



**School of  
Engineering**

CAI Centre for  
Artificial Intelligence

## **Projektarbeit** **(Wirtschaftingenieurwesen)**

Analyse des Einflusses von drei  
Parametern auf Chat-GPT-4 Antworten  
mittels Variierung entsprechender  
Argumente

---

**Autoren**

Gizem Gross

---

**Hauptbetreuung**

Prof. Dr. Mark Cieliebak

---

**Datum**

30.01.2024

## Erklärung betreffend das selbstständige Verfassen einer Bachelorarbeit an der School of Engineering

### Erklärung betreffend das selbstständige Verfassen einer Bachelorarbeit an der School of Engineering

Mit der Abgabe dieser Bachelorarbeit versichert der/die Studierende, dass er/sie die Arbeit selbständig und ohne fremde Hilfe verfasst hat. (Bei Gruppenarbeiten gelten die Leistungen der übrigen Gruppenmitglieder nicht als fremde Hilfe.)

Der/die unterzeichnende Studierende erklärt, dass alle zitierten Quellen (auch Internetseiten) im Text oder Anhang korrekt nachgewiesen sind, d.h. dass die Bachelorarbeit keine Plagiate enthält, also keine Teile, die teilweise oder vollständig aus einem fremden Text oder einer fremden Arbeit unter Vorgabe der eigenen Urheberschaft bzw. ohne Quellenangabe übernommen worden sind.

Bei Verfehlungen aller Art treten die Paragraphen 39 und 40 (Unredlichkeit und Verfahren bei Unredlichkeit) der ZHAW Prüfungsordnung sowie die Bestimmungen der Disziplinarmaßnahmen der Hochschulordnung in Kraft.

**Ort, Datum:**

Zürich, 30. Januar 2024

**Name Studierende:**



## Zusammenfassung

OpenAI ist ein Forschungsunternehmen aus den USA und hat sich insbesondere durch Softwareentwicklungen im Bereich der Generativen Künstlichen Intelligenz einen Namen gemacht. Die Einführung des KI-Chatbots ChatGPT-3 im November 2022 erregte weltweit Aufmerksamkeit in verschiedenen Bereichen und hatte weitreichende Auswirkungen, dies auch im Bildungssektor, was die Bedeutung dieser Arbeit unterstreicht. Die vorliegende Projektarbeit befasst sich mit der Untersuchung des Einflusses bestimmter Parameter - temperature, Kontextanzahl und Prompt-Art - auf die Leistungsfähigkeit von ChatGPT-4, einem Large Language Model von OpenAI. Es wird ein systematischer Ansatz verfolgt, um sechs Experimente durchzuführen, die speziell darauf abzielen, die Auswirkungen dieser Parameter auf die Genauigkeit der vom Modell generierten Antworten zu ermitteln. Unter Einsatz eines Datensatzes, der Fragen und Kontexte zur Infektionskrankheit Masern beinhaltet, erstrecken sich die Experimente von der Generierung einfacher Ja/Nein-Antworten bis hin zur Erstellung ausführlicher Antworttexte, wobei die temperature-Einstellungen variiert werden, um die Präzision des Modells zu bewerten. Die Ergebnisse deuten darauf hin, dass ein ausgewogenes Verhältnis zwischen Kontextanzahl und geeigneten temperature-Einstellungen wesentlich zur Verbesserung der Antwortqualität des Modells beiträgt. Das heisst, dass die ein Einfluss der Parameter temperature und Kontextanzahl festzustellen ist. Im Rahmen der Untersuchung für die angeführte Projektarbeit ist kein Einfluss der Prompt-Art festzustellen. Die Arbeit schliesst mit einer Diskussion über die Experimenten für zukünftige Anwendungen von Large Language Models ab und fügt zuletzt einen Ausblick an.

# Inhaltsverzeichnis

<b>1. Einleitung</b>	<b>1</b>
1.1 Ausgangslage	2
1.2 Zielsetzung	3
1.3 Vorgehen	3
<b>2. Theoretische Grundlagen</b>	<b>4</b>
2.1 Natural Language Processing, NLP	5
2.2 Das Large Language Modell von OpenAI	6
2.2.1 Pre-Trained Large Language Modelle	7
2.2.2 Transformer-Architektur	8
2.2.3 Reinforcement Learning (RLHF)	10
2.3 ChatGPT-4 und seine Vorgänger	11
2.3.3 Vergleich GPT-3 und GPT-4	12
2.4 Einflussparameter	13
2.4.1 Erläuterung relevanter Parameter	13
3.1 Experimente	15
3.1.1 Experiment 1	15
3.1.2 Experiment 2	16
3.1.3 Experiment 3	17
3.1.4 Experiment 4	18
3.1.5 Experiment 5	18
3.1.6 Experiment 6	19
<b>4. Resultate</b>	<b>20</b>
4.1 Experiment 1	20
4.2 Experiment 2	21
4.3 Experiment 3	22
4.4 Experiment 4	24
4.5 Experiment 5	25
4.6 Experiment 6	25
<b>5. Diskussion</b>	<b>26</b>

5.1 Interpretation der Ergebnisse . . . . .	26
5.3 Ausblick . . . . .	26
<b>6. Verzeichnisse . . . . .</b>	<b>27</b>
6.3 Abkürzungsverzeichnis . . . . .	31
<b>7. Appendix . . . . .</b>	<b>32</b>
7.1 Python-Code Dokumentation . . . . .	32

# 1. Einleitung

In einem Zeitalter, indem künstliche Intelligenz (KI) einen zunehmend starken Einfluss auf verschiedene Bereiche des menschlichen Lebens ausübt, wird grossen Sprachmodellen (Large Language Models, LLMs) eine signifikante Bedeutung beigemessen. LLMs nutzen KI-Algorithmen, die auf modernen Deep Learning-Architekturen basieren und mittels Methoden des maschinellen Lernen trainiert werden, wobei sie dabei auf grosse Datenmengen angewiesen sind. Bereits im Jahr 2018 stellte Google mit BERT ein grosses Sprachmodell vor, welches in darauffolgenden Jahren eine Basis für Experimente zur Verarbeitung natürlicher Sprache<sup>1</sup> geworden ist. In der breiten Öffentlichkeit wurde die Aufmerksamkeit für grosse Sprachmodelle jedoch erst durch die Veröffentlichung des Chatbots “ChatGPT-3” vom Forschungsunternehmen OpenAI im November 2022 gewonnen, welches signifikant zur Popularisierung dieser Technologie beitrug [1]. Der Chatbot, welcher auf der Technologie namens GPT basiert, stellte einen revolutionären Durchbruch im Bereich des NLPs dar. Die in GPT verwendete Transformer- Architektur ist ein revolutionärer Fortschritt und stellt eine wesentliche Verbesserung im Vergleich zu früheren Ansätzen im Bereich der natürlichen Sprache (NLP) dar [2].

LLMs haben das Potential, die Interaktion zwischen Mensch und Maschine, Methoden der Informationsbeschaffung sowie die Lernweisen grundlegend zu verändern und bieten weitreichende Nutzungsmöglichkeiten für diverse Anwendungsbereiche. Es ist jedoch anzumerken, dass grosse Sprachmodelle anfällig für die Generierung voreingenommener oder irreführender Antworten sein können und zudem eine begrenzte Fähigkeit bei der Kontexterfassung aufweisen können [1]. Diese Aspekte unterstreichen die Notwendigkeit einer sorgfältigen Überwachung und fortlaufender Optimierung von LLMs. Im Rahmen dieser Arbeit wird auf die möglichen negativen Auswirkungen von grossen Sprachmodellen jedoch nicht weiter eingegangen.

---

<sup>1</sup>Natural Language Processing (NLP)

## 1.1 Ausgangslage

Der von OpenAI entwickelte “Generative Pretrained Transformer” (GPT) ist ein solches Large Language Model. Es wurde mittels von KI und maschinellem Lernen auf Grundlage einer ausserordentlich grossen Datenmenge aus dem Internet trainiert. Das Sprachmodell besitzt ein sogenanntes Rahmenwerk, welches die Feinabstimmung durch Variierung von diversen Parametern auf der Grundlage von u.a. Verstärkungslernen durch menschliches Feedback, bekannt als Reinforcement Learning from human feedback (RLHF), ermöglicht. Im Gegensatz zu denjenigen Sprachmodellen, welche vor der Veröffentlichung von ChatGPT bekannt waren, weist das GPT-Modell ein sogenanntes Konversationsgedächtnis auf. Dieses ermöglicht die Beantwortung komplexerer Fragestellungen wie z.B. mögliche Lösungswege von Layoutproblemen, Lösung von Optimierungsproblemen sowie das Schreiben von Computercode [2].

Um das Modell für Endnutzer zugänglich zu machen, wurde ein sogenannter Chatbot entwickelt. Chatbots sind so konzipiert, dass über eine dialogbasierte Programmierschnittstelle (APIs) Informationen geliefert und Fragen beantwortet werden. Dabei spielt NLP also der Prozess der natürlichen Sprachverarbeitung eine zentrale Rolle, damit der Chatbot menschliche Sprache verstehen und erzeugen kann [2].

Chat-GPT hat bereits eine Woche nach seiner Lancierung mehr als eine Million Nutzer verzeichnet [3]. Die ursprüngliche Version von ChatGPT basierte auf dem GPT-3.5 Modell. Seit März 2023 verwendet ChatGPT das optimierte GPT-4 Modell, welches gemäss Auswertungen von OpenAI 82% weniger auf Anfragen nach unzulässigen Inhalten antwortet und zu 40% wahrscheinlicher sachliche Antworten liefert [4].

## 1.2 Zielsetzung

Die vorliegende Projektarbeit widmet sich der Untersuchung der Generierung von Antworten von ChatGPT-4. Die konkrete Leitfrage der Projektarbeit lautet:

**Welchen Einfluss haben die drei ausgesuchten Parameter auf die generierten Antworten von ChatGPT-4?**

Nachfolgende Unterleitfragen dienen zur Orientierung bei der Recherche, Aufbereitung der Thematik und tragen zur Beantwortung der Leitfrage bei.

1. Welche Technologien sowie Disziplinen liegen grossen Sprachmodellen zugrunde und wie funktioniert das GPT-Modell von OpenAI?
2. Welche Parameter können angepasst werden ?
3. Welchen Einfluss haben die Anzahl Kontexte, die dem System übergeben werden ?
4. Inwiefern beeinflusst das Argument temperature das Resultat der Experimente ?
5. Inwiefern ist das System in der Lage, Fragen korrekt zu beantworten, wenn der korrekte Antworttext als einer der Kontexte übergeben wird ?
6. Wie stark kann die Prompt-Art geändert werden, damit man die gleichen Resultate für die entsprechenden Fragestellungen erhält ?

## 1.3 Vorgehen

Zur Beantwortung der ersten Unterleitfrage soll ein theoretisches, literaturgestütztes Vorgehen gewählt werden, bei dem in einem ersten Schritt eine umfassende Literaturrecherche durchgeführt wird. Die gefundene Literatur wird gesichtet, ihrer Relevanz entsprechend geordnet und relevante Inhalte werden für Kapitel 2 aufbereitet. Die in Kapitel 1.2 vorgestellten Unterleitfragen sollen empirisch beantwortet werden. Dazu werden methodisch aufeinander aufbauende Experimente entwickelt und deren Zwischenresultate analysiert werden (Kapitel 3). Die Endresultate werden in Kapitel 4 vorgestellt, woraufhin im fünften Kapitel die Leitfrage beantwortet werden soll.

## 2. Theoretische Grundlagen

Um die Technologie hinter ChatGPT zu ergründen wird ein Einblick in Schlüsselthemen wie Natural Language Processing (NLP), Large Language Models (LLMs) und die Entwicklungsgeschichte von ChatGPT gegeben. In den nachfolgenden Unterkapiteln wird mithilfe eines Top-Down Ansatzes das Verständnis für die Funktionsweise von ChatGPT vertieft, insbesondere auch im Hinblick auf deren Relevanz für die angeführte Projektarbeit. Aufgrund der Komplexität wird im nachfolgenden Abschnitt zu Beginn die Grundidee von ChatGPT erläutert.

Die grundlegende Funktionsweise von LLMs basiert darauf, dass bei einem gegebenen Text passende Wörter erzeugt werden bis die erforderliche Menge an Worten erreicht wurde. Jedes Wort in einem Text respektive in einem Satz besitzt eine Liste von Nachfolgewörtern, welche aufgrund des Kontextes in ihrer Nachfolgewahrscheinlichkeit variieren. Die nachfolgende Darstellung illustriert anhand eines einfachen Satzes diesen Grundgedanken. Das System steht dabei für ein LLM wie z.B. das GPT-Modell von OpenAI [5].

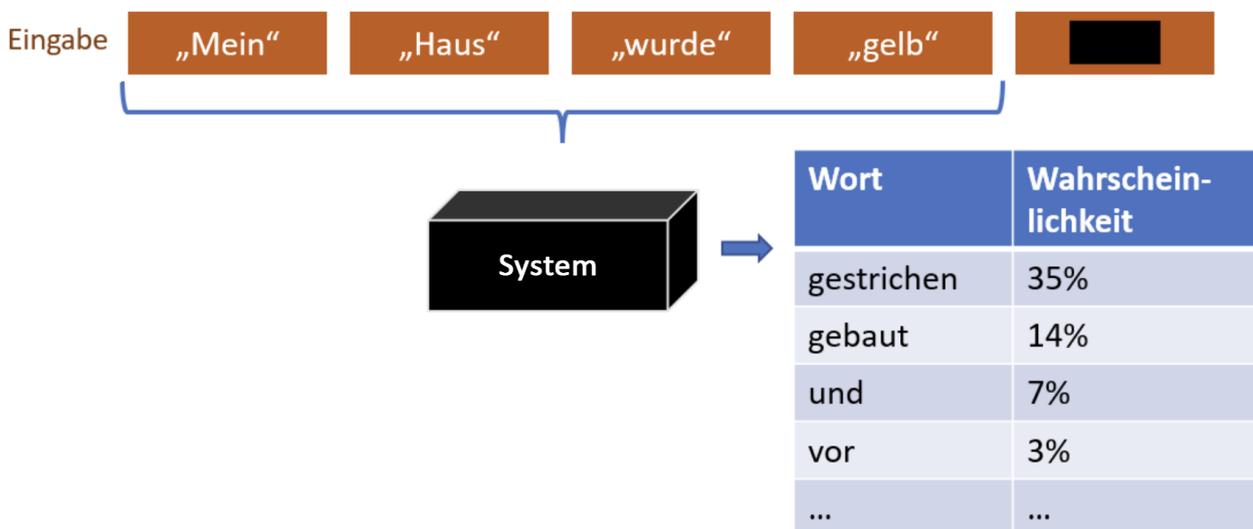


Abbildung 1: Das Konzept von grossen Sprachmodellen basiert darauf, dass für einen gegebenen Satz eine Wahrscheinlichkeitsverteilung für das nachfolgende Wort zu berechnen, welche in der Grafik in blauen Kästchen in Prozent angegeben ist [5].

## 2.1 Natural Language Processing, NLP

Der Ursprung von Sprachmodellen liegt in der Zählung von Wortpaaren innerhalb eines Textes und der Analyse ihrer Häufigkeit. Lange Zeit wurde angenommen, dass der erfolgsversprechendste Ansatz in der Entwicklung von Worttaxonomien, der Analyse von Sätzen mittels Parse-Bäumen, der Kategorisierung von Wortarten sowie der Verwendung von Vorlagen und Regeln zur Interpretation der Semantik läge. Die Entwicklung von sogenannten N-Grammen, welche die Zählung von Wortsequenzen bis zu einer bestimmten Länge beinhalten, verbesserte die Präzision der Sprachmodelle. Eine N-Gramm-Analyse beantwortet die Frage, wie wahrscheinlich es ist, dass auf eine bestimmte Buchstaben- respektive Wortfolge ein bestimmter Buchstabe bzw. ein bestimmtes Wort folgt. Allerdings wies die N-Gramm-Analyse kein Verständnis für die Semantik eines Textes auf. Die Darstellung von Wörtern als Vektoren in einem hochdimensionalen Raum stellte einen Durchbruch im Bereich des statistischen NLP dar. Dadurch wurde ermöglicht, die Beziehung zwischen Wörtern über ihre Positionierung in einem multidimensionalen Raum zu definieren, was zu einer differenzierteren und flexibleren Darstellung von Wortbedeutungen führte [5], [6].

Die Techniken von Word Embeddings, welche eine Reihe von Sprachmodellierungs- und Merkmalstechniken umfasst, bei denen Wörter oder Sätze auf Vektoren reeller Zahlen abgebildet werden, erlaubten es, Ähnlichkeiten und Zusammenhänge zwischen Wörtern und Konzepten effizient zu erfassen [6]. Wie in Abbildung XY dargestellt werden jedem Wort zufällig eine Position sowie ein Kontextvektor im semantischen Raum zugeordnet. Der Kontextvektor repräsentiert die Position des Wortes im semantischen Raum, an dem die Vorgänge- respektive Nachfolgewörter vermutet werden. Als Folge des Modelltrainings mit zahlreichen Beispielsätzen liegen Wörter mit ähnlicher Semantik im semantischen Raum eng beieinander. Doch in der menschlichen Sprache existieren Wörter, welche abhängig von ihrem Kontext eine andere Bedeutung aufweisen<sup>1</sup>. Daher stellte diese Methodik nur eine grobe Annäherung an ein Textverständnis dar [5].

Mit dem im Jahr 2017 veröffentlichte Paper “Attention is all you need” von Google wurde eine revolutionäre Transformer-Architektur vorgestellt, welche die sequenzielle Verarbeitung bisheriger Modelle durch eine Parallelverarbeitungsfähigkeit der Architektur ablöste. Durch

---

<sup>1</sup>In der Linguistik bezeichnet man ein Wort, welches unterschiedliche Bedeutungen aufweisen kann, als Homonym.

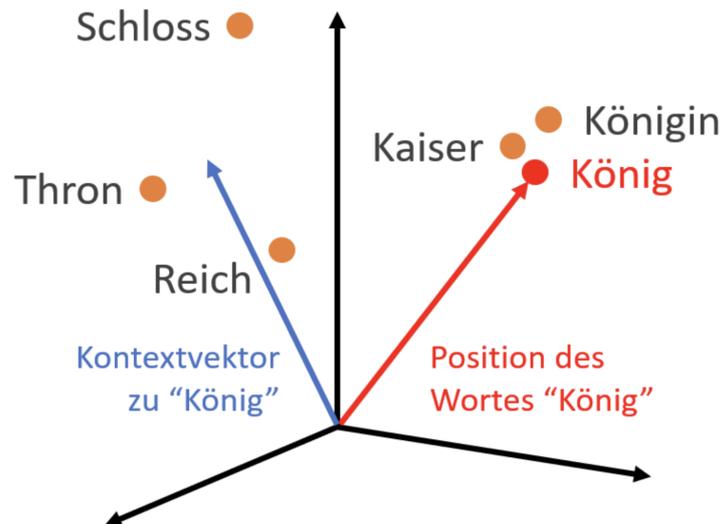


Abbildung 2: Die Grafik stellt eine Vektorrepräsentation von Wörtern im semantischen Raum dar. Die Nähe des Wortes “Kaiser” zu “König” suggeriert, die inhaltliche Verwandtheit und werden oft in ähnlichen Kontexten verwendet. Der blaue Vektor, der als Kontextvektor zu “König” quantifiziert die Beziehung zwischen “König” und den anderen Wörtern im semantischen Raum [3].

die neue Transformer-Architektur wurde die Effizienz bei der Lösung von NLP-Aufgaben sowie die Leistungsfähigkeit von LLMs signifikant gesteigert. OpenAI entwickelte das Konzept der Transformer-Architektur weiter, welches im nachfolgenden Kapitel 2.2 näher erläutert wird [7].

## 2.2 Das Large Language Modell von OpenAI

OpenAI stellte 2018 das von ihr entwickelte Large Lanuguage Modelle namens GPT vor, welches fortlaufend durch neue Versionen verbessert wird. In den nachfolgenden Unterkapiteln werden die verschiedenen Phasen, welche in Abbildung 3 dargestellt sind für die Erstellung eines finalen LLM-Modells wie GPT-3/4 vorgestellt [8].

