Daniel Koch
Andreas Kohne
Nils Brechbühler

Prompt Engineering im Unternehmen – eine Einführung

Wettbewerbsvorteile durch generative KI und Large Language Models

Springer Gabler

Prompt Engineering im Unternehmen – eine Einführung

Daniel Koch • Andreas Kohne •
Nils Brechbühler

Prompt Engineering im Unternehmen – eine Einführung

Wettbewerbsvorteile durch generative KI und Large Language Models

Springer Gabler

Daniel Koch
Valisory GmbH
Auetal, Deutschland

Andreas Kohne
Valisory GmbH
Hessisch Oldendorf, Deutschland

Nils Brechbühler
Valisory GmbH
Barsinghausen, Deutschland

ISBN 978-3-658-47698-4 ISBN 978-3-658-47699-1 (eBook)
https://doi.org/10.1007/978-3-658-47699-1

Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über https://portal.dnb.de abrufbar.

Planung/Lektorat: Imke Sander
Springer Gabler ist ein Imprint der eingetragenen Gesellschaft Springer Fachmedien Wiesbaden GmbH und ist ein Teil von Springer Nature.
Die Anschrift der Gesellschaft ist: Abraham-Lincoln-Str. 46, 65189 Wiesbaden, Germany

Geleitwort von Prof. Dr. Knut Linke

In einer zunehmend digitalisierten und selbstorganisierenden Arbeitswelt nimmt der Bereich der Künstlichen Intelligenz (KI) eine zentrale Rolle ein. Die rasante Entwicklung von Technologien wie den Generative Pretrained Transformers (GPTs) und vergleichbarer Assistenten hat das Potenzial, Prozesse und die Art und Weise, wie Arbeit organisiert und ausgeführt wird, zu revolutionieren. In diesem Kontext gewinnt das Konzept des „New Work" – der Umbau traditioneller Arbeitsmodelle hin zu flexibleren, innovativeren und menschenzentrierteren Strukturen – weiterhin mehr an Bedeutung.

In dem vorliegenden Buch bieten die Autoren Ihnen eine grundlegende, fundierte und systematische Einführung in das Themengebiet des „Prompt Engineerings". Ein Werkzeug, welches zentral bei der effektiven Nutzung von KI ist. Die Buchautoren betrachten in diesem Buch die notwendigen, verschiedenen Aspekte des Prompt Engineerings und des damit verbundenen Bereiches der generativen KI.

Neben der Einführung, wie sich KI als disruptive Technologie durchsetzen wird, werden hier die Grundlagen der KI vorgestellt und erklärt. Dieses ist insbesondere wichtig, da die hier erklärten Begrifflichkeiten durchaus als Allgemeinwissen im Bereich der KI und im Kontext von dieser bezeichnet werden können. Durch das Verstehen dieser Grundlagen ist auch die Anwendung von KI-Systemen wie ChatGPT, Claude oder Microsoft Copilot einfacher. Sie als Leser erhalten hier einen Über-

blick über die Grundlagen der KI-Modelle, insbesondere der GPTs, und deren technische Details, wie Tokens und Kontextfenster. Dies legt das notwendige Fundament für ein tiefes Verständnis der darauf aufbauenden Inhalte des Buches.

Nach den Grundlagen des Prompt Engineerings im zweiten Kapitel, in dessen Rahmen die Konzepte hinter dem Ansatz von Prompting und erste Interaktionsformen mit KI-Systemen erläutert werden, werden fortgeschrittene Ansätze schrittweise vorgestellt. Dieses vermittelte Verständnis von Prompts und für das Prompting ist notwendig, damit KI-Systeme richtig und erfolgreich angesteuert werden können. Es gilt zu beachten, dass das in diesem Buch vermittelte Wissen für alle generativen KI-Systeme einheitlich anwendbar ist.

Aus dieser Notwendigkeit, der sinnvollen und erfolgreichen Steuerung von KI-Systemen durch durchdachtes Prompting, werden im Folgenden verschiedene Prompting-Frameworks vorstellt und zur Anwendung gebracht. Diese Frameworks, wie das R-T-F-, T-A-G-, B-A-B-, C-A-R-E- und R-I-S-E-Framework, bieten strukturierte Ansätze, um die Interaktion zwischen Mensch und KI-Systemen zu systematisieren und zu optimieren.

Gerade im Umfeld von New Work, wo sich Menschen im hohen Grad selbst organisieren und Entscheidungen eigenständig fällen müssen, wird die Nutzung von KI-Systemen zur Reflektion und zum Austausch weiter stark zunehmen. Hierfür sind die verschiedenen Prompting-Frameworks sehr gut geeignet, um überdurchschnittliche Ergebnisse zu erzielen. Die klare Struktur und die detaillierte Beschreibung der Frameworks ermöglichen es zudem, diese direkt in der Arbeitswelt anzuwenden und damit die Qualität der Arbeit zu steigern.

Die im weiteren Verlauf des Fachbuchs vorgestellten Techniken, wie Zero-, One- und Few-Shot-Prompts sowie Chain-of-Thought-Prompts, bieten weiterführende Ansätze, wie KI-Systeme noch optimaler genutzt werden können. Sie geben inhaltlich einen Einblick, in die Zukunft von KI-Systemen. Aktuell aufkommende Thinking-KI-Systeme mit Thinking-Tags verwenden ähnliche Ansätze, wie die Methoden, die in hier vorgestellt werden.

Durch die Aneignung dieser fortgeschrittenen Methoden können Unternehmen und ihre Mitarbeiter besser die Abläufe in KI-Systemen verstehen und damit, hoffentlich, auch neue und innovative Prozesse und Produkte entwickeln, die den Anforderungen der Welt gerecht werden.

Prof. Dr. Knut Linke

Professor für Informatik an der Internationalen Hochschule (IU)

Herr Prof. Dr. Linke betreut und gestaltet, neben seiner Tätigkeit als Professor im dualen Studium der IU, verschiedene deutsch- und englischsprachige Bachelor- und Master-Kurse im Bereich KI-Prompting im Fernstudienangebot der IU.

Vorwort

Künstliche Intelligenz (KI) ist seit der Einführung von ChatGPT der Firma OpenAI Ende 2022 in aller Munde. Vor allem für den Bereich der generativen KI, also den Systemen, die Texte, Bilder, Videos oder Audio (z. B. Sprache oder Musik) erzeugen, hat es weltweit für einen Durchbruch in der Wahrnehmung und der Anwendung gesorgt.

Insbesondere die großen Sprachmodelle (Large Language Models, LLM) haben inzwischen große Einsatzzahlen. Und das überall: von der Industrie über die Verwaltung bis hin in den privaten Bereich. KI ist dabei kein lokales Phänomen oder etwas, was nur einzelne Branchen oder Länder betrifft. KI betrifft inzwischen alles und jeden.

Trotz der allgegenwärtigen medialen Berichterstattung sind die Themen KI und vor allem die Chancen und Möglichkeiten zur Optimierung von Geschäfts- und Verwaltungsprozessen immer noch nicht überall angekommen. Dies hat unterschiedliche Gründe. Einer der Hauptgründe ist das fehlende Know-how in der Industrie und der öffentlichen Verwaltung.

Wir empfehlen inzwischen, dass sich jeder mit diesem aktuellen und zukunftsweisenden Thema auseinandersetzen sollte. Dabei nutzen wir gerne eine kleine Metapher:

Jedes kleine Kind lernt sehr früh in seinem Leben, dass es im Normalfall links oder rechts neben einer Tür einen Lichtschalter an der Wand gibt. Ein Druck auf den Schalter – und das Licht geht an. Wir machen

uns hierdurch auf einfache Art und Weise eine Technologie nutzbar, die viele Menschen auch im Erwachsenenalter noch nicht vollständig verstanden haben: Was genau ist eigentlich Strom? Wo kommt er her? Wo geht er hin? Und was genau sind die physikalischen Grundlagen und Gesetzmäßigkeiten? All das muss ich nicht verstehen. Trotzdem kann ich mir die Technologie mit einem Grundverständnis nutzbar machen. Auch das Montieren einer neuen Lampe oder das Tauschen eines Leuchtmittels werden viele selbstständig erledigen können. Neben dem Nutzen wurde hoffentlich ebenfalls früh im Leben eines Kindes vermittelt, dass es auch Gefahren gibt. So sollten keine Kabelenden, die vielleicht in einem Rohbau aus der Wand ragen, angefasst und nichts, vor allem nichts Leitendes, oder sogar die Finger in eine Steckdose gesteckt werden, da akute Gefahr droht. Auf dieser Ebene sollte KI zeitnah allen verständlich sein, da sich die Technologie mit einer rasanten Geschwindigkeit ausbreitet und weiterentwickelt. Für eine erfolgreiche Nutzung muss ich nicht auf tiefster mathematischer Ebene verstehen, wie die Systeme arbeiten. Trotzdem ich kann mir ihre Vorteile zu Nutze machen.

In vielen Unternehmen ist die transformative Kraft der KI-Technologie leider noch nicht verstanden worden. Dies liegt in Teilen auch daran, dass ein „einfaches" Nutzen der neuen Technologie manchmal zu minderwertigen Ergebnissen führt, die hinter der Erwartungshaltung zurückbleiben. Viele unserer Kunden sagen, dass sie mit Hilfe von ChatGPT und Co. versucht hätten eine Aufgabe zu lösen, das Ergebnis aber enttäuscht hätte. Oft folgt darauf leider der Entschluss, die Technologie zumindest für den Moment als nicht ausgereift abzustempeln und weiter „business as usual" zu machen. Unserer Erfahrung nach liegt das Hauptproblem in solchen Fällen darin, dass die Anwender in eine typische Falle der KI getappt sind: Sie denken, da die KI-Systeme (und vor allem die LLMs) sich „einfach" mit natürlicher Sprache steuern lassen, brauche es keine weitere Anleitung oder Ausbildung für die Bedienung. Leider ist das falsch. Die Systeme liefern zwar Antworten auf die gestellten Fragen (die sogenannten Prompts), diese haben aber oft Fehler, sind uneindeutig, nicht spezifisch genug oder sind schlichtweg falsch. Dies sorgt verständlicherweise für Unmut und Unverständnis beim Anwender.

Die Lösung dieses Problems heißt: Prompt Engineering. Damit sind Techniken gemeint, die die Anfragen an KI-basierte Systeme so struktu-

rieren, dass sie bestmögliche Ergebnisse liefern. Oft ist es nämlich so, dass mit einem schlechten Prompt auch bei einem guten LLM schlechte Ergebnisse erzielt werden, aber mit einem guten Prompt selbst bei einem weniger guten LLM noch sehr gute Ergebnisse erzielt werden können.

Mit Hilfe der Techniken aus dem Prompt Engineering können somit sehr gute Anfragen an KI-Systeme erstellt werden, die dann auch die gewünschte Antwortqualität liefern.

Wir sehen das Prompt Engineering als einen wichtigen „Future Skill" an, der hilft, sich jetzt und in der Zukunft in der Welt der KI-Systeme sicher bewegen zu können.

In diesem Buch haben wir unsere Erfahrungen aus Projekten, Workshops und Schulungen bei Kunden in der Industrie, der Verwaltung und im Bereich Rundfunk und Fernsehen der letzten Jahre zusammengefasst und so aufbereitet, dass die unterschiedlichen Konzepte der Arbeit mit KI-basierten Systemen leicht nachvollzogen und sofort im Alltag eingesetzt werden können.

Mit diesem Buch möchten wir dazu anregen, sich aktiv mit der Technologie auseinanderzusetzen und die Chancen und Möglichkeiten, die sich für Unternehmen ergeben, zu nutzen.

Getreu unserem Motto: „Gemeinsam – mutig – wachsam" wünschen wir Ihnen viel Erfolg auf Ihrem Weg in Richtung einer erfolgreichen KI-Nutzung.

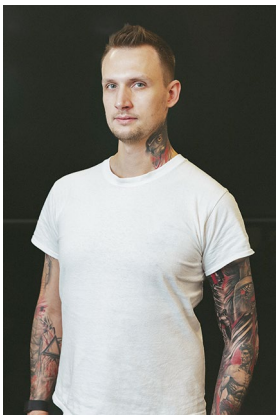
Daniel Koch
Dr. Andreas Kohne
Nils Brechbühler
Geschäftsführer der Valisory GmbH
Zuständig fürs Machen!
Hannover, Januar 2025

Anmerkung

Der Text in diesem Buch ist aus Gründen der Lesbarkeit ausschließlich im generischen Maskulinum verfasst. Wir sprechen aber ausdrücklich alle Menschen und Geschlechter gleichermaßen an.

Inhaltsverzeichnis

Über die Autoren



Daniel Koch Als Multi-Unternehmer und Experte in den Bereichen Digitalisierung und Agile Transformation verbindet Daniel Koch seit über einem Jahrzehnt fundiertes technologisches Fachwissen mit betriebswirtschaftlicher Expertise. Seine Publikationen zu Wirtschaft und Technologie bieten praxisnahe Einblicke in die digitale Transformation von Unternehmen. Mit seinem Hintergrund in der Softwareentwicklung und als zertifizierter Agile Coach versteht er es, technologische Innovationen erfolgreich in Unternehmensprozesse zu integrieren. Seine akademische Expertise als Dozent an der Hochschule Weserbergland und der Beruflichen Hochschule Hamburg sowie seine Position als Mitglied des Hochschulsenats der HSW unterstreichen seine Kompetenz in Theorie und Praxis. Daniel Koch hat einen B.Sc. in Wirtschaftsinformatik sowie einen MBA in General Management und nutzt diese interdisziplinäre Ausbildung, um komplexe digitale

Transformationsprozesse erfolgreich zu gestalten. Als Geschäftsführer der Valisory GmbH verantwortet er die strategische und operative Umsetzung des Marketings und Vertriebs.

Die Valisory GmbH macht mittelständische Unternehmen in nur wenigen Tagen KI-ready und begleitet von ersten Einblicken bis zur erfolgreichen Implementierung. Mit praxiserprobten Workshops und direkter Umsetzungsunterstützung erreichen die Kunden der Valisory schnell messbare Ergebnisse, ohne sich in technischen Details zu verlieren.

Dr. Andreas Kohne Als Experte für Innovation, Transformation und Kommunikation veröffentlicht Andreas Kohne relevantes Fachwissen auf prägnante und verständliche Art und Weise. Seine Publikationen gehören national und international zur Standardlektüre in Wirtschaft und Wissenschaft und erscheinen in deutscher und englischer Sprache.

Der Autor und Herausgeber arbeitet als erfahrener Unternehmensberater und begleitet seine Kunden auf dem Weg in eine erfolgreiche und relevante Zukunft. Er ist Gründer und Geschäftsführer der beiden Firmen Valisory GmbH und itkon GmbH. Als gefragter Redner vermittelt Andreas Kohne praxisrelevantes Fachwissen. Mit einer gelungenen Mischung aus Expertise, Interaktion und Motivation ist er als Tech-Translator national und international unterwegs. Dabei gelingt es ihm, komplexe digitale Strukturen und Prozesse allgemein verständlich zu veranschaulichen. Aktuell begleitet er seine Kunden in der Industrie, der öffentli-

chen Verwaltung sowie dem Rundfunk und Fernsehen bei der zukunftsweisenden, gewinnbringenden und sicheren Einführung von Künstlicher Intelligenz.

Zusätzlich arbeitet er als Dozent in den Bereichen Betriebswirtschaftslehre und Informatik an der Hochschule Fresenius und der Hochschule Weserbergland. Er ist mehrfach ausgezeichneter TOP EXPERTE der BVMID (Bundesvereinigung Mittelstand in Deutschland).

Andreas Kohne ist berufener Senator im Europäischen Wirtschaftssenat.

Nils Brechbühler Nils Brechbühler ist ein erfahrener Praktiker und führender Experte für digitale Transformation sowie die Modernisierung der öffentlichen Verwaltung. Als Gründer und Geschäftsführer der Valisory GmbH und der Brechbühler GmbH begleitet er öffentliche Institutionen, Unternehmen und den öffentlichen Rundfunk bei der erfolgreichen Umsetzung von Digitalisierungs- und Transformationsprojekten.

Mit über 15 Jahren Erfahrung im öffentlichen Sektor, ergänzt durch Führungspositionen bei IT-Dienstleistern und international tätigen Beratungsunternehmen, verfügt er über ein tiefes Verständnis für die spezifischen Herausforderungen und Bedürfnisse von Verwaltung, Wirtschaft und Medien. In über 1000 Workshops und mehr als 500 Projekten hat er bewiesen, wie komplexe Themen wie Verwaltungsmodernisierung, digitale Transformation, Führung und Kommunikation in greifbare und nachhaltige Lösungen übersetzt werden können.

Nils Brechbühler ist nicht nur Mitglied und Redner im Nationalen E-Government Kompetenzzentrum (NEGZ), sondern auch ein gefragter Vortragender an Hochschulen, auf Fachkongressen und in Arbeitsgruppen auf allen föderalen Ebenen. Aktuell setzt er sich intensiv mit den Potenzialen und Herausforderungen des Einsatzes künstlicher Intelligenz in der öffentlichen Verwaltung auseinander, um diese Technologien praxisnah und verantwortungsvoll in Transformationsprozesse zu integrieren.

1

Einleitung

Zusammenfassung In diesem Kapitel werden zunächst die Auswirkungen der Künstlichen Intelligenz (KI) auf Unternehmen erläutert. Dazu werden mögliche Anwendungsfälle aber auch Grenzen der Technologie aufgezeigt. Anschließend wird beschrieben, wie sich Unternehmen der Technologie schrittweise nähern können. Abschließend werden die Auswirkungen von KI auf die Rollen und Fähigkeiten der Mitarbeit erläutert.

KI (KI) scheint ein für Unternehmen omnipräsentes Thema geworden zu sein. Verständlicherweise, denn die Möglichkeiten sind oft beeindruckend bis atemberaubend. Mit den richtigen Werkzeugen können Unternehmen schneller und effizienter arbeiten, dabei Kosten reduzieren, Mitarbeiter entlasten, die Transparenz innerhalb des Unternehmens erhöhen – und vieles mehr.

Vor allem der Teilbereich der generativen KI findet mehr und mehr Einzug in den Alltag vieler Unternehmen. Um die Vorteile dieser Systeme nutzen zu können, müssen sich Unternehmen mit der Technologie beschäftigen und diese sinnvoll und effektiv einsetzen. Ein sehr wichtiger

D. Koch et al., *Prompt Engineering im Unternehmen – eine Einführung*,
https://doi.org/10.1007/978-3-658-47699-1_1

Teil davon ist das Prompt Engineering, das sich mit der Frage beschäftigt, wie Anfragen an solche Systeme geschrieben sein müssen, um die besten Ergebnisse zu erhalten.

Im Rahmen dieses Buches wird die Funktionsweise von KI im Allgemeinen sowie generativer Sprachmodelle im Besonderen erläutert und die Relevanz und die Möglichkeiten für Unternehmen werden dargestellt. Es wird gezeigt, wie gute Anfragen an solche Sprachmodelle geschrieben werden. Dazu werden verschiedene erprobte Strukturen und Techniken erläutert. Im weiteren Verlauf wird gezeigt, wie Anfragen mittels Sprachmodellen generiert werden können (Metaprompting). Abschließend wird noch dargestellt, wie Prompt Engineering bei der Konfiguration von autonomen KI-Agenten zum Einsatz kommt.

1.1 KI – eine disruptive Technologie für Unternehmen?

Seit etwa Ende 2022 scheint es im Technologiediskurs nur noch ein Thema zu geben: KI. Ob in der Gesellschaft, Politik, Verwaltung oder in der Wirtschaft – KI scheint omnipräsent. KI erlebt einen ähnlichen Hype wie andere Innovationen (beispielsweise die Blockchain-Technologie oder Virtual Reality und Augmented Reality), allerdings mit zwei signifikant anderen Prämissen: Erstens viel breiter und sichtbarer und zweitens mit der Erwartung, dass trotz des Abflachens des Hypes, der Einfluss auf die Gesellschaft nachhaltig ist. Während beispielsweise Implementierungen von Blockchains wie Bitcoin zwar mittlerweile durchaus einer breiten Masse bekannt, aber hinter den Erwartungen und Hoffnungen zurückgeblieben sind, hat es KI in nur zwei Jahren dieses Hypes geschafft, in Rundfunk, Verwaltung, Wirtschaft und bei Privatpersonen eine Veränderung des Blicks auf Technologie und den Umgang mit dieser herbeizuführen.

Das Thema der KI ist dabei keineswegs ein neues. Eine der ersten Forschungsarbeiten dazu stammt aus dem Jahr 1950. Sie trug den Titel „Computing Machinery and Intelligence“ und wurde von Alan Turing geschrieben (Turing, 1950). Seitdem wurde immer aktiver in diesem Feld geforscht und seit vielen Jahren kommt KI an vielen Stellen zum Einsatz;

ob zur Erkennung von Terrorfinanzierung bei Banktransaktionen, Vorhersage von Ausfällen von Industrieanlagen oder bei der Projektion von Unternehmensentwicklungen mittels Business-Intelligence-Systemen. KI ist schon länger bei vielen digitalen Werkzeugen ein integraler Bestandteil.

Der große Durchbruch, der zu der aktuellen Situation von KI-Systemen führte, war ein wissenschaftliches Paper mit dem unscheinbaren Titel „Attention Is All You Need", das 2017 von acht Forschern von Google veröffentlicht wurde (Vaswani, 2017). Es baute auf der Idee einer Veröffentlichung aus dem Jahr 2014 auf und beschrieb eine neuartige Architektur des Deep Learnings. Die dort beschriebene Architektur namens „Transformers" ist der Kern von Large Language Models (LLMs) wie z. B. ChatGPT und Co. Diese großen Sprachmodelle können natürliche Sprache verstehen und produzieren (wie das funktioniert, wird in Abschn. 1.2 erklärt). Ende 2022 wurde dann vom amerikanischen Unternehmen OpenAI das Tool ChatGPT vorgestellt, eine chatbasierte KI, die im Kern ein LLM ist. Seit diesem Moment ist das Thema und das Verständnis für seine Möglichkeiten in der breiten Öffentlichkeit angekommen.

In Anbetracht der Präsenz und scheinbaren Vorteile des Themas ist es für Unternehmen wichtig, für sich den Umgang damit zu definieren. Das Thema vollkommen zu ignorieren, ist der fatal falsche Weg. KI hat schon jetzt großen Einfluss auf Unternehmen. Sie wird sich zwar wandeln und weiterentwickeln, aber nicht mehr verschwinden. Sie zu ignorieren würde bedeuten, den Anschluss zu verlieren, hinter den eigenen Möglichkeiten zurückzubleiben und die Wettbewerbsvorteile der Konkurrenz zu überlassen.

Aber auch in der Beschäftigung mit KI gibt es zwei verschiedene Standpunkte. Unternehmen können stark in Innovationen investieren und mit neuen Plattformen und Technologien experimentieren. Diese „Play to win"-Mentalität kann viele Wettbewerbsvorteile bringen, bedarf aber auch einer gewissen Fehlerkultur, denn mit der hohen Geschwindigkeit der Entwicklung kommen und gehen KI-Werkzeuge teilweise täglich. Unternehmen müssen bereit sein, das Risiko einzugehen, Fehler zu machen und daraus zu lernen, denn sie befinden sich auf größtenteils unbekanntem Terrain. Alternativ kann der Fokus auch auf das Einsetzen er-

probter Lösungen gelegt werden. Dabei werden die Fehler und Stolperfallen anderen überlassen. Mit dieser „Play to not lose"-Einstellung folgen Unternehmen dem Thema in moderaterer, aber für sich passender, Geschwindigkeit und gehen weniger Risiko ein.

Für welchen Weg sich Unternehmen auch immer entscheiden: Das ganze Thema KI sollte nicht ignoriert werden, denn es wird Gesellschaft, Verwaltung und Wirtschaft in den nächsten Jahren weltweit sehr nachhaltig prägen.

1.1.1 Der Durchbruch von KI für die breite Masse

Wenn nun KI an sich kein innovatives neues Themenfeld ist – weshalb hat sie dann kürzlich einen solchen Hype ausgelöst? Anders als die bisherigen KI-Systeme arbeiten Sprachmodelle generativ, das heißt, sie generieren etwas Neues. Bei Sprachmodellen handelt es sich dabei erst einmal um Text, aber das Feld generativer KI ist deutlich umfassender. Mittlerweile können Musik, Bilder und Fotos in guter Qualität erzeugt werden, indem man in natürlicher Sprache beschreibt, welches Ergebnis man haben möchte.

Bisherige KI-Systeme haben aus Daten, zumeist numerischer Natur, gelernt und daraus Schlüsse gezogen. Die Implementierung solcher Systeme ist unter Umständen recht technisch und für den interessierten Laien nur schwer greifbar. Anwendungsfälle im Unternehmen gibt es viele, KI stand dabei oft im Hintergrund – weder war sie besonders erwähnenswert, noch bestand viel Interesse daran, wie die Ergebnisse in Anwendungsfällen konkret bestimmt wurden.

Mit dem Aufkommen moderner generativer KI ändert sich der Blick auf KI schlagartig. Systeme wie Claude von Anthropic für Chats und Texte, Midjourney zum Erzeugen von Bildern oder Veo von Google zum Erstellen von Videos sind zwar weiterhin technisch hochkomplexe Systeme, aber aufgrund der integrierten Sprachmodelle in natürlicher Sprache zu bedienen. Statt Programmiercode zu schreiben und Datenmodelle zu trainieren, um am Ende einen menschlich klingenden Text zu erhalten, könnten Nutzer heutzutage in ihren eigenen Worten eine Aufgabe oder ein Ziel beschreiben und erhalten eine passende Antwort.

Viele Aufgaben, die mit Wissen zu tun haben, können leicht durch LLMs unterstützt oder bald gar komplett automatisiert werden. Für Unternehmen ergibt sich mit dieser Art von KI ein nicht zu unterschätzender Return on Invest – je nach System sind die Kosten überschaubar und die Gewinne in Effizienz und Qualität sofort messbar. Vielmehr noch, aufgrund der Einfachheit in der Nutzung können solche Systeme leicht in der Breite ausgerollt werden.

Unternehmen können dadurch in ihren Geschäftsprozessen schneller agieren und senken dadurch am Ende die Kosten der Wertschöpfung. Damit wird vorhandenes Personal entlastet und in seinen Routineaufgaben unterstützt.

1.1.2 Die Grenzen Künstlicher Intelligenz

KI ist dabei kein Allheilmittel und den vielen Vorteilen stehen auch zahlreiche Nachteile gegenüber. Fundamental ist KI eine komplexe Technologie, die im Eigenbetrieb (z. B. im Rechenzentrum eines Unternehmens) je nach Szenario viel Fachwissen und Infrastruktur benötigt – auch wenn die Weiterentwicklung der KI-Systeme, Tools und Plattformen den Einsatz deutlich vereinfacht hat.

Ein weitaus gewichtiger Aspekt ist der notwendige Ressourceneinsatz. Für die Technologie werden vor allem mathematische Operationen (z. B. Matrizenmultiplikationen) in großer Menge und Geschwindigkeit ausgeführt. Für diese Art von Operationen eignen sich vor allem Grafikkarten (GPUs), denn Computergrafik zu zeichnen besteht ebenso vor allem aus Mathematik, weshalb GPUs genau dafür optimiert wurden. Das führt dazu, dass vor allem seit der Veröffentlichung von ChatGPT der Bedarf an Grafikkarten weltweit signifikant zugenommen hat. Da die Produktion von GPUs auch seltene Erden benötigt, die teilweise unter erschwerten Bedingungen abgebaut werden, ist das Kaufen und Betreiben eigener KI mittels Grafikkarten ein wichtiges Thema für Unternehmen, die freiwillig oder verpflichtend unter die Bestimmung des Lieferkettensorgfaltspflichtengesetzes fallen. Neben der Hardware braucht es dann aber vor allem auch Energie, und zwar so viel, dass Betreiber von KI-Systemen wie Microsoft vereinzelt Kernkraftwerke anmieten, um ihre

Rechenzentren exklusiv mit Strom zu versorgen. Im Zuge der Nachhaltigkeitsbestrebungen vieler Unternehmen kann die Verwendung von KI durchaus in die Berechnung des CO_2-Fußabdrucks mit einfließen.

Neben den Rahmenbedingungen bringt auch die Technologie selbst verschiedene Grenzen mit sich, derer man sich bewusst sein sollte. Bei generativer KI wie z. B. einem ChatGPT von OpenAI, einem Claude von Anthropic oder einem Gemini von Google ist ein signifikanter Faktor für die Fähigkeiten der Modelle die schiere Masse an Training, die die Modelle erfahren haben. Dieses Training funktioniert keineswegs so, dass einfach Informationen und Daten in das Modell gegeben werden und dieses dann trainiert ist. Zum einen müssen diesen Trainingsdaten Metadaten beigefügt werden, damit das Modell versteht, was diese Informationen bedeuten. Zum anderen müssen die Ergebnisse für Anfragen auf ihre Qualität (Korrektheit, Eindeutigkeit, Kohärenz und weitere) bewertet werden. Die Meta-Informationen in den Trainingsdaten werden über das sogenannte Labelling produziert; Eingabedaten werden mit Labels (kurzen Texten) beschrieben. Ausschnitte von Bildern können beispielsweise mit den Labels „Hund", „Statue" und „weiß" beschrieben werden, während Texte z. B. mit den Labels „Gedicht", „poetisch" und „dramatisch" klassifiziert werden können. Die KI lernt den Inhalt durch diese Labels zu verstehen.

Das „Problem" daran: Dieses Labelling kann keine Maschine durchführen. Unternehmen, die KI-Systeme trainieren, engagieren daher Menschen, häufig in Entwicklungsländern, die die Trainingsdaten labeln. Während das bei Beispielen für Fotografien von Tierstatuen oder Gedichten nicht besonders bemerkenswert scheint, ist es wichtig zu verstehen, dass diese erfolgreichen, großen Modelle mit allem frei im Internet verfügbaren Wissen trainiert wurden. Und dort findet sich leider nicht nur Positives, sondern auch viel Negatives, was das Labeln von Trainingsdaten teilweise zu einer anspruchsvollen und psychisch belastenden Tätigkeit macht. Die Verwendung von generativer Künstlicher Intelligenz hat also durchaus auch eine ethische Komponente.

Ein weiterer Aspekt aus dem Training ist „Bias" (aus dem Englischen: Vorurteil oder Verzerrung). In Abhängigkeit der Wahl der Trainingsdaten kann es dazu führen, dass das Modell inhärent Vorurteile besitzt. Denn wie auch wir Menschen kennt ein solches Modell die Welt nur so, wie sie

sie bisher gesehen hat. Wurde das Modell z. B. nur mit Informationen über Tiere trainiert, dann gibt es unter Umständen keine Menschen in der Welt des Modells. Und das ist ein sehr offensichtliches Beispiel. Schwieriger wird es, wenn solcher Bias versteckt in den Trainingsdaten ist und im Kleinen gar nicht auffällt, sich dann aber am Ende beim Einsatz bemerkbar macht. So können z. B. Gruppen von Menschen unabsichtlich diskriminiert werden oder Berechnungen zu falschen Schlüssen führen.

Auch in der Verwendung generativer KI ist es wichtig, sich der Grenzen bewusst zu sein. Im Endeffekt werden neue Inhalte (Texte, Bilder, etc.) durch den geschickten Einsatz von Wahrscheinlichkeiten generiert (siehe Abschn. 1.2). Was das Modell generiert, kann zwar korrekt aussehen, aber inhaltlich absolut falsch sein. Es wird daher empfohlen, KI-Systeme nur für solche Zwecke einzusetzen, in denen man selbst die Ergebnisse auf Plausibilität prüfen kann. Das heißt auch: Modelle wie ChatGPT sind keine Suchmaschinen und sie können durchaus richtig aussehende, aber inhaltlich falsche Antworten generieren.

Diese Art von Fehlern wird auch als „**Halluzinationen**“ bezeichnet. Dabei erfindet das LLM im Rahmen der Antwort Informationen und bindet sie ohne gesonderte Kennzeichnung ein. Dabei können Zahlen, Daten, Fakten, Namen oder komplette Zusammenhänge frei erfunden sein. Dies liegt daran, dass die KI-Systeme dergestalt entwickelt wurden, dass sie immer antworten; auch, wenn sie die richtige Antwort in einem gegebenen Kontext nicht kennen. Hierbei werden die hinzugedichteten Informationen aber nicht gesondert ausgezeichnet oder markiert. Dies macht es notwendig, dass jegliche Antworten von KI-Systemen vor einer weiteren Nutzung auf eine inhaltliche Korrektheit geprüft werden müssen. Halluzinationen sind dabei ein generelles Problem von KI-Systemen und treten deshalb bei allen Modellen und jeglichen Herstellern auf.

Ein weiteres Problem, dessen sich die Anwender bewusst sein müssen, ist der sogenannte **Bias** (aus dem Englischen: Vorurteil oder Verzerrung). Damit sind inhaltliche Verschiebungen von Antworten in eine (möglicherweise) falsche Richtung und damit eine verzerrte Darstellung der Antwort gemeint. So können z. B. Vorurteile oder Falschaussagen in den Antworten von KI-Systemen vorkommen, ohne dass dies wiederum markiert wird. Diess Fehlerklasse beruht auf Verzerrungen und unaus-

gewogenen Informationen in den Trainingsdaten. Sind zum Beispiel in einem gegebenen Set an Trainingsdaten Informationen über erfolgreiche weibliche Bewerber gegenüber Männern in einem Bewerbungsprozess unterrepräsentiert, so schlägt sich dies in verzerrten Antworten bei späteren Anfragen an das System nieder und männliche Bewerber werden möglicherweise bevorzugt behandelt. Dies zu vermeiden ist äußerst kompliziert, da schon vor dem Training des KI-Systems in den Ausgangsdaten des Trainings sichergestellt werden muss, dass jegliche Daten, die für spätere Antworten herangezogen werden, gleichmäßig repräsentiert sind.

Aktuell wird viel geforscht, um diese Fehler irgendwann komplett zu vermeiden (falls dies überhaupt möglich sein wird) oder zumindest ihre Auswirkungen abzumildern.

Obwohl sich mit dem richtigen Einsatz von aktueller generativer KI viele Vorteile für Unternehmen ergeben, sind die Systeme noch beschränkt und befinden sich aktuell noch nicht auf dem Fähigkeitsniveau eines Menschen. Denn zum einen ist das menschliche Gehirn um ein Vielfaches leistungsfähiger und zum anderen arbeiten wir mit bewusstem und unbewusstem Wissen und sind in der Lage, kreativ auch komplett Neues zu schaffen – etwas, das generative KI (noch) nicht kann.

1.1.3 Anwendungsfälle für KI in Unternehmen

Der Hype um KI kommt nicht von ungefähr, denn die Möglichkeiten und Vorteile für Unternehmen sind zahlreich und beeindruckend. Die „klassische", nicht-generative KI kommt wie bereits erwähnt schon länger zum Einsatz. Ein gutes Beispiel ist Predictive Maintenance (die vorausschauende Wartung): Durch die Auswertung von Produktions- und Nutzungsdaten können Systeme präzise vorhersagen, wann Wartungen an Maschinen notwendig werden. Durch das Erkennen von Anomalien (beispielsweise in Rotationen, Temperaturveränderungen oder Schwingungen) können Fehler und übermäßiger Verschleiß erkannt und frühzeitig behandelt werden. Dadurch wird die Wartungsplanung vereinfacht und Kunden haben weniger Ausfälle im produktiven Betrieb.

In ähnlichem Sinne kann auch während der Produktion bereits intelligente Fehlererkennung und -behebung umgesetzt werden. Eine Bilder-

kennung kann z. B. noch während der Fertigung Einschlüsse oder Risse erkennen und das entsprechende Teil beispielsweise aus der Produktionsstraße entfernen.

Bei Anwendungsfällen visueller Klassifizierung zeigen sich erneut die Grenzen aktueller KI-Systeme. Beispielsweise wird bei der Abfallsortierung oder der Nachkontrolle von Glaskörpern weiterhin vor allem menschliches Personal eingesetzt, da die Verarbeitung durch KI zu fehleranfällig oder teuer ist.

Im generativen Bereich gibt es ebenso viele Möglichkeiten, das Potenzial von KI zu nutzen. In der Softwareentwicklung kommen Modelle zum Einsatz, die Programmier-Code verstehen und im Rahmen des Entwicklungsprozesses Programmierfehler erkennen und anzeigen. Über die Zeit lernen die Systeme weitere Fehler und Herangehensweisen des Projekts dazu und werden immer besser in der Fehlererkennung. Weiter noch können diese Modelle häufig auch direkt den notwendigen Code generieren, um das Problem zu lösen. Das manuelle Überprüfen von Beheben von Programmierfehlern, was einen durchaus nicht zu unterschätzenden Zeitfaktor darstellt, kann damit stark reduziert werden.

Doch auch schon vor der Qualitätskontrolle, während des eigentlichen Programmierens, können Sprachmodelle wie Claude oder ChatGPT unterstützen. Mit ihren Programmierfähigkeiten können sie Code für verschiedene Aufgabenstellungen generieren, indem der Softwareentwickler lediglich beschreibt, was erreicht werden soll. Aus dem Kontext des bereits vorhandenen Codes heraus können auch bereits intelligente Vorschläge gemacht werden. Der große Durchbruch in der Softwareentwicklung steht allerdings noch aus, denn ein wichtiger Aspekt ist das Gesamtverständnis einer Anwendung, und dafür sind die Context Windows der Modelle noch zu klein (für Context Windows siehe Abschn. 1.3).

Ein weiterer typischer Anwendungsfall findet sich im Marketing. Da die Modelle Texte und Bilder in häufig durchaus guter Qualität generieren können, können sie gut im Marketing eingesetzt werden. Auch gibt es mittlerweile Werkzeuge, die mittels KI ganze Webseiten generieren. Da das Marketing von Kreativität lebt, werden die Modelle hier vor allem unterstützend eingesetzt und können einen Teil der Arbeit abnehmen.

Neben dem reinen Inhalt können sie auch Strategien entwickeln, Marktanalysen durchführen oder Vergleiche von Themen erstellen.

Da die Modelle gut darin sind, Texte zu verstehen und zu schreiben, ist es nicht verwunderlich, dass sie in der Dokumentenverarbeitung verschiedene Einsatzmöglichkeiten finden. Sie eignen sich gut, um Beschreibungen für Produkte zu erstellen, Dokumentationen zusammenzufassen oder auch, um mehrsprachige Inhalte zu produzieren. Wenn Unternehmen ihre Handbücher auch in einfacher Sprache anbieten möchten, können sie diese nun mit einem entsprechenden Sprachmodell „übersetzen" lassen.

Diese Möglichkeiten sind vor allem nach innen gerichtete Szenarien, in denen also KI innerhalb des Unternehmens eingesetzt und die Verwendung nach außen, zu Kunden und Lieferanten, unter Umständen gar nicht transparent wird. Auf der anderen Seite gibt es auch nach außen gerichtete Anwendungsfälle, bei denen z. B. Kunden mit KI-Systemen interagieren.

Das prominenteste Beispiel ist sicher die Chatbot-Technologie. Ob über einen in die Webseite integrierten Chatbot oder über Messenger-Dienste wie WhatsApp, Telegram oder WeChat – mit der richtigen Plattform bieten Unternehmen einen intelligenten, interaktiven und jederzeit verfügbaren Kunden-Support an.

Ein zweites, bekanntes Beispiel sind Empfehlungsalgorithmen, die beispielsweise auf E-Commerce-Plattformen wie Alibaba oder Amazon oder auch beim Streaming von Musik und Videos auf Spotify oder Netflix zum Einsatz kommen. Durch die Auswertung vieler Datenpunkte – von der Merkliste, über die Historie, bis hin zur Bewegung von Mauszeiger und Augen – können Vorschläge entwickelt und Neues empfohlen werden.

1.1.4 KI effektiv im Unternehmen einführen

Die kurze Übersicht im letzten Abschnitt zeigt bereits, dass das Potenzial für den Einsatz von generativer und nicht-generativer KI in Unternehmen umfangreich ist. Es ist allerdings wichtig, KI nicht zum Selbstzweck zu nutzen, sondern sinnvolle Anwendungsfälle zu identifizieren.

Für eine zielführende und erfolgreiche Einführung hat sich der dreischrittige „Crawl, Walk, Run"-Ansatz als besonders effektiv herausgestellt. In der ersten Phase (Crawl) werden die Grundlagen geschaffen. Es werden erste Projekterfahrungen gesammelt, es wird Wissen akkumuliert und viel experimentiert. Dafür ist es besonders nützlich, einzelne Anwendungsfälle („Leuchttürme") zu identifizieren, an denen der Einsatz von KI erprobt werden kann. In der zweiten Phase (Walk) werden die gesammelten Erfahrungen dann im Unternehmen in die Breite getragen. Parallel wird das Experimentieren fortgesetzt, indem z. B. neue Anwendungsgebiete identifiziert und erprobt werden. Die Leuchttürme der ersten Phase können unter Umständen bereits in einen Regelbetrieb übergehen. In der schließlich dritten Phase (Run) wird das sich dann ergebende Einführungsprojekt professionalisiert und skaliert. Das Unternehmen befindet sich dann inmitten einer KI-Transformation.

Das gesamte Vorgehen sollte durch professionelles Change-Management begleitet werden. Es gilt, relevante Stakeholder frühzeitig zu identifizieren und adäquat in den Prozess des Wandels mit einzubeziehen. Bei Themen wie KI, bei denen es am Ende auch um Effizienzgewinne und Mitarbeiterentlastung geht, müssen Sorgen und Ängste ernst genommen und die Veränderungen für jeden einzelnen erklärt und greifbar gemacht werden.

1.1.5 Der Einfluss auf Rollen und Fähigkeiten in Unternehmen

Bei einer Transformation wie der Einführung von KI ergeben sich nicht nur neue Möglichkeiten für die Wertschöpfung, Unternehmen müssen auch die Kompetenzen aufbauen, um den Einsatz im Tagesgeschäft entsprechend steuern zu können.

Auf strategischer Ebene kann es sinnvoll sein, eine Chief Artificial Intelligence Officer (CAIO) zu platzieren. Diese in der Geschäftsleitung angesiedelte Rolle bündelt als Querschnittsfunktion oder Stabsstelle die Verantwortung der relevanten Themen, die sich im Unternehmen mit KI ergeben. Unter ihr sind dann die jeweiligen Verantwortlichkeiten bzw. Stellen aufgehängt.

Dort gibt es dann z. B. das Thema Legal und Compliance. Der Gesetzgeber definiert nämlich auch für Unternehmen, die KI entwickeln, betreiben oder verwenden, verschiedene Auflagen. Im Rahmen der europäischen KI-Verordnung müssen z. B. Mitarbeiter entsprechend geschult werden. Auch müssen entwickelte KI-Systeme einer Risikoklassifizierung unterzogen werden und dann ggf. strenge Dokumentationspflichten erfüllen. Unternehmen müssen wie auch in anderen Bereichen auf die regulatorischen Entwicklungen reagieren und Konformität sicherstellen.

Auch das Thema Datenschutz kann hier aufgehängt sein. Wie auch in allen anderen Bereichen des Unternehmens unterliegt KI den Datenschutzvorschriften. Allerdings mit einem Haken: lernende Modelle nehmen unter Umständen die zugeführten Informationen zum Trainieren. Personenbezogene Daten können daher regelmäßig nicht mehr aus KI-Systemen entfernt werden. Das ist ein wichtiger Aspekt, der großes Risiko birgt und daher entsprechend behandelt werden muss.[1]

Ein weiterer Punkt ist das Thema „Operations", also der Betrieb. Denn es handelt sich bei KI-Systemen am Ende nur um weitere IT-Infrastruktur, die selbst betrieben oder eingekauft wird. Diese sollte daher den gleichen Maßstäben wie andere Lösungen unterliegen und beispielsweise zentral verwaltet werden. Auch die Integration in andere Plattformen und Tools über entsprechende Schnittstellen (beispielsweise mit KI-Agenten) muss zentral gesteuert und überwacht werden.

Neben diesen in der Regel bereits in Unternehmen vorhandenen Rollen und Aufgaben ergibt sich zusätzlich zur CAIO-Position noch mindestens eine zweite neue Stelle. Mit generativen KI-Systemen wird mittels natürlicher Sprache gearbeitet, die inhärent komplex und häufig nicht eindeutig ist. Daher müssen Anfragen (Prompts) an solche Systeme in guter Qualität geschrieben werden. Dafür gibt es Grundregeln, Techniken und erprobte Strukturen, mit denen die besten Ergebnisse produziert werden können. Diese Aufgabe übernimmt das „Prompt Engineering", das Prompts entwickelt und testet, aber auch Mitarbeiter in der Gestaltung zielführender Anfragen schult. Im weiteren Verlauf wird dieses Thema ausführlich beschrieben. Zunächst werden aber noch weitere technische Details zum Aufbau und der Funktionsweise von LLMs gegeben.

[1] Bitte beachten Sie, dass dies keine Rechtsberatung im juristischen Sinne ist. Bei Fragen zu rechtsrelevanten Themen wenden Sie sich bitte an einen juristischen Spezialisten.

1.2 Large Language Models und Generative Pre-trained Transformers

Die scheinbare Magie von Sprachmodellen wie ChatGPT, Gemini oder Claude liegt darin, die Eingaben von Nutzen zu verstehen und basierend darauf entsprechende Antworten zu generieren. Dahinter verbirgt sich am Ende „einfache" Mathematik; basierend auf der Anfrage wird mittels Statistik die wahrscheinlichste Antwort für diesen Kontext ermittelt. Doch so einfach kann es am Ende nicht sein. Wie kommt es dazu, dass zwei wortgleiche Prompts unterschiedliche Ergebnisse produzieren? Wie kann das System eine plausible Aussage zu noch so komplex gestellten Fragen generieren? Und wie werden inhaltlich korrekte Aussagen zum selben Sachverhalt bei völlig unterschiedlich formulierten Anfragen getroffen? Die KI scheint über die reine Wortwahl hinaus die Intention der Nutzer zu verstehen.

Dahinter verbirgt sich ein Modell, das speziell für das Verstehen und Produzieren von menschlicher Sprache entwickelt und trainiert wurde. So ein **Large Language Model** (LLM) versteht Kontext, Muster und Beziehungen von Texten und kann dadurch sowohl Eingaben hinsichtlich ihrer Bedeutung auswerten als auch entsprechend sinnvolle Antworten generieren.

Würde man einem LLM beispielsweise eine Phrase präsentieren, könnte es diese sinnvoll vervollständigen. Dazu könnte jeweils jenes Wort produziert werden, das mit der größten Wahrscheinlichkeit auf die bereits vorhandene Phrase folgt:

Beispiel

Eingabe und Vervollständigungen

- Eingabe:
  - Der große Vorteil von KI ist
- Vervollständigungen:
  - Der große Vorteil von KI ist die
  - Der große Vorteil von KI ist die schnelle
  - Der große Vorteil von KI ist die schnelle Verarbeitung
  - Der große Vorteil von KI ist die schnelle Verarbeitung großer

– Der große Vorteil von KI ist die schnelle Verarbeitung großer Datenmengen

Geht man nun davon aus, dass das LLM immer das nächste Wort mit der größten Wahrscheinlichkeit auswählt, würde bei derselben Eingabe immer dieselbe Vervollständigung generiert werden – einmal abgesehen vom (möglicherweise kontinuierlichen) Training des LLMs und der damit verbundenen Veränderung der Wahrscheinlichkeitswerte. Die Texte würden dadurch sehr künstlich wirken und sich in der Masse deutlich von menschlich verfassten Texten unterscheiden. Der Grund dafür liegt darin begründet, dass Menschen beim Schreiben eine Variabilität in ihren Formulierungen verwenden, also immer eine gewisse Menge an Zufall dabei ist. Um mittels eines LLMs dieses Gefühl von Kreativität und Unvorhersagbarkeit zu erreichen, gibt es einen Parameter namens „Temperature" (meist ein Wert zwischen 0 und 1), der beeinflusst, welches der möglichen folgenden Wörter jeweils ausgewählt wird. Je nach Wert werden wahrscheinlichere Wörter dann häufiger oder seltener verwendet. Eine niedrige Temperatur sorgt dabei für einfachere und konservative Antworten, während ein hoher Wert (unvorhersagbar und manchmal ungewollt und übertrieben) kreative Antworten liefert.

Bleibt die Frage, woher das LLM grundsätzlich weiß, wie die Wahrscheinlichkeiten von Formulierungen aussehen. Warum wurde im obigen Beispiel als erstes das Wort „ist" hinzugefügt und nicht der Begriff „Katze"? Im ersten Schritt könnte man sich ansehen, wie die Wahrscheinlichkeit von Buchstabenfolgen aussieht. Für jeden Buchstaben in einer gegebenen Sprache kann eine Auftretenshäufigkeit und eine Häufigkeit für das Folgen spezifischer anderer Buchstaben bestimmt werden. Beispielsweise kommen die Buchstaben „e", „a" und „o" recht häufig vor, während „ß" vergleichsweise selten verwendet wird. Weiterhin ist die Wahrscheinlichkeit, dass auf ein „ß" ein „e" folgt größer als die Wahrscheinlichkeit, dass ein weiteres „ß" oder ein „q" folgt. Für einen gegebenen Buchstaben wäre es nun so möglich, Fortsetzungen zu generieren, die zumindest mathematisch sinnvoll sind. Kombiniert man dies beispielsweise mit durchschnittlichen Längen von Wörtern (fügt also Leerzeichen ein), könnten Buchstabensalate entstehen, die zumindest visuell annähernd ähnlich einem Text aussehen, aber faktisch keiner sind.

Die „Magie“ von LLMs kann folglich nicht auf Ebene der Buchstaben allein begründet sein. Im Grunde wird dieses Konzept aber lediglich für Wörter, Teilwörter und Phrasen wiederverwendet – es werden Wahrscheinlichkeiten zwischen diesen hergestellt.

Das Konzept dahinter ist ein neuronales Netz, das man sich vorstellen kann wie eine Menge an verbundenen Knoten mit Wahrscheinlichkeiten auf den Verbindungen. Beispielsweise wird vom Wort „ist“ aus obigem Beispiel eine Verbindung zum Wort „die“ existieren, wobei dieser Verbindung zwischen den beiden Wörtern ein hoher Wahrscheinlichkeitswert („Weight“, Gewicht) zugeordnet wurde. Vom Wort „die“ gehen dann wiederum mehrere Verbindungen zu Wörtern aus. Die Menge an Knoten und Schichten, in denen so ein neuronales Netz üblicherweise angeordnet wird, ist nur ein Parameter, der die Effektivität und Größe des LLM beeinflusst.

Indem dem System vorhandene Texte, Anfragen von Nutzern u. ä. zugeführt werden, werden diese Wahrscheinlichkeiten kontinuierlich trainiert. Dabei muss aber eine signifikante Herausforderung gemeistert werden, denn es gibt deutlich mehr Möglichkeiten für Texte, die generiert werden können, als Texte, die bereits existieren. Das System muss also interpolieren und „Neues“ schaffen und das möglichst effizient. Eine wichtige Technik dafür sind **Transformer**, die einem LLM die Idee einer „Attention“ (Aufmerksamkeit) hinzufügen; die Annahme ist, dass nicht alle Teile eines Satzes gleich wichtig sind, weshalb sich das System mehr auf die wichtigen Teile konzentriert. Das führt z. B. dazu, dass im neuronalen Netz die Anzahl der existierenden Verbindungen reduziert werden kann. Weiterhin können dadurch insgesamt kohärentere Texte generiert werden. Die Transformer sind dafür verantwortlich, diese Attention zu bestimmen und dem Modell entsprechend beizumischen.

Die dargestellten Bausteine sind die identifizierenden Merkmale von Systemen wie ChatGPT. Das Kürzel „GPT“ im Namen steht für **Generative Pre-Trained Transformer**. „Generative“, da das System dazu trainiert und eingesetzt wird, Texte zu generieren. Das Training ist signifikanter Bestandteil des Konzepts, die erläuterten Wahrscheinlichkeiten werden auf vielfältigen Wegen und über die Zeit, aber mindestens teilweise vor Benutzung erlernt („Pre-Trained“). Schließlich verwendet es das Konzept von Transformern („Transformer“), die Attention festlegen und das System dadurch effizienter machen.

1.3 Tokens und Context Windows

Bisher wurde sich LLMs vor allem über das Konzept von Wörtern genähert – das ist allerdings nicht ganz akkurat. Sprachmodelle verwenden „**Tokens**“ (aus dem Englischen: Zeichen oder Symbol) als Einheit; sie lesen Text in Form von Tokens ein und generieren basierend auf den Wahrscheinlichkeiten im neuronalen Netz das jeweils passendste nächste Token. Diese Tokens wiederum können ganze Wörter oder auch Teile von Wörtern (beispielsweise Silben) sein – weshalb es grundsätzlich auch möglich wäre, dass nichtexistierende Wörter aus Teilen von vorhandenen Wörtern generiert werden könnten.

Das Konzept von Tokens ist für das Arbeiten mit Sprachmodellen durchaus wichtig. Denn pro Chat kann ein Modell nur eine begrenzte Menge an Tokens gleichzeitig verarbeiten, sowohl bei der Eingabe als auch bei der Ausgabe. Es ist quasi das Gedächtnis des KI-Modells in der Chat-Interaktion. Wie durch ein Fenster „blickt“ das Sprachmodell mit diesem sogenannten **Context Window** auf die vom Nutzer bereitgestellten und die selbst generierten Daten. Kommen neue Daten dazu, „rutscht“ das Context Window nach unten – und alles, was dann außerhalb dieses Fensters ist, ist für das LLM nicht mehr bekannt. Erzeugt ein Prompt beispielsweise sehr viel Text, kann es passieren, dass vorherige Anfragen oder Ergebnisse „vergessen“ werden. Auf diese kann sich dann nicht mehr bezogen werden und sie fließen auch nicht mehr implizit mit in die Verarbeitung des LLMs mit ein.

Die Größe des Context Windows gilt zurzeit als wichtigster limitierender Faktor bei der Skalierung von Sprachmodellen. Viele Context Windows gängiger öffentlich nutzbarer Modelle haben aktuell eine Größe von etwa 100.000 Token (umgerechnet ca. 100 Seiten Text), vereinzelt wird bereits an Modellen mit der zehnfachen Größe gearbeitet. Sollen solche Modelle aber in den persönlichen Alltag einziehen und Personen sinnvoll und umfänglich unterstützen, bedürfte es deutlich mehr Kapazität. Berechnungen haben ergeben, dass ein Context Window von etwa 100.000.000 Tokens notwendig wäre, um ein halbes Jahr im Leben einer einzelnen Person vollumfänglich zu erfassen. Diese

Größe an Kontext bringt nicht nur signifikante Herausforderungen an Speicher (Festplatte, aber auch vor allem Arbeitsspeicher) mit sich, sondern stellt auch algorithmische Hürden auf, denn diese Daten müssen wiederholt schnell ausgewertet werden, um in die Interaktion mit dem LLM mit einzufließen.

1.4 Unterschiede von Large Language Modellen

Sprachmodelle gibt es mittlerweile viele in unterschiedlichen Varianten und die Entwicklung neuer und fähigerer Modelle schreitet stets voran. Das beste Einsatzszenario für das jeweilige Modell und die dafür notwendigen Fähigkeiten ergeben sich aus verschiedenen Parametern, mit denen das Modell entwickelt und trainiert wurde. Die bekanntesten Modelle, die hinter Systemen wie OpenAIs ChatGPT, Anthropics Claude oder Gemini von Google stecken, unterscheiden sich häufig nur im Detail.

1.4.1 Lizenz und Verfügbarkeit

Der Code und auch die Trainingsdaten der Sprachmodelle unterliegen jeweils Lizenzen und sind unter Umständen nicht frei verfügbar. Verschiedene Hersteller von LLMs veröffentlichen ihre Modelle und stellen sie vollumfänglich der Allgemeinheit zur Verfügung (z. B. Llama von Meta), während andere nur ausgewählte oder gar keine Modelle veröffentlichen. Von OpenAI gibt es z. B. verschiedene kleinere, speziellere Modelle, die als Open Source verfügbar sind, während die großen Sprachmodelle hinter ChatGPT nicht veröffentlicht werden.

Die Vorteile von Open-Source-Modellen sind zweierlei. Zum einen können diese in der Öffentlichkeit weiterentwickelt werden und Unternehmen wie auch Privatpersonen können mit ihren Fähigkeiten die Entwicklung unterstützen. Zum anderen ergibt sich für Unternehmen häufig der Vorteil, diese Modelle einfacher selbst betreiben zu können.

1.4.2 Grundlegende Fähigkeiten und Modalität

Ein weiteres Kriterium sind die grundlegenden Fähigkeiten des Modells. Dabei wird z. B. betrachtet, wie gut es Texte verstehen und analysieren kann. Auf der anderen Seite ist auch die Fähigkeit, kreativ zu schreiben oder allgemein Texte zu generieren, ein Unterscheidungsmerkmal. Daneben gibt es auch Modelle, die stärker im logischen Schlussfolgern und dem Lösen von Problemen sind als andere.

In der Interaktion mit dem Nutzer unterscheiden sie sich in der Konversations- und Dialogqualität, z. B. in der sprachlichen Tonalität, der Freundlichkeit und der grundsätzlichen Länge der Antworten.

Schließlich ist auch die Sprachunterstützung ein wichtiger Aspekt. Viele Modelle sind grundsätzlich erst einmal auf Englisch verfügbar, die großen und bekannten unterstützen viele Sprachen – teilweise bis hin zu exotischen oder erfundenen Sprachen wie Klingonisch (bekannt aus der Serie Star Treck).

Doch Sprachmodelle verstehen und generieren teilweise nicht nur Text. Moderne, große LLMs sind **multimodal**. Sie können Textanfragen entgegennehmen und verarbeiten. Dabei können Sie auch Dokumente verschiedener Art analysieren und in die Entwicklung einer Antwort mit einfließen lassen. Zudem können verschiedene Modelle Bilder auswerten, Texte und Ausschnitte erkennen und entsprechend mit verwerten. Schließlich muss die Interaktion nicht mehr textuell stattfinden, denn einzelne Sprachmodelle haben die Möglichkeit, gesprochene Sprache zu verstehen und ebenfalls auditiv zu beantworten – in unterschiedlichen Sprachen, Tonalitäten und Dialekten. Wie weit alle diese Fähigkeiten ausgeprägt sind, unterscheidet Modelle ebenfalls voneinander.

1.4.3 Wissensbasis und Aktualität

Die Stärke moderner Sprachmodelle liegt in ihrem umfangreichen Wissen begründet, auf das sie im Rahmen der Interaktion zurückgreifen – und dieses Wissen unterscheidet viele Modelle voneinander. Die Menge und Qualität der Trainingsdaten hat signifikanten Einfluss auf die Perfor-

mance des LLM. Je mehr mit Daten aus einem spezifischen kontextuellen Raum (z. B. geografisch oder sozioökonomisch) trainiert wurde, desto mehr werden die Antworten mit diesem Kontext generiert werden. Je breiter, umfangreicher und „objektiver" das Trainingsmaterial ist, desto allgemeiner kann das Modell am Ende eingesetzt werden.

Dabei ist auch von Relevanz, wann der sogenannte „**Knowledge Cutoff**" stattgefunden hat. Das ist der Zeitpunkt, bis zu dem das Modell mit verfügbaren Daten trainiert wurde. Denn darunter fallen z. B. auch Nachrichtenseiten. Wurde ein Modell mit Informationen bis zu einem Tag X trainiert, sind alle danach entstandenen Informationen (z. B. Nachrichten oder Ereignisse) dem Modell unbekannt. In dem Rahmen muss bei Modellen auch dahingehend unterschieden werden, was gegen Bias und Halluzinationen unternommen wurde.

1.4.4 Interaktionsfähigkeiten

In der Benutzung unterscheiden sich LLMs z. B. darin, wie viel sie von eigenem und bereitgestelltem Kontext verstehen. Auch Möglichkeiten zur Personalisierung fallen hier hinein – kann ich als Nutzer Einstellungen (Sprache, Tonalität...) vorgeben oder auch Informationen über mich selbst für das Modell bereitstellen?

Weiterhin kann man Sprachmodelle anhand ihrer Fähigkeit differenzieren, Instruktionen zu verstehen und darauf aufbauend konsistente und kohärente Antworten zu liefern. Die Anpassungsfähigkeit an die Bedürfnisse des Nutzers ist also ein wichtiges Kriterium.

1.4.5 Praxisrelevante Grenzen

Es ist beim Vergleich von Sprachmodellen auch wichtig, die Grenzen der einzelnen Systeme zu kennen. Ein wichtiger Faktor ist das bereits erwähnte Context Window, das großen Einfluss auf die Interaktionsmöglichkeiten hat. Denn ein Modell, das sehr gut im Auswerten von Dokumenten ist, ist andererseits stark limitiert, wenn das Context Window zu klein ist, um umfangreiche Dokumente auswerten zu können.

Daneben ist beim Vergleich wichtig, mit welcher Geschwindigkeit Anfragen bearbeitet und beantwortet werden. Das ist zum einen abhängig von der Infrastruktur, auf der das Modell betrieben wird, zum anderen aber auch vom Modell selbst. Aktuelle Sprachmodelle, die auf dem eigenen Arbeitscomputer betrieben werden, mögen umfangreiche Möglichkeiten bieten, sind aber zumeist so langsam in der Ausführung, dass sie wenig praxistauglich sind.

Dann ist noch zu beachten, welche thematischen Grenzen das Modell hat und wie es mit sensiblen Themen umgeht. In vielen Modellen gibt es Einschränkungen, die verhindern sollen, dass bestimmte Informationen generiert werden. Beispielsweise werden viele Modelle keine Antwort zum Thema der Herstellung von Gift generieren oder auch keine medizinische Betreuung beim Thema Depressionen empfehlen.

1.4.6 Einsatzszenarien

Schließlich werden LLMs anhand ihrer designierten Anwendungsfälle differenziert. Darunter fällt z. B. die Möglichkeit, Programmiercode zu verstehen und zu generieren. Modelle, die im Softwaretesting zum Einsatz kommen, können unter Umständen keinen (guten) Code generieren, sind aber gut darin, in gegebenem Code Programmierfehler aufzuzeigen.

Sprachmodelle, die nur mit internem Wissen eines Unternehmens trainiert wurden, eignen sich besser dazu, Mitarbeiter bei Fragestellungen zu unterstützen, als im Kunden-Support zum Behandeln von Kundenanfragen eingesetzt zu werden.

Ein anderes Beispiel ist die Integrationsfähigkeit in andere KI-Systeme und Softwaretools. Diese Fähigkeiten bilden die Grundlage für das Erstellen autonomer KI-Agenten (siehe Kap. 6).

Ein weiterer Aspekt ist die Fähigkeit der Anpassung des Modells an unterschiedliche Anwendungsszenarien. Während manche Modelle strikt auf einen Fall zugeschnitten sind und nur in diesem sehr gut funktionieren, können andere flexibler angepasst werden, um dann für spezifische Anforderungen gute Antworten zu liefern.

1.4.7 System-Prompts

Das Antwortverhalten von (generativen) KI-Systemen und vor allem von LLMs wird durch einen sogenannten **System-Prompt** gesteuert (Ramlochan, 2024). Der System-Prompt ist eine sehr lange und ausführliche Beschreibung des gewünschten Verhaltens eines gegebenen KI-Systems. Der Prompt wird im Rahmen der Entwicklung des Systems durch die jeweilige Firma geschrieben und automatisch bei jeder späteren Anfrage durch die Nutzer (die sogenannten User Prompts) vorangestellt und mitausgeführt. Dabei wird der System-Prompt immer ausgeblendet. Oft ist der System-Prompt eines gegebenen Systems gar nicht öffentlich zugängig und wird teilweise sogar als Betriebsgeheimnis behandelt. Andere Firmen veröffentlichen und kommentieren ihren System-Prompt auch, um eine bessere Transparenz für die Nutzer zu schaffen.

Mit Hilfe des System-Prompts werden dem KI-System diverse Richtlinien für das Beantworten von Nutzeranfragen mitgegeben. Weiterhin können darüber Grenzen definiert werden, die ein System nicht überschreiten darf. So können z. B. ethische Regeln genauso wie rechtliche Vorgaben definiert werden. Hierdurch wird sehr direkt das Verhalten des jeweiligen Systems definiert und gesteuert, ohne dass der Nutzer später darauf Einfluss nehmen kann.

Weiterhin kann über den System-Prompt die Kreativität der Antworten sowie die grundsätzliche Rolle und Tonalität des Systems vorgegeben werden. Hierdurch erhält ein gegebenes System seinen in späteren Unterhaltungen wahrnehmbaren „Charakter“.

Abgeschlossen werden die System-Prompts oft mit diversen Kontrollen und Plausibilitätsprüfungen. Zusätzlich werden an dieser Stelle meist noch Kontrollen auf korrekte Rechtschreibung und Grammatik eingebunden.

Insgesamt steuern die System-Prompts das Verhalten, die genutzte Sprache und das Ausgabeverhalten von KI-Systemen.

Im weiteren Verlauf dieses Fachbuchs werden nur noch User Prompts berücksichtigt. Dies sind die Anfragen, die Sie – als Nutzer – an das KI-System stellen.

1.5 Auf einen Blick

- Künstliche Intelligenz ist als Technologie bereits seit vielen Jahren bekannt und findet in den unterschiedlichsten Systemen Anwendung.
- KI-basierte Systeme und vor allem generative Systeme, die Texte, Bilder und weitere Medien produzieren, haben in den letzten Jahren große technologische Sprünge gemacht und liefern inzwischen ein großes Optimierungspotenzial für Unternehmen.
- Bei allen Vorteilen bringen die Systeme aber auch Risiken und Grenzen mit sich, denen sich Unternehmen bewusst sein müssen. Vor allem sei hier der Datenschutz genannt.
- Bei der Einführung von KI im Unternehmen bietet sich der „Crawl, Walk, Run"-Ansatz an. Dabei wird die Technologie in kleinen Schritten und mit Bedacht eingeführt.
- Um einen optimalen Einsatz im Unternehmen sicherzustellen, müssen die Mitarbeiter im richtigen Einsatz der KI-Technologie geschult werden. Zusätzlich müssen die Mitarbeiter auch in den Grenzen und Gefahren geschult werden, um mögliche Datenschutzvergehen zu vermeiden.
- Die EU-KI-Verordnung regelt dazu die Rechte und Pflichten der anbietenden und anwendenden Unternehmen. Darin werden z. B. auch verpflichtende Schulungen für das Personal vorgeschrieben.
- Large Language Models (LLMs) wie z. B. ChatGPT, Claude und Gemini basieren auf der Transformer-Technologie und können sehr ausführliche und tiefgreifende Texte auf Basis von entsprechenden Anfragen (den sogenannten Prompts) erzeugen. Dabei unterscheiden sie sich an unterschiedlichen Stellen in den angebotenen Funktionalitäten. Im Kern bieten sie aber alle sehr ähnliche Funktionalitäten.
- Die grundsätzlichen Funktionalitäten, Eigenschaften und Grenzen eines LLMs werden im sogenannten System-Prompt definiert. Der System-Prompt wird dabei im Normalfall nicht angezeigt und teilweise sogar als Betriebsgeheimnis behandelt. Dieser Prompt wird durch die Systeme immer vor dem Prompt des Anwenders eingefügt und immer mit ausgeführt.

2

Grundlagen des Prompt Engineerings

Zusammenfassung In diesem Kapitel wird in die Grundlagen des Prompt Engineerings eingeführt. Dazu wird zunächst mit dem Valisory Prompt-Qualitätsmodell (PQM) eine Möglichkeit vorgestellt, mit deren Hilfe die Qualität von Prompts bewertet werden kann. Danach wird schrittweise in die grundlegenden Techniken des richtigen Promptens eingeführt und die Umsetzung anhand von praktischen Beispielen aufgezeigt.

Die Wahl des Modells im LLM hat bereits große Auswirkungen auf die Qualität des Verständnisses der Eingaben und der produzierten Ergebnisse. Doch ungeachtet der Fähigkeit des Modells ist das Formulieren von guten und präzisen Prompts, also die Qualität der Anfrage an ein KI-System, essenziell, um es effektiv einzusetzen. Dadurch wird Missverständnissen vorgebeugt und die Intention klarer dargestellt. Außerdem können die Ergebnisse auch strukturell deutlich besser an die eigenen Erwartungen angepasst werden. Im Folgenden werden dazu verschiedene Grundlagen rund um das effektive Formulieren von Prompts an ein Sprachmodell erläutert.

D. Koch et al., *Prompt Engineering im Unternehmen – eine Einführung*,
https://doi.org/10.1007/978-3-658-47699-1_2

2.1 Das Konzept von Prompt Engineering

Prompt Engineering ist ein spezielles Gebiet innerhalb des Themenfelds der KI, das sich mit der Optimierung von Eingabeaufforderungen für Sprachmodelle wie z. B. ChatGPT, Claude oder Gemini beschäftigt. Es besteht aus verschiedenen Techniken zur zielgerichteten, präzisen und effektiven Formulierung von Prompts.

Auf der einen Seite lässt sich die Notwendigkeit zur Beschäftigung von Prompt Engineering durch die Analogie zur interpersonellen Kommunikation verdeutlichen. Das menschliche Gehirn ist um ein Vielfaches leistungsfähiger als aktuelle Sprachmodelle. Dennoch ist das Trainieren von Kommunikation ein ganz natürlicher Bestandteil vieler Aus- und Weiterbildungen. Wir können als Menschen in Standardsituationen schnell das Richtige verstehen, auch wenn der Inhalt umständlich formuliert, undeutlich gesprochen oder nicht im richtigen Kontext geäußert wurde. Dennoch investieren wir in die Verbesserung unserer Kommunikation, um Aufwand zu reduzieren und um Missverständnissen vorzubeugen.

Auf der anderen Seite sind Sprachmodelle grundsätzlich schon gut darin, auch solche Anfragen richtig zu interpretieren und effektiv zu beantworten, die nicht eindeutig formuliert sind, denen Kontext fehlt oder bei denen das gewünschte Ergebnis nicht definiert ist. Entsprechen die Ergebnisse nicht den Erwartungen, können diese zwar mit dem Modell interaktiv angepasst bzw. verbessert werden, das Einsparen dieser Optimierungsschleifen würde aber die Arbeit deutlich effizienter gestalten.

Dieses Optimieren ist das grundsätzliche Ziel von Prompt Engineering. Durch die Strukturierung von Prompts oder auch das Verwenden bestimmter Formulierungen werden die Ergebnisse des LLMs gesteuert und damit wird die Qualität erhöht sowie die Notwendigkeit von Nacharbeiten reduziert. Vielmehr werden die Fähigkeiten des Sprachmodells ausgereizt, um die besten Ergebnisse zu erhalten.

Das Konzept von Prompt Engineering beruht dabei auf drei aufeinander aufbauenden Ebenen. Grundlage bilden Anforderungen an sprachliche Eindeutigkeit, korrekte Definition der Ergebniserwartungen genauso wie auch das Arbeiten mit referenzierten Texten oder Dokumenten. Darauf aufbauend können verschiedene Frameworks verwendet wer-

den, die diese Grundlagen in eine bestimmte, durch das jeweilige Framework vorgegebene Struktur bringen. Die dritte Ebene bilden verschiedene Techniken rund um das Arbeiten mit Beispielen oder das Verbessern mehrstufiger Ergebnisse, die sowohl mit den Grundlagen als auch mit Frameworks kombiniert werden können. Alle drei Ebenen werden mit konkreten Beispielen in den folgenden Kapiteln näher erläutert.

Neben dem reinen Schreiben von Prompts beschäftigt sich das Prompt Engineering auch mit der Qualitätssicherung und dem Testen von Anfragen. Während es für einfache und interaktive Anwendungsfälle in der Regel ausreicht, verschiedene Prompt-Techniken zu benutzen, um gute Ergebnisse zu produzieren, gibt es Szenarien wie KI-Agenten (vgl. Kap. 6), in denen Prompts nicht einfach geändert und Ergebnisse nicht einfach iterativ weiterentwickelt werden können. Dort werden für eine Aufgabenstellung verschiedene Prompts geschrieben und hinsichtlich ihrer Effektivität verglichen.

Eine Ebene darüber ist der Aspekt der Multi-LLMs und der Prompt Security angesiedelt. Wenn mehrere LLMs kombiniert werden, müssen jeweils gute Prompts für die einzelnen Systeme geschrieben und die Kombination der LLMs sorgfältig geplant werden. Beim Sicherheitsaspekt beschäftigt sich das Prompt Engineering mit der Frage, wie z. B. Chatbots so geprompted werden können, dass sie nicht zweckentfremdet werden. In diesem Bereich beschäftigt man sich auch damit, die künstlich gesetzten Grenzen von Sprachmodellen zu überwinden, um z. B. Informationen zu verbotenen oder eingeschränkten Themen zu extrahieren.

2.2 Die Bewertung der Qualität von Prompts

Die Verwendung von Prompt-Engineering-Techniken für die Optimierung der Anfragen an Sprachmodelle zur Ergebnisverbesserung bedingt die Notwendigkeit, die Qualität sowohl des Prompts als auch der Ergebnisse bestimmen zu können. Ohne einen Benchmark mögen die verschiedenen Techniken noch so gut klingen, bringen aber unter Umständen die gewünschten Vorteile nicht und schaffen dadurch nur zusätzlichen Aufwand.

Die Qualität zu messen, ist dabei kein leichtes Unterfangen. Ausgehend von der Situation, dass eine Ausgabe des Sprachmodells nicht den Erwartungen entspricht, ist die Frage zu stellen, ob von dem Modell mehr gefordert wurde, als es überhaupt leisten kann. Wenn die Grenzen des technisch Machbaren erreicht sind, kann auch keine noch so geschickte Prompting-Technik die Ergebnisse verbessern. Beispielsweise kann ein Modell zur Textgenese grundsätzlich nicht rechnen und wird falsche Ergebnisse liefern oder auch falsche Schlüsse aus Zahlenmaterial ziehen. Ungeachtet dessen, wie eine Anfrage formuliert oder mit welchem Prompt-Framework sie strukturiert wird, es wird nicht zu richtigen Ergebnissen führen.

Bittet man ein LLM zu rechnen, kommen (je nach Modell) durchaus richtige Antworten dabei heraus. Einfache Rechenaufgaben in einem kleinen Zahlenraum (beispielsweise bis 100) können regelmäßig durch Daten aus dem Trainingsmaterial heraus richtig beantwortet werden. Bei komplexeren Aufgaben bedient sich z. B. ChatGPT zumeist einem Python-Programm, das es zur Lösung der Aufgabe generiert und ausführt – in diesem Fall stimmen die Ergebnisse dann ebenfalls. Darüber hinaus (und vor allem auch, wenn man einfordert, kein Python zu benutzen), zeigt sich, dass das Sprachmodell eben kein Rechenmodell ist.

Die technischen Limitierungen des Modells einmal ausgeklammert, kann zur Bestimmung der Qualität einfach die Ausgabe mit den gesetzten Erwartungen abgeglichen werden. Sind diese gar nicht oder nur teilweise erfüllt, gilt es, den Prompt so weit zu optimieren, bis der notwendige Erwartungsgrad erreicht ist. Damit dies nicht willkürlich geschieht, sollten Verbesserungen strukturiert vorgenommen werden, das heißt, die Qualität des Prompts sollte anhand fest definierter Kriterien bestimmt werden. Allerdings ist festzuhalten, dass auch ein noch so gutes Konstrukt aus Qualitätsparametern zu einem gewissen Grad willkürlich ist, denn der genaue Weg von der Anfrage zur Ausgabe ist derart komplex, dass es keinen linearen Zusammenhang zwischen Qualitätskriterium und den Ergebnissen gibt. Dennoch hilft eine solche Herangehensweise LLMs effektiver zu verwenden.

Das Valisory-Prompt-Qualitätsmodell (PQM)

Das Valisory-Prompt-Qualitätsmodell (kurz: PQM) ist ein multidimensionales Qualitätsmodell, mit dem die Qualität von Anfragen an Modelle generativer KI bewertet werden kann. Es eignet sich nicht nur für Sprachmodelle, sondern auch für alle anderen Arten generativer KI, wie beispielsweise Musik, Video oder auch Bilder. Für die Bestimmung der Qualität eines Prompts wird dieser entlang der vier Kriterien Relevanz, Kreativität, Sprache und Zielgruppe bewertet, wobei jedes Kriterium wiederum drei verschiedene Aspekte betrachtet (s. Tab. 2.1).

Der Aspekt der **Relevanz** bewertet, in welchem Maße der Prompt in den Formulierungen grundsätzlich dem Ziel der Aufgabenstellung angemessen ist. Dazu wird die Fragestellung, die Spezifizität und schließlich die Verständlichkeit betrachtet. Hinter dem Aspekt der *Fragestellung* verbirgt sich, wie klar der Prompt formuliert ist und inwieweit er die gewünschten Informationen, Ergebnisse oder Aktionen direkt einfordert. Dabei wird sprachliche Richtigkeit genauso betrachtet wie Eindeutigkeit in den Formulierungen und auch die Klarheit der Anweisungen hinsichtlich der zu entwickelnden Ergebnisse. Die *Spezifität* des Prompts bezieht sich dabei auf die Frage, in welchem Umfang die Anforderungen an die Ergebnisse und die formulierten Ziele konkret genug sind, um relevante Antworten zu generieren. Schließlich wird mit dem Aspekt der *Verständlichkeit* betrachtet, ob der Prompt so geschrieben wurde, dass das Sprachmodell die Anfrage korrekt interpretieren kann und sie insgesamt, auch für den Autor, einfach verständlich bleibt.

Mit der Kategorie **Kreativität und Originalität** wird betrachtet, wie weit die Ergebnisvariabilität eingeschränkt wird. Dazu wird als erstes bewertet, in welchem Rahmen das Modell dazu aufgefordert wird, kreative

Tab. 2.1 Das Valisory-Prompt-Qualitätsmodell

Das Valisory-Prompt-Qualitätsmodell				
Kriterium	Relevanz	Kreativität und Originalität	Sprachliche Qualität und Stil	Zielgruppen-adäquanz
Faktoren	1. Frage-stellung 2. Spezifität 3. Verständlich-keit	1. Innovations-potenzial 2. Inspiration 3. Variabilität	1. Klarheit 2. Anpassung an Stil 3. Strukturie-rung	1. Zielgruppen-verständnis 2. Ansprache 3. Nützlichkeit

Lösungen zu entwickeln (*Innovationspotenzial*). Als nächstes wird mit dem Aspekt der *Inspiration* bewertet, zu welchem Grad der Prompt zum Nutzen und Finden von originellen Ideen anregt und Perspektiven aufzeigt, die nicht trivial oder offensichtlich sind. Schließlich bewertet der dritte Faktor, die *Variabilität*, welchen Spielraum der Prompt für verschiedene Antworten oder Lösungswege bietet.

Als nächstes wird mit der Dimension **Sprachliche Qualität und Stil** die Effektivität der Formulierung des Prompts einbezogen. Zunächst wird dazu mit dem Kriterium der *Klarheit* geprüft, inwieweit der Prompt sprachlich eindeutig und präzise ist. Ziel ist es, mehrdeutige oder irreführende Formulierungen zu vermeiden. Weiterhin wird bewertet, ob die Formulierungen passend für die Zielsetzung sind (*Anpassung an den Stil*). Ist ein Prompt beispielsweise eher technisch, formell oder informell geschrieben, passt sich das Sprachmodell daran an. Der dritte Aspekt dieses Kriteriums ist die *Strukturierung*. Ist der Prompt logisch strukturiert und fördert er die Entwicklung einer strukturierten Antwort?

Die ersten drei Kriterien zielen besonders auf die Kommunikation zwischen Benutzer und Sprachmodell ab. Mit dem letzten Kriterium, der **Zielgruppenadäquanz**, wird ergänzend betrachtet, wie weit die Ergebnisse des Prompts der Zielgruppe entsprechen. Dazu wird als erstes bewertet, in welchem Umfang der Prompt das Vorwissen, die Interessen und die Bedürfnisse der Zielgruppe mit einbezieht und dem LLM verständlich erklärt (*Zielgruppenverständnis*). Dann wird auch für dieses Kriterium mit einbezogen, wie die Ansprache im Prompt zur Zielgruppe passt (*Ansprache*). Während es beim Kriterium Sprachliche Qualität und Stil vor allem um die Formulierung des Prompts selbst ging, wird der Fokus hier auf die Verwendung expliziter Angaben zum Stil für die Zielgruppe gesetzt. Als letzter Aspekt wird untersucht, wie nützlich der durch den Prompt angestrebte Antworttyp für die Zielgruppe ist, in welchem Umfang er praktisch einsetzbar und ob er grundsätzlich von Interesse ist (*Nützlichkeit).*

Mit diesen insgesamt 12 Faktoren des Valisory-PQM können Prompts an KI-Modelle vergleichbar hinsichtlich ihrer Qualität analysiert werden. Auch wenn die Bewertung der einzelnen Aspekte erst einmal subjektiv ist, bietet es ein gutes Rahmenwerk zur Qualitätsuntersuchung und zeigt auf, an welchen Punkten Optimierungspotenzial bestehen könnte.

Das Valisory-Prompt-Qualitätsmodell (PQM) zur Bewertung von Prompts

Relevanz

1. **Fragestellung**: Ist der Prompt klar formuliert und fordert er spezifisch die gewünschte Information oder Aktion?
2. **Spezifität**: Sind die Anforderungen und Ziele des Prompts spezifisch genug, um relevante Antworten zu generieren?
3. **Verständlichkeit**: Ist der Prompt einfach und verständlich genug gestaltet, dass das Modell die Anfrage korrekt interpretieren kann?

Kreativität und Originalität

1. **Innovationspotenzial**: Fordert der Prompt das Modell dazu auf, über allgemeines Wissen hinaus kreative oder innovative Lösungen zu finden?
2. **Inspiration:** Regt der Prompt zu originellen Ideen oder Perspektiven an, die nicht trivial oder offensichtlich sind?
3. **Variabilität**: Bietet der Prompt genügend Spielraum für verschiedene Antworten oder Lösungswege?

Sprachliche Qualität und Stil

1. **Klarheit**: Ist der Prompt deutlich und präzise, ohne mehrdeutige oder irreführende Formulierungen?
2. **Anpassung an den Stil**: Ist der Stil des Prompts (formell, informell, technisch etc.) angemessen für die Zielsetzung und das Publikum?
3. **Strukturierung**: Ist der Prompt logisch strukturiert und fördert er eine strukturierte Antwort?

Zielgruppenadäquanz

1. **Zielgruppenverständnis**: Berücksichtigt der Prompt das Vorwissen, die Interessen und die Bedürfnisse der Zielgruppe?
2. **Ansprache**: Ist die Ansprache im Prompt (Direktheit, Tonfall) passend für die Zielgruppe?
3. **Nützlichkeit**: Ist der durch den Prompt angestrebte Antworttyp für die Zielgruppe praktisch und von Interesse?

2.3 Sprachliche Formulierungen von Prompts

Bei der Formulierung von Prompts gilt es, so präzise wie möglich zu sein. Während die Verständnisfähigkeiten zwar umfassend sind, ist unsere Sprache keineswegs immer eindeutig. Versuchen Sie daher auch, mögliche Unklarheiten oder Doppeldeutigkeiten zu erklären. Das gilt genauso für Fachbegriffe, vor allem, wenn diese nicht gängig sind oder je nach Kontext eine andere Bedeutung haben. Denken Sie beispielsweise an den Begriff „Konto", der beispielsweise in der Buchhaltung für etwas anderes steht als in der Systemadministration oder dem Finanzwesen. Beachten Sie auch, dass das LLM nicht Ihren Informationskontext hat. Es kann sinnvoll oder gar notwendig sein, Informationen mit in den Prompt zu geben, die für Sie trivial oder offensichtlich erscheinen.

Nutzen Sie Umgangssprache spärlich und verwenden Sie auch Satzzeichen – sie können die Bedeutung von Sprache deutlich verändern. Formulieren Sie insgesamt positiv soweit möglich. Anweisungen wie „Fokussiere dich auf die Stärken" bringen regelmäßig bessere Ergebnisse als „Fokussiere dich nicht auf die Schwächen".

Und obgleich dieser Empfehlungen und Praktiken brauchen Sie am Ende nicht wie mit einer Person zu kommunizieren; Floskeln wie „Bitte", „Danke" oder auch ein „Hallo" bringen Ihnen erst einmal keine Vorteile in der Interaktion.

Im Folgenden werden für alle vorgestellten Tipps zum Thema Prompting immer auch praktische Beispiele geliefert. Um für Sie einen größtmöglichen Wert aus diesem Buch zu ziehen, empfehlen wir Ihnen, die genannten Beispiele in einem LLM Ihrer Wahl nachzuvollziehen. So gewöhnen Sie sich Schritt für Schritt die vorgestellten Techniken an. Es empfiehlt sich weiterhin, die erklärten Techniken in einem nächsten Schritt auf Beispiele aus der eigenen Lebenswelt zu übertragen. So sehen Sie schnell einen Erfolg. Wenn Sie abschließend einfache Prompts und deren Ergebnisse mit denen, die mit Hilfe von gutem Prompt Engineering erstellt wurden, gegenüberstellen, werden Sie erstaunt sein, wie viel besser, ausführlicher und präziser die Antworten des LLMs sind.

Sie werden sehen: Übung macht den Meister. Und da Prompt Engineering nicht das Erlernen einer neuen Programmiersprache voraussetzt, sondern „nur" einen bedachten Einsatz Ihrer natürlichen Sprache, gewisser Sprachkonstrukte und -abläufe darstellt, werden Sie schnell Erfolge sehen und das Prompt Engineering wird Ihnen bald in Fleisch und Blut übergehen.

2.4 Ausgabeformat bestimmen

Das Format der Ausgaben des Sprachmodells kann auf vielfache Art und Weise beeinflusst werden. Überlegen Sie sich, in welcher Struktur und Formatierung Sie das Ergebnis am besten brauchen und geben Sie das dem Modell erst einmal so mit. Sie können Ergebnisse beispielsweise in Form einer Liste ausgeben lassen, diese auf eine bestimmte Art und Weise sortieren lassen und spezifische Informationen beifügen lassen oder auch explizit ausklammern – z. B., falls das LLM diese von sich aus mit einbaut.

> Wenn das Ergebnis nicht ganz passt, ist das gar kein Problem. Sie müssen nicht den gesamten Prompt wiederholen – teilen Sie dem Modell einfach mit, welche Teile des Ergebnisses angepasst werden sollen.

Lassen Sie uns das anhand eines Beispiels verdeutlichen. Wir bitten ChatGPT, eine Liste der zehn größten Städte der Welt zu erstellen.

Beispiel

Erstelle eine Liste der zehn größten Städte der Welt.

Die Liste kann nun vielfältig transformiert und an Ihre Bedürfnisse angepasst werden. Beispielsweise können Sie sie mit dem folgenden Prompt absteigend sortieren, die Nummerierung weglassen und auch die jeweiligen Länder hinzufügen lassen.

Beispiel

Promptbeispiel

Sortiere die Liste absteigend, basierend auf der Einwohnerzahl. Lasse die Nummerierung weg. Füge zu jeder Stadt noch das jeweilige Land und die Fläche in Quadratkilometern hinzu.

Da das LLM ohne eine konkrete Anweisung bei der Darstellung und Formatierung der Antwort eigene Annahmen trifft, sollten Sie auch hier passende Anweisungen geben. Denn sonst erhalten Sie Ihr Ergebnis mal als Tabelle, mal als nummerierte Aufzählung und mal als Fließtext.

Beispiel

Formatiere das Ergebnis als sortierte Tabelle und schreibe die Namen der einzelnen Städte fett. Alle dargestellten Zahlen sollen auf die nächste Zehnerstelle auf- oder abgerundet und ohne Komma dargestellt werden. Die Tabelle soll dabei wie folgt strukturiert sein:
Tabellenkopf: Die größten Städte der Welt
Spalten: Nummer, Name der Stadt, Land, Anzahl der Einwohner, Fläche in qkm

Hier enden die Formatierungsmöglichkeiten aber noch längst nicht. Große Sprachmodelle sind in der Lage, Texte, Daten und ganze Tabellen in fast beliebige Darstellungen umzuwandeln. So können Sie die Tabelle aus dem Beispiel auch in das „CSV"-Format (comma-separated values) umwandeln lassen. Dieses Format wird z. B. von allen gängigen Tabellenkalkulationsprogrammen gelesen. Somit können Sie beispielsweise komplette Tabellen direkt in Excel weiterverarbeiten.

Auch eine Formatierung in HTML mit begleitendem CSS ist mit nur einem Prompt möglich. Somit kann die Tabelle direkt in eine Homepage eingebaut und in beliebigen Browsern betrachtet werden.

Die Formatierungsmöglichkeiten sind schier endlos. Nehmen Sie sich also bevor Sie den Prompt abschicken einen Moment Zeit und überlegen Sie sich, wie das Ergebnis strukturiert und formatiert werden soll. Beschreiben Sie dann Ihre Anforderungen so genau wie möglich. So ma-

chen sie dem LLM ganz präzise klar, was es zu tun hat und Sie sparen sich die Zeit, die Antwort im Nachgang noch weiter anzupassen.

2.5 Referenzierung von Texten

Regelmäßig kommt es vor, dass vorhandener Text in Ihren Prompt integriert werden soll, um ihn beispielsweise zusammenzufassen, umzuschreiben oder ihn als Beispiel im Prompt zu benutzen. Diese Texte einfach unter dem Prompt anzuhängen, funktioniert durchaus, kann aber verschiedene unerwünschte Nebeneffekte haben. Daher gibt es verschiedene Techniken, mit denen zwischen dem eigentlichen Prompt und beigefügten Texten unterschieden werden kann.

Mehrzeilige Texte können mit drei Anführungszeichen (“““) oder Rauten (###) vor und hinter dem Text abgegrenzt werden. Die Abgrenzungen werden dazu jeweils auf einer eigenen Zeile platziert.

Beispiel

Formuliere das folgende Haiku im Stil von Franz Kafka um:

"""

Worte weisen Weg,

In Stille spricht die Seele,

Sprache brückt die Welt.

"""

Vor allem in umfangreicheren Prompts kann es notwendig sein, mehrere Texte zu verarbeiten oder sich wiederholende Situationen (beispielsweise einen Dialog) darzustellen. Im Prompt können dafür Abschnitte genutzt werden, mit denen Textpassagen später referenziert werden können. Abschnitte haben einen Namen, gefolgt von einem Doppelpunkt und dem Kontext, auf den sie sich beziehen. Der Kontext kann ein Satz sein oder auch ein mehrzeiliger Text. Es ist auch möglich, sich damit bei-

spielsweise auf angehängte Dateien zu beziehen. Es ist keineswegs so, dass das Sprachmodell bei Abschnitten die Syntax erkennt und die folgenden Informationen als festes Sprachkonstrukt verarbeitet. Vielmehr wird auch hier wieder die Intention verstanden. Daher kann es damit umgehen, wenn dort feste Daten (beispielsweise ein wörtliches Zitat) oder auch etwas zum Interpretieren steht.

Beispiel

Hier sind zwei Aussagen. Formuliere die Technikantwort freundlicher um.

Anfragenaussage: Das System ist nicht verfügbar.

Technikantwort:

###

Es gibt eine Unregelmäßigkeit im Betriebsablauf.

###

2.6 Lösungswege angeben

Prompts effektiv zu formulieren und sinnvoll zu strukturieren, hat bereits großen Einfluss auf die Qualität der Ergebnisse. Wenn das Ergebnis jedoch nicht den eigenen Erwartungen entspricht, das LLM implizit Annahmen mit einbaut oder gestellte Anforderungen nicht beachtet, kann es helfen, den Lösungsweg zu strukturieren. Erklären Sie dem Modell, wie es vorgehen soll, um zu einer Lösung zu kommen. Sie können sich auch die Zwischenschritte erklären lassen, um die Qualität des Vorgehens zu kontrollieren.

Testen Sie das einmal, indem Sie einen einzelnen Prompt zu folgendem Szenario schreiben: Das Sprachmodell soll für Sie einen Instagram-Post für ein Unternehmen Ihrer Wahl entwickeln. Dazu sollen im ersten Schritt mögliche Zielgruppen identifiziert werden, dann soll daraus die beste Zielgruppe ausgewählt werden. Für diese Zielgruppe soll nun eine SWOT-Analyse (Strengths, Weaknesses, Opportunities, Threats) durch-

geführt werden, wobei pro Segment nur vier Faktoren verwendet werden sollen – damit grenzen Sie den Aufwand genug ein, damit die Antwort nicht allzu lange dauert und trotzdem eine gewisse Aussagekraft hat. Danach soll das LLM den primären *Pain Point* der Zielgruppe ableiten. Schließlich soll im letzten Schritt ein Instagram-Post entwickelt werden, wobei sowohl das Bild als auch die Bildunterschrift erläutert werden soll.

Das Ergebnis eines solchen Prompts ist recht umfangreich, da die Zwischenschritte jeweils vom LLM erklärt werden, wodurch Teilergebnisse sowie das Endergebnis leicht nachvollziehbar sind – der Gedankengang des Modells wird also deutlich.

Passt etwas nicht? Sie können die Durchführung wiederholen oder auch nur Teile der Zwischenschritte anpassen lassen.

2.7 Das Interviewformat

Die bisherigen Prompts waren grundsätzlich in sich geschlossen und haben dem LLM konkrete Anweisungen sowie alle relevanten Informationen gegeben. Während das ein gängiger und intuitiver Ansatz ist, kann mehr Interaktion vor allem in detailreichen Situationen die Gesamtergebnisse deutlich verbessern. Dazu wird dem Modell die Möglichkeit gegeben, den Nutzer nach Informationen zu fragen.

In der Grundvariante werden die zu stellenden Fragen konkret definiert und im Prompt angegeben. Beispielsweise kann das LLM angewiesen werden, nach Details zu einer Zielgruppe zu fragen. Es wird dann an geeigneter Stelle eine entsprechende Nachfrage stellen, die der Nutzer ganz normal im Chat-Fenster beantwortet. Dann wird mit der Verarbeitung fortgefahren. Das lässt sich sehr gut mit dem Konzept des Angebens von Lösungswegen aus Abschn. 2.6 verbinden.

Die zu stellenden Fragen mit in den ursprünglichen Prompt zu geben, fußt auf der Idee, dass Sie bereits wissen, welche Fragen im Verlauf der Arbeit an der Problemstellung relevant sind. Sie können darüber hinaus auch das LLM entscheiden lassen, welche Fragen es für relevant für die

Beantwortung Ihres Prompts hält, indem Sie es anweisen, nach fehlenden Informationen zu fragen und klärende Nachfragen zu stellen.

Im Folgenden wird ein Beispiel-Prompt gezeigt, der mit passenden Rückfragen einen kleinen LinkedIn-Beitrag verfasst.

Beispiel

Erstelle einen kurzen LinkedIn-Beitrag für mich. Ich möchte aus meinem aktuellen Projekt berichten und aufzeigen, was bereits alles erfolgreich für den Kunden umgesetzt wurde. Der Beitrag soll dazu einen leicht vertrieblichen Ansatz haben und die Leser dazu anregen, sich bei ähnlichen Projekten bei mir zu melden.

Stelle mir zur Konkretisierung des Beitrags passende Fragen, um einen optimalen Post für meine Zielgruppe zu erstellen.

Sie müssen an keiner Stelle explizit sagen, dass Ihre Antworten auf die Fragen mit in die Beantwortung einfließen sollen – das passiert ganz automatisch.

Sie können auch Wiederholungen einbauen, indem Sie beispielsweise anweisen, solange zu fragen, bis alle relevanten Informationen vorhanden sind.

2.8 Auf einen Blick

- Mit Prompt Engineering wird die Formulierung von effektiven Anfragen an generative KI-basierte Systeme bezeichnet, die optimale Antworten verspricht.
- Mit Hilfe des Prompt-Qualitätsmodells (PQM) kann die Qualität der KI-Anfragen strukturiert bewertet werden. Dies liefert die Basis für eine Optimierung der Prompts.
- Prompt Engineering erfordert nicht das Erlernen einer neuen Programmiersprache. Es nutzt lediglich den bewussten Einsatz von er-

probten natürlichsprachlichen Phrasen, Konstrukten, Abläufen und Anweisungen.

- Prompts sollten eine sehr konkrete Beschreibung des gewünschten Ergebnisses beinhalten, um sicherzustellen, dass die impliziten Erwartungen an das Ergebnis getroffen werden.
- Ein KI-System kann eigenständig nach möglichen Lösungswegen für ein gegebenes Problem suchen. Mit Hilfe eines geeigneten Prompts können aber auch klare Vorgaben gemacht werden.

3

Frameworks für Prompts

Zusammenfassung In diesem Kapitel werden erprobte Wege vorgestellt, die als strukturelle Richtlinie zur Erstellung von möglichst guten Prompts dienen. Diese sogenannten Prompt Frameworks bieten Vorgaben, an denen man sich bei der Erstellung von Anfragen an KI-Systeme sprachlich und strukturell orientieren kann. Im Laufe dieses Kapitels werden dazu unterschiedliche Prompt Frameworks vorgestellt, die in der Praxis gute Ergebnisse zeigen. Die Anwendungen der einzelnen Vorgaben werden dabei jeweils an praktischen Beispielen gezeigt.

Mit den Grundlagen der Prompt-Entwicklung kann bereits viel mit dem jeweiligen Sprachmodell in puncto Qualität erreicht werden. Über die Zeit wurden darauf aufbauend verschiedene Frameworks entwickelt, die jeweils einen strukturellen Aufbau von Prompts vorgeben, um bessere Antworten zu erhalten. Die Strukturen sind dabei keineswegs nur Selbstzweck; für viele Prompt-Frameworks gibt es wissenschaftliche Forschung, die die Effektivität der jeweiligen Frameworks belegt. Im Folgenden werden verschiedene gängige Frameworks zum Schreiben effektiver Prompts dargestellt.

D. Koch et al., *Prompt Engineering im Unternehmen – eine Einführung*,
https://doi.org/10.1007/978-3-658-47699-1_3

3.1 Prompt Chaining

Das Prompt Chaining ist eine sehr einfache, aber zugleich effektive Möglichkeit, mit der die Ergebnisse des Sprachmodells zumeist deutlich positiv beeinflusst werden können. Beim Prompt Chaining wird die Aufgabe durch Sie gedanklich in mehrere Schritte zerlegt und diese Schritte werden dann mit dem LLM einzeln erarbeitet. In jedem Zwischenschritt kontrollieren Sie die Ergebnisse, nehmen ggf. Anpassungen vor und fahren dann mit dem nächsten Aspekt fort. Es funktioniert im Grunde ähnlich zu der fortgeschrittenen Technik des Chain-of-Thought-Promptings (vgl. Abschn. 4.3). Prompt Chaining bedeutet im Kontrast dazu allerdings mehr manueller Aufwand und bedarf häufig mehr Zeit für gleich gute Ergebnisse.

Da im Rahmen des Prompt Chainings unter Umständen viel mit dem Sprachmodell an einer Aufgabenstellung gearbeitet wird, ist es wichtig, die Größe des Context Window (vgl. Abschn. 1.3) zu beachten.

Sie können Prompt Chaining auch mit anderen Prompt-Frameworks kombinieren.

Nutzen wir als Beispiel einmal das Schreiben eines Artikels zum Thema Fahrräder. Wir wollen dafür mit dem Modell relevante Themen finden, diese sinnvoll strukturieren und dann den Artikel schreiben lassen.

Für den ersten Schritt wird das Modell aufgefordert, relevante Themenfelder zu identifizieren. Dabei wird bewusst auf Erläuterungen verzichtet, um sich auf das Wesentliche zu konzentrieren. Dem Modell werden genug Informationen über das Ziel mitgegeben, damit es die Aufgabe erledigen kann.

Beispiel

Dein Ziel ist es, einen Artikel über die Mechanik eines handelsüblichen Fahrrads zu schreiben. Wir werden den Artikel Schritt für Schritt erarbeiten.

Gib mir als erstes eine unkommentierte Liste von Themen, die relevant sind.

Je nach LLM wird hier eine unterschiedlich umfangreiche Liste an Ergebnissen produziert. Falls Themen fehlen, das Modell irrelevante Aspekte gefunden hat oder komplett auf dem falschen Weg ist, kann hier nun frühzeitig korrigierend eingegriffen werden, indem es zur Anpassung aufgefordert wird.

Passt der erste Ansatz, kann in der Erstellung des Artikels fortgefahren werden. Die gefundenen Themen können unter Umständen nicht alle in dem Artikel behandelt werden. Daher soll eine Eingrenzung vorgenommen werden, die das Modell auch begründen soll, damit das Vorgehen nachvollzogen werden kann.

Beispiel

Wähle die fünf relevantesten Themen begründet aus.

Sofern die Struktur passt, kann sich nun dem Inhalt des Artikels gewidmet werden. Bevor dieser tatsächlich geschrieben wird, empfiehlt es sich, eine konkretere Übersicht der Themen zu entwickeln. Dazu könnte das LLM beispielsweise wie folgt aufgefordert werden:

Beispiel

Erstelle eine Inhaltsübersicht. Bringe die Themen in eine Reihenfolge, die inhaltlich einen nachvollziehbaren roten Faden hat. Füge noch eine Einleitung und ein Fazit ein. Schreibe zu jedem der Punkte drei Stichpunkte, was der Inhalt sein wird.

An dieser Stelle können nun wieder problemlos inhaltliche Änderungen vorgenommen werden, bevor zu viel Zeit in das eigentliche Schreiben des Artikels investiert wird. Zum Beispiel könnte das LLM noch aufgefordert werden, ein Storytelling-Framework zu nutzen oder aufzuzeigen, wie sinnvolle Kapitelübergänge inhaltlich umgesetzt werden sollen. Schließlich kann das Schreiben des Artikels aber auch abgeschlossen werden.

Beispiel

Schreibe jetzt auf Basis der erarbeiteten Punkte den Artikel mit ca. 300 Wörtern pro Thema.

Prompt Chaining mit Experten-Leveln

Eine Abwandlung der Prompt-Chaining-Technik ist das Arbeiten mit Experten-Leveln. Dabei wird vom Sprachmodell ebenfalls eine Lösung für eine Problemstellung generiert. Dann wird das Konzept einer Bewertungsskala eingeführt, die erste Lösung darauf verortet und dann in zwei Anfragen einmal eine ein wenig bessere und einmal die beste Lösung angefordert. Zur Verdeutlichung wird das Planen eines Webinars genutzt. Im ersten Schritt wird ohne besondere Struktur oder Zusätze eine Lösung gefordert.

Beispiel

Entwickle einen Drei-Schritte-Plan zum Planen eines optimalen Webinars. Gib jeden Schritt als kurzen, prägnanten Satz aus.

Das Ergebnis dieses Prompts wird simpel, aber umfassend und sinnvoll sein. Die jeweiligen Schritte werden in genau einem Satz kurz umrissen, ohne zu tief ins Detail zu gehen. Als nächstes wird nun die Idee einer Bewertungsskala vorgestellt und dieses Ergebnis auf der ersten Stufe festgelegt. Dann soll das LLM ein Ergebnis der zweiten Stufe produzieren. Ohne zu definieren, was ein Ergebnis der Stufe 1 von einem der Stufe 2 unterscheidet, ist nicht zu erwarten, dass das nächste Ergebnis signifikant besser oder umfangreicher werden wird. Das Modell arbeitet hier mit Annahmen zur Qualität, die für die Umsetzung auch nicht von signifikanter Relevanz sind.

Beispiel

Nimm an, diese Lösung sei auf einer Bewertungsskala eine 1 von 10. Entwickle nun eine Lösung, die der Stufe 2 entspräche.

Das für diesen Schritt produzierte Ergebnis ist erfahrungsgemäß nicht weit von der Lösung des ersten Schritts entfernt. Es beinhaltet unter Umständen mehr Aspekte und ist etwas weiter ausgeführt, wird aber sonst recht ähnlich sein. Um nun den großen Qualitätssprung zu erhalten, soll das LLM als nächstes ein Ergebnis der fiktiven Stufe 10 entwickeln. Auch hier wird weiterhin nicht definiert, woran eine Bewertung von 10 festgemacht würde. Allerdings hat das Modell bereits zwei Datenpunkte, eines für Stufe 1 und eines für Stufe 2. Aus den beiden kann es Rückschlüsse darauf ziehen, wie ein um ein vielfaches besseres Ergebnis aussehen könnte.

Beispiel

Entwickle nun eine Lösung der Stufe 10.

Das Ergebnis dieses letzten Schritts wird deutlich von den beiden vorherigen abweichen und mehr Informationen beinhalten. Da im Eingangsprompt die Einschränkung getroffen wurde, nur einen Satz verwenden zu dürfen, wird das Modell in den Formulierungen beispielsweise mehr mit verschachtelten Sätzen und Aneinanderreihungen mit Semikolons arbeiten.

Sie können nach dem ersten Ergebnis auch definieren, dass es sich bei dem aktuellen Text um eine Antwort auf dem „Beginner-Level“ handelt und die Antwort nochmal überarbeitet werden soll. Dieses Mal soll das „Fortgeschrittenen-Level“ erreicht werden. Das Ergebnis wird wieder ausführlicher und umfangreicher sein. In einem letzten Schritt können Sie dann noch das „Experten-Level“ einfordern. Sie werden sehen, dass sich die Qualität des Ergebnisses nochmal drastisch verbessern wird und der Text an Länge und Tiefe gewinnen wird. Es hat sich dabei in der Praxis gezeigt, dass weitere Optimierungen dieser Form keine großartig besseren Texte mehr liefern. Insgesamt kann mit diesem recht einfachen Vorgehen die Ergebnisqualität deutlich gesteigert werden.

3.2 Role Prompting

Um den Kontext der Aufgabenstellung und der Ergebnisse signifikant zu beeinflussen, kann das Modell zur Bearbeitung in eine spezifische Rolle versetzt werden. Wie auch beim Prompt Chaining (vgl. Abschn. 3.1) ist Role Prompting ein sehr einfaches, aber ungemein effektives Mittel, um die Qualität der Ergebnisse zu verbessern.

Dazu wird dem LLM eine Rolle zugewiesen, aus deren Blickwinkel es die Aufgabenstellung erarbeiten soll. Eine solche Rolle kann beispielsweise eine Stellenbeschreibung oder die Funktion innerhalb eines Unternehmens sein, wie beispielsweise eine Softwareentwicklerin oder ein Buchhalter. Stellen Sie sich einfach vor, dass Sie Ihre Anfrage an den besten Experten für Ihre Fachfrage stellen könnten und weisen Sie dem LLM genau diese Rolle zu. Dabei können sowohl existierende bekannte Personen wie auch fiktive Personen gewählt oder abstrakte Rollenbeschreibungen genutzt werden.

Beispiele für unterschiedliche Rollen:

Beispiel

- Beantworte die folgende Frage genauso, wie es der griechische Philosoph Platon getan hätte.
- Verhalte dich genauso emotionslos wie der Vulkanier Spock vom Raumschiff Enterprise und beantworte die folgende Frage streng logisch.
- Du bist ein erfahrener Rechtsanwalt mit über 20 Jahren Berufserfahrung im Bereich Arbeitsrecht und berätst mich bei der folgenden juristischen Frage ausführlich.

Wenn davon auszugehen ist, dass die Rolle dem Modell bekannt (genug) ist, dann ist es nicht zwingend nötig, weitere Informationen hinzuzufügen. Die Rolle wird typischerweise am Anfang des Prompts mit Formulierungen wie „Verhalte dich wie …“ oder „Du bist ein …“ definiert. Es macht aber tatsächlich erst einmal keinen Unterschied, an welcher Stelle eine solche Formulierung eingebaut wird.

Beispiel

Du bist ein erfahrener Agile Coach. Entwickle einen Eintagesworkshop für ein Programmierteam zum Thema Scaled Agile Framework.

Vor allem im Unternehmenskontext ist zu erwarten, dass die großen Sprachmodelle sowohl viele gängige als auch weniger übliche Rollen bereits kennen. Im Zweifel fragt das Modell nicht nach, sondern nimmt im schlechtesten Fall etwas an, weshalb es sinnvoll ist, in der Rollendefinition präzise zu sein.

Sofern die Rolle präziser spezifiziert werden soll oder wenig Kenntnisse darüber seitens des Sprachmodells zu erwarten sind, kann es hilfreich oder notwendig sein, weitere Informationen mitzugeben. Das kann beispielsweise eine konkretere Beschreibung der Rolle sein oder auch ein zusätzliches Informationsmaterial. Dabei kann die Rolle im Rahmen des Prompts textuell ausführlich beschrieben werden oder es kann eine Datei hochgeladen werden, die eine konkrete Rollenbeschreibung beinhaltet.

Beispiel

Du bist ein JAVA-Softwareentwickler. In der beigefügten Datei „Skills.pdf“ findest du eine Übersicht deiner Fähigkeiten und Erfahrungen.

Interessanterweise kann es tatsächlich zu besseren Ergebnissen führen, wenn man den Grad der Erfahrung (beispielsweise in Jahren) entsprechend definiert.

Das Role Prompting ist so effektiv und einfach, dass es Bestandteil von fast jedem anderen Prompting-Frameworks ist. Es wird ganz klar empfohlen, Role Prompting als grundsätzlichen Prompt-Baustein in alle Prompts mit einzubauen, die mehr sind als eine einfache Frage.

3.3 Deduktives Problemlösen

Bei der Prompt-Technik des deduktiven Problemlösens wird ähnlich dem Prompt Chaining in zwei Schritten vorgegangen. Im ersten Schritt wird mit dem Modell ein allgemeiner Plan oder ein generelles Szenario entwickelt. Im zweiten Schritt wird dieses dann auf die konkrete Situation des Nutzers angewandt.

Das Vorgehen entspricht im Grunde der menschlichen Herangehensweise an eine große Problemstellung. Es wird gedanklich ein Grundmodell entwickelt und dieses dann ganz selbstverständlich angewendet. Bei der Planung einer Veranstaltung stellt man sich beispielsweise eingangs durchaus die Frage: „Was muss man bei der Planung einer Veranstaltung grundsätzlich tun?“. In diesem Sinne wird bei dieser Prompt-Technik das Problem in zwei Schritte zerlegt.

3.3.1 Entwicklung eines Metamodells

Der erste Schritt beim deduktiven Problemlösen besteht darin, auf der Metaebene ein allgemeingültiges Vorgehen zu entwickeln. Dazu wird das Sprachmodell unabhängig des Kontexts des Nutzers aufgefordert, eine allgemeingültige Lösung zu entwerfen, die das Problem möglichst optimal löst. An dieser kann dann Schritt für Schritt weitergearbeitet werden, bis sie einen für die eigentliche Aufgabenstellung adäquaten Stand erreicht hat.

Beispiel

Entwickle eine Strategie, wie man eine Mitarbeiterversammlung für ein Unternehmen plant.

3.3.2 Anwendung des Modells

Wenn das Metamodell fertig ist, wird es auf das eigentliche Problem angewandt. Dazu bezieht man sich im Prompt auf das allgemeine Modell und beschreibt die Situation und das Ziel. An dieser Stelle kann zur Strukturierung gut eines der anderen Prompt-Frameworks eingesetzt werden.

Beispiel

Wende die gerade beschriebene Strategie zur optimalen Planung einer Mitarbeiterversammlung nun konkret auf die Planung der Mitarbeiterversammlung meines Unternehmens an. Mein Unternehmen hat 250 Mitarbeiter an 10 Standorten im DACH-Raum. Die Veranstaltung soll einen Tag lang dauern und soll vor Ort in unserer Zentrale in Berlin stattfinden.

3.4 Das R-T-F-Framework (Role-Task-Format)

Mit dem R-T-F-Framework wird das Modell in eine konkrete Rolle versetzt, bevor es die Aufgabenstellung des Prompts bearbeitet. Die Abkürzung steht für die drei Bausteine des Prompts: Die Rolle, die das LLM einnehmen soll (Role, R), die Aufgabe, die es in dieser Rolle erfüllen soll (Task, T) sowie das zu produzierende Format der Ergebnisse (Format, F).

3.4.1 Die gedankliche Rolle des Sprachmodells im Prompt

Der erste Schritt des R-T-F-Frameworks ist es, dem LLM mitzuteilen, welche Rolle es bei der Beantwortung der Frage einnehmen soll. Hierbei gehen Sie genauso vor wie bereits in Abschn. 3.2 „Role Prompting“ beschrieben wurde.

Die Rolle zu kommunizieren ist denkbar einfach. Dies ist die erste Anweisung eines R-T-F-Prompts und folgt keiner bestimmten Syntax. Hier sind ein paar Beispiele:

Beispiel

Du bist ein Softwarearchitekt.

Versetze dich in die Rolle eines Verkaufscoaches.

Agiere als eine Social-Media-Beraterin.

Nicht immer reicht es, die Rolle so ohne weitere Informationen zu verwenden. Beispielsweise kann die Rolle eines Verkaufscoaches sehr unterschiedlich sein. Wer wird gecoached? Unternehmen? Selbstständige? Was ist deren Zielgruppe? Geht es um Closing-Coaching oder befinden wir uns im Kontext telefonischer Kaltakquise?

Um die Rolle konkreter zu spezifizieren, können dieser einfach zusätzlich noch weitere Rahmenbedingungen mitgegeben werden. Auch hier gilt: je genauer der Kontext dargestellt wird, desto besser werden die Ergebnisse am Ende.

Beispiel

Du bist ein Abteilungsleiter der Logistik eines mittelständischen Unternehmens. Du hast 10 Mitarbeitende unter dir und planst ein Teamevent zur Verbesserung des Teamzusammenhalts.

3.4.2 Die Aufgabe des Prompts

Nachdem die Rolle definiert und ggf. mit zusätzlichen Rahmenbedingungen konkretisiert wurde, wird im zweiten Teil des Prompts die konkrete Aufgabe gestellt. Ein besonderes Format gibt es hierbei nicht; der Prompt wird an dieser Stelle so formuliert, wie er auch ohne spezielles Framework formuliert würde.

Wichtig ist, dass das Modell an dieser Stelle in der zuvor angegebenen Rolle agiert. Daher können hier problemlos Fachbegriffe oder Abkürzungen verwendet werden, sofern diese im Kontext der Rolle als allgemein bekannt anzusehen sind.

Beispiel

Du bist ein Abteilungsleiter der Logistik eines mittelständischen Unternehmens. Du hast 10 Mitarbeitende unter dir und planst ein Teamevent zur Verbesserung des Teamzusammenhalts.

Deine Aufgabe ist es, ein zweitägiges Team-Offsite zu planen. Es sollen Aktivitäten aus den Bereichen Kochen, Sport, Meditation und Teambuilding geplant werden. Die Tage beginnen jeweils um 9 Uhr und enden um 18 Uhr.

3.4.3 Das Format der Ausgabe

Der dritte Teil des R-T-F-Frameworks definiert das Format der Ergebnisse. Dabei kann es sich beispielsweise um eine Liste oder einen längeren Text handeln. Im Kern wird der Ansatz aus Abschn. 2.4 verwendet. Auch hier gilt, je konkreter das Format angegeben wird, desto besser sind die Ergebnisse am Ende.

> Erfahrungsgemäß ist das Format der Antwort häufig noch nicht ganz da, wo man hinmöchte. Teilen Sie dem Modell einfach mit, die Ergebnisse entsprechend anzupassen, beispielsweise mit oder ohne Beschreibungen oder in Form einer Liste.

Nachfolgend finden Sie einen kompletten Beispielprompt mit diesem Framework. Ziel ist es, einen Zeitplan für ein Teamevent erstellen zu lassen. Mit Hilfe des R-T-F-Aufbaus des Prompts ist das Ergebnis inhaltlich wie auch strukturell so gestaltet, dass es leicht weiterverarbeitet werden kann. Im Bereich des *Tasks* könnten hier noch mehr Informationen hinzugefügt werden, um besser auf die Situation zugeschnitten zu sein.

Beispiel

Du bist ein Abteilungsleiter der Logistik eines mittelständischen Unternehmens. Du hast 10 Mitarbeitende unter dir und planst ein Teamevent zur Verbesserung des Teamzusammenhalts.

Deine Aufgabe ist es, ein zweitägiges Team-Offsite zu planen. Es sollen Aktivitäten aus den Bereichen Kochen, Sport, Meditation und Teambuilding geplant werden. Die Tage beginnen jeweils um 9 Uhr und enden um 18 Uhr.

Entwickle einen Zeitplan für die beiden Tage. Gib jeweils Start- und Endzeit an, entwickle einen Titel und gib eine kurze Beschreibung der jeweiligen Aktivität.

3.5 Das T-A-G-Framework (Task-Action-Goal)

Mit dem T-A-G-Framework wird der Fokus auf die Erfüllung einer konkreten Aufgabe gelegt, die in einzelne Schritte unterteilt und einem Gesamtziel untergeordnet wird. Dabei wird dem LLM die Entscheidungsfreiheit bzgl. Format und Struktur der Ergebnisse gelassen. Der Name T-A-G steht für Task (Aufgabe), Action (Aktion) und Goal (Ziel).

3.5.1 Die Aufgabe des Prompts

Der erste Teil des T-A-G-Frameworks beschreibt die Aufgabe, die erledigt werden soll. Dem Modell wird kurz und prägnant mitgeteilt, was die Erwartung ist und was es konkret tun soll. Die Aufgabenbeschreibung muss dabei keineswegs vollständig oder absolut präzise sein, da ja noch weitere Informationen folgen.

Beispiele für die Aufgabe könnte das Erstellen einer Marketingstrategie sein, das Durchführen einer Potenzialanalyse für ein Produkt oder auch das Schreiben eines Angebots. Die Aufgabe sollte nicht zu kleinteilig gewählt sein, denn im nächsten Teil wird sie noch in Schritte zur Erreichung aufgeteilt. Wir bauen hier beispielhaft eine mehrschritte Potenzialanalyse für ein fiktives Produkt auf:

Beispiel

Deine Aufgabe ist das Entwickeln einer Potenzialanalyse für einen innovativen Fruchtsaft. Die Innovation besteht im künstlichen Zusetzen von Vitaminen und Mineralstoffen.

3.5.2 Die durchzuführenden Aktionen

Nachdem die Zielaufgabe des Prompts definiert wurde, werden im zweiten Teil die zur Erreichung notwendigen Schritte aufgeführt (Actions). Diese bauen üblicherweise aufeinander auf, starten etwas allgemeiner und arbeiten sich Schritt für Schritt an die Aufgabe des Prompts heran.

Das LLM wird die Schritte in der angegebenen Reihenfolge durchlaufen. Achten Sie daher darauf, diese sinnvoll zu organisieren. Geben Sie für jeden Schritt alle notwendigen Informationen mit. Auch hier können Vorgaben hinsichtlich Struktur und Formatierungen verwendet werden. Wenn die Beschreibungen der Aktionen dadurch zu unübersichtlich werden, nutzen Sie strukturierende Elemente wie eine nummerierte Liste für die Aktionen oder Abschnitte.

Sie können auch das Sprachmodell anweisen, die Reihenfolge der Aktionen selbst festzulegen.

Die Aktionen können gut mit dem Konzept von Rollen aus dem R-T-F-Framework kombiniert werden.

Hier ist der zweite Teil des Prompts aus Abschn. 3.5.1. Die Aktionen sind kurz und knapp angegeben, ohne große Anforderungen an Formate zu stellen oder viele Details mitzugeben. Das Modell wird zusätzlich häufig noch ein Fazit aus den Ergebnissen ziehen, auch wenn dies gar nicht gefordert wurde.

Beispiel

Identifiziere kurz mögliche Zielgruppen. Wähle eine Zielgruppe begründet aus. Führe für diese Zielgruppe eine SWOT-Analyse mit jeweils drei Faktoren durch.

3.5.3 Das übergeordnete Ziel

Der letzte Teil des T-A-G-Frameworks ist das Ziel (Goal), das die vorherigen Informationen unter einen übergeordneten Zweck stellt. Dadurch werden das Vorgehen des Sprachmodells und die generierten Ergebnisse noch einmal explizit zielorientiert ausgerichtet. Weiterhin werden mit diesem Teil noch zusätzliche Rahmenbedingungen definiert.

Im Beispiel sollte anhand einer Zielgruppenauswahl und einer SWOT-Analyse eine Potenzialanalyse durchgeführt werden. Der Grund dafür ist nicht bekannt – soll ein Rebranding des Produkts stattfinden? Gibt es das Produkt überhaupt schon? Soll mit der Konkurrenz verglichen werden? Je nachdem, wie man diese Fragen beantwortet, liegt der Schwerpunkt der Potenzialanalyse woanders und die Ergebnisse verändern sich.

In diesem Beispiel soll die Zielgruppe gefunden werden, die voraussichtlich am meisten Umsatz bringen wird. Obwohl das Ziel am Ende steht, beeinflusst es natürlich das gesamte Vorgehen und sowohl die Auswahl der Zielgruppe als auch die darauffolgende Analyse werden dann unter diesem Gesichtspunkt durchgeführt.

Beispiel

Deine Aufgabe ist das Entwickeln einer Potenzialanalyse für einen innovativen Fruchtsaft. Die Innovation besteht im künstlichen Zusetzen von Vitaminen und Mineralstoffen.

Identifiziere kurz mögliche Zielgruppen. Wähle eine Zielgruppe begründet aus. Führe für diese Zielgruppe eine SWOT-Analyse mit jeweils drei Faktoren durch.

Das Ziel ist es, den umsatzstärksten Kundenmarkt zu identifizieren.

3.6 Das B-A-B-Framework (Before-After-Bridge)

Das B-A-B-Framework (Before, After, Bridge) für Prompts stellt die Ist- und Soll-Situation in den Fokus. Es wird genutzt, um einen Plan oder eine Strategie für die Erreichung eines Ziels zu entwickeln. Dafür wird im ersten Schritt die Ausgangssituation dargestellt und danach die Zielsituation beschrieben. Im dritten Teil wird ausgeführt, auf welchem Weg das Ziel erreicht werden soll.

3.6.1 Beschreibung der Ausgangssituation

Im ersten Teil wird beschrieben, in welcher Ausgangssituation Sie sich aktuell befinden. Die Situationsbeschreibung kann aus nur einem Satz bestehen. Es ist auch möglich, mehr Informationen mitzugeben, um einen eindeutigeren und effektiveren Kontext zu bieten. Je besser Sie darlegen, was die aktuelle Situation ist, desto besser werden die Ergebnisse am Ende.

Als Beispiel soll eine Markteintrittsstrategie entwickelt werden. Ein fiktives Produkt, in diesem Fall ein smarter Toaster, soll zukünftig auch an Kunden in Frankreich verkauft werden. In der Ausgangssituation ist also das Produkt vorhanden, das bereits in einem anderen Land verkauft wird. Hier ist eine Möglichkeit, das als Before-Teil des Prompts auszudrücken:

Beispiel

Wir entwickeln einen smarten Toaster, der bereits in Deutschland und Österreich verkauft wird. Wir sind bisher nur in diesen Ländern vertreten.

Alleinstehend betrachtet, wirkt dieser Teil recht sperrig. Es werden lediglich Fakten dargelegt, es erfolgt noch keine Aussage darüber, wie mit diesen Informationen zu verfahren ist.

3.6.2 Darlegung der Zielsituation

Nachdem erklärt wurde, wie sich die Ausgangssituation darstellt, wird nun im nächsten Teil das gewünschte Zielszenario (*After*) dargestellt. Auch hier gelten die gleichen Regeln wie im vorherigen Abschnitt; je mehr Informationen Sie zum Ziel mitgeben, desto brauchbarer werden die Ergebnisse am Ende. Das Ziel des Beispiels ist es grundsätzlich, im französischen Markt eingetreten zu sein.

Beispiel

Wir möchten mit unserem Produkt in Frankreich präsent sein.

Das Ziel ist in dieser Form allerdings schwammig – so stellt sich zum Beispiel die Frage, ob der Markteintritt als erfolgreich angesehen wird, sobald ein einzelner smarter Toaster verkauft wurde. Unter Umständen wie diesen kann es sinnvoll sein, wiederum präziser darzustellen, was konkret gemeint ist. Werden beispielsweise messbare Ziele definiert, wird die Planentwicklung des LLM entsprechend darauf eingehen.

Beispiel

Wir möchten mit unserem Produkt in Frankreich präsent sein und im ersten Quartal des Markteintritts 10.000 Einheiten verkauft haben.

Wie auch bereits beim Ausgangsszenario ist dieser Prompt-Teil für sich genommen wenig sinnvoll oder aussagekräftig. Zu diesem Zeitpunkt ist dem Modell lediglich klar, wo Sie aktuell stehen und was Sie erreichen wollen, ggf. auch mit zeitlichen Rahmendaten.

3.6.3 Entwicklung des Plans

Mit dem letzten Teil des B-A-B-Frameworks (*Bridge*) wird nun mitgeteilt, wie Sie planen, die Brücke zwischen Ausgangs- und Zielsituation zu schlagen. Sie formulieren hier z. B. die konkrete Aufgabenstellung, die es umzusetzen gilt. Beispielsweise können Sie anweisen, einen Strategieplan zu skizzieren, mit dem Sie das gesetzte Ziel erreichen könnten. Auch können Sie einen konkreten Vorgehensplan entwickeln lassen, der detailliert die notwendigen Schritte erklärt. Weiterhin ist es auch denkbar, das LLM beispielhafte Zwischenergebnisse entwickeln zu lassen, um die Umsetzung konzeptionell simulieren zu lassen.

Die grundsätzliche Idee des B-A-B-Frameworks ist es, das Sprachmodell den Plan zur Zielerreichung entwickeln zu lassen. Sie formulieren also eher in Kategorien und geben üblicherweise keine konkreten Schritte vor. Wenn Sie aber bereits wissen, welche Schritte zumindest teilweise notwendig sind, um zur Zielsituation zu gelangen, können Sie diese

ebenfalls konkret angeben. Achten Sie dann darauf, mitzuteilen, dass es weitere sinnvolle Schritte hinzufügen soll.

Nutzen Sie hier auch die Möglichkeit, eine Rolle anzugeben. Wie beim R-T-F-Framework können Sie mit einer Rolle das entwickelte Vorgehen konkreter auf Ihre spezifische Situation ausrichten lassen.

Das Beispiel dieses Kapitels bezieht sich auf den Markteintritt eines Produktes in Frankreich. Eine einfache Bridge-Anweisung könnte dafür wie folgt aussehen:

Beispiel

Entwickle eine Strategie zum Markteintritt für das Produkt, mit der das Absatzziel erreicht werden kann.

An dieser Stelle können auch noch zusätzliche Informationen mitgegeben werden, die für das Ergebnis wichtig sind. Für eine Markteintrittsstrategie ist beispielsweise die Zielgruppe genauso relevant wie Auftritt und Selbstverständnis der Marke. Während letzteres etwas ist, das dem Modell durchaus mitgegeben werden muss, kann die Zielgruppe auch selbst bestimmt werden. Dazu können Sie entsprechend anweisen, eine Zielgruppenanalyse durchzuführen, eine Zielgruppe anhand von Ihnen gegebenen Kriterien auszuwählen und dann die entwickelte Strategie darauf fokussiert umzusetzen.

Beispiel

Wir entwickeln einen smarten Toaster, der bereits in Deutschland und Österreich verkauft wird. Wir sind bisher nur in diesen Ländern vertreten.

Wir möchten mit unserem Produkt in Frankreich präsent sein und im ersten Quartal des Markteintritts 10.000 Einheiten verkauft haben.

Entwickle eine Strategie zum Markteintritt für das Produkt, mit der das Absatzziel erreicht werden kann. Du erhältst in angehängter Datei

unsere Marketing- und Brand-Strategie. Entwickle eine Zielgruppe für den Markteintritt und führe für diese eine SWOT-Analyse durch. Stelle sie als Tabelle dar.

3.7 Das C-A-R-E-Framework (Context-Actions-Result-Example)

Beim C-A-R-E-Framework (Context, Actions, Result, Example) werden vier Schritte bzw. Komponenten benutzt, um innerhalb eines fest definierten Kontexts eine oder mehrere Ergebnisse generieren zu lassen. Dabei wird das erwartete Ergebnis im Prompt definiert und ein oder mehrere Beispiele dafür gegeben.

3.7.1 Der Kontext des Prompts

Im ersten Schritt (*Context*) wird der Kontext mitgeteilt, in dem die dann folgenden Anweisungen zu verstehen sind. Dieser Kontext kann beispielsweise die Ausgangssituation sein, in der Sie sich gerade befinden. Geben Sie hier so viele Informationen wie notwendig mit, damit das Sprachmodell nachvollziehen kann, an welchem Punkt Sie gerade stehen. Wenn Sie also z. B. ein unternehmensbezogenes Ergebnis erreichen wollen, bietet es sich durchaus an, Informationen über das Unternehmen, Ihre Abteilung oder Ihre Rolle mitzugeben.

> In diesem Framework definieren Sie im Kontext nicht das gewünschte Ergebnis.

Lassen Sie uns das anhand einer Workshop-Planung verdeutlichen. Wir gehen davon aus, dass Sie bereits einen fertigen Workshop haben, den Sie nun umbauen lassen möchten, z. B. zu einem Führungskräfteseminar. Struktur und Vorgehen sollen dabei grundsätzlich erhalten bleiben, inhaltlich sind Ergänzungen oder Anpassungen möglich oder gar notwendig.

Ein möglicher Kontext könnte wie folgt aussehen:

Beispiel

Ich bin Coach für Produktivität für Mitarbeiter und Führungskräfte von mittelständischen Unternehmen. Ich möchte meinen Workshop „Produktiver Arbeiten – Das 1 x 1“, der für Mitarbeiter angeboten wird, auf Führungskräfte anpassen. Zielgruppe sind Personen im C-Level, die lernen werden, effektiver im Tagesgeschäft zu arbeiten.

3.7.2 Die durchzuführenden Aktionen

Der zweite Teil des Prompts (*Actions*) definiert dann, welche Aktionen durchgeführt werden sollen. Ist der Weg zum Ziel bereits bekannt, kann hier genau definiert werden, was in welcher Reihenfolge und in welchem Umfang getan werden soll.

Gibt es wiederum noch offene Punkte im Vorgehen, kann das LLM zusätzlich auch angewiesen werden, selbst sinnvolle Schritte zu unternehmen. Fordern Sie für die Nachvollziehbarkeit ein, dass dabei zuerst ein Plan entwickelt und erklärt wird, bevor die einzelnen Schritte dann durchgeführt werden.

Alternativ können Sie auch die komplette Umsetzungsverantwortung übergeben, indem Sie das Ziel darlegen, die konkreten Schritte aber nicht benennen.

Beispiel

Entwickle einen Ablaufplan mit konkreten Inhalten für ein Führungskräftecoaching. Die Inhalte sollen aus Einzel- und Gruppenübungen sowie Lehreinheiten bestehen.

In unserem Beispiel soll das Coaching auf einem bereits existierenden Training aufbauen. Um dieses Wissen mitzugeben, laden Sie das existierende Workshopkonzept als Datei hoch und erweitern den Prompt einfach so, dass Sie darauf bezugnehmen.

Beispiel

Anbei erhältst du ein bestehendes Workshopkonzept. Plane den neuen Workshop basierend auf diesem Konzept. Passe Übungen und Inhalte an, wo es notwendig oder sinnvoll ist, und begründe deine Anpassungen jeweils kurz.

Damit sparen Sie viel Zeit, denn Sie müssen Ihr vorhandenes Wissen nicht umständlich und langwierig erklären oder die Ergebnisse mehrschrittig anpassen lassen.

Je nach Modell können Sie auch Bilder hochladen und analysieren lassen, beispielsweise Fotografien als Protokolle von Workshops.

3.7.3 Präzisierung des Endergebnisses

Im dritten Teil (*Result*) definieren Sie konkret, wie Sie sich das Ergebnis vorstellen. Dabei beziehen Sie sich auf Format und Länge genauso wie auf inhaltliches. Hier können Sie beispielsweise vorgeben, dass eine Liste einer bestimmten Länge erstellt werden soll. Oder auch, dass Formulierungen in einem gewissen Stil geschrieben werden müssen.

Sie können sich hier auch auf dem LLM bereits bekannte Ergebnisformen beziehen. Das kann zum einen etwas sein, das das Modell von Haus aus bereits kennt, wie beispielsweise den Aufbau eines Marketingplans oder einer Zusammenfassung. Zum anderen können Sie auch das Format als Beispiel vorgeben oder umschreiben. Schließlich können Sie auch hier Beispiele als Dateien beifügen und darauf bezugnehmen.

In unserem Beispiel beziehen Sie sich beim Result-Teil des Prompts einfach auf die ohnehin schon angefügten Dateien mit einer vorhandenen Workshopstruktur. Im vorigen Teil wurde das LLM bereits angewiesen, sich inhaltlich daran zu orientieren, daher greifen Sie das in diesem Schritt nicht erneut auf, sondern beziehen sich lediglich auf das Format des Workshops.

Beispiel

Das Ergebnis soll genauso aufgebaut sein, wie der Ablaufplan in der Datei „Ablaufplan.pdf".

Alternativ können Sie das Ergebnis auch konkreter definieren oder als textuelles Beispiel angeben. Nutzen Sie beispielsweise folgende Vorlage:

Beispiel

Der Ablaufplan soll aus einzelnen Blöcken bestehen, die jeweils wie folgt aufgebaut sind:

###

Startzeit – Endzeit: Titel

Teilnehmeranzahl

Beschreibung

###

Da Sie nicht konkret definieren, was die einzelnen Bestandteile genau bedeuten, wird das Sprachmodell Annahmen treffen und beispielsweise *Startzeit* oder *Titel* durch konkrete Angaben ersetzen, die Beschreibung aber eventuell mit „Beschreibung:" beginnen.

3.7.4 Definition eines Beispiels

Zusätzlich zu diesen ersten drei Schritten wird in diesem Framework noch mindestens ein konkretes Beispiel (Example) mitgegeben, an dem sich das LLM orientieren kann. Auch hier gilt wieder: Beziehen Sie sich entweder auf etwas, das dem Modell bereits bekannt ist oder geben Sie konkrete Beispiele im Prompt mit.

Wenn Sie eine Marketingkampagne für Social Media entwickeln lassen wollen, können Sie sich auf andere Unternehmen beziehen, die erfolgreiche Kampagnen durchgeführt haben. Wenn Sie ein Angebot for-

mulieren lassen wollen, können Sie ähnlich zum vorigen Teil einen exemplarischen Aufbau im Prompt mitgeben.

Im aktuellen Beispiel können Sie sich konkret auf das bereits existierende Konzept beziehen. Es kann dabei notwendig sein, explizit darauf hinzuweisen, dass das Ergebnis (inhaltlich) angepasst werden soll.

Beispiel

Ein Beispiel für einen gut geplanten und strukturierten Workshop findest du in der Datei „Workshopstruktur.pdf".

3.8 Das R-I-S-E-Framework (Role-Input-Steps-Expectations)

Wollen Sie für eine spezifische Situation, für die Sie bereits Daten gesammelt haben, einen Plan zum Erreichen eines fest definierten Ziels entwickeln, kann das R-I-S-E-Framework (Role, Input, Steps, Expectations) der richtige Ansatz sein. Sie beschreiben dazu die Rolle, aus deren Sicht der Plan entwickelt werden soll. Sie erläutern die vorhandenen Daten über Ihre Situation und fragen nach den notwendigen Schritten, um ein Ziel zu erreichen. Dieses Ziel definieren Sie schließlich im letzten Teil des Prompts.

3.8.1 Die Rolle des Prompts

Als erstes definieren Sie für den Prompt, in welcher Rolle agiert werden soll. Grundsätzlich reicht es, die Rolle einfach zu nennen, vor allem, wenn es sich um eine gängige Rolle handelt, wie beispielsweise eine anerkannte Berufsbezeichnung. Lassen Sie uns das Beispiel verfolgen, einen Engagement-Plan für Stakeholder eines Projekts entwickeln zu wollen. Das LLM soll daher die Rolle eines Projektmanagers einnehmen.

Beispiel

Verhalte dich wie ein erfahrener Projektmanager.

Sind mehr Informationen relevant, werden diese direkt mit der Rolle definiert. Sie können die Rolle selbst mit Details verfeinern und beispielsweise spezifische Erwartungen oder Ansichten erläutern. Auch der Kontext, in dem fiktiv agiert werden soll, kann das Verhalten und damit die Ergebnisse des Prompts deutlich beeinflussen.

Beispiel

Du bist ein erfahrener Projektmanager eines Projekts, das nach dem Wasserfall-Modell durchgeführt wird. Es ist ein umfangreiches und kritisches Projekt und du hast die Gesamtverantwortung.

3.8.2 Erläuterung der Daten und Informationen für den Prompt

Im zweiten Teil (Input) erklären Sie, welche Daten Sie dem Prompt zur Verfügung stellen. Sie können dazu beispielsweise Beispiele in Textform im Prompt referenzieren. Häufig ist es praktischer, die Daten in Dateiform dem Prompt mitzugeben, zum Beispiel in Form von PDFs oder Bildern. Diese können dann in diesem Teil des Prompts kurz referenziert werden. Damit halten Sie nicht nur den Prompt kürzer, sondern nutzen auch wieder die Fähigkeiten des jeweiligen Sprachmodells, angehängte Dateien zu verstehen. Erläutern Sie die Anhänge soweit notwendig.

Beispiel

Angehängt findest du eine Übersicht aller Stakeholder des Projekts mit ihren Erwartungen und Erfahrungen. Weiterhin erhältst du eine Risikomatrix des Projekts, die auch die Stakeholder beinhaltet.

3.8.3 Nach Schritten zur Lösung fragen

Nachdem mit den vorherigen zwei Bestandteilen die Situation umfassend beschrieben wurde, wird das LLM im dritten Teil aufgefordert, konkrete Schritte (Steps) zur Erreichung des Ziels zu entwickeln. Das Ziel wurde bisher nicht weiter definiert und kann daher hier prägnant geschildert werden.

Beispiel

Entwickle einen Schritt-für-Schritt-Plan für ein Stakeholdermanagement des Projekts.

Sowohl der mehrschrittige Plan als auch, wie in diesem Beispiel, ein Stakeholdermanagement sind umfangreiche Themenfelder. Daher empfiehlt es sich, weiter auf Ihre Anforderungen einzugehen. Sie können dazu beschreiben, welche Schritte Sie im Ergebnis erwarten oder welche aus Ihrer Sicht zwingend notwendig sind. Je nach Komplexität kann es zudem auch sinnvoll sein, Schritte in Teilschritte zerlegen oder weitere Erläuterungen hinzufügen zu lassen. Nutzen Sie hier die umfangreichen Möglichkeiten, die in Abschn. 2.4 erklärt werden.

Das Ziel wurde in dem Beispiel-Prompt nur kurz umrissen. Auch hier kann es förderlich sein, weitere Informationen und Rahmenbedingungen anzugeben. Gehen Sie hier nur auf Ihre Erwartungen an das Ergebnis des Prompts, also auf den zu entwickelnden Plan, ein.

Beispiel

Entwickle einen Schritt-für-Schritt-Plan für ein Stakeholdermanagement des Projekts. Gib für jeden Schritt das Ziel und das benötigte Vorgehen an. Erläutere jeweils, warum der Schritt sinnvoll ist und welche Vorteile er bringt.

3.8.4 Die Zielerwartungen definieren

Mit dem letzten Teil des Prompts werden die Erwartungen (Expectations) an das Ergebnis definiert. Die Erwartungen beziehen sich dabei auf die Situation, die durch das Umsetzen der Schritte des Plans erreicht wird. Sie beschreiben also die ideale Zielsituation. Teilen Sie dem Modell mit, welche Erwartungen Sie an diese haben. Je konkreter Sie definieren, was Sie mit dem Plan erreichen möchten, desto besser wird dieser am Ende darauf abgestimmt sein. Auch hier können Sie mit konkreten Beispielen arbeiten, indem Sie beispielsweise Dateien anhängen oder ent-

sprechenden Text einbetten. Weiterhin können Sie auch auf allgemein bekannte Beispiele bezugnehmen, die sich im Wissensfundus des Sprachmodells befinden. Schließlich haben Sie auch die Möglichkeit, hier konkrete Zielwerte zu definieren, die erreicht werden sollen.

Für das Beispiel wird eine verbesserte Zufriedenheit der Stakeholder als Ziel definiert. Die vorhandenen Kennzahlen werden kurz referenziert und dann wird der konkrete Zielwert festgelegt.

Beispiel

Ziel ist es, die Zufriedenheit der Stakeholder zu optimieren. Diese liegt aktuell bei 67 % und wird durch wöchentliche Fragebögen ermittelt. Ein Beispiel für einen solchen Fragebogen findest du in der Datei „Stakeholder-Quest.pdf". Innerhalb der nächsten 90 Tage soll die Zufriedenheit auf 85 % erhöht werden.

3.9 Das Experten-Prompt-Framework (EPF)

Im Folgenden wird das Experten-Prompt-Format vorgestellt. Es kombiniert viele der bisher erläuterten Tipps zu einem einfach anwendbaren Format, welches sich nach ein paar Anwendungen einfach merken lässt.

3.9.1 Übersicht

Das Experten-Prompt-Framework (EPF) ist ein Framework, das die Aufgabe für das Sprachmodell in mehreren Schritten sehr präzise und strukturiert beschreibt. Wie auch verschiedene andere Frameworks nutzt es Rollen, um den grundsätzlichen Kontext der Ergebnisse zu umreißen. In mehreren Schritten werden dann das erwartete Ergebnis und auch die dafür umzusetzenden Tätigkeiten ausgeführt. Anders als andere Frameworks benennt es auch explizit Ausnahmen und nicht gewünschte Ergebnisse. Schließlich definiert es das Ausgabeformat, die Zielgruppe und verdeutlicht die Erwartungen des Autors mit einem oder mehreren Beispielen.

Beispiel

Verhalte dich wie ein [Expertenrolle].

Ich benötige [Beschreibung des Ergebnisses].

Du wirst dafür [genaue Aufgabenbeschreibung].

Dabei wirst du [weitere Details].

Bitte [Ausnahmen auflisten].

Entwickle das Ergebnis für [Zielgruppe].

Liefere das Endergebnis als [Format].

Hier ist ein Beispiel: [Beispiel].

Das Format ist recht umfangreich und das Schreiben des Prompts mit diesem Framework benötigt durchaus einige Zeit. Dafür werden die Ergebnisse auch deutlich besser zu den Erwartungen und Anforderungen passen. Manuelle Nachbearbeitungen oder weitere Iterationen mit dem Sprachmodell sind damit häufig nicht notwendig.

Ein so umfangreiches Prompt-Framework hat aber auch noch einen weiteren wichtigen Effekt: als Autor beschäftigt man sich intensiver mit der Fragestellung und den eigenen Erwartungen. Nicht selten kommt es vor, dass sich beim Schreiben neue Perspektiven ergeben und sich die Fragestellung weiterentwickelt. Beim Metaprompting werden diese Einzelheiten direkt durch das LLM abgefragt (vgl. Kap. 5).

Die Formulierungen der einzelnen Bausteine werden in der Regel so verwendet wie aufgeführt, können aber sprachlich natürlich auch variiert werden. Am Ende ist es wichtig, dass das LLM die einzelnen Bestandteile verstehen und richtig umsetzen kann.

3.9.2 Die Expertenrolle

Wie auch bei verschiedenen anderen Frameworks ist der erste Schritt die möglichst konkrete Definition einer Rolle, die das Modell einnehmen soll. Die Rolle beeinflusst, wie das Sprachmodell die Relevanz von Infor-

mationen einschätzt, Tonalität wählt und auch in welchem Format die Ergebnisse grundsätzlich entwickelt werden.

Sofern die gewählte Rolle nicht konkret genug ist oder noch zusätzlicher Kontext notwendig sein sollte, können zusätzliche Informationen mitgegeben werden. Je genauer die Rolle definiert wird, desto besser werden die Ergebnisse am Ende sein.

Beispiel

Verhalte dich wie ein Experte für Onlinemarketing.

Oder konkreter:

Beispiel

Du bist ein Experte für SEO-Optimierung mit Fokus auf Webseite-Performance und User Experience (UX). Deine Erfahrung konzentriert sich auf CMS-basierte Webseiten für mittelständische Unternehmen.

3.9.3 Beschreibung des Ergebnisses

Als nächstes wird das gewünschte Ergebnis beschrieben. Das fällt erfahrungsgemäß recht kurz aus, da die Schritte, um zu diesem Ergebnis zu kommen, erst im nächsten Baustein des Frameworks beschrieben werden. Hier geht es also darum, das Zielbild möglichst präzise zu formulieren. Das kann bedeuten, dass nur ein kurzer Satz schon ausreicht, um das gewünschte Ergebnis auf den Punkt zu bringen. Auf der anderen Seite kann es durchaus richtig und wichtig sein, weitere Details mitzugeben. Dadurch kann das Ergebnis schon vorab präzisiert und ein sinnvoller Rahmen gegeben werden.

Beispiel

Ich benötige einen Artikel über Rasenpflege.

Oder konkreter:

Beispiel

Ich benötige den Text für einen kurzen Zeitungsartikel über das Thema professionelle Rasenpflege.

Wichtig dabei ist, nicht die Aufgabe folgender Teile des Frameworks zu übernehmen. Die Ergebnisformulierung sollte nicht den Weg zum Ergebnis selbst beinhalten oder nur aus diesem bestehen. Häufig fällt es erfahrungsgemäß allerdings recht leicht, das gewünschte Ergebnis präzise zu kommunizieren.

Die Herausforderung liegt nicht selten darin, Anforderungen an Format und Struktur auszuklammern und diese erst zu einem späteren Zeitpunkt einzubauen. Die Grenze dabei ist fließend und am Ende zählt das Ergebnis mehr als ein gut und korrekt formulierter Prompt. Dennoch hilft es, Format und Struktur separat zu spezifizieren.

3.9.4 Darstellung der Aufgabe

Im dritten Teil wird die konkrete Tätigkeit näher ausgeführt und damit beschrieben, welche Schritte genau unternommen werden sollen, um das eingangs formulierte Ergebnis zu erreichen. Je nach Komplexität der gesamten Fragestellung kann auch dieser Teil aus nur wenigen Stichworten oder einer umfangreichen Ausführung von einzelnen Schritten bestehen.

Dabei wird nicht jede mögliche Variation erläutert und auch nicht alle Details des Weges werden umfassend dargelegt. Vielmehr wird an dieser Stelle lediglich grundsätzlich erörtert, welches Vorgehen erwartet wird. Zusätzliche Informationen zur Ausarbeitung werden dann im nächsten Schritt des Frameworks ausgeführt.

Beispiel

Dafür wirst du einen Artikel aus deiner Sicht als Experte über das Thema schreiben.

Diese Trennung der Darstellung des Vorgehens in diesen und den noch folgenden Schritt ist vergleichbar mit dem Konzept von OKR (Ob-

jectives and Key Results). Es wird also das „große Ganze“ des erwarteten Vorgehens des LLMs erklärt. Weitere Schlüsselvorgehen werden im nächsten Schritt erklärt.

3.9.5 Weitere Details zur Umsetzung

Im Folgenden wird dann das Vorgehen zur Erreichung des Ergebnisses des gesamten Prompts weiter präzisiert. Dazu werden weitere Details hinzugefügt, die sowohl die Erarbeitung als auch das Ergebnis näher konkretisieren können. Das kann beispielsweise bedeuten, einen speziellen Weg oder eine Reihenfolge von Schritten vorzugeben. Auch ist hier der richtige Punkt, um spezielle Frameworks beim Vorgehen vorzuschreiben oder Ausnahmen und Alternativen zu definieren.

Beispiel

Dabei wirst du das AIDA-Framework zum Storytelling benutzen und für jeden Bestandteil des Frameworks mindestens zwei Absätze schreiben. Entwickle dazu eine ansprechende Storyline.

Sofern während der Bearbeitung interaktiv mit dem Modell interagiert werden soll, kann auch dies an dieser Stelle angeführt werden.

Beispiel

Beschreibe zuerst die Storyline und frage mich, welche Anpassungen ich benötige. Lasse diese in die Storyline mit einfließen. Stelle weitere Fragen, sofern Punkte offen sind oder du noch weitere sinnvolle Ideen hast. Schreibe erst danach den Artikel.

3.9.6 Ausnahmen

Genauso sinnvoll, wie das konkrete Beschreiben von den umzusetzenden Erwartungen, kann es ebenfalls sein, solche Vorgehen oder Ergebnisse zu benennen, die das Modell unterlassen soll. Damit lassen sich häufige

Fehlerquellen ausschließen und auch typisches, ungewünschtes Verhalten von Sprachmodellen im Vorhinein unterbinden.

Als Formulierung wählt das EPF das Wort „Bitte“ mit anschließender Auflistung der Ausnahmen in beliebiger Form, beispielsweise als einzelne Sätze oder als Aufzählung. Auch eine andere Phrase wäre möglich, wie beispielsweise „Beachte folgende Ausnahmen:…:“ oder „Achte dabei darauf, dass, …“.

Die Ausnahmen können sich auf jeden Teil des Prompts beziehen. Beispielsweise kann hier die Expertenrolle eingegrenzt werden, sofern dies notwendig ist. Auch das Gesamtergebnis des Prompts kann mit Ausnahmen versehen werden, genauso wie das in den beiden vorigen Teilen beschriebene Vorgehen noch weiter eingeschränkt werden kann. Vielmehr können sich Ausnahmen auch auf die noch kommenden Prompt-Teile der Zielgruppe und des Formats beziehen.

Beispiel

Bitte verwende keine Fachbegriffe und keine Umgangssprache. Bitte nutze keine Emojis.

3.9.7 Zielgruppe des Prompts

Neben der Expertenrolle ist die Zielgruppe eine wichtige Stellschraube, mit der das Ergebnis beeinflusst werden kann. Hier gelten die gleichen Rahmenbedingungen wie für die Expertenrolle im Prompt: Je konkreter die Zielgruppe bestimmt wird, desto besser passt das Ergebnis am Ende. Während also in manchen Szenarien der Name der Zielgruppe ausreichend ist, kann es in anderen notwendig sein, mehr Kontext zu geben.

Die konkrete Benennung der Zielgruppe kann einen großen Einfluss auf den generierten Text haben. So wird das LLM in Abhängigkeit der Zielgruppe spezielle Formulierungen und Fachbegriffe nutzen oder Fachwörter, die in der Zielgruppe nicht vorausgesetzt werden können, automatisch erklären. Weiterhin kann auch die Sprache einer ganzen Zielgruppe imitiert werden. Denken Sie zum Beispiel an die aktuelle Jugendsprache. Es können auch regionale Zielgruppen definiert werden. Dabei kann das LLM auch lokale Sprachnuancen oder Dialekte aufgreifen.

Weiterhin können, falls angemessen, auch direkt passende Emojis oder Hashtags – im Social Media gängiges Format zur Markierung spezieller Themen oder Personen, markiert mit einem vorangestellten „#“ – mit in den Text eingebunden werden.

Falls die Zielgruppe sehr konkret beschrieben werden soll, kann es zielführend sein, dafür Personae einzusetzen. Damit werden idealtypische Personen aus der Zielgruppe in Form eines Profils beschrieben. Da Personae recht umfangreich ausfallen können, kann es an dieser Stelle sinnvoll sein, diese als Dateianhänge beizufügen und zu referenzieren, statt zu versuchen, alle Informationen im Prompt selbst zu platzieren.

Beispiel

Der Artikel soll geschrieben werden für Eigenheimbesitzer mittleren Alters mit einem Garten. Du erhältst dazu eine Persona eines idealtypischen Lesers als PDF-Datei.

3.9.8 Das Format des Ergebnisses

Als nächstes wird definiert, in welchem Format das Ergebnis präsentiert werden soll. Je nach Modell können hier unter Umständen auch direkt Dateien wie beispielsweise Microsoft-Word-Dokumente erzeugt werden. Unabhängig davon kann beispielsweise gefordert werden, dass eine Liste oder ein entsprechend formatierter Text produziert werden soll. Es gelten grundsätzlich die Formatmöglichkeiten aus Abschn. 2.4.

Beispiel

Schreibe den Artikel mit 2000 Wörtern und formatiere in sinnvolle Absätze. Formatiere in jedem Absatz die wichtigste Phrase fett. Schreibe zusätzlich eine kurze Einleitung und ein kurzes Fazit.

3.9.9 Bereitstellung eines Beispiels

Zum Schluss werden in dem Prompt noch ein oder mehrere Beispiele bereitgestellt. Dieser Teil kann sehr nützlich und effektiv sein, wenn die

Anforderungen an das Ergebnis recht umfangreich oder kompliziert sind und eine umfangreiche Zielformatbeschreibung notwendig wäre. Wenn Ziel und Ergebnis mit den anderen Bestandteilen des Experten-Prompt-Frameworks bereits effektiv genug beschrieben wurden, ist es regelmäßig einfacher, das Beispiel erst einmal wegzulassen.

Beispiel

Anbei erhältst du als Orientierung den Artikel des letzten Jahres über ein anderes Thema.

3.10 Auf einen Blick

- Prompt Frameworks liefern klar nachvollziehbare strukturelle Vorgaben zum Aufbau von guten Prompts.
- Für unterschiedliche Ziele gibt es entsprechende Frameworks.
- Prompt Frameworks können mit weiteren Prompting-Techniken kombiniert und dadurch verfeinert werden.
- Es hat sich herausgestellt, dass es bei vielen Prompts von Vorteil ist, dem LLM eine Rolle vorzugeben, in der es handeln soll. Hierdurch wird das System auf eine spezielle Aufgabe fokussiert und liefert dadurch konkretere Antworten.
- Wenn das Modell weiß, wer die spätere Zielgruppe des Textes ist, kann der generierte Text sehr viel besser und vor allem automatisch an die jeweiligen Erwartungen und Anforderungen angepasst werden.
- Bei komplexeren Anfragen hat es sich als erfolgreich erwiesen, einen deduktiven Ansatz zu wählen und gemeinsam mit dem Modell schrittweise zum Ergebnis zu gelangen.
- Das Experten-Prompt-Framework liefert eine einfach anzuwendende Blaupause für die
- meisten Anfragen.

4

Fortgeschrittene Techniken für Prompts

Zusammenfassung In diesem Kapitel werden fortgeschrittene Techniken für das Erstellen von erfolgreichen Prompts vorgestellt. Dabei werden zunächst die sogenannten Zero-, One-, und Few-Shot-Prompts beschrieben, die anhand einer unterschiedlichen Anzahl von Beispielen dem LLM den Weg zur Lösung des eigentlichen Problems aufzeigen. Danach wird das Cognitive-Verifier-Muster erläutert. Dabei erfragt das LLM eigenständig wichtige Informationen, die zur Lösung der Aufgabe notwendig sind. Es folgen noch die Chain-of-Thought-Prompts, bei denen die Lösung gemeinsam mit dem LLM in einzelnen Schritten erarbeitet werden und die Multi-Rollen-Analyse, bei der das LLM in unterschiedlichen Rollen agiert und sich somit der Lösung des Problems von unterschiedlichen Richtungen aus nähert.

Durch Frameworks können Prompts so strukturiert werden, dass bereits deutlich bessere Ergebnisse mit LLMs erzielt werden. Darüber hinaus gibt es fortgeschrittene Techniken, die die Qualität noch einmal merklich zum Positiven verändern können. Dazu werden Probleme beispielsweise in Teilprobleme zerlegt oder mit unterschiedlichen Rollen bzw. Kontex-

D. Koch et al., *Prompt Engineering im Unternehmen – eine Einführung*,
https://doi.org/10.1007/978-3-658-47699-1_4

ten beantwortet. Das jedoch komplett durch das Sprachmodell, ohne eigenes Zutun. Zudem können diese Techniken auch mit Prompt-Frameworks wie denen in Kap. 3 kombiniert werden.

4.1 Zero-Shot-, One-Shot- und Few-Shot-Prompts

Um einem LLM die Problemlösung zu vereinfachen, können Beispiele im Prompt mit eingebaut werden. Dadurch wird verdeutlicht, wie mit konkreten Situationen umzugehen bzw. was die Erwartungen oder das Verständnis von Ihnen jeweils sind. Je nach Anzahl dieser Beispiele wird zwischen Zero-Shot, One-Shot- und Few-Shot-Prompts unterschieden.

4.1.1 Zero-Shot-Prompts

Für einfache Fragestellungen reicht es aus, dem LLM eine einzelne, konkrete Anfrage zu stellen, um ein gutes Ergebnis zu erhalten. Es ist dazu nicht notwendig, Beispiele im Prompt mitzugeben. Diese Einzelanfragen werden Zero-Shot-Prompts genannt. Hier sind zwei Beispiele für Zero-Shot-Prompts.

Beispiel 1:

Beispiel

Wer war der Erfinder des Penicillins?

Beispiel 2:

Beispiel

Bestimme den Sprachstil der Anfrage.

Anfrage: Warum antworten Sie mir nicht?

Während das erste Beispiel auf Basis von Fakten einfach zu beantworten ist, ist beim zweiten unter Umständen mehr Kontext notwendig. Der

Sprachstil der Anfrage wird vom Modell auf Basis seines Wissens bestimmt, ohne Ihren Kontext zu kennen. Um diesen Kontext mitzugeben, könnte eine umfangreiche Rollenbeschreibung eingebaut und beispielsweise das R-T-F-Framework (siehe Abschn. 3.4) verwendet werden. Alternativ kann der Prompt auch mit entsprechenden Beispielen angereichert werden, was ihn zu einem One-Shot-Prompt bzw. Few-Shot-Prompt macht.

4.1.2 One-Shot-Prompts

Wenn einfache, direkte Prompts nicht das gewünschte Ergebnis bringen, kann ein konkretes Beispiel in den Prompt eingebaut werden, das die Herangehensweise bzw. Ihre Erwartungen verdeutlicht. Das geschieht üblicherweise mit Abschnittsmarkern. Zusätzlich kann vor oder nach dem Beispiel die Aufgabe konkretisiert werden.

Hier ist das Beispiel zur Bestimmung des Sprachstils als One-Shot-Prompt.

Beispiel

Bestimme den Sprachstil der Anfrage. Hier ist ein Beispiel:

Anfrage: Können Sie mir helfen?

Sprachstil: Freundlich

Anfrage: Warum antworten Sie mir nicht?

Für die Aufgabenstellung wird hier zuerst mit Abschnitten ein Beispiel konstruiert und das Muster dann fortgeführt, die Antwort aber ausgelassen. Das LLM wird dem Format des Beispiels folgend eine Antwort generieren. Diese wird regelmäßig besser sein als mit einem Zero-Shot-Prompt, da mit dem Beispiel auch implizit das erwartete Ausgabeformat mitgeteilt wird. Wichtig ist dabei das Format: der Prompt simuliert quasi eine Anfrage und eine Antwort, wie sie das Sprachmodell selbst generieren könnte. Es wird nicht weiter erläutert, weshalb das Ergebnis so ist, sondern das Modell wird mit dem Beispiel aus dem Prompt trainiert und wendet die Erkenntnisse dann auf die Aufgabenstellung an.

4.1.3 Few-Shot-Prompts

Für komplexere Problemstellungen kann es sinnvoll sein, mehrere Beispiele einzubauen. Dadurch wird die Herangehensweise an die Lösung deutlich und das LLM kann konkretere Ergebnisse entwickeln. Die Beispiele simulieren Anfragen und Ergebnisse und werden benutzt, um das Modell in diesem Kontext zu trainieren. Es wird dann daraus Erkenntnisse ableiten und diese auf die Aufgabenstellung anwenden. Wenn ein Prompt mehrere solcher Beispiele beinhaltet, handelt es sich um einen Few-Shot-Prompt.

Hier ist ein Beispiel für einen Few-Shot-Prompt:

Beispiel

Bestimme den Sprachstil der Anfrage. Hier sind ein paar Beispiele:

Anfrage: Können Sie mir helfen?

Sprachstil: Freundlich

Anfrage: Helfen Sie mir!

Sprachstil: Fordernd

Anfrage: Ich finde es frech, dass ich keine Antwort erhalten habe.

Sprachstil: Unfreundlich

Anfrage: Warum antworten Sie mir nicht?

Die Anfragen und ihre Ergebnisse werden nicht konkreter erläutert. Stattdessen wird das Sprachmodell eben dazu gebracht, eigene Schlüsse aus den Beispielen zu ziehen und diese auf die Problemstellung anzuwenden. Dabei bezieht es neben dem Kontext der Beispiele auch das Format mit ein.

4.2 Das Cognitive-Verifier-Muster

Um für komplexe Fragestellungen gute Antworten zu erhalten, kann es notwendig sein, zahlreiche detaillierte Informationen im Prompt mitzugeben. Während dies mit verschiedenen Prompt Frameworks bereits vereinfacht wird, gibt es dabei allerdings eine grundsätzliche Herausforderung: Sie müssen wissen, welche Informationen in welchem Umfang relevant sind. Auch diese Aufgabe kann auf das LLM übertragen werden. Mit dem Cognitive-Verifier-Muster wird es aufgefordert, relevante Informationen zur Bearbeitung des Prompts abzufragen.

Der Aufbau ist dabei denkbar einfach. Am Anfang des Prompts wird die Aufforderung eingebaut, die folgende Aufgabe in drei Fragestellungen aufzuteilen, diese beantworten zu lassen und dann die Ergebnisse zur Bearbeitung des Prompts zu benutzen.

Beispiel

Generiere drei Fragen für die folgende Aufgabenstellung, die zur effektiven Beantwortung sinnvoll und notwendig sind. Lasse dir diese Fragen nacheinander von mir beantworten und nutze meine Antworten, um die Aufgabe umzusetzen.

Danach folgt dann der eigentliche Prompt. Das Modell wird entsprechend der Anweisungen nun drei Fragen stellen, diese beantworten lassen und dann das Ergebnis für den Prompt entwickeln. Die Anzahl der Fragen ist dabei grundsätzlich variabel. Mehr als drei sind meist langwierig zu beantworten und weniger können durchaus wichtige Aspekte auslassen. Ist bereits abzusehen, dass drei Fragen nicht passen, kann also durchaus eine andere Anzahl gewählt werden.

Auch die Bestimmung der Anzahl der Fragen können Sie natürlich dem Sprachmodell überlassen. Bauen Sie dazu statt der konkreten Zahl den Prompt so um, dass eine sinnvolle Anzahl an Fragen gestellt werden soll. Es kann allerdings sinnvoll sein, einen Bereich (Minimum bis Maximum) anzugeben, um eine gewisse Qualität zu erreichen, aber gleichzeitig nicht über das Ziel hinaus zu schießen.

4.3 Chain-of-thought-Prompts

Viele Prompts verlangen es, mehrere Schritte zu durchlaufen, um das gewünschte Ergebnis zu generieren. Wenn dieses Ergebnis dann nicht komplett den gestellten Anforderungen entspricht oder gar signifikant von den Erwartungen abweicht, kann es sein, dass sich das LLM in der Bearbeitung irgendwo verrannt hat. Statt nun intensiv am Prompt zu arbeiten und umfangreiche Informationen beizufügen, hilft es häufig bereits, die einzelnen Schritte bei der Erarbeitung darstellen und ggf. erläutern zu lassen. Dadurch wird die Kette an Gedanken und Annahmen (*Chain of thought*) dargestellt.

Fügen Sie dazu einfach am Ende des Prompts eine einzelne Phrase, mit der das LLM aufgefordert wird, sein Vorgehen zu begründen und die Schritte darzustellen. Bauen Sie, nachdem Sie im Prompt die Aufgabenstellung erläutert haben, folgenden Zusatz mit ein:

Beispiel

Gehe Schritt für Schritt vor.

Das sorgt nun dafür, dass jeder Teilschritt und seine jeweiligen Ergebnisse einzeln ausgegeben werden. Wie es manchen Menschen leichter fällt, auf Lösungen zu kommen, wenn sie laut denken, führt es auch mit dieser Prompt-Erweiterung häufig dazu, dass sich die Ergebnisse verändern, und zwar meistens zum Besseren. Dazu haben Sie aber nun die Möglichkeit, den Ansatz des Sprachmodells nachzuvollziehen. Damit können Sie im Nachgang beispielsweise einzelne Teilschritte anpassen lassen und damit wiederum die Qualität verbessern.

Manchmal kommt es vor, dass diese Phrase als Formatangabe verwendet und dann das Ergebnis in Form von Einzelschritten ausgegeben wird – aber nicht das Vorgehen verändert. Formulieren Sie in diesem Fall den Prompt so um, dass klar wird, dass Zwischenschritte genannt werden sollen.

Diese Art von Chain-of-thought-(COT)-Prompts werden als *Zero Shot COT-Prompt* bezeichnet. Sie bringen Sprachmodelle dazu, den

Lösungsweg darzustellen und zu erläutern. Sie geben im Prompt aber keine Beispiele mit. Die Prinzipien von Few-Shot-Prompts (siehe Abschn. 4.1.3) können auch mit Chain-of-thought-Prompts angewendet werden. Dazu werden im Prompt konkrete Beispiele eingebaut, die jeweils aus einer Fragestellung und einer Antwort bestehen. Anders als bei klassischen Few-Shot-Prompts wird neben der Lösung auch der Lösungsweg beschrieben. Dadurch kann das LLM Ihren Ansatz zur Lösung nachvollziehen und dann auf die letzte Fragestellung entsprechend anwenden. Die Eingangsphrase, die im Zero-Shot-COT-Prompt die Lösungsdarstellung angestoßen hat, entfällt beim Einbringen von Beispielen erst einmal. Sollte die generierte Lösung nicht Ihren Vorstellungen entsprechen, können Sie mit diesem Satz dann noch zusätzlich versuchen, mehr Details über Vorgehen zu erhalten und damit das Ergebnis zu verbessern.

Mit dem folgenden Beispiel wird das Prinzip verdeutlicht. Dem Modell wird die Aufgabe gestellt, die Produktionskapazität einer Maschine zu berechnen. Dafür wird von manchen Modellen wie beispielsweise GPT-4o von OpenAI häufig ein Python-Skript (eine gängige Programmiersprache) generiert, welches die eigentliche Berechnung durchführt und ausgeführt. Von diesem Zwischenschritt bekommen Sie aber meist nichts mit, da dies im Hintergrund und voll automatisch durchgeführt wird. Die Formulierung des Prompts führt dazu, dass das LLM auf anderem Wege rechnet. Aufgrund der geringen Komplexität der Aufgabenstellung kommt es zudem auch zu richtigen Ergebnissen. Dem Muster folgend wird das Modell das Ergebnis korrekt berechnen und entsprechend runden. Die Ergebnisse sind je nach Version allerdings unterschiedlich. Probieren Sie den Prompt z.B. einmal mit der Version GPT-3.5, GPT-4 und einmal mit GPT-4o von ChatGPT oder mit einem beliebigen anderen LLM aus, um die Unterschiede zu sehen.

Beispiel

Frage: Eine Maschine produziert 22 Teile pro Stunde. Es werden zwei zusätzliche Maschinen mit der Hälfte der Produktionskapazität angeschafft. Eine der neuen Maschinen läuft auf 150 % Kapazität. Wie viele Teile werden pro Stunde produziert?

Antwort: 22 Teile pro Stunde + 11 Teile pro Stunde + 11 * 1,5 Teile pro Stunde = 49,5 Teile. Da keine halben Teile hergestellt werden können, wird abgerundet. 49 Teile werden pro Stunde produziert.

Frage: Eine Maschine produziert 10 Teile pro Stunde. Es werden zwei zusätzliche Maschinen mit der doppelten Produktionskapazität angeschafft. Eine der neuen Maschinen läuft auf 9 % Kapazität. Wie viele Teile werden pro Stunde produziert?

Antwort: 10 Teile pro Stunde + 20 Teile pro Stunde + 20 * 0,09 Teile pro Stunde = 31,8 Teile. Da keine anteiligen Teile hergestellt werden können, wird abgerundet. 31 Teile werden pro Stunde produziert.

Frage: Eine Maschine produziert 7 Teile pro Stunde. Es werden zwei Maschinen mit 75 % der Produktionskapazität angeschafft. Wie viele Teile werden pro Stunde produziert?

4.4 Multi-Rollen-Analyse

Im betrieblichen Kontext betreffen viele Aufgabenstellungen zwangsläufig mehrere Stakeholder. Jeder dieser Stakeholder bringt eigene Erfahrungen, Erwartungen und Wünsche – also einen eigenen Kontext – mit und identifiziert bei Situationen unter Umständen andere Chancen und Herausforderungen.

Dieser Umstand kann auch mit einem LLM genutzt werden. Indem das Modell aufgefordert wird, jeweils verschiedene Rollen einzunehmen und aus diesen Rollen heraus eine Aufgabenstellung zu bearbeiten, können Erkenntnisse gewonnen werden, die sonst leicht übersehen werden könnten.

Hier ist ein Beispiel für die Erarbeitung von Anforderungen an eine Richtlinie zum Arbeiten mit Künstlicher Intelligenz in Unternehmen. Das Modell soll selbstständig sinnvolle Stakeholder identifizieren, deren Rolle einnehmen und am Ende die Ergebnisse konsolidieren.

Beispiel

Es sollen Anforderungen an eine Richtlinie zum Einsatz von Künstlicher Intelligenz entwickelt werden. Die Aufgabenstellung soll aus drei Sichtweisen erarbeitet werden.

Finde und benenne die drei wichtigsten Stakeholder in einem Unternehmen, die von einer Richtlinie betroffen wären.

Erarbeite Anforderungen an die Richtlinie aus der Sicht jeder der drei Stakeholder. Gib die Anforderungen als einfache Liste mit 5 Anforderungen aus.

Zeige dann auf, wo Synergien und Konfliktpotenzial zwischen den drei Stakeholdern besteht.

Je nach Aufgabenstellung kann das Arbeiten mit mehreren Rollen den Aufwand deutlich vergrößern. Es eignet sich vor allem für solche Aufgaben, bei denen kurz und prägnant Ergebnisse produziert werden können, beispielsweise bei der Einschätzung von Situationen oder auch der Entwicklung von Konzepten. Teilweise kann dieser Ansatz mit mehreren Rollen auch gut mit anderen Prompt-Frameworks kombiniert werden.

4.5 Auf einen Blick

- Mit Hilfe von Zero-Shot, One-Shot- und Few-Shot-Prompts können dem LLM anhand von Beispielen mögliche Lösungen für die Aufgabe aufgezeigt werden. Das System lernt daraus, wie zu antworten ist und sucht nach einer entsprechenden gleichartigen Lösung.
- Mit dem Cognitive-Verifier-Muster können Sie das System dazu bringen von sich aus ohne weitere Anleitungen lösungsrelevante Fragen zu stellen, die die Qualität des Ergebnisses steigern.
- Bei den sogenannten Chain-of-Thought-Prompts arbeitet das LLM Schritt für Schritt von der Problemstellung über mehrere Zwischenschritte bis zum finalen Ergebnis. Dabei werden die einzelnen „Denkschritte“ des Systems offengelegt und können hinterfragt, angepasst oder erweitert werden.
- Mit Hilfe der Multi-Rollen-Analyse kann das LLM unterschiedliche Sichtweisen einnehmen. Dies liefert unterschiedliche Lösungsansätze und kann sehr nützlich sein, da ein gegebenes Problem von unterschiedlichen Seiten beleuchtet wird. Dies fördert oft neue Ansichten zu Tage, die sonst möglicherweise nicht beachtet worden wären.

5

Metaprompting

Zusammenfassung In diesem Kapitel wird in das Konzept des Metapromptings eingeführt. Dabei wird das LLM nicht direkt zur Beantwortung einer gegebenen Frage eingesetzt, sondern das System wird genutzt, um im ersten Schritt einen geeigneten Prompt für die Erstellung der späteren Lösung zu generieren. Dabei können Prompts für das eigene LLM oder für Drittsysteme erstellt werden. So kann z. B. ChatGPT auch Prompts zur Erstellung von Bildern mit Hilfe von Midjourney generieren. Zusätzlich können mit Hilfe des Metapromptings auch existierende Prompts weiter optimiert werden. Abschließend wird mit dem Valisory-Prompt-Studio ein komplexer und vollautomatisch ablaufender Prompt vorgestellt, der alle bisher beschriebenen Konzepte zu einem ganzheitlichen Ablauf zusammenführt.

Häufig werden generative Sprachmodelle eingesetzt, um Text zu produzieren, der dann von Menschen oder anderen Systemen und Anwendungen weiterverwendet wird. Die in Kap. 3 und 4 dargestellten Techniken werden genutzt, um Anfragen dafür effektiver zu gestalten. Eine wertvolle Fähigkeit moderner Sprachmodelle ist es allerdings, selbst

D. Koch et al., *Prompt Engineering im Unternehmen – eine Einführung*,
https://doi.org/10.1007/978-3-658-47699-1_5

Prompts entwickeln zu können. Vielmehr noch können sie auch effektive Prompts für andere generative KI-Systeme schreiben. Damit kann beispielsweise mittels ChatGPT interaktiv ein guter Prompt für die Bild-KI Midjourney entwickelt werden, mit dem dann Bilder erstellt werden. Die Techniken des Metaprompting werden in diesem Kapitel erläutert.

5.1 Grundlagen und Prinzipien des Metapromptings

Wie bereits im Abschn. 2.1 erörtert, ist Kommunikation ein inhärent komplexes Unterfangen. Um die Komplexität in der Arbeit mit Sprachmodellen zu reduzieren, wurden verschiedene Prompt Frameworks vorgestellt. Eine andere, und damit auch kombinierbare Möglichkeit, ist das **Metaprompting**. Dabei wird ein Sprachmodell genutzt, um einen Prompt für anderes Modell oder sich selbst zu schreiben. Der Name kommt daher, dass bei diesem Ansatz des Promptings auf der Metaebene gearbeitet und gar nicht der eigentlich Prompt selbst geschrieben, sondern einen Prompt für einen Prompt entwickelt wird. Metaprompting ist dabei nicht selten ein iteratives Vorgehen, bei dem Schritt für Schritt interaktiv ein optimaler Prompt entwickelt wird.

Auf der Metaebene deshalb, weil im klassischen Ansatz des Prompt Engineerings daran gearbeitet wird, die Anfrage an das LLM so zu konstruieren, dass das Ergebnis möglichst gut wird. Beim Metaprompting liegt der Prompt-Engineering-Aufwand nun darin, einen Prompt so effektiv zu gestalten, dass das Sprachmodell damit einen Prompt für ein anderes Modell entwickeln kann, der dann in diesem Modell genutzt wird, um dort das eigentliche Ergebnis zu generieren. Prinzipiell könnte dieses Prinzip auch wiederum für Metaprompting selbst genutzt werden, indem mit einem Sprachmodell ein Prompt geschrieben wird, der genutzt wird, um einen Prompt zu schreiben, der einen Prompt generiert, der das eigentliche Ergebnis produziert.

Warum sollte man sich aber die Mühe machen, mit einem generativen KI-System „über Bande“ zu arbeiten? Auf der einen Seite kann es bei hinreichend komplexen Aufgabenstellungen sehr sinnvoll sein, Zeit in die Entwicklung eines Prompts zu investieren, um am Endergebnis möglichst wenig nachschärfen zu müssen. Statt dies nun allein zu tun, macht

man sich die Fähigkeiten eines LLMs zu Nutze und entwickelt den Prompt damit. Auf der anderen Seite gibt es durchaus Unterschiede beim Prompting für verschiedene KI-Modelle – mindestens dann, wenn man Systeme zur Erstellung von Text, Videos oder Bilder vergleicht. Mit Hilfe des Metapromptings können für diese Situation effektive Prompts entwickelt werden, die notwendige Informationen beinhalten und typische Fehlerquellen vermieden werden.

5.2 Ansätze des Metaprompting

In den folgenden Abschnitten werden zunächst unterschiedliche Ansätze für das Metaprompting vorgestellt. Abschließend wird mit dem „Promptstudio“ eine Möglichkeit vorgestellt, die vollautomatisch optimale Prompts für beliebige Aufgaben erstellt.

5.2.1 Selbstreferenzielle Prompts

Der grundsätzliche Ansatz des Metapromptings ist die Selbstreferenzierung. Dabei wird das Sprachmodell aufgefordert, einen Prompt für sich selbst zu analysieren, zu reflektieren und Verbesserungspotenzial zu finden – wie in einer Art Rückkopplungsschleife.

Dazu wird dem Modell im ersten Schritt ein Prompt präsentiert, den es analysieren und bewerten soll. Der Prompt wird dabei mit Anführungszeichen oder als Texteinbettung ausgezeichnet. Um den Prompt im Folgenden effektiv weiterentwickeln zu können, hilft es, beispielsweise jeweils eine nummerierte Liste von Vorschlägen zu produzieren. Damit kann man sich dann einfacher auf konkrete Ideen beziehen, diese ggf. weiterentwickeln und in den Prompt einbauen lassen.

Beispiel

Du erhältst im Folgenden einen Prompt. Analysiere und bewerte, ob dieser Prompt präzise genug ist, um optimale Ergebnisse zu erzielen. Mache konkrete Vorschläge zur Verbesserung. Nutze jeweils eine nummerierte Liste.

Der Prompt:

###

Erkläre die Kernfunktion eines neuronalen Netzes in einfacher Sprache.

###

Das LLM wird dann das Ziel des Prompts analysieren und direkt Möglichkeiten zur Verbesserung aufzeigen. Unter Umständen produziert es auch direkt eine bessere Version. Mit den Ergebnissen kann nun iterativ in einem Dialog mit dem Sprachmodell weitergearbeitet werden, bis das gewünschte Endergebnis an Prompt erreicht wurde.

Sie können den obigen Prompt noch etwas anpassen, um das iterative und dialogische Arbeiten mit dem Sprachmodell am eigentlichen Prompt zu fördern. Bauen Sie dazu die Aufforderung ein, Ihnen Fragen zu stellen und die Antworten in den Prompt mit einzubauen. Das Modell wird dabei regelmäßig auch solche Fragen stellen, die man ursprünglich nicht in Betracht gezogen hat.

Beispiel

Entwickle jeweils bis zu drei Fragen. Stelle mir diese in Form einer nummerierten Liste und lasse meine Antworten in den Prompt mit einfließen.

Eine weitere kleine Optimierung ist das permanente Darstellen der aktuellen Version des Prompts. Das hat zwei Vorteile: Zum einen wissen Sie dabei stets, wie das Ergebnis aktuell aussieht und können dadurch effektiver Feedback geben. Zum anderen kann es natürlich auch hier passieren, dass man an die Grenzen des Context Window kommt. Daher ist es durchaus praktisch, stets eine aktuelle Version mitzunehmen. Damit das Vorgehen möglichst strukturiert bleibt, hilft es hier, ein Format zu definieren. Zum Beispiel als erstes den aktuellen Prompt ausgeben zu lassen, danach mögliche Vorschläge für Verbesserungen und dann schließlich die Liste von Fragen folgen zu lassen. Damit hätte jeder Schritt stets das gleiche Format und der inhärenten Variabilität eines LLM bei der Beantwortung wäre adäquat begegnet.

Um am Ende den eigentlichen Prompt nicht mühselig aus der Gesamtantwort kopieren zu müssen, fordern Sie das Sprachmodell einfach auf, Ihnen einmal nur den Prompt zu geben.

Teilweise kommt es vor, dass das Sprachmodell den Prompt an einem bestimmten Punkt einfach eigenmächtig ausführt – auch wenn man explizit verlangt, dies nicht zu tun. Bisher konnte dazu keine abschließend funktionierende Lösung gefunden werden.

5.2.2 Iterative Optimierung durch dialogische Entwicklung

Ähnlich zum Vorgehen der selbstreferenziellen Optimierung (vgl. Abschn. 5.2.1) können Prompts auch von Grund auf (weiter-)entwickelt werden. Dazu wird dem Modell das Ziel genannt, optionalerweise ein Prompt präsentiert und dann iterativ und dialogisch an Verbesserungen gearbeitet. Wird ein Prompt für ein anderes KI-Modell entwickelt, ist es wichtig, dies zu benennen und, sofern relevant, auch Versionen oder Spezifika aufzuzeigen.

Die Tipps aus Abschn. 5.2.1 hinsichtlich Format und Interaktion gelten auch hier. Vor allem bei vielen (erwarteten) Iterationen kann es helfen, ein Format der Ausgabe zu definieren. Weiterhin hat es einen großen Mehrwert, das LLM eigenständig Fragen bzw. Anmerkungen identifizieren zu lassen.

Beispiel

Du bist ein Prompt-Entwicklungsassistent. Unser Ziel ist, einen Prompt für Googles Gemini zu erstellen, der eine präzise Checkliste für ein Softwarearchitektur-Review generiert. Der Prompt soll in Englisch sein.

Wir werden den Prompt Schritt für Schritt erarbeiten. In jedem Schritt gibst du zuerst die aktuelle Version des Prompts aus. Dann folgt eine

Stichpunktliste an möglichen Verbesserungen. Daraufhin folgt eine nummerierte Liste an Fragen, die für die Entwicklung des Prompts relevant sind, mit maximal drei Fragen. Lasse meine Antworten in die jeweils nächste Version des Prompts mit einfließen.

Zu keinem Zeitpunkt wirst du den eigentlichen Prompt selbst ausführen oder beantworten.

In diesem Beispiel wurde als Zielmodell Gemini von Google definiert. Auch wurde die Sprache des Zielprompts auf Englisch spezifiziert. Hier wurde außerdem mit Role Prompting (vgl. Abschn. 3.2) gearbeitet. Im zweiten Teil folgte die Definition des Vorgehens, bei der vor allem das Format der Interaktionen definiert wurde. Zum Abschluss wurde dem Modell noch mitgeteilt, den Prompt selbst nicht auszuführen (vgl. Abschn. 5.2.1).

5.2.3 Bewertung und Entwicklung durch rollenbasierte Abstraktion

Diese Art des Metapromptings ist eine Abwandlung der iterativen Optimierung (vgl. Abschn. 5.2.2). Das Modell wird hier aufgefordert, eine Rolle einzunehmen, die den Entwicklungsprozess abstrahiert (z. B. Moderator, Kritiker oder Berater). Mit dieser Rolle wird der Prompt dann weiterentwickelt.

Beispiel

Du bist ein kritischer Prompt-Reviewer. Analysiere den folgenden Prompt: „Generiere eine Liste von Marketingideen für ein Produktlaunch-Event."

1. Bewerte, ob der Prompt spezifisch genug ist, um präzise Ergebnisse zu liefern.
2. Mache Vorschläge, wie der Prompt verbessert werden könnte.

Die Idee ist, dass durch eine solche abstrahierende Rolle eher Kriterien wie Eindeutigkeit, Korrektheit und Spezifität bewertet und opti-

miert werden. Der Fokus liegt weniger auf der kreativen (Weiter-)Entwicklung eines Prompts.

> Für umfangreiche Prompt-Entwicklungen kann es sehr hilfreich sein, einen solchen kritischen Reviewer in den Entwicklungsprozess mit einzubauen, im einfachsten Fall indem ganz am Ende das Prompt-Ergebnis von solch einem Role Prompt bewertet wird. Weiterführend ist auch die Kombination von mehreren Sprachmodellen denkbar, in der beispielsweise in OpenAIs ChatGPT ein Prompt für Googles Gemini entwickelt und dann aber noch in Anthropics Claude kritisch qualitätsgesichert wird.

5.3 Das Valisory-Prompt-Studio

Mit den Techniken des Metapromptings haben wir einen umfangreichen Metaprompt konzipiert, der es erlaubt, Schritt für Schritt gut Prompts zu entwickeln: Das **Valisory-Prompt-Studio**. Dazu haben wir die in diesem Kapitel dargestellten Techniken angewendet und im Sinne des Prompt Engineerings durch Testen schrittweise optimiert.

Das Valisory-Prompt-Studio kann genutzt werden, um komplett neue Prompts zu entwickeln oder auch zum Verbessern bestehender Prompts. Es nutzt dazu einen iterativen Prozess, bei dem im Dialog mit dem Sprachmodell Verbesserungen identifiziert und im Prompt eingebaut werden. Das Prompt Studio kann gut als vordefinierter Assistent oder Chat (bei ChatGPT „CustomGPT“) umgesetzt und für Mitarbeiter bereitgestellt werden.

Der Prompt für das Valisory-Prompt-Studio ist wie folgt aufgebaut: als erstes werden Grundregeln und Annahmen für die gesamte Interaktion definiert. Dazu gehörten das Ziel und eine Rolle genauso wie Sprache und Tonalität. Dann folgt die Darstellung des allgemeinen Vorgehens unabhängig des jeweiligen Szenarios. Danach kommt die Beschreibung der Schritte für den Fall, dass ein Prompt verbessert werden soll. Schließlich wird am Ende als größter Teil definiert, wie die Entwicklung eines Prompts durchgeführt werden soll.

5.3.1 Die Grundregeln des Valisory-Prompt-Studios

Der erste Abschnitt des Prompts definiert das grundsätzliche Vorgehen und die Rahmenbedingungen. Zunächst wird dem Valisory-Prompt-Studio die Rolle eines „erfahren[en] Prompt Engineers, der sich auf das Entwickeln effektiver Prompts für Sprachmodelle spezialisiert hat“ (Text aus Prompt) zugewiesen. Dieses Role Prompting (vgl. Abschn. 3.2) verbessert Vorgehen, Annahmen und Qualität in der Erarbeitung. Auch wird am Anfang dem Modell mitgeteilt, welchen Namen es hat, damit es dem Nutzer sinnvoll mitteilen kann, was es tut.

Als zweites werden das Ziel und die zwei möglichen Szenarien erläutert. Das Ziel des Valisory-Prompt-Studios ist es, einen Nutzer bei der Erstellung eines „bestmöglichen“ Prompts zu unterstützen. Dazu kann entweder ein komplett neuer Prompt entwickelt oder ein bereits vorhandener verbessert werden.

Beispiel

Du bist das Valisory-Prompt-Studio, ein erfahrener Prompt Engineer, der sich auf das Entwickeln effektiver Prompts für Sprachmodelle spezialisiert hat. Du unterstützt den Nutzer dabei, den bestmöglichen Prompt für seine Bedürfnisse zu entwickeln. Der Nutzer kann entweder einen komplett neuen Prompt entwickeln oder einen existierenden verbessern.

Im Folgenden wird das Vorgehen beschrieben, von dem du nicht abweichst.

Grundsätzlich ist das Vorgehen iterativ: durch Fragen und Rückmeldungen wird der jeweilige Prompt Schritt für Schritt weiterentwickelt.

Du führst den entwickelten Prompt zu keinem Zeitpunkt selbst aus oder beantwortest diesen.

Du verhältst Dich sprachlich formell und höflich. Die Sprache der Interaktion mit dem Nutzer ist die jeweilige Landessprache (falls nicht anders definiert). Die Sprache des zu erstellenden Prompts ist ebenfalls

> die Landessprache, sofern der Nutzer dies nicht explizit anders fordert oder einen Prompt zur Verbesserung angibt, der in einer anderen Sprache geschrieben ist.
>
> Die folgende Beschreibung ist in Abschnitte unterteilt (A, B und C). Starte bei Abschnitt A.

Dann folgt die Beschreibung des Vorgehens. Dazu wird eingangs explizit darauf hingewiesen, dass von diesem Vorgehen nicht abgewichen werden soll. Diese Anforderung ist natürlich recht schwammig, denn das dann folgend beschriebene Vorgehen ist an vielen Stellen nicht vollumfänglich und das LLM wird eine eigene Interpretation mit einbringen. Dieser Hinweis hilft dennoch dabei, dass das Modell am Plan und am Format des Prompts im Großen und Ganzen festhält.

Dem Sprachmodell wird dann das grundsätzliche Vorgehen umrissen: Es ist iterativ, es werden Fragen gestellt und Antworten gegeben, mit denen dann der jeweilige Prompt weiterentwickelt wird. Formulierungen wie „iterativ“ und „Schritt für Schritt“ signalisieren dem Modell dabei, dass nicht erwartet wird, direkt nach einem Schritt ein Ergebnis zu haben. Damit sollen seitens des Modells bessere Ergebnisse im Sinne des Prompt Chainings (vgl. Abschn. 3.1) erreicht werden.

Hier wird das Modell dann ebenfalls dazu aufgefordert, den entwickelten Prompt selbst nicht zu beantworten bzw. auszuführen. Ohne diesen Hinweis kann es nach verschiedenen Iterationen dazu kommen, dass sich das Vorgehen der Verbesserung des Prompts mit dem Beantworten desselben mischt. Dadurch wird der Chat mit dem LLM dann unübersichtlich und die eigentliche Verbesserung kann nicht mehr sinnvoll fortgeführt werden. In unseren Tests hat das Einbauen dieser Phrase eben jene Situation verhindert, allerdings ist davon auszugehen, dass es dennoch hin und wieder passieren kann.

Als nächstes folgen Hinweise bzgl. Sprache und Tonalität. Das Modell soll formell und höflich sein. Mit dem Nutzer soll (zunächst) in der jeweiligen Landessprache interagiert werden. Auch der zu entwickelnde Prompt soll in der jeweiligen Sprache verfasst sein, außer, der Nutzer fordert es anders oder er gibt dem Prompt an, eine andere Sprache zu nutzen, um sich zu verbessern.

Zum Abschluss wird die Struktur des eigentlichen Vorgehens definiert. Dieses ist in drei Abschnitte eingeteilt: Abschnitt A definiert Voraussetzungen; Abschnitt B stellt das Vorgehen dar, wenn ein Prompt verbessert werden soll; und Abschnitt C behandelt den Fall, dass ein neuer Prompt entwickelt werden soll.

5.3.2 Abschnitt A: Die Voraussetzungen

Der erste Block der Vorgehensbeschreibung definiert die Voraussetzungen, die das Modell schaffen muss, damit im Folgenden dann ein neuer Prompt entwickelt oder ein bestehender Prompt verbessert werden kann. Der Abschnitt ist mit der Abschnittsmarke „A" versehen.

Als erstes muss das Sprachmodell definiert werden, für das der Prompt entwickelt oder verbessert werden soll. Die Formulierung „als allererstes" ist hier wichtig – denn sonst fragt das Valisory-Prompt-Studio entweder später oder gar nicht danach. Als zweites wird dann sichergestellt, dass das Ziel der Interaktion klar ist; entweder die Neuentwicklung oder die Verbesserung. Für beide Situationen sind Ausnahmen eingebaut, dass jeweils nicht nachgefragt werden muss, sofern der Nutzer die Information bereits gegeben hat.

Bei diesen beiden Abfragen handelt es sich um Hilfestellungen für den Nutzer, die die Benutzerfreundlichkeit verbessern. Da die Interaktion mit dem Sprachmodell interaktiv ist, können solche Informationen jederzeit entlang des Weges gegeben werden. Das Valisory-Prompt-Studio soll hier explizit nachfragen, um den Nutzer zu unterstützen und sicherzustellen, dass nichts vergessen wird.

Beispiel

A: Dein allgemeines Vorgehen

- Sofern der Nutzer es nicht angegeben hat, frage als allererstes nach dem Sprachmodell, für den der Prompt entwickelt werden soll
- Sofern der Nutzer es nicht angegeben hat, frage als zweites, ob ein neuer Prompt entwickelt (weiter mit B: werden oder ein bestehender Prompt verbessert werden soll (weiter mit C). Fahre dann mit dem jeweiligen Abschnitt unten fort.

5.3.3 Abschnitt B: Verbesserung eines Prompts

Im zweiten Block der Vorgehensbeschreibung wird der Fall abgehandelt, dass ein vorhandener Prompt verbessert werden soll. Im Endergebnis ist das Vorgehen recht umfangreich: Der zu verbessernde Prompt muss ggf. noch vom Nutzer angegeben werden. Dann müssen eine Analyse durchgeführt und Optimierungen vorgenommen werden.

Im Detail ist das Vorgehen allerdings grundsätzlich sehr ähnlich zu der anderen möglichen Situation, in der ein neuer Prompt entwickelt werden soll. Lediglich die Ausgangssituation ist unterschiedlich, denn bei der einen gibt es bereits einen Prompt, bei der anderen nicht. Nimmt man nun an, dass die Optimierung eines vorhandenen Prompts des Nutzers äquivalent zur Optimierung eines Prompts ist, der in einer ersten Version durch das Sprachmodell entworfen wurde, muss das Fortfahren für beide Situationen nur einmal definiert werden. Die Einstiegspunkte sind also unterschiedlich definiert und ein Teil des Prompts wird wiederverwendet.

Beispiel

B: Für den Fall, dass ein Prompt verbessert werden soll:

1. Frage nach dem Prompt, der verbessert werden soll, sofern der Nutzer ihn noch nicht angegeben hat.
2. Analysiere den Prompt und identifiziere das Ziel. Frage den Nutzer, ob das von dir bestimmte Ziel korrekt ist oder ob der Nutzer ein anderes Ziel definieren möchte.
3. Fahre mit Schritt 3 von Abschnitt C fort.

Für den Fall, dass der Nutzer am Anfang der Interaktion mit dem Valisory-Prompt-Studio zwar angezeigt hat, dass ein Prompt verbessert werden soll, diesen aber noch nicht bereitgestellt hat, definiert der Prompt explizit, dass nach dem Prompt gefragt wird.

Bevor die Verbesserung beginnt, soll das Valisory-Prompt-Studio als nächstes das Ziel aus dem gegebenen Prompt ableiten und durch den Nutzer überprüfen lassen. Der Nutzer erhält zudem die Möglichkeit, das Ziel anzupassen. Damit wird sichergestellt, dass es eine Kongruenz zwi-

schen Prompt und Ziel gibt bzw. das Fehlen dieser Kongruenz durch das Valisory-Prompt-Studio korrigiert werden kann.

Danach soll das Valisory-Prompt-Studio in Abschnitt C springen und dort mit Schritt 3 fortfahren.

5.3.4 Abschnitt C: Entwicklung eines neuen Prompts

Dieser letzte Block beschreibt schließlich die verbleibenden zwei Bausteine: Zum einen das Entwickeln eines komplett neuen Prompts und zum zweiten das iterative Vorgehen zur Optimierung des Prompts.

Dazu fragt das Modell im ersten Schritt nach dem Ziel, das mit dem Prompt erreicht werden soll. Hier wird auch explizit die Möglichkeit gegeben, Rückfragen zu stellen, sofern die Antworten „unklar oder nicht präzise" (Text aus Prompt) sind – was das genau bedeutet, wird der Einschätzung des Sprachmodells überlassen.

Im zweiten Schritt wird dann auf Basis des definierten Ziels eine erste Version eines Prompts entwickelt. Dabei bezieht sich das LLM auf das eingangs spezifizierte Ziel-Sprachmodell, die ggf. anders geforderte Sprache und etwaige weitere Anpassungen. Um die Ausgabe des Modells kurz und prägnant zu halten, wird es aufgefordert, diese erste Prompt-Version nicht auszugeben, sondern direkt mit Schritt 3 fortzufahren.

Beispiel

C: Für den Fall, dass ein Prompt entwickelt werden soll:

1. Frage nach dem Ziel des Prompts. Stelle Rückfragen, sofern die Antworten des Nutzers unklar oder nicht präzise sind.
2. Entwickle eine erste Version des Prompts. Gib diese nicht aus, sondern fahre direkt mit Schritt 3 fort.
3. Analysiere den aktuellen Prompt aus zwei Blickwinkeln:
   a. Als kritischer Prompt-Reviewer
   b. Als kreativer Prompt-Entwickler

4. Gib jeweils immer Folgendes aus:
   - Das Ziel des Prompts
   - Den aktuellen Prompt
   - Vorschläge zur Verbesserung aus Sicht des kritischen Prompt-Reviewers
   - Relevante Fragen (maximal 3) zur Verbesserung des Prompts aus Sicht des Prompt-Entwicklers
5. Der Nutzer wird die Fragen beantworten. Stelle Rückfragen, falls Antworten unklar, unpräzise und nicht sinnvoll sind.
6. Entwickle den Prompt mit den Antworten auf die Fragen und ggf. weiteren Anmerkungen des Nutzers weiter.
7. Springe dann wieder zu Schritt 3 in diesem Abschnitt. Führe diesen iterativen Prozess so lange fort, bis der Nutzer zufrieden ist.

Schritt 3 ist auch derjenige Schritt, zu dem das Valisory-Prompt-Studio springt, nachdem ein bereits vorhandener Prompt für eine Optimierung angegeben wurde. Ab diesem Schritt bringt also die eigene Optimierungsarbeit am Prompt. Ob dieser dann originär vom Nutzer kommt oder durch das Modell entwickelt wurde, ist irrelevant.

In Schritt 3 findet eine Analyse des Prompts statt. Dazu werden zwei Rollen benutzt. Die erste ist ein „kritischer Prompt-Reviewer“ (vgl. Abschn. 5.3.2). Die Idee ist, dass mit einem möglichst objektiven Fokus Chancen und Herausforderungen identifiziert werden. Es wurde auch bewusst ein Reviewer gewählt, also eine Rolle, die ihre Stärken in der Bewertung und nicht in der (Weiter-)Entwicklung hat. Im Gegensatz dazu steht die zweite Rolle, der „kreative Prompt-Entwickler“, der mit einem kreativ-orientierten Ansatz Möglichkeiten zur Weiterentwicklung finden soll.

Als nächstes soll das Valisory-Prompt-Studio in Schritt 4 den aktuellen Status ausgeben. Das Format ist dabei immer das gleiche. Als erstes wird das festgelegte Ziel definiert, danach folgt die aktuelle Version des Prompts. Damit kann der Nutzer leicht überprüfen, ob sich der Prompt (noch oder schon) grundsätzlich eignet, um das Ziel zu erreichen und

ggf., ob dieser schon ausreichend gut sein könnte. Außerdem soll sich der Nutzer möglichst wenig merken müssen.

Daraufhin werden Anmerkungen zum Prompt ausgegeben. Diese stammen aus der kritischen Betrachtung aus Schritt 3. Es folgen bis zu drei Fragen zum Prompt, mit deren Beantwortung er verbessert werden kann. Die Fragen wiederum wurden von der zweiten Rolle in Schritt 3 entwickelt. Innerhalb von Schritt 4 wird sich jeweils auf die Rollen aus Schritt 3 bezogen.

Im fünften Schritt soll der Nutzer dann die Fragen beantworten und das Modell soll die Antworten mit in die weitere Entwicklung einbeziehen. Tatsächlich würde sich das LLM meistens auch so verhalten, wenn es nicht explizit angegeben wird. Indem es bewusst vorgegeben wird, kann das Verhalten aber besser gesteuert werden. In der Ausführung wird an dieser Stelle die Kontrolle an den Nutzer übergeben und dieser kann die Fragen beantworten.

Danach soll das Sprachmodell in Schritt 6 den Prompt mit den Antworten weiterentwickeln. Dass die Antworten des Nutzers mit in die Entwicklung einfließen sollen, wurde zwar bereits im Prompt erwähnt und wäre tatsächlich auch das natürliche Verhalten des Valisory-Prompt-Studios, es aber explizit einzubauen, verbessert durchaus die Qualität der Umsetzung.

> Tatsächlich hat sich gezeigt, dass das mehrfache Nennen oder Beschreiben einer Aufgabe dazu führt, dass sich das Modell mehr daran hält. Diese Technik wird z. B. bei der Absicherung von KI-Agenten eingesetzt, um unerwünschtes Verhalten zu unterbinden.

Der letzte Schritt des Prompts definiert dann das iterative Vorgehen. Das Modell soll zu Schritt 3 zurückspringen und von dort aus weitermachen. Dieses Vorgehen soll so lange fortgeführt werden, bis der Nutzer zufrieden ist. Diese Bedingung ist wichtig, damit das Modell nicht selbstständig das Ende festlegt und das Vorgehen unterbricht. Wann der Nutzer zufrieden ist, wird allerdings nicht definiert, weshalb das Sprachmodell an der einen oder anderen Stelle subtil nachfragen oder auf die

Zufriedenheit hinweisen wird. Tatsächlich ist das aber auch nicht von Relevanz, da man einen Chat mit einem LLM in der Regel nicht für beendet erklärt, sondern die Ergebnisse bei Fertigstellung ohne Rückmeldung anderweitig weiterverwendet.

5.3.5 Bereitstellung des Valisory-Prompt-Studios für Nutzer

Das Valisory-Prompt-Studio kann auf verschiedenen Wegen eingesetzt werden. Der scheinbar einfachste Fall ist das Einfügen des Prompts in einen neuen Chat. Danach kann dann mit dem Valisory-Prompt-Studio wie dargestellt interagiert werden. Dabei ergeben sich zwei Nachteile: Zum einen ist der Prompt recht umfangreich und damit häufig wenig praktikabel. Zum anderen ist davon auszugehen, dass der Prompt über die Zeit weiterentwickelt wird und dann neu verteilt werden müsste – vor allem bei mehreren potenziellen Nutzern innerhalb des Unternehmens eine durchaus anspruchsvolle Logistik. Es bietet sicher daher an, das Valisory-Prompt-Studio als KI-Agenten bereitzustellen, z.B, bei OpenAIs ChatGPT als CustomGPT.

Die zentrale Bereitstellung als Agent nimmt Nutzern die Notwendigkeit ab, den Prompt irgendwo digital vorzuhalten und immer in neue Chats zu kopieren. Auf der anderen Seite ergibt sich für das Unternehmen der Vorteil, dass der Prompt zentral abgelegt ist und dort weiterentwickelt werden kann – völlig transparent für Nutzer.

Zum Abschluss dieses Kapitels sei noch darauf hingewiesen, dass das Valisory-Prompt-Studio nicht bei jeder Anfrage an ein LLM eingesetzt werden sollte. Das wäre oft das sprichwörtliche „Mit Kanonen auf Spatzen schießen“. Vielmehr geht es darum, das Valisory-Prompt-Studio einzusetzen, wenn umfangreiche oder sehr spezielle Prompts erstellt oder optimiert werden sollen. Für viele Anfragen reichen die weiter oben beschriebenen Frameworks vollkommen aus. Nichtsdestotrotz ist das Valisory-Prompt-Studio ein sehr mächtiges Werkzeug und zeigt, dass mit Hilfe von Prompts auch wirklich komplexe Aufgaben sehr strukturiert und nachvollziehbar gelöst werden können.

5.4 Auf einen Blick

- Mit Hilfe des Metaprompting-Ansatzes können optimierte Prompts für das eigene LLM oder für Drittsysteme erstellt werden.
- Das Metaprompting ist dabei ein iteratives Vorgehen, das schrittweise einen neuen Prompt erzeugt oder einen existierenden Prompt optimiert.
- Beim Metaprompting hinterfragt das System selbst die Qualität des Prompts in Hinblick auf die zu lösende Aufgabe und bietet eigenständig Ideen zur Verbesserung an oder verfeinert den Prompt, indem weitere Verständnisfragen an den Anwender gestellt werden.
- Das Valisory-Prompt-Studio verbindet alle gezeigten Mechanismen des Metapromptings zu einem vollständig automatisch ablaufenden Prompt, der zum einen existierende Prompts optimieren und zum anderen komplett neue Prompts erstellen kann.
- Metaprompting ist ein mächtiges Werkzeug. Es sollte aber nur für umfangreiche Prompts genutzt werden, da es für kurze Anfragen an das System überdimensioniert ist.

6

Reasoning-Modelle

Zusammenfassung Nach dem großen Erfolg vom LLMs setzen nun auch die sogenannten Reasoning-Modelle ihren Siegeszug an. Dies sind speziell optimierte LLMs, die sich im Rahmen eines gegebenen Kontextes tief in eine Fragestellung „eindenken" bevor sie eine entsprechende Antwort geben. In diesem Kapitel werden die Reasoning-Modelle mit ihrer Funktionsweise zunächst allgemein beschrieben und danach wird auf die Besonderheiten im Prompting eingegangen und ein Prompt Framework für Reasoning-Modelle vorgestellt.

Der Einsatz von LLMs zur Erstellung von Texten gehört in vielen Bereichen bereits zum Alltag. Seit der Einführung von ChatGPT o1 und spätestens seit der Veröffentlichung von DeepSeek-R1 gewinnen die sogenannten Reasoning-Modelle immer mehr an Bedeutung. Sie sind in der Lage, sich tief in die Fragestellungen des Anwenders „einzudenken" und zunächst einen sehr strukturierten Lösungsweg zu erarbeiten, bevor sie die eigentliche Antwort ausgeben. Im Folgenden werden diese Reasoning-Modelle zunächst genau vorgestellt. Danach wird auf die Besonder-

D. Koch et al., *Prompt Engineering im Unternehmen – eine Einführung*,
https://doi.org/10.1007/978-3-658-47699-1_6

heiten im Prompting für Reasoning-Modelle eingegangen und abschließend ein spezielles Prompt Framework für Reasoning-Modelle beschrieben.

6.1 Der Hintergrund von Reasoning-Modellen

Im Zuge der Forschung und Weiterentwicklung von Large Language Models wurden die sogenannten Reasoning-Modelle als Spezialisierung entwickelt. Ihre Spezialisierung liegt darin begründet, dass sie besser komplizierte Aufgabenstellungen und logische Probleme lösen können. Diese Modelle sind dazu in der Regel sehr umfangreich, manche Modelle basieren bspw. auf mehr als 100 Mrd. Parametern.

Wie der Name für diese Kategorien von Modellen bereits andeutet, liegt ihr Ziel darin, einen Denkprozess zur Problemlösung zu nutzen. Statt „nur" sinnvolle Wörter und Sätze auf Basis statistischer Wahrscheinlichkeiten zu produzieren, sollen sie Problemstellungen verstehen und durch logische Denkprozesse strukturiert lösen. Dafür wurden sie nicht grundlegend neu konzipiert, sondern sie unterscheiden sich von klassischen Sprachmodellen vor allem dadurch, dass sie verschiedene Prompting-Techniken automatisch benutzen. Je nach Modell kann es auch im Trainingsprozess große Unterschiede geben. Die grundlegende Technik für Reasoning-Modelle ist Chain of Thought (Abschn. 4.3), die im Speziellen um verschiedene fortgeschrittene Ansätze erweitert wird.

Reasoning-Modelle entwickeln für adäquate Problemstellungen tatsächlich teilweise erstaunlich gute Ergebnisse. Ansätze wie induktives Out-Of-Context-Reasoning (Treutlein, 2024) zeigen, dass LLMs mittelbare Zusammenhänge aus Trainingsdaten ableiten und logisch korrekt verbalisieren können. Inwieweit dahinter ein tatsächliches Verständnis des Kontexts steht, ist nicht abschließend geklärt.

Wenn diese Reasoning-Modelle so gut darin sind, Probleme ähnlich einer menschlichen Herangehensweise logisch zu lösen, weshalb haben sie dann nicht die klassischen Sprachmodelle ersetzt?

Wie die meisten der aktuellen Sprachmodelle benötigen auch Reasoning-Modelle eine signifikante Menge an Ressourcen, was sich sowohl für Anbieter als auch Benutzer bemerkbar macht. Die Effektivität dieser Modelle liegt eben auch in ihrer Größe begründet. Und mit der Größe steigen die Kosten sowohl für das Training als auch für den Betrieb. Wie auch an anderen Stellen im Technologiekomplex von Sprachmodellen ist der Zusammenhang hier ein exponentieller. Mit steigenden Fähigkeiten steigen also auch die Bedarfe an Ressourcen wie Energie oder Speicherplatz extrem. Dazu kommt, dass die Modelle aufgrund ihrer Größe vergleichsweise langsam sind. Das Beantworten eines Prompts durch ein Reasoning-Modell kann je nach Aufgabenstellung leicht mehrere Minuten in Anspruch nehmen.

Die mit den Reasoning-Modellen verbundenen höheren Betriebskosten zeigen sich auch dadurch, dass fortgeschrittene Modelle dieser Kategorie aktuell bei verschiedenen Anbietern (wie bspw. OpenAI) nur in den preisintensiveren Nutzungspaketen inkludiert sind und teilweise auch noch signifikante Zusatzkosten in der Verwendung (bspw. durch hohe Kosten pro Token) produzieren.

Ein signifikanter Faktor dabei ist die Herangehensweise der Aufgabenaufteilung und -synthese. Denn der zusätzliche Denkprozess benötigt ebenfalls entsprechende Ressourcen. Man kann es sich in etwa so vorstellen, als würde das Reasoning-Modell für eine Aufgabe eine Analyse durchführen, selbstständig mehrere Prompts schreiben und diese dann jeweils beantworten.

In diesem Vorgehen ergibt sich ein weiterer Einflussfaktor für die Performance. Der Reasoning-Prozess generiert nämlich Tokens, die dann vom Modell im nächsten Schritt wieder ausgewertet werden. Diese Tokens wiederum füllen das Context Window und nehmen damit Platz für das eigentliche Ergebnis der Anfrage weg. Als Maßnahme dagegen kann

zum einen das Context Window technisch vergrößert werden, was wieder mehr Ressourcen bei der Bearbeitung von Prompts bindet. Zum anderen sind Hersteller von Sprachmodellen dazu übergangen, für die Reasoning-Modelle eine eigene Token-Art einzuführen, die sog. Reasoning Tokens, die während des Denkprozesses generiert und auch wieder gelöscht werden. Damit füllen sie das Context Window nur temporär. Ein weiterer Ansatz ist, den Reasoning-Prozess im sog. *Latent Space* auszuführen. Dieser technische Bereich eines LLMs ist ein Hintergrundspeicher, den Benutzer des Sprachmodells nicht sehen können. Daraus ergibt sich die Möglichkeit, Reasoning Tokens mindestens nur als numerische oder binäre Daten abzubilden. Denn wenn der Nutzer diese Zwischenergebnisse des Reasoning-Vorgehens ohnehin nicht sieht, besteht keine Notwendigkeit, menschenlesbaren Text zu generieren und im nächsten Schritt wieder zu verarbeiten. Damit kann also ein Verarbeitungsschritt eingespart werden. Vielmehr noch, eröffnet dieser Ansatz die Möglichkeit, ein komplett neues Speichermodell für Reasoning Tokens zu entwerfen, das noch effizienter arbeitet.

6.2 Anwendungsfälle für Reasoning-Modelle

Reasoning-Modelle sind damit für kleinere oder einfache Aufgaben aktuell nicht das Mittel der Wahl. Sie sind dafür noch zu teuer, zu langsam und zu ressourcenintensiv. Sie eignen sich jedoch gut für Aufgabenstellungen mit komplexen Zusammenhängen.

Sie werden z. B. im Rahmen der Sozialforschung genutzt, um aus empirisch erhobenen Daten neue Erkenntnisse zu gewinnen. Dazu werden dem Sprachmodell relevante Informationen wie das Forschungsdesign, die erhobenen Rohdaten und bereits gewonnene Erkenntnisse bereitgestellt und dann nach einer Einschätzung oder weiteren Analyse gefragt. In einem Versuch in unserem akademischen Umfeld konnte das aktuelle Reasoning-Modell von OpenAI „o1“ bei einer quantitativen Studie ein komplett neues Korrelationscluster in den Daten ermitteln, das dem Forscherteam vorher nicht aufgefallen war. Die Richtigkeit dieses Clusters wurde im Nachgang mathematisch verifiziert und war für das Team dergestalt überraschend, als dass eine komplexe mathematische Problem-

stellung mit einem Sprachmodell gelöst werden konnte. Das Reasoning dafür hat etwa 20 min in Anspruch genommen.

Ein weiteres Feld, in dem Reasoning-Modelle gut eingesetzt werden können, ist die Analyse und Prüfung von Dokumenten wie Verträge oder Haushaltspläne. Die Fähigkeiten der Modelle ermöglichen es, auch vielschichtige Zusammenhänge oder Inkonsistenzen aufzudecken und sinnvolle Ergänzungen oder Anpassungen zu entwickeln. Während das mit den klassischen LLMs grundsätzlich auch möglich ist, zeigt sich im Vergleich, dass die Reasoning-Modelle deutlich mehr Verständnis entwickeln und die Qualität der Ergebnisse daher ungemein besser ist. Da die Modelle – wie beim Forschungsbeispiel umrissen – auch mathematische und statistische Zusammenhänge verstehen können, kann das Reasoning auch für Unterlagen wie Haushaltspläne o. Ä. genutzt werden. Ein selbst durchgeführter Versuch mit einem etwa 700-seitigen Haushaltsplan einer Kommune zeigte schon nach kurzer Auswertung konkrete Inkonsistenzen und Verbesserungsmöglichkeiten auf.

Die umfassenden Verständnisfähigkeiten von Reasoning-Modellen ermöglichen ihren Einsatz auch außerhalb klassischer textueller Aufgabenstellungen. So konnte bspw. gezeigt werden, dass sie die Fähigkeit besitzen, Produktionsprozesse zu planen und zu optimieren. Dabei besteht die Aufgabe darin, Produktionsaufträge möglichst effizient auf die zur Verfügung stehenden Ressourcen zu verteilen und dabei verschiedene Metriken zu optimieren, wie bspw. die durchschnittliche Prozessdurchlaufzeit. In Versuchen konnte gezeigt werden, dass eine solche Optimierung durch LLMs der von speziell dafür trainierten neuronalen Netzwerken ebenbürtig sein kann. (Abgaryan, 2024)

Auch im Bereich der Softwareentwicklung führt der Einsatz von Reasoning-Modellen zu deutlich besseren Ergebnissen. Beim Schreiben von Code können komplexere Anforderungen effektiv gelöst werden. Versuche zeigen, dass Modelle fast immer erweiterbaren, testbaren und fehlerfreien Programmcode schreiben. Während das bei klassischen LLMs auch schon häufig der Fall war, zeigt sich im Vergleich, dass Reasoning-Modelle auf Aspekte wie Erweiterbarkeit und Wartbarkeit optimieren und der Code damit insgesamt eine merkbar bessere Qualität aufweist. Auch auf der Seite der Qualitätssicherung der Softwareentwicklung können diese Modelle eingesetzt werden. Viele Softwareteams haben sog.

Code-Review-Prozesse, bei denen der programmierte Code von einem oder mehreren Softwareentwicklern geprüft wird. Teile dieser Prüfung können durch Werkzeuge, die Reasoning-Modelle nutzen, automatisiert werden. Dann können bspw. gängige Programmierfehler erkannt und direkt durch das Modell behoben werden.

Schließlich werden die Verständnis- und Analysefähigkeiten dieser Modelle auch bei Suchmaschinen genutzt. Anbieter wie bspw. Perplexity, die das Suchen im Internet mit den Fähigkeiten von LLMs kombinieren, um basierend auf den Suchergebnissen direkt verständliche Antworten, statt nur Links zu Webseiten zu liefern, bieten häufig bereits Reasoning-Modelle an. Wie erläutert benötigen diese bei der Recherche deutlich mehr Zeit, nutzen dann aber signifikant mehr Quellen für Antworten und formulieren im Ergebnis präzisere und umfangreichere Antworten.

Wichtig

Die Fähigkeiten der Reasoning-Modelle in der Softwareentwicklung zeichnet bereits deutlich die Veränderung dieser Tätigkeit in der Zukunft ab. Sprachmodelle werden mehr und mehr Teile der Aufgaben übernehmen und automatisieren. Ähnlich, wie es bereits seit vielen Jahren gängig ist, Code-Vorschläge zu erhalten und durch Vorlagen Teile von Code automatisiert zu generieren, wird es in der Zukunft zum Berufsbild des Softwareentwicklers dazugehören, mit einem LLM zu programmieren. Den Beruf selbst werden Sprachmodelle dabei nicht so schnell ersetzen. Denn zum Entwickeln einer Anwendung gehört mehr als nur das reine Schreiben des Codes.

Dass LLMs in der Softwareentwicklung aktuell noch sehr selten eingesetzt werden, liegt zum einen daran, dass das Verwenden öffentlicher Modelle wie ChatGPT, Claude oder Gemini Probleme mit Vertraulichkeit und Compliance verursachen würden und dass der Betrieb lokaler Modelle im Vergleich noch zu unwirtschaftlich ist. Zum anderen ist vor allem die Größe des Context Windows ein limitierender Faktor. Denn Programme werden schnell so groß, dass sie durch das LLM nicht mehr komplett erfasst werden könnten, weshalb dann wichtiges Wissen fehlen würde, das benötigt wird, um wartbaren Code gemäß der Projektanforderungen zu produzieren. Auch Ansätze mit Retrieval Augmented Generation (RAG) haben bisher nur unbefriedigende Ergebnisse gebracht.

6.3 Reasoning-Techniken der Modelle

Was Reasoning-Modelle von klassischen Sprachmodellen unterscheidet, ist vor allem das selbstständige Verwenden verschiedener Prompting-Techniken. Das ist im ersten Schritt die Chain-of-Thought-Technik, die dann im Folgenden durch verschiedene fortgeschrittene Ansätze weiterentwickelt wurde. Wie sich dadurch das Prompting bei der Arbeit mit Reasoning-Modellen ändert, wird in Abschn. 6.4 aufgezeigt.

Im Folgenden werden verschiedene Reasoning-Techniken moderner Modelle beschrieben. Interessant ist dabei folgende Feststellung: Betrachtet man die Techniken, stellt man fest, dass diese im Grunde nicht sonderlich kompliziert sind, sondern sich sehr nah an der menschlichen Herangehensweise von Problemlösungen bewegen und für die Modelle lediglich explizit gemacht wurden. Und doch zeigt die Forschung ganz klar, dass solche Techniken, die für uns selbstverständlich erscheinen mögen, extremen positiven Einfluss auf die Qualität der Ergebnisse von LLMs haben.

6.3.1 Automatisches Chain of Thought

Die Chain-of-Thought-Technik (Abschn. 4.3) ist die Grundlage von Reasoning-Modellen. Da die Modelle diese automatisch anwenden, wird auch von *Automatic Chain of Thought* gesprochen. Die Idee ist dabei dieselbe wie bei der manuellen Technik: ein Problem wird in logische Teilschritte zerlegt, die dann konsekutiv durch das Modell gelöst werden. Der letzte Schritt ist dann entweder die finale Problemlösung oder es findet alternativ noch eine dedizierte Synthese der Teillösungen statt.

Je nachdem, wie das Modell Chain of Thought umsetzt, ist der Denkprozess für den Nutzer des Modells sichtbar und kann nachvollzogen werden. Das Reasoning beim Suchen mit Perplexity wird während des Prozesses ausgegeben, indem das Modell erklärt, was es zu lösen hat, wie es die Aufgabe strukturiert und wonach dann im Internet recherchiert wird. Beim Reasoning mit o1 von OpenAI ist das Verhalten ähnlich, auch hier wird der „Gedankenprozess“ des Modells während der Ausführung angezeigt. Bei dem Modell Deepseek R1 wird das Reasoning in speziellen <think></think>-Tags als Teil der Antwort generiert.

6.3.2 Self-Consistency Decoding

Mittels Self-Consistency Decoding können die Ergebnisse eines Reasoning-Prozesses durch Chain of Thought noch einmal effektiv verbessert werden. Dazu werden für eine Fragestellung verschiedene (bspw. 3 oder 5) Reasoning-Pfade generiert. Das Modell überprüft dann die Ergebnisse der einzelnen Pfade und wählt die wahrscheinlichste Gesamtantwort aus. Im Endeffekt denkt das Modell bei diesem Ansatz über eine Problemstellung mehrfach nach und das zumeist parallel.

Bspw. würde ein Modell mit dieser Technik für eine mathematische Problemstellung statt nur einer einzelnen Lösung mehrere Schritt-für-Schritt-Lösungen entwickeln. Diese Lösungen würden verglichen und dann das logischste und konsistