik TODO Vorläufig muss überarbeitet werden!!!!

Als Grundlage zur Erstellung von Testfällen dienen Anforderungsspezifikationen. Wichtig ist dabei, dass man sowohl Anforderungsspezifikationen als auch manuell erstellte Testfälle, die aus diesen Anforderungsspezifikationen erstellt wurden, findet.

Step No.	Module	Type of Testing	Test Cases	Pre-Condition	Expected Result	Post-Condition	Actual Result	Test Data	Status
1	Compatibility Testing		Checking the URL in Different Browsers	Run the link in Different Browsers	Should run in different Browsers	Browsers should be display	Found as per expectation	Chrome Internet Explorer	Passed
2	UI Testing		Checking the color, size & text alignment of different elements should match with their requirement	Run the link	Should be as per the requirement	Run the link	Found as per expectation	N/A	Passed
3			Checking icons that uses in login page are perfect or not	Run the link	Should be perfect	Run the link	Found as per expectation	N/A	Passed
4			Checking the title of login page	Run the link	Should be correct	Run the link	Found as per expectation	N/A	Passed
5			Check the spelling error	Run the link	Should be correct	Run the link	Found as per expectation	N/A	Passed
6			Checking grammatical error	Run the link	No grammatical error	Run the link	Found as per expectation	N/A	Passed
7			Checking the login page is responsive & aligns properly on different screen and devices	Run the link	Should be run properly	Run the link	Found as per expectation	N/A	Passed
8			Check the Login button	Click on the "Login" button.	The login form should be displayed	Display the form	Found as per expectation	N/A	Passed
9			Enter a valid email & a valid password	Enter a valid email and password & click the Submit button.	Successful login	User should be able to see the login page	Successful login	Email:ankasir@hy 41321@gmail.com	Passed
10			Enter a valid email & invalid password	Enter a valid email and invalid password & click the Submit button.	It will show an error message "invalid email or password"	error message "invalid email or password"	Unsuccessful login	Email:ankasir@hy 41321@gmail.com	Failed
11			Enter an invalid email & valid password	Enter invalid email and valid password & click the Submit button	It will show an error message "invalid email or password"	error message "invalid email or password"	Unsuccessful login	Email:xxxxxxx Password:1234@.AB	Failed
12			Enter invalid email and invalid	Enter invalid email and invalid	It will show an error				

Abbildung 4.1: Eine Anforderungsspezifikation mit Informationen und Anweisungsschritten

Nur so können diese Testfälle mit den KI generierten Testfällen valide verglichen und ausgewertet werden. Eine Quelle für Daten können Opensource Projekte auf Github sein, welche zu finden sind.

Eine Anforderungsspezifikation enthält dabei Informationen und Anweisungsschritte, aus denen dann das LLM einen geeigneten Testfall generieren soll.

Test Scenario ID	Reference	Test Scenario Description	Priority	Number of Test Cases
TS_MA_001		Validate the URL in Different Browsers	P1	
TS_MA_002		Verify that the login functionality on the MobileAction website works as expected.	P1	
TS_MA_003		Validate the working of 'Contact' functionality	P1	
TS_MA_004		Validate the working of 'Go to SearchAds' functionality	P4	
TS_MA_005		Validate the working of 'Schedule a demo' functionality	P4	
TS_MA_006		Validate the working of 'products' functionality	P2	
TS_MA_007		Validate the working of 'Solutions' functionality	P2	

Abbildung 4.2: Aus der Anforderungsspezifikation erzeugten manuelle Testfälle

Nachdem die vorbereiteten Schritte abgeschlossen sind, beginnt der praktische Teil der Arbeit. Die Testfälle werden erzeugt und miteinander verglichen.

Für die Auswertung der von der KI erstellten Testfällen und den Vergleich zwischen ihnen und den manuell erstellten Testfällen dient das Werk von (Wang und Zhu, 2024) als Inspiration. Für die Evaluation der erstellten Testfälle gibt es eine Metrik mit vier Kategorien:

True Positives (TP)
False Positives (FP)
True Negatives (TN)
False Negatives (FN)

In dieser Arbeit wird ChatGPT verwendet.

TODO AUSFORMULIEREN WARUM CHATGPT UND WELCHES MODELL WELCHE STATISTIKEN

Wenn es LLM gibt, welche Testfälle generieren können, welche die Struktur und den Inhalt der Dokumente gut abdecken können, ist die Forschungsfrage positiv beantwortet. Am Ende werden die Ergebnisse kritisch eingeordnet und mögliche zukünftige Entwicklungen genannt.

4.1 Verwendetes LLM

Dieser Abschnitt beschreibt das verwendete LLM, welche Eigenschaften es besitzt und aus welchen Gründen es für diese Arbeit verwendet wird. TODO LLM beschreiben und Daten finden

4.2 Testdaten

Die Testdaten welche für die Durchführung der Testfälle verwendet werden, werden in diesem Abschnitt erläutert. TODO Testdaten beschreiben und zeigen

4.3 Prompt und verwendete Methoden

Der Plan der vorliegenden Arbeit gestaltet sich wie folgt: Zuerst wird eine. TODO Prompt zeigen und Methoden erklären

4.4 Qualitätskriterien

Der Output der KI wird mit folgenden Qualitätskriterien bewertet: TODO Qualitätskriterien finden, definieren und erläutern

5 Ergebnisse

In Kapitel 5 werden die Ergebnisse der von der KI erstellten Anforderungstestfälle dokumentiert, beschrieben und eingeordnet.

5.1 Vorliegende Ergebnisse

Dies sind die Ergebnisse der KI TODO Ergebnisse zeigen

5.2 Auswertung der Ergebnisse

Die von der KI generierten Testfälle schneiden wie folgt ab: TODO Ergebnisse auswerten und beschreiben

5.3 Diskussion

Der Grund warum die Testfälle so abschneiden wie sie abschneiden, kann an folgenden Punkten liegen: TODO Nach der Auswertung Ergebnisse einordnen und Grund dafür nennen

6 Fazit und Ausblick

Über die Einordnung der Ergebnisse dieser Arbeit und mögliche Bedeutungen für die Zukunft wird in Kapitel 6 Fazit und Ausblick Auskunft gegeben.

Literatur

- Aysolmaz, B., Leopold, H., Reijers, H. A., & Demirörs, O. (2018). A semi-automated approach for generating natural language requirements documents based on business process models. *Information and Software Technology*, 93, 14–29. <https://www.sciencedirect.com/science/article/abs/pii/S0950584917302069?via%3Dihub>
- Barmi, Z. A., Ebrahimi, A. H., & Feldt, R. (2011). Alignment of Requirements Specification and Testing: A Systematic Mapping Study. *2011 IEEE Fourth International Conference on Software Testing, Verification and Validation Workshops*. <https://ieeexplore.ieee.org/document/5954452>
- Bhandari, J., Knechtel, J., Narayanaswamy, R., Garg, S., & Karri, R. (2024). LLM-Aided Testbench Generation and Bug Detection for Finite-State Machines. *Computer Science > Hardware Architecture*, 1. <https://arxiv.org/abs/2406.17132>
- Bhatt, C., Kumar, I., Vijayakumar, V., Singh, K. U., & Kumar, A. (2021). The state of the art of deep learning models in medical science and their challenges. *Multimedia Systems*, 4, 599–613. <https://link.springer.com/article/10.1007/s00530-020-00694-1>
- Bozic, S. (2022). *Künstliche Intelligenz im Testprozess* [abgerufen am 03.11.2024]. <https://bbv-software.de/insights/blog/ai-testing/>
- Chen, Y., Hu, Z., Zhi, C., Han, J., Deng, S., & Yin, J. (2024). ChatUniTest: A Framework for LLM-Based Test Generation. *FSE 2024: Companion Proceedings of the 32nd ACM International Conference on the Foundations of Software Engineering*, 572–576. <https://dl.acm.org/doi/10.1145/3663529.3663801>
- Dymatrix. (2018). *Deep-Learning im Digital Business* [abgerufen am 13.1.2025]. <https://www.dymatrix.de/dymatrixblog/deep-learning-definition>
- Enderli, D. (2019). *Testfalldesign für SAP Applikationen* [abgerufen am 16.1.2025]. <https://community.sap.com/t5/technology-blogs-by-members/testfalldesign-f%C3%BCr-sap-applikationen/ba-p/13451295>
- Gao, Y., Xiong, Y., Gao, X., Jia, K., Pan, J., Bi, Y., Dai, Y., Sun, J., Wang, M., & Wang, H. (2024). Retrieval-Augmented Generation for Large Language Models: A Survey. *Computer Science > Computation and Language*, 5. <https://arxiv.org/abs/2312.10997>
- Gu, S., Fang, C., Zhang, Q., Tian, F., Zhou, J., & Chen, Z. (2024). TestART: Improving LLM-based Unit Test via Co-evolution of Automated Generation and Repair Iteration. *Computer Science > Software Engineering*, 5. <https://arxiv.org/abs/2408.03095>
- Hecker, D., Döbel, I., Petersen, U., Rauschert, A., Schmitz, V., & Voss, A. (2018). Zukunftsmarkt Künstliche Intelligenz. <https://kmu-digital.eu/de/service-kompetenz/publikationen/dokumente/studien/198-zukunftsmarkt-kuenstliche-intelligenz-potenziale-und-anwendungen>
- Honroth, T., Siebert, J., & Kelbert, P. (2024). *Retrieval Augmented Generation (RAG): Chatten mit den eigenen Daten* [abgerufen am 13.1.2025]. <https://www.iese.fraunhofer.de/blog/retrieval-augmented-generation-rag/>
- Iqbal, T., & Qureshi, S. (2022). The survey: Text generation models in deep learning. *Journal of King Saud University – Computer and Information Sciences*, 34(6), 2515–2528. <https://doi.org/10.1016/j.jksuci.2020.04.001>

- Jama Software. (2024). *Functional vs. nonfunctional requirements: What's the difference?* [abgerufen am 16.1.2025]. <https://www.jamasoftware.com/requirements-management-guide/writing-requirements/functional-vs-non-functional-requirements>
- Kerner, S. M. (2024). *Large Language Model (LLM)* [abgerufen am 03.11.2024]. <https://www.computerweekly.com/de/definition/Large-Language-Model-LLM>
- Khan, F. I., Mahmud, F. U., & Hoseen, A. (2024). A New Approach of Software Test Automation Using AI. *Journal of Basic Science and Engineering*, 21(2), 559–570. https://www.researchgate.net/publication/380459206_A_NEW_APPROACH_OF_SOFTWARE_TEST_AUTOMATION_USING_AI
- Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W., Rocktäschel, T., Riedel, S., & Kiela, D. (2021). Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. *Computer Science > Computation and Language*, 4. <https://arxiv.org/abs/2005.11401>
- Liu, Z., Chen, C., Wang, J., Chen, M., Wu, B., Che, X., Wang, D., & Wang, Q. (2024). Make LLM a Testing Expert: Bringing Human-like Interaction to Mobile GUI Testing via Functionality-aware Decisions. *ICSE '24: Proceedings of the IEEE/ACM 46th International Conference on Software Engineering*, 46(100), 1–13. <https://dl.acm.org/doi/10.1145/3597503.3639180>
- Miesle, P. (2023). *Retrieval-Augmented Generation (RAG) Explained: Understanding Key Concepts* [abgerufen am 13.1.2025]. <https://www.datastax.com/guides/what-is-retrieval-augmented-generation>
- Mocko, M., & Bitton-Bailey, A. (2021). *Areas of Artificial Intelligence* [abgerufen am 13.1.2025]. <https://ufl.pb.unizin.org/artificialintelligence/chapter/six-area-of-artificial-intelligence/>
- Möllers, N. (2024). *Künstliche Intelligenz und Datenschutz* [abgerufen am 03.11.2024]. <https://keyed.de/blog/kuenstliche-intelligenz-und-datenschutz/>
- Mustafa, A., Wan-Kadir, W. M. N., Ibrahim, N., Shah, M. A., Younas, M., Khan, A., Zareei, M., & Alanazi, F. (2021). Automated Test Case Generation from Requirements: A Systematic Literature Review. *Computers, Materials Continua*, 67, 1819–1833. <https://www.techscience.com/cmc/v67n2/41326>
- Ouédraogo, W. C., Kaboré, K., Tian, H., Song, Y., Koyoncu, A., Klein, J., Lo, D., & Bissyandé, T. F. (2024). Large-scale, Independent and Comprehensive study of the power of LLMs for test case generation. *Computer Science > Software Engineering*, 1(1). <https://arxiv.org/abs/2407.00225>
- Roscher, K., & Guderitz, A. (2025). *Künstliche Intelligenz (KI) und maschinelles Lernen* [abgerufen am 13.1.2025]. <https://www.iks.fraunhofer.de/de/themen/kuenstliche-intelligenz.html>
- Salemi, A., & Zamani, H. (2024). Evaluating Retrieval Quality in Retrieval-Augmented Generation. *SIGIR '24: Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2395–2400. <https://dl.acm.org/doi/10.1145/3626772.3657957>
- Schmid, A., & Sommer, M. (2024). *Retrieval Augmented Generation – RAG: Generative künstliche Intelligenz-Lösungen mit Unternehmensdaten anreichern* [abgerufen am 13.1.2025]. <https://aracom.de/retrieval-augmented-generation/>
- Siebert, J. (2024). *Halluzinationen von generativer KI und großen Sprachmodellen (LLMs)* [abgerufen am 22.11.2024]. <https://www.iese.fraunhofer.de/blog/halluzinationen-generative-ki-llm/>

- Statistisches Bundesamt. (2024). *Jedes fünfte Unternehmen nutzt künstliche Intelligenz* [abgerufen am 05.11.2024]. [https://www.destatis.de/DE/Presse/Pressemitteilungen/2024/11/PD24_444_52911.html#:~:text=444%20vom%2025.,November%202024&text=WIESBADEN%20%E2%80%93%20Jedes%20f%C3%BCnfte%20Unternehmen%20\(20,Einheiten%20mit%20mindestens%20zehn%20Besch%C3%A4ftigten.](https://www.destatis.de/DE/Presse/Pressemitteilungen/2024/11/PD24_444_52911.html#:~:text=444%20vom%2025.,November%202024&text=WIESBADEN%20%E2%80%93%20Jedes%20f%C3%BCnfte%20Unternehmen%20(20,Einheiten%20mit%20mindestens%20zehn%20Besch%C3%A4ftigten.)
- Streim, A., & Hecker, J. (2023). *Deutsche Wirtschaft drückt bei Künstlicher Intelligenz aufs Tempo* [abgerufen am 03.11.2024]. <https://www.bitkom.org/Presse/Presseinformation/Deutsche-Wirtschaft-drueckt-bei-Kuenstlicher-Intelligenz-aufs-Tempo>
- Visure. (2024). *Was ist Anforderungsspezifikation: Definition, beste Tools und Techniken* [abgerufen am 15.1.2025]. <https://visuresolutions.com/de/blog/requirements-specification/>
- Wang, X., & Zhu, D. (2024). Validating LLM-Generated Programs with Metamorphic Prompt Testing. *Computer Science > Software Engineering*, 1. <https://arxiv.org/abs/2406.06864>
- Weingartz, R., & Suleymanov, N. (2024). *Wie man bessere Tests mit KI schreibt* [abgerufen am 03.11.2024]. <https://aqua-cloud.io/de/ai-to-write-tests/>
- Wu, S., Xiong, Y., Cui, Y., Wu, H., Chen, C., Yuan, Y., Huang, L., Liu, X., Kuo, T., Guan, N., & Xue, C. J. (2024). Retrieval-Augmented Generation for Natural Language Processing: A Survey. *Computer Science > Computation and Language*, 2. <https://arxiv.org/abs/2407.13193>
- Yuan, W., Neubig, G., & Liu, P. (2021). BARTSCORE: Evaluating Generated Text as Text Generation. *Computer Science > Computation and Language*, 2. <https://arxiv.org/abs/2106.11520>
- Zhu, L., Spachos, P., Pensini, E., & Plataniotis, K. (2021). Deep learning and machine vision for food processing: A survey. *Current Research in Food Science*, 4, 233–249. <https://www.sciencedirect.com/science/article/pii/S2665927121000228?via%3DIihub>

Eidesstattliche Erklärung

Hiermit erkläre ich eidesstattlich, dass die vorliegende Arbeit von mir selbstständig und ohne unerlaubte Hilfe angefertigt wurde, insbesondere, dass ich alle Stellen, die wörtlich oder annähernd wörtlich oder dem Gedanken nach aus Veröffentlichungen und unveröffentlichten Unterlagen und Gesprächen entnommen worden sind, als solche an den entsprechenden Stellen innerhalb der Arbeit durch Zitate kenntlich gemacht habe, wobei in den Zitaten jeweils der Umfang der entnommenen Originalzitate kenntlich gemacht wurde. Ich bin mir bewusst, dass eine falsche Versicherung rechtliche Folgen haben wird.

Ort, Datum

Unterschrift

Anhang

In diesem Anhang befinden sich die von der KI erzeugten Testfälle.