### ① Pre-Training

### ② Transformer

### ③ Fine-Tuning & Reward Model

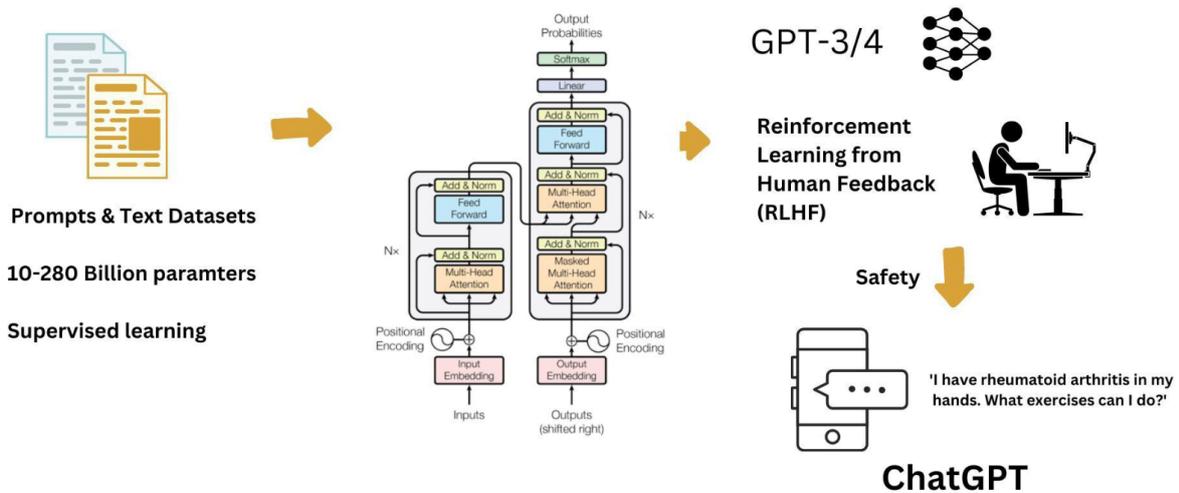


Abbildung 3: Die Grafik zeigt schematisch den Aufbau und die grobe Funktionsweise des GPT-3/4 Sprachmodells in drei Hauptphasen [7].

#### 2.2.1 Pre-Trained Large Language Modelle

Das Ziel eines vortrainierten Sprachmodells besteht darin, eine Art Blackbox zu erschaffen, welche natürliche menschliche Sprache versteht und fähig ist Sprachverarbeitungsaufgaben zu lösen. Dies ist von essenzieller Bedeutung, da es dem Modell ermöglicht, sich ein allgemeines Verständnis von Sprachmustern verschiedener Datenquellen aufzubauen und sich ein breites Wissen über Grammatik, Semantik und Syntax anzueignen. Zu Beginn wird das Sprachmodell mit einer grossen Menge an Daten gefüttert. Es wird ein Algorithmus auf markierte Daten trainiert, von denen die korrekten Ausgaben bereits bekannt sind (Supervised Learning). Während des Trainings erlernt der Algorithmus die Zuordnung von Eingabemerkmalen und den respektiven Ausgabebezeichnungen. Das Ziel dieses Verfahrens ist die Erstellung eines Modells, welches die Ausgabe für neue Eingaben genau vorhersagen kann. Die Phase des Vortrainings bildet wie in Abbildung 4 dargestellt die erste Phase. Anschliessend wird das vortrainierte Sprachmodell durch die Transformations-Architektur und der Feinabstimmung verbessert, welche in den nachfolgenden Kapiteln 2.2.2 und 2.2.3 genauer erläutert wird [6].

## 2.2.2 Transformer-Architektur

Beim Transformationsprozess und der Feinabstimmung wird das beim Vortraining des LLMs erlernte Wissen, welches in Abbildung 4 als erste Phase dargestellt ist, genutzt und an die jeweiligen Aufgaben angepasst. Dies geschieht in weiteren zwei Phasen. In der zweiten Phase kommt ein sogenannter Transformer zum Einsatz. Das Konzept eines Transformers wurde erstmals 2017 von Ashish Vaswani vorgestellt und stellt einen bedeutenden Schritt in der Weiterentwicklung von Deep-Learning Modellen dar. Ein Transformer besteht aus zwei Hauptkomponenten, dem Encoder und dem Decoder wie in Abbildung 4 dargestellt, welche typischerweise durch unüberwachtes Lernen (Unsupervised learning) trainiert werden. Der Encoder nimmt eine Eingabesequenz auf und transformiert diese in eine Zwischenrepräsentation, welche die wesentlichen Merkmale der Eingabesequenz enthält und eine feste Zeichenlänge besitzt. Diese Repräsentation dient als Initialzustand für den Decoder. Der Decoder wiederum generiert Ausgabesequenzen, wobei sowohl die vom Encoder bereitgestellte Repräsentation als auch die bereits generierten Ausgabesequenzen bei jedem Bearbeitungsschritt beachtet werden. Dieser Mechanismus ermöglicht es dem Modell, die Bedeutung einzelner Teile der Eingabesequenz differenziert zu gewichten; bei der Erzeugung der Ausgabesequenz wird die Signifikanz von jedem Element berücksichtigt [9].

### ② Transformer

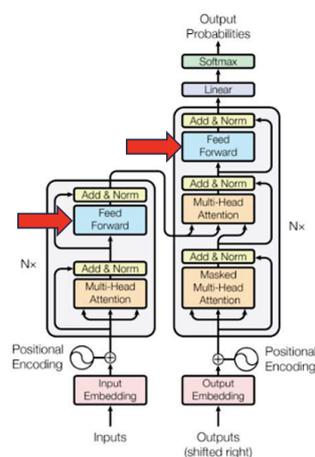


Abbildung 4: Die Grafik illustriert den Einsatz (rote Pfeile) des Feed-Forward-Netzwerkes in der Transformer-Architektur und deutet auf die elementare Bedeutung dieses Netzwerkes hin [10].

Ergänzt wird dies durch ein sogenanntes Feed-Forward-Netzwerk, welches in jeden Schritt des Encoders und Decoders integriert ist. Die Eingabedaten werden durch die Schichten des Feed-Forward-Netzwerks geleitet, wobei jede Schicht (vgl. Abbildung 5: Input Layer, Hidden Layer und Output Layer) die Daten in eine bestimmte Art und Weise umwandelt. Jedes Schicht kommuniziert nur mit der Schicht vor ihr und nach ihr, wobei sich der Informationsfluss nur in eine Richtung bewegt; von der Eingabeschicht zur Ausgabeschicht. Das Feed-Forward-Netzwerk ist ein elementarer Teil der Transformer-Architektur und ermöglicht es dem Modell komplexe Muster und vorallem non-lineare Beziehungen in den Daten zu erlernen, was letztlich zu einer effizienten Verarbeitung und Transformation der Daten beiträgt [9] [10].

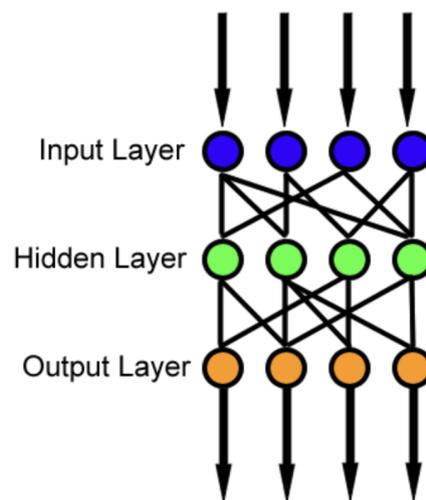


Abbildung 5: Die Grafik stellt im Wesentlichen den Prozess der Weiterleitung von Informationen durch das Netz vom Input Layer über die Hidden Layer bis hin zum Output Layer [10].

### 2.2.3 Reinforcement Learning (RLHF)

In der dritten Phase kommt Reinforcement Learning zum Einsatz; dabei handelt es sich genauer gesagt um Reinforcement Learning from Human Feedback (RLHF), d.h. es wird händisch überprüft. Die Überprüfer werden auch als Trainer bezeichnet. Die menschlichen Trainer stimmen über die Relevanz der Modellausgaben ab und erstellen einen Vergleichsdatensatz, auf Basis dessen ein Reward Model (RM) trainiert wird. Dies ist in Abbildung 6 in Schritt 2 illustriert. Das RM ist eine einfache Kopie des Supervised Finetuning Modell, welches aus dem Schritt 1 in Abbildung XY resultiert. Das RM-Reward-Modell wird unter Zuhilfenahme der Policy-Gradient-Methode sowie der Proximal Policy Optimization (PPO) optimiert. Der PPO-Algorithmus ist ein „on-policy“-Algorithmus, der eine aktuelle Policy lernt und aktualisiert, indem er sich direkt auf die erhaltenen Aktionen und Belohnungen stützt. Ein wesentlicher Teil des PPO Algorithmus ist die sogenannte Vorteilsfunktion, mit welcher versucht wird zu schätzen, ob bestimmte Handlungen besser sind als andere mögliche Handlungen. Dadurch wird ein neues Modell, das sogenannte Policy Model, generiert. Mit diesem Policy Modell kann das ursprüngliche Reward-Modell verbessert werden und ein neuer Vergleichsdatensatz erstellt werden [8, 10]. Anschliessend können die beiden Schritte in einer Schleife wiederholt werden, um die Qualität des Modells kontinuierlich zu steigern. Diese iterative Methode ermöglicht, dass das Modell zunehmend präzisere und relevantere Zusammenfassungen generiert, was die Effektivität des RLHF-Prozesses weiter steigert und zu einem robusteren und zuverlässigeren Endprodukt führt [12].

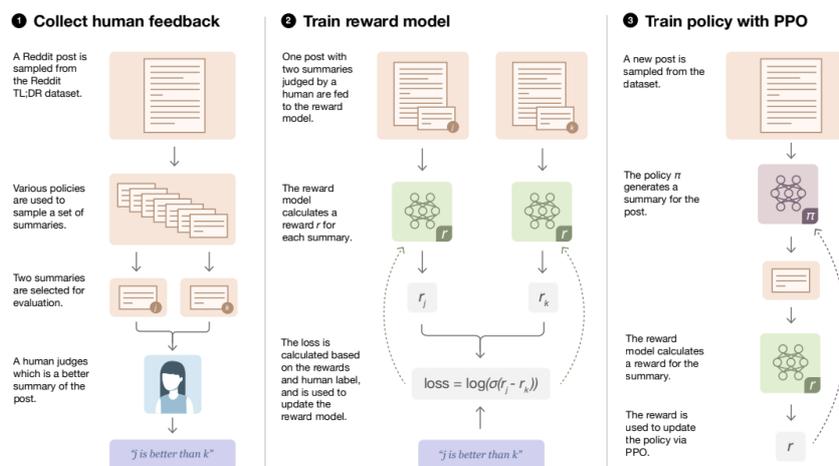


Abbildung 6: Die Grafik illustriert einen dreistufigen Prozess für die Feinabstimmung des Modells, [12].

## 2.3 ChatGPT-4 und seine Vorgänger

ChatGPT-4, entwickelt von OpenAI, ist das neueste Modell in einer Reihe von Generative Pre-trained Transformers (GPTs). Der erste GPT wurde 2018 eingeführt, gefolgt von GPT-2 im Jahr 2019 und GPT-3 im Jahr 2020. Jedes Modell baute auf dem vorherigen auf, wobei GPT-3 mit 175 Milliarden Parametern einen bedeutenden Durchbruch darstellte. Die Entwicklung von GPT-4 markierte einen weiteren Fortschritt in der Fähigkeit, komplexe Aufgaben zu bewältigen und die Häufigkeit unerwünschter oder schädlicher Antworten zu verringern. GPT-4 kennzeichnet sich dadurch, dass die Anzahl Parameter nahezu verdoppelt wurde (vgl. Abbildung 7) [13].

Ein weiterer Unterschied besteht bei der in Kapitel 2.2.2 genauer erläuterten Transformer-Architektur, bestehend aus Encoder und Decoder, welche wiederum je nach Modellkomplexität 12 bis 24 sogenannte Feed-Forward-Schichten<sup>1</sup> enthalten. Im Gegensatz zu den vorherigen Modellen kann der Transformer von GPT-4 bis zu 128 Schichten aufweisen, was die erfolgreiche Bewältigung sprachlich komplexer Aufgaben enorm steigert [13].

	<b>GPT-1</b>	<b>GPT-2</b>	<b>GPT-3</b>	<b>GPT-4</b>
Parameters	117 million	1.5 billion	175 billion	300 billion
Decode Layers	12	48	96	128
Hidden Layers	768	1600	12,288	20,480
Context Token Size	512	1024	2048	4096
Fine Tuning Datasets	Limited	More	Many	Extensive
Fine Tuning Tasks	Few	More	Many	Extensive
Language Understanding	Limited	Improved	Advanced	Highly Advanced
Text Generation	Basic	Advanced	Very Advanced	Exceptional
Sentiment Analysis	Not Supported	Not Supported	Supported	Enhanced
Text Summarization	Not Supported	Not Supported	Supported	Enhanced
Text Correction	Not Supported	Not Supported	Supported	Enhanced

Abbildung 7: Vergleichsübersicht der fortschreitenden Entwicklungsstufen der Generative Pre-trained Transformer Modelle (GPT) von OpenAI [13].

In Abbildung 8 ist ein Vergleich der Leistungsfähigkeit verschiedener LLMs abgebildet, wobei deren Fähigkeiten in Textklassifikation, Texterzeugung und Textzusammenfassungen bewertet werden. Auf der Abszisse sind die Sprachmodelle gelistet und auf der Ordinate ihre jeweilige Leistungsfähigkeit (in Prozent von 0% bis 100%). Es ist ersichtlich, dass GPT-3, XLNet und Megatron über alle drei Metriken hinweg die die höchste Leistungsfähigkeit aufweisen [13].

<sup>1</sup>Feed-Forward-Schichten sind in der Lage verschiedene Ebenen sprachlicher Mehrdeutigkeit sowie Komplexität, Interpunktion, Morphologie, Synthax, Semantik und komplizierte Interaktionen zu erfassen

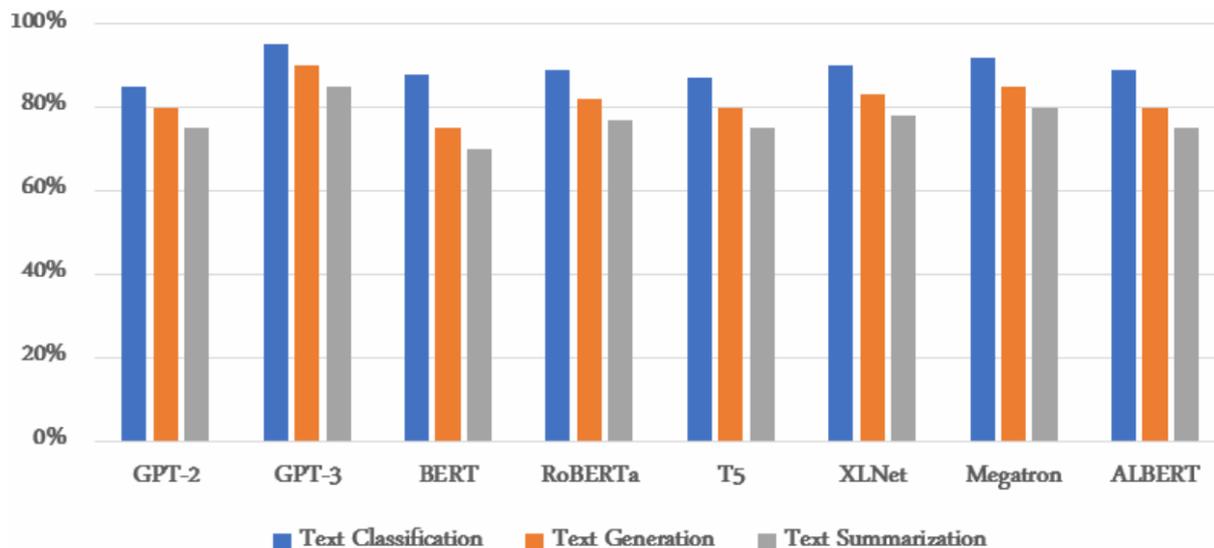


Abbildung 8: Ein Leistungsvergleich von diversen Sprachmodellen in verschiedenen NLP-Aufgaben [13].

### 2.3.3 Vergleich GPT-3 und GPT-4

Ein Hauptmerkmal von GPT-4 ist die Erweiterung der Fähigkeit, verschiedene Medienformate wie z.B Bilder und Videos zu verarbeiten, was es zu einem multifunktionalen System für verschiedene Anwendergruppen macht. Im Gegensatz dazu ist der GPT-3 hauptsächlich auf die Verarbeitung von Sprache beschränkt und zeigt Einschränkungen in der Handhabung von Multimedia-Inputs. Einen bedeutenden Fortschritt bietet GPT-4 zum einen in der differenzierten Verarbeitung und Generierung von Sprachvarianten (insbesondere von Dialekten) und zum anderen in der Reaktion auf emotionale Nuancen in Texten. Gemäss der Studie von Alawida, Mejri und Mehmood ist GPT-4 in der Lage, emotionale Ausdrucksformen besser zu erfassen. Dies zeigte sich dadurch, dass die Nutzer das System als empathische wahrgenommen haben. Des Weiteren zeichnet sich GPT-4 durch die markante Fähigkeit aus, Informationen aus eine Vielzahl von Quellen zu aggregieren, was die Bewältigung von komplexeren Fragestellungen vereinfacht; GPT-3 ist dahingehend weitaus eingeschränkter. Zudem besitzt GPT-4 eine verbesserte Kompetenz in der Erstellung kreativer Inhalte. Abschliessend ist die Erhöhung der maximalen Tokenanzahl<sup>1</sup> bei GPT-4 hervorzuheben, welche 32'000 Tokens umfasst (entspricht ca. 25'000 Wörtern). Zum Vergleich: Die Token-Grenze von GPT-3 lag bei 4'000 Tokens respektive ca. 3'125 Wörtern. GPT-4 kann dadurch deutlich umfangreichere und komplexere Texte generieren [13].

<sup>1</sup>siehe auch Kapitel 2.4

## 2.4 Einflussparameter

Die nachfolgenden Parameter beeinflussen massgeblich die Leistungsfähigkeit von ChatGPT-4 und werden daher kurz erläutert. In nachfolgenden Kapitel 2.4.1 werden die für die angeführte Projektarbeit ausgewählte Parameter vorgestellt.

### Maximale Token-Anzahl

Begrenzt die Länge der vom Modell generierten Antworten. Eine höhere Token-Anzahl erlaubt längere und möglicherweise detailliertere Antworten, kann aber auch zu ausschweifenderen und damit unfokussierteren Ergebnissen führen [14].

### Stop-Sequenzen

Bestimmen, wann das Modell die Answerzeugung beenden soll. Diese können genutzt werden, um zu verhindern, dass das Modell über ein bestimmtes Ende hinaus antwortet oder um die Generierung von Inhalten in einem bestimmten Format zu steuern [15].

### 2.4.1 Erläuterung relevanter Parameter

#### 1) Parameter: temperature

Der Parameter “temperature” in ChatGPT steuert die Zufälligkeit bzw. Kreativität der generierten Antworten. Eine niedrige Temperatur führt zu konsistenteren und vorhersehbareren Antworten, während eine höhere Temperatur das Modell dazu veranlasst, vielfältigere und weniger vorhersehbare Antworten zu generieren [16].

Bei einer niedrigen Temperatur (z. B. nahe bei Null) tendiert das Modell dazu, die wahrscheinlichste Fortsetzung basierend auf dem gelernten Trainingsdatensatz zu wählen. Dies kann in Situationen nützlich sein, in denen Genauigkeit und Klarheit entscheidend sind. Bei einer höheren Temperatur (z.B. näher beim Wert 1) nimmt die Wahrscheinlichkeit zu, dass das Modell unerwartete oder kreative Antworten generiert. Dies kann für kreative Schreibaufgaben oder zur Generierung von Ideen hilfreich sein [16].

Es ist wichtig zu beachten, dass eine höhere Temperatur auch das Risiko erhöht, ungenaue oder unzusammenhängende Antworten zu erhalten, da das Modell stärker von den wahrscheinlichsten Antworten abweicht. Die Wahl der richtigen Temperatur hängt daher stark vom Kontext der Anwendung und den spezifischen Anforderungen des Endnutzers ab [16].

## **2) Parameter: Anzahl Kontexte**

Der Parameter “Anzahl Kontexte” bezieht sich auf die Menge an Kontextinformationen, welche dem Modell für die Generierung von Antworten zur Verfügung gestellt werden. Dieser Kontext kann in Form von vorhergehenden Nachrichten, Hintergrundinformationen oder spezifischen Anweisungen bestehen. Der Einfluss des Kontextes auf die Antwortgenerierung ist signifikant und vielschichtig. Ein umfangreicher Kontext ermöglicht es dem Modell, präzisere und relevantere Antworten zu generieren, da es einen tieferen Einblick in das Diskussionsthema oder die spezifische Anfrage ermöglicht. Beispielsweise kann das Modell bei einer technischen Support-Anfrage bessere Lösungsvorschläge liefern, wenn es Informationen über das verwendete Betriebssystem, frühere Problemlösungsversuche und spezifische Fehlermeldungen hat. In der Praxis bedeutet dies, dass die Anzahl und Qualität der Kontextinformationen sorgfältig abgewogen werden muss, um die Effektivität von ChatGPT-4 zu maximieren. Ein ausgewogener Kontext, der relevante und aktuelle Informationen enthält, unterstützt das Modell dabei, genaue und nützliche Antworten zu generieren.

## **Parameter: Prompt**

Wie in Kapitel 1.1 beschrieben, wird das LLM dem Endnutzer durch ein Chatbot zugänglich gemacht<sup>1</sup>. Der Prompt ist dabei, die dem System gestellte Frage oder Anweisung. Es existieren zahlreiche Prompt-Methoden für diverse Anwendungsbereiche. In den Experimenten 1 bis 4 (siehe Kapitel 3) wird die Schritt-für-Schritt Methode angewendet und in den Experimenten 5 und 6 wurde eine neue Prompt-Methode, der Rollenzuteilung, hinzugefügt, um zu eruieren, welchen Einfluss dies auf die von ChatGPT-4 generierten Antworten hat [17].

---

<sup>1</sup>Daher auch der Name *ChatGPT*

# 3. Methodik Die Durchführung der Experimente basiert auf einem ausgewählten Textdokument über ein spezifisches Thema. Hierzu ist ein FAQ-Dokument im pdf-Format über die Infektionskrankheit Masern, welches auf der offiziellen Homepage vom Bundesamt für Gesundheit (BAG) zu finden ist. Aus den 60 vorhandenen Fragen im Dokument sind 50 Fragen ausgewählt worden. 10 Fragen wurden weggelassen, weil die Beantwortung der Fragen mittels Tabellen oder Grafiken beantwortet wurde und daher nicht geeignet sind für vorliegende empirische Untersuchung. Des Weiteren wurde ein Excel-Dokument erstellt, in welchem alle 70 Kontexte vorhanden sind, wobei 50 Kontexte die 50 Antworten aus dem FAQ-Masern Dokument entnommen wurden und die restlichen 20 Kontexte bilden irrelevante Inhalte, welche mittels Klick auf “Zufälliger Artikel” von Wikipedia entnommen wurden.

### **3.1 Experimente**

Für die Beantwortung der in Kapitel 1.2 formulierten Leitfragen werden insgesamt sechs Experimente durchgeführt, welche in den folgenden Unterkapiteln näher erläutert sind. Dabei werden die drei Parameter “Anzahl Kontexte”, “temperature” und “Prompt” variiert. Die resultierenden Ergebnisse sind in Kapitel 4 angeführt. In den nachfolgenden Kapiteln wird der Terminus “System” gleichbedeutend mit ChatGPT-4 verwendet.

#### **3.1.1 Experiment 1**

Das erste Experiment soll die Unterleitfrage 3 (vgl. Kapitel 1.2) beantworten. Dazu wird ein schrittweises Verfahren angewendet, bei dem das System sequenziell mit 50 Fragen konfrontiert wird. Beginnend mit einem Kontexten, steigt die Anzahl der Kontexte in jedem Durchlauf um einen an, bis zu 50 Kontexten. 49 irrelevante Kontexte werden zufällig aus 70 möglichen Kontexten nach dem Prinzip “Ziehen ohne Zurücklegen” ausgewählt. Dies bedeutet, dass innerhalb einer Abfrage kein Kontext mehrfach vorkommen kann. Zusätzlich dazu wird jeweils der korrekte Kontext bei jedem Durchlauf übergeben, dies bedeutet, dass das System im ersten Durchlauf zwei Kontexte und im letzten Durchlauf aus insgesamt 50 Kontexten auswählen soll, ob der passende Kontext zur Beantwortung der Fragestellung vorhanden ist. Diese Prozedur wird bei jeder der 50 Fragen wiederholt.

Das Design von Experiment 1 ist schematisch in Abbildung 9 dargestellt. Es ist vorgesehen repräsentativ die Ergebnisse für vier ausgewählte Fragestellungen durch ein Balkendiagramm darzustellen, wobei die x-Achse die Anzahl Ja-Antworten und die y-Achse die Anzahl Kontexte anzeigen soll. Des Weiteren werden alle Resultate in ein Excel-Dokument eingefügt. Der Parameter temperature wird auf 0.5 gesetzt. Das System soll für jeden Durchlauf nachfolgenden task ausführen. Dies entspricht auch direkt dem Parameter Prompt.

task = f"“Führe folgende Schritte durch:

1 - Analysiere die Kontexte.

2 - Falls, du einen passenden Kontext findest, antworte mit: Ja + passende Kontextnummer.

Falls du keinen passenden Kontext findest, antworte mit: Nein.”“

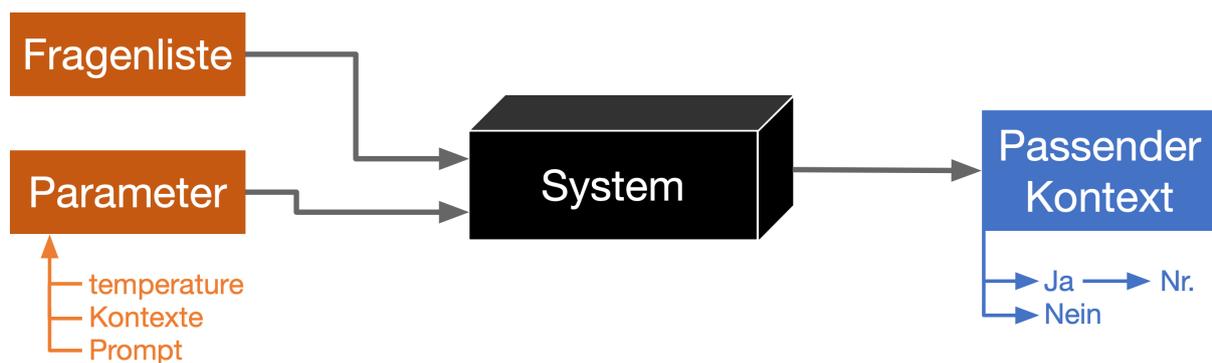


Abbildung 9: Als Inputparameter werden dem System temperature, Verschiedene Kontexte und ein Prompt übergeben. Als Output generiert das System zu jeder Frage pro Anzahl Kontexte eine Ja- oder Nein-Antwort.

### 3.1.2 Experiment 2

Das zweite Experiment soll Resultate zur Beantwortung der Unterleitfrage 4 liefern (vgl. Kapitel 1.2). Der Ansatz ist, dass erste Experiment erneut durchzuführen, jedoch mit einem temperature-Wert von 0. Die Ergebnisse sollen visualisiert werden und in einem Excel-Dokument abgespeichert werden.

### 3.1.3 Experiment 3

Das dritte Experiment soll die Unterleitfrage 5 beantworten (vgl. Kapitel 1.2) und sieht vor, dass 50 Fragen beantwortet werden. Statt einer Ja/Nein-Antwort soll das System ein Antworttext in Prosa schreiben. Anschliessend wird eine Stichprobenanalyse durchgeführt, um die Genauigkeit der Antworten zu überprüfen. Durch den Vergleich der Genauigkeit und der Kontext Adäquanz der Antworten, mit denjenigen aus dem ersten Experiment wird die Systemzuverlässigkeit beurteilt. Der Parameter temperature wird auf 0.5 gesetzt. Das Design von Experiment 3 ist in Abbildung 10 schematisch dargestellt. Es ist eine Visualisierung der Ergebnisse geplant. Die Ergebnisse werden in einem Excel-Dokument abgespeichert. Dem System wird nachfolgender Prompt übergeben.

task = f” “Führe folgende Aktionen durch:

1 - Analysiere die Kontexte.

2 - Falls, keiner der Kontexte die Informationen enthält, um die Frage korrekt zu beantworten, wähle genau einen Kontext aus, der die meisten Informationen enthält, um die Frage zu beantworten.

3 - Formulieren anhand des ausgewählten Kontextes deine Antwort in eigenen Worten.”“ “

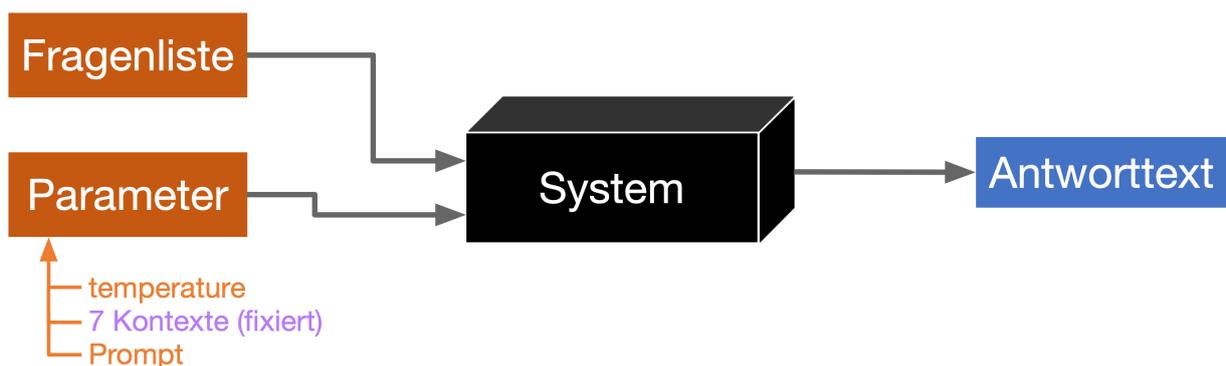


Abbildung 10: Als Inputparameter werden dem System temperature, eine fixe Anzahl Kontexte und ein Prompt übergeben. Als Output generiert das System einen Antworttext, welcher die Fragestellung konkret beantworten soll.

### 3.1.4 Experiment 4

Das vierte Experimente ist eine Anpassung des dritten Experiments. Der Parameter temperature wird von 0.5 auf 0 minimiert. Die Reliabilitätsprüfung erfolgt durch den Vergleich der neuen Ergebnisse mit denjenigen aus Experiment 2. Dies dient ebenso wie Experiment 2 zur Beantwortung der Unterleitfrage 4 (vgl. Kapitel 1.2). Die Resultate sollen in ein Excel-Dokument eingefügt und abgespeichert werden.

### 3.1.5 Experiment 5

Zur Beantwortung der in Kapitel 1.2 formulierten Unterleitfrage 6 wird Experiment 5 durchgeführt. Im Rahmen des Experiments werden erstmals zwei unterschiedliche Prompt's (task\_1, task\_2) verwendet, welche die Schritt-für-Schritt Methode und die Storytelling-Methode als Prompt-Art einsetzen. Folglich liefert das System zwei Antworttexte. Das Design von Experiment 5 ist in Abbildung 11 schematisch dargestellt. Die daraus resultierenden Ergebnisse werden in ein Excel-Dokument eingefügt und abgespeichert. Es erfolgt eine manuelle Stichprobenanalyse, um Unterschiedenen in den beiden Antworttexten zu eruieren.

Der nachfolgende Prompt stellt **task\_1** dar:

task\_1 = f"“Führe folgende Aktionen durch:

1 - Analysiere die Kontexte.

2 - Falls, keiner der Kontexte die Informationen enthält, um die Frage korrekt zu beantworten, wähle genau einen Kontext aus, der die meisten Informationen enthält, um die Frage zu beantworten.

3 - Formulieren anhand des ausgewählten Kontextes deine Antwort in eigenen Worten.”“ “

Der nachfolgende Prompt stellt **task\_2** dar:

task\_2 = Du bist ein Software-Engineer, der Fragen zu Masern korrekt beantworten soll. Du verfügst über ein medizinisches Basiswissen. Um die Aufgabe erfolgreich zu bewältigen, werden dir verschiedene Kontexte mit relevanten und irrelevanten Inhalten gegeben, die du analysieren sollst und anschliessend einen Kontext auswählst, der die Information zur Beantwortung der Frage beinhaltet. Danach sollst du in einem zweiten Schritt die Frage in eigenen Worten anhand des Inhaltes des ausgewählten Kontextes, korrekt beantworten.

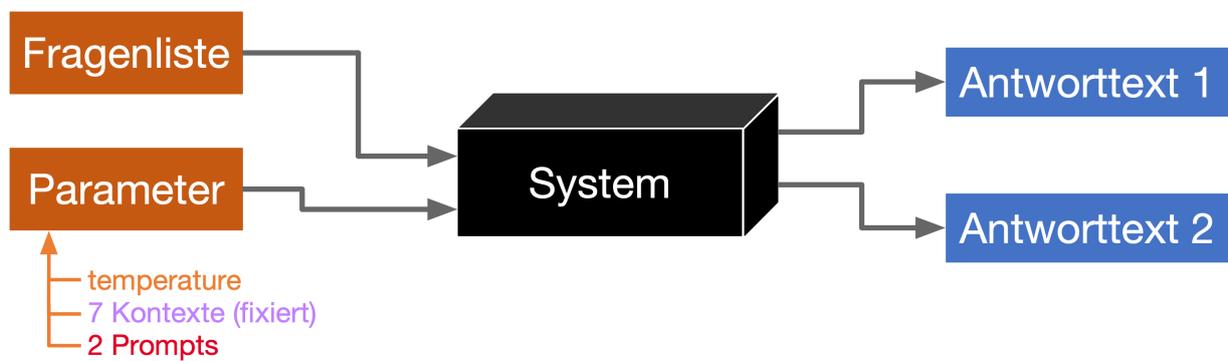


Abbildung 11: Als Inputparameter werden dem System temperature, eine fixe Anzahl Kontexte und zwei Prompt übergeben. Als Output generiert das System zwei Antworttexte, welcher die Fragestellung konkret beantworten sollen

### 3.1.6 Experiment 6

Das sechste Experiment stellt eine angepasste Version des fünften Experiments dar. Der Parameter temperature wird von 0.5 auf 0 reduziert. Mit den resultierenden Ergebnissen soll analog zum zweiten und vierten Experiment die Unterleitfrage 4 (vgl. Kapitel 1.2) beantwortet werden.

## 4. Resultate

Es ist für die beschriebenen Experimente zu betonen, dass eine Auswahl eines unzutreffenden Kontextes durch das System, also ChatGPT-4, nicht notwendigerweise zu einer fehlerhaften Beantwortung der Frage führt. Jeder Kontext bezüglich Masern repräsentiert eine Antwort auf eine spezifische Frage aus dem FAQ-Dokument zu Masern. Daher kann es vorkommen, dass bestimmte Kontexte sehr ähnliche Informationen enthalten, wodurch die Wahl eines nicht exakten Kontextes nicht unmittelbar eine inkorrekte Antwort nach sich zieht.

### 4.1 Experiment 1

Die nachfolgende Abbildung 12 visualisiert die Gesamtergebnisse des ersten Experiments in Form eines Balkendiagramms. Die x-Achse repräsentiert die Anzahl der Kontexte, während die y-Achse die Häufigkeit der “Nein”-Antworten auf die Frage nach der Verfügbarkeit eines passenden Kontextes für die Beantwortung der jeweiligen Fragestellung angibt. Die über den Balken angebrachten Zahlenwerte aggregieren die Häufigkeit der “Nein”-Antworten über sämtliche Fragen hinweg, bezogen auf die jeweilige Kontextanzahl. Diese “Nein”-Antworten reichen von null bis siebzehn. Mit zunehmender Kontextanzahl ist zunächst ein Rückgang der “Nein”-Antworten zu beobachten, bis bei sieben Kontexten keine “Nein”-Antworten mehr verzeichnet werden. Darauf folgt eine erneute Zunahme der “Nein”-Antworten. Bei der Kontextanzahl von fünfundzwanzig, siebenundzwanzig und neunundzwanzig werden analog zu sieben Kontexten keine “Nein”-Antworten festgestellt. In Anbetracht der Tokenlimitierung des GPT-4-Modells und der erhöhten Rechenzeit bei umfangreicheren Kontextmengen erscheint eine Begrenzung auf sieben Kontexte gegenüber fünfundzwanzig, siebenundzwanzig oder neunundzwanzig als die optimalere Wahl.

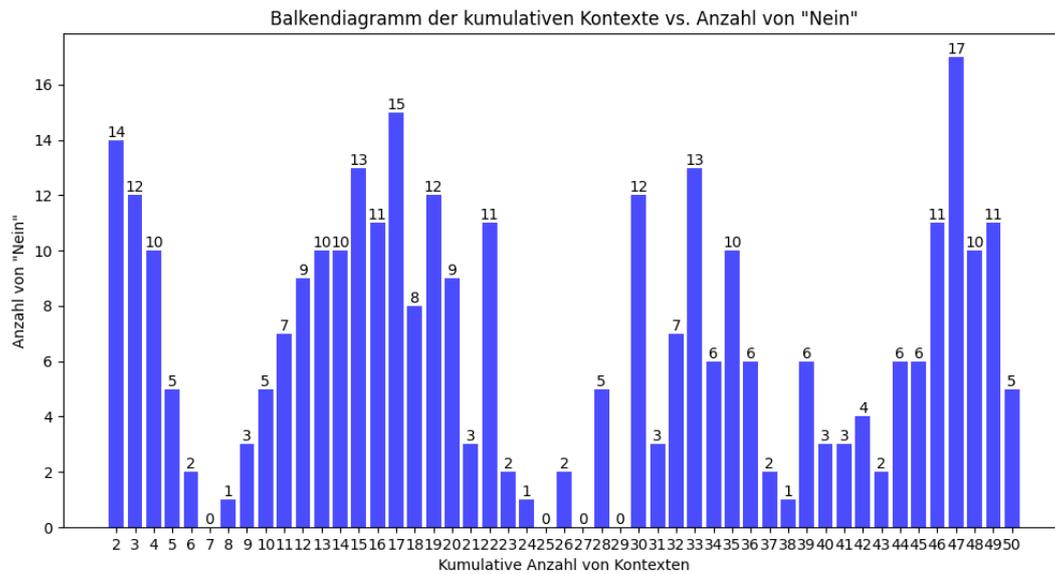


Abbildung 12: Balkendiagramm der resultierenden Ergebnisse aus Experiment 1.

## 4.2 Experiment 2

Die nachfolgende Abbildung 13 stellt ausgewählte Ergebnisse des zweiten Experiments in einer Tabelle dar. Aufgrund der geringen Variationsbreite der Werte für "Nein"-Antworten, die zwischen null und zwei liegen, wurde von einer Visualisierung als Balkendiagramm abgesehen, da diese keine signifikante Informationsvermittlung bieten würde. Die Zeilen entsprechen den einzelnen gestellten Fragen, während die Spalten die jeweilige Anzahl der zugeordneten Kontexte widerspiegeln. Die letzte Zeile der Tabelle fasst die Häufigkeit der "Nein"-Antworten über alle Fragen hinweg zusammen, basierend auf der Anzahl der Kontexte, wobei die Kontextanzahl in der Darstellung bis zu 14 reicht. Ein Wert von null wurde insgesamt 27 Mal erreicht, was teilweise durch eine gelbe Markierung in der Darstellung hervorgehoben wird. Es ist zu bemerken, dass bei einer Kontextanzahl von sieben genau zwei "Nein"-Antworten verzeichnet wurden. Angesichts der marginalen Differenz in der Anzahl der "Nein"-Antworten in Bezug auf die verschiedenen Kontextanzahlen und der Möglichkeit, den Code konsistent mit einer identischen Kontextanzahl für nachfolgende Experimente zu verwenden, wurde entschieden, eine Kontextanzahl von sechs für alle weiteren Experimente beizubehalten.



der Krankheit zu verhindern und die Kontaktpersonen zu identifizieren und ihren Immunstatus zu überprüfen.

Basierend auf Kontext 3, sollte eine gesunde, nicht immune Person, die mit einem Masernfall in Kontakt war, folgende Massnahmen ergreifen:

1. Sie sollte ihren Immunstatus überprüfen lassen, um festzustellen, ob sie gegen Masern geschützt ist oder nicht. 2. Falls sie nicht immun ist, sollte sie ihren Arzt oder ihre Ärztin kontaktieren, um weitere Anweisungen zu erhalten. Dies kann die Empfehlung einer sofortigen Impfung oder anderer vorbeugender Massnahmen umfassen. 3. Sie sollte auf mögliche Symptome der Masern achten, wie Fieber, Husten, Bindehautentzündung und rote Flecken auf der Haut. 4. Um die Verbreitung der Krankheit zu verhindern, sollte sie soziale Distanzierung praktizieren und enge Kontakte mit anderen Personen vermeiden, insbesondere mit solchen, die ein erhöhtes Risiko für Komplikationen haben, wie Säuglinge, Schwangere und immunsupprimierte Personen.

Es ist wichtig, dass sie diese Massnahmen ernst nimmt, da Masern eine hochansteckende Krankheit sind und schwerwiegende Gesundheitsprobleme verursachen können.”

**Frage 29:** “Was soll ich tun, wenn ich meinen Impfausweis nicht finde / keinen Impfausweis habe?”

**Antwort (mit falschem Kontext) ChatGPT-4:**

“Keiner der angegebenen Kontexte enthält spezifische Informationen darüber, was zu tun ist, wenn man seinen Impfausweis nicht findet oder keinen Impfausweis hat. Der Kontext, der am ehesten nützlich erscheint, um die Frage im Allgemeinen zu beantworten, ist Kontext 1, da er sich mit Impfeempfehlungen und der Möglichkeit der Suche nach Antikörpern beschäftigt.

Basierend auf Kontext 1 würde ich folgende Antwort formulieren:

Wenn Sie Ihren Impfausweis nicht finden oder keinen Impfausweis haben, sollten Sie Ihren Arzt oder Ihre Ärztin kontaktieren. Sie können mit Ihrem Arzt oder Ihrer Ärztin besprechen,

ob eine Nachholimpfung sinnvoll ist oder ob eine Antikörpersuche durchgeführt werden sollte, um festzustellen, ob Sie bereits immun gegen die betreffende Krankheit sind. Insbesondere für Personen, die nach 1963 geboren sind und sich nicht sicher sind, ob sie die Masern durchgemacht haben, wird eine Nachholimpfung empfohlen. Die Impfung einer bereits immunen Person stellt kein zusätzliches Risiko dar, aber die individuelle Situation sollte mit einem medizinischen Fachpersonal besprochen werden.”

#### 4.4 Experiment 4

Aus den resultierenden Daten von Experiment 4 geht hervor, dass für die Beantwortung einer Fragestellung , der Frage 15, der falsche Kontext herangezogen wird. Die restlichen Fragen, mit Ausnahme der Frage 45, bei welchem das System zwei Kontexte zur Beantwortung der Fragestellung herangezogen hat, wählt das System den passenden Kontext aus respektive generiert die korrekten die Antworten. Im nachfolgenden Abschnitt wird exemplarische die Frage 15 mit der entsprechenden Antwort angeführt.

**Frage 15:** “Welche Massnahmen muss eine gesunde, nicht immune Person treffen, nachdem sie mit einem Masernfall in Kontakt war?”

##### **Antwort (mit falschem Kontext) ChatGPT-4:**

“Der Kontext 3 enthält die meisten Informationen, die relevant sind, um die Frage zu beantworten, welche Massnahmen eine gesunde, nicht immune Person treffen muss, nachdem sie mit einem Masernfall in Kontakt war.

Basierend auf dem ausgewählten Kontext lautet die Antwort in eigenen Worten wie folgt:

Eine gesunde, nicht immune Person, die mit einem Masernfall in Kontakt gekommen ist, sollte umgehend ihren Immunstatus überprüfen lassen, um festzustellen, ob sie vor Masern geschützt ist oder nicht. Da Masern hochansteckend sind, ist es wichtig, die Ausbreitung der Krankheit zu verhindern. Die betroffene Person sollte sich an den Arzt oder die Ärztin wenden, um die

notwendigen Schritte zu besprechen. Dies kann bedeuten, dass die Person sich isolieren muss, um keinen weiteren Kontakt mit anderen nicht immunen Personen zu haben, insbesondere mit solchen, die ein erhöhtes Risiko für Komplikationen haben. Zudem sollte die Person ihren Impfstatus überprüfen und gegebenenfalls eine Impfung in Betracht ziehen, falls dies noch möglich ist und vom Arzt empfohlen wird.”

## 4.5 Experiment 5

Die Analyse der Daten aus Experiment 5 für `task_1` zeigt, dass das System bei der Beantwortung von Frage 15 nicht den richtigen Kontext, konkret Kontext 1, auswählt. Weiterhin wird festgestellt, dass das System bei Frage 32 keinen der verfügbaren Kontexte als zutreffend betrachtet, obwohl ein geeigneter Kontext zugänglich ist.

In Bezug auf `task_2` in Experiment 5 ergibt sich aus den Daten, dass das System bei keiner Fragestellung einen unangemessenen Kontext identifiziert, was darauf hinweist, dass nach Systemanalyse stets ein passender Kontext vorhanden ist. Dennoch wählt das System für die Fragen 15 und 32 einen nicht korrekten Kontext für die Beantwortung aus, was jedoch nicht notwendigerweise eine inkorrekte Antwort auf die Fragen bedeutet.

## 4.6 Experiment 6

Die Daten aus Experiment 6 für Aufgabe 1 verdeutlichen, dass das System die Kontexte für die Fragen 15, 29 und 32 nicht als passend einstuft, obgleich jeweils ein geeigneter Kontext verfügbar ist. Zudem wird erkannt, dass das System bei der Beantwortung von Frage 30 zwei Kontexte, nämlich Kontext 1 (der korrekte Kontext) und Kontext 6, als geeignet betrachtet.

Die Ergebnisse aus Experiment 6 für Aufgabe 1 deuten darauf hin, dass das System bei den Fragen 15 und 32 keinen der vorgegebenen Kontexte als geeignet für die Beantwortung identifiziert hat. Für die übrigen Fragen liefert das System jedoch jeweils den passenden Kontext zusammen mit der korrekten Antwort.

## 5. Diskussion

Die resultierenden Ergebnisse unterstreichen die Bedeutung eines ausgewogenen Verhältnisses zwischen den untersuchten Parametern- temperature, Kontextanzahl, Prompt-Art - für die Modelleistung. In den nachfolgenden Kapiteln wird eine detaillierterer Überblick über die Ergebnisse angeführt.

### 5.1 Interpretation der Ergebnisse

Anhand des Vergleiches zwischen den Resultaten eins und zwei, drei und vier sowie fünf und sechs ist festzustellen, dass bei jedem der Vergleiche die inkorrekte Antwort, das heisst keinen passenden Kontext oder eine falsche Kontextauswahl für die Beantwortung der Frage, mit Reduzierung des Parameters temperature abnimmt. Folglich kann die Frage, ob die temperature einen Einfluss auf die Generierung der Antworten hat mit Ja beantwortet werden. Des Weiteren ist zu beobachten, dass die Anzahl “Nein”-Antworten mit zunehmender Kontextanzahl abnimmt und danach erneut wieder steigt. Folglich ist ein Einfluss der Kontextanzahl auf die Generierung der Antworten von ChatGPT-4 festzustellen. Ferner ist bei der Untersuchung, ob die Prompt-Art einen Einfluss auf die generierten Antworten hat, kein markanter Unterschied festzustellen und daher besteht im Rahmen dieser Projektarbeit keine Evidenz, dass ein Einfluss besteht.

### 5.3 Ausblick

Durch den Vergleich der Ergebnisse aus den Paaren der Experimente 1 und 2, 3 und 4 sowie 5 und 6 wird deutlich, dass die Häufigkeit inkorrekturer Antworten – also das Fehlen eines passenden Kontextes oder eine fehlerhafte Kontextwahl – mit der Verringerung des Temperatur-Parameters abnimmt. Daraus lässt sich ableiten, dass die Temperatur tatsächlich einen Einfluss auf die Antwortgenerierung hat. Zudem zeigt sich, dass die Anzahl der “Nein”-Antworten mit steigender Kontextanzahl zunächst sinkt und anschliessend wieder ansteigt, was den Einfluss der Kontextanzahl auf die Generierung der Antworten unterstreicht. Hingegen wird bei der Betrachtung des Einflusses der Prompt-Art auf die generierten Antworten kein signifikanter Unterschied erkannt, was darauf hindeutet, dass innerhalb dieser Projektarbeit kein nachweisbarer Einfluss der Prompt-Art festgestellt werden kann.

## 6. Verzeichnisse

- [1] F. Haerer, “Conceptual model interpreter for Large Language Models”, in: Companion Proceedings of the 42nd International Conference on Conceptual Modeling: ER Forum, 7th SCME, Project Exhibitions, Posters and Demos, and Doctoral Consortium, Nov. 2023, Lisbon, Portugal.
- [2] G. Yenduri, M. Ramalingam, S.G. Chemmalar, Y. Supriya, G. Srivastava, Praveen Kumar Reddy Maddikunta, G. Deepti Raj, Rutvij H Jhaveri, B. Prabadevi, Weizheng Wang, V. Athanasios Vasilakos, Thippa Reddy Gadekallu, “GPT (Generative Pre-trained Transformer) - Comprehensive Review on Enabling Technologies, Potential Applications, Emerging Challenges, and Future Decisions”, Mai 2023.
- [3] R. Brandl, C. Ellis (19.07.2023) ChatGPT-Statistiken 2024. Alle aktuellen Zahlen über den Chatbot von OpenAI [Online] URL: <https://www.tooltester.com/de/blog/chatgpt-statistiken/> [Stand: 14.01.2024]
- [4] OpenAI (14.03.2023) Research. GPT-4 [Online] URL: <https://openai.com/research/gpt-4> [Stand: 14.01.2024]
- [5] H. Linde (Februar 2023) So funktioniert ChatGPT. Ein Deep Dive von Helmut Linde [Online] URL: <https://www.golem.de/news/kuenstliche-intelligenz-so-funktioniert-chatgpt-2302-171644.html> [Stand: 15.12.2023]
- [6] V. Nigam (Mai 2019) Natural Language Processing: From Basics to using RNN and LSTM. A detailed introduction to all the concepts prevalent in the world of Natural Language Processing [online] URL: <https://medium.com/analytics-vidhya/natural-language-processing-from-basics-to-using-rnn-and-lstm-ef6779e4ae66> [Stand: 11.12.2023]

[7] H. Linde (Februar 2023) So funktioniert ChatGPT. Ein Deep Dive von Helmut Linde [Online] URL: <https://www.golem.de/news/kuenstliche-intelligenz-so-funktioniert-chatgpt-2302-171644-3.html> [Stand: 15.12.2023]

[8] OpenAI (11.06.2018) Research. Improving language understanding with unsupervised learning [Online] URL: <https://openai.com/research/language-unsupervised> [Stand: 14.01.2024]

[9] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin, “Attention Is All You Need”, 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, USA, Dez. 2017.

[10] K. Gomez (22.12.2023) The Feedforward Demystified: A Core Operation of Transformers [Online] URL: <https://medium.com/@kyeg/the-feedforward-demystified-a-core-operation-of-transformers-afcd3a136c4c#:~:text=The%20Feedforward%20operation%20is%20a%20fundamental%20bu> [Stand: 15.01.2024]

[11] T. Huegle, “The wide range of opportunities for large language models such as ChatGPT in rheumatology”, RMD Open, Jahr 2023, Nr. 9, Apr. 2022.

[12] I. Kargar (07.01.2023) Reinforcement Learning from Human Feedback, InstructGPT, and ChatGPT [Online] URL: <https://medium.com/aiguys/reinforcement-learning-from-human-feedback-instructgpt-and-chatgpt-693d00cb9c58> [Stand: 15.12.2023]

[13] M. Alawida, A. Mejri, A. Mehmood, B. Chikhaoui, O. I. Abiodun, “A Comprehensive Study of ChatGPT: Advancements, Limitations, and Ethical Considerations in Natural Language Processing and Cybersecurity”, Information, Jg. 2023, Nr. 14, S. 462, 16.08.2023.

[14] OpenAI (xx.xx.xxxx) Fine-tuning [Online] URL: <https://platform.openai.com/docs/guides/fine-tuning/use-a-fine-tuned-model> [Stand: 14.01.2024]

[15] OpenAI (xx.xx.xxxx) How do I use Stop Sequences? [Online] URL: <https://help.openai.com/en/articles/5072263-how-do-i-use-stop-sequences> [Stand: 15.01.2024]

[16] OpenAI (xx.xx.xxxx) Authentication [Online] URL: <https://platform.openai.com/docs/api-reference/authentication> [Stand: 14.01.2024]

[17] PaperTrue (03.11.2023) 100+ Creative Writing Prompts for Masterful Storytelling [Online] URL: <https://www.papertrue.com/blog/writing-prompts/> [Stand: 15.01.2024]

[18] OpenAI (xx.xx.xxxx) Introduction [Online] URL: <https://platform.openai.com/docs/api-reference> [Stand: 14.12.2023]

## **Hintergrundlektüren**

S. Li, X. Puig, C. Paxton, Y. Du, C. Wang, L. Fan, T. Chen, D. Huang, E. Akyurek, A. Anandkumar, J. Andreas, I. Mordatch, “Pre-Trained Language Models for Interactive Decision-Making”, 36th Conference on Neural Information Processing Systems, New Orleans, Okt. 2022.

P. Dahal (xx.xx.xxxx) RLHF [Online] URL: <https://notesonai.com/RLHF+-+Reinforcement+Learning+with+Human+Feedback> [Stand: 20.12.2023]

OpenAI (xx.xx.xxxx) GPT-4 is OpenAI’s most advanced system, producing safer and more useful responses [online] URL: <https://openai.com/gpt-4> [Stand: 17.12.2023]

P. Ganesh (Dez 2019) Pre-trained Language Models: Simplified. Sesame street of the NLP world [Online] URL: <https://towardsdatascience.com/pre-trained-language-models-simplified-b8ec80c62217> [Stand: 13.12.2023]

# Abbildungsverzeichnis

1	Das Konzept von grossen Sprachmodellen basiert darauf, dass für einen gegebenen Satz eine Wahrscheinlichkeitsverteilung für das nachfolgende Wort zu berechnen, welche in der Grafik in blauen Kästchen in Prozent angegeben ist [5]. . . . .	4
2	Die Grafik stellt eine Vektorrepräsentation von Wörtern im semantischen Raum dar. Die Nähe des Wortes “Kaiser” zu “König” suggeriert, die inhaltliche Verwandtheit und werden oft in ähnlichen Kontexten verwendet. Der blaue Vektor, der als Kontextvektor zu “König” quantifiziert die Beziehung zwischen “König” und den anderen Wörtern im semantischen Raum [3]. . . . .	6
3	Die Grafik zeigt schematisch den Aufbau und die grobe Funktionsweise des GPT-3/4 Sprachmodells in drei Hauptphasen [7]. . . . .	7
4	Die Grafik illustriert den Einsatz (rote Pfeile) des Feed-Forward-Netzwerkes in der Transformer-Architektur und deutet auf die elementare Bedeutung dieses Netzwerks hin [10]. . . . .	8
5	Die Grafik stellt im Wesentlichen den Prozess der Weiterleitung von Informationen durch das Netz vom Input Layer über die Hidden Layer bis hin zum Output Layer [10]. . . . .	9
6	Die Grafik illustriert einen dreistufigen Prozess für die Feinabstimmung des Modells, [12]. . . . .	10
7	Vergleichsübersicht der fortschreitenden Entwicklungsstufen der Generative Pre-trained Transformer Modelle (GPT) von OpenAI [13]. . . . .	11
8	Ein Leistungsvergleich von diversen Sprachmodellen in verschiedenen NLP-Aufgaben [13]. . . . .	12
9	Als Inputparameter werden dem System temperature, Verschiedene Kontexte und ein Prompt übergeben. Als Output generiert das System zu jeder Frage pro Anzahl Kontexte eine Ja- oder Nein-Antwort. . . . .	16
10	Als Inputparameter werden dem System temperature, eine fixe Anzahl Kontexte und ein Prompt übergeben. Als Output generiert das System einen Antworttext, welcher die Fragestellung konkret beantworten soll. . . . .	17

11	Als Inputparameter werden dem System temperature, eine fixe Anzahl Kontexte und zwei Prompt übergeben. Als Output generiert das System zwei Antworttexte, welcher die Fragestellung konkret beantworten sollen . . . . .	19
12	Balkendiagramm der resultierenden Ergebnisse aus Experiment 1. . . . .	21
13	Excel-Ausschnitt der resultierenden Ergebnisse aus Experiment 2 . . . . .	22
14	Python-Code-Ausschnitt Funktion Anweisung für Experiment 1 und 2 . . . . .	33
15	Python-Code-Ausschnitt Funktion Anweisung für Experiment 5 und 6 . . . . .	33
16	Python-Code-Ausschnitt der Funktion für die Übergabe der Inputparameter ans System . . . . .	34

### 6.3 Abkürzungsverzeichnis

KI	Künstliche Intelligenz
LLM	Large Language Modelle
NLP	Natural Language Processing
GPT	Generative Pretrained Transformer
RLHF	Reinforcement Learning from Human Feedback
API	Application Programming Interfaces
RM	Reward Model
PPO	Proximal Policy Optimization
CLIP	Contrastive Language-ImagePre-training

## 7. Appendix

### 7.1 Python-Code Dokumentation

Einleitend werden die erforderlichen Python-Bibliothek für die Ausführung des Codes vorgestellt. Der Python-Code wurde in der webbasierten Umgebung Jupyter Notebook ausgeführt.

#### **Openai Bibliothek (pip install openai)**

Die openai Bibliothek ist das offizielle Python-Paket von OpenAI, das Entwicklern den Zugang zu OpenAIs API ermöglicht. Diese API bietet Zugriff auf fortschrittliche KI-Modelle wie GPT, DALL-E und CLIP, und ermöglicht es, verschiedenste Aufgaben wie Textgenerierung, Textanalyse, Bildgenerierung und Bildanalyse durchzuführen. Die Bibliothek ist darauf ausgelegt, eine einfache und effiziente Integration dieser Modelle in Python-Anwendungen zu ermöglichen, um KI-basierte Lösungen zu entwickeln. Anhand einer API sind Interaktionen über HTTP-Anfragen möglich, welche eine Authentifizierung erfordert. Diese wird durch eine API respektive einem API-Schlüssel ermöglicht und kann auf dem persönlichen OpenAI Konto generiert werden. Aus Sicherheitsgründen empfiehlt es sich den API-Schlüssel als Umgebungsvariable in das Python-Skript zu laden [18].

#### **Dotenv-Bibliothek (pip install dotenv)**

Die dotenv Bibliothek wird verwendet, um Umgebungsvariablen aus einer .env-Datei zu laden. Wenn man `load_dotenv()` aus dem dotenv Paket aufruft, liest diese Funktion die Schlüssel-Wert-Paare aus der .env-Datei und setzt sie als Umgebungsvariablen. Damit kann der persönliche API-Schlüssel für die Verwendung der Openai-Bibliothek sicher in das Python-Skript geladen werden [18].

## Funktion create-task für Experiment 1 und 2

Die Funktion `create_task` nimmt jeweils die Parameter `question`, `contexts` als Liste entgegen und erzeugt eine formatierte Anweisung respektive Aufgabenstellung.

```
def create_task(question, contexts):
    task = "Führe folgende Schritte durch:\n"
    task += "1 - Analysiere die Kontexte.\n"
    task += "2 - Falls, du einen passenden Kontext findest, antworte mit: Ja + passende Kontextnummer. Falls du kein"
    task += f"Frage: {question}\n"
    for i, context in enumerate(contexts):
        task += f"Kontext {i+1}: {context}\n"
    return task
```

Abbildung 14: Python-Code-Ausschnitt Funktion Anweisung für Experiment 1 und 2

## Funktion create-task für Experiment 5 und 6

Die Funktion `create_task` nimmt jeweils die Parameter `question`, `contexts` und `task_type` entgegen. Im Vergleich zur `create_task` Funktion von Experiment 1 und 2 werden zwei `tasks` mittels `for-elif`-Schleife übergeben. Der Rest der Funktion ist analog zur `create_taks` Funktion von Experiment 1 und 2.

```
def create_task(question, contexts, task_type):
    if task_type == "task_1":
        task = f""Führe folgende Aktionen durch:
        1 - Analysiere die Kontexte.
        2 - Falls, keiner der Kontexte die Informationen enthält, um die Frage korrekt zu beantworten, wähle genau e
        3 - Formuliere anhand des ausgewählten Kontextes deine Antwort in eigenen Worten.""
    elif task_type == "task_2":
        task = ""Du bist ein Software-Engineer, der Fragen zu Masern korrekt beantworten soll. Du verfügst über ein
    else:
        raise ValueError("Ungültiger task_type. Verwenden Sie 'task_1' oder 'task_2'.")

    task += f"\nFrage: {question}\nKontexte:\n"
    for i, context in enumerate(contexts):
        task += f"{i+1}: {context}\n"
    return task
```

Abbildung 15: Python-Code-Ausschnitt Funktion Anweisung für Experiment 5 und 6

## Funktion send-task

Die Funktion `send_task` wird in jedem Experiment aufgerufen und verwendet `task` und `temperature` als Inputparameter. Mittels `load_dotenv` wird der API-Schlüssel als Umgebungsvariable geladen. Innerhalb der `while`-Schleife wird über die `client.chat.completions.create` Funktion versucht eine Antwort von ChatGPT-4 respektive dem zugrunde liegenden Modell `gpt-4-1106-preview`, welches 128'000 annehmen kann. Im Erfolgsfall gibt die Funktion die Antwort des Modells zurück. Andernfalls prüft der Code, ob es sich um einen Rate-Limit-Fehler handelt (zu viele Anfragen in kurzer Zeit). Falls ja, wartet das Skript 60 Sekunden mittels, was mittels `time.sleep(60)` generiert wird.

```
def send_task_to_openai(task, temperature):
    load_dotenv(env_file)
    client = OpenAI()
    while True:
        try:
            response = client.chat.completions.create(
                model="gpt-4-1106-preview",
                messages=[{"role": "user", "content": task}],
                temperature=temperature
            )
            return response.choices[0].message.content.strip()
        except Exception as e:
            # Check if it's a rate limit error
            if 'rate limit' in str(e).lower():
                print(f"Rate limit reached, waiting for 60 seconds to retry: {e}")
                time.sleep(60) # Wait for 60 seconds before retrying
            else:
                print(f"Ein Fehler ist aufgetreten: {e}")
                return "Fehler"
```

Abbildung 16: Python-Code-Ausschnitt der Funktion für die Übergabe der Inputparameter ans System

## Funktion perform-experiment

```
def perform_experiment(contexts_df, questions_df, temperature, questionrange, maxrange):
    experiment_results = []

    # Äußere Schleife für die Anzahl der Fragen
    for question_index in range(3, questionrange):
        question_results = [] # Liste für die Ergebnisse dieser Frage
        # Innere Schleife für die Anzahl der Kontexte
        for num_contexts in range(1, maxrange):
            selected_contexts = contexts_df.sample(n=num_contexts)['Inhalt'].tolist()
            question = questions_df.iloc[question_index]['Frage']
            task = create_task(question, selected_contexts)
            result = send_task_to_openai(task, temperature)
            question_results.append(result) # Ergebnis für diese Frage und Anzahl von Kontexten speichern
        experiment_results.append(question_results) # Ergebnisse für alle Fragen speichern

    return experiment_results
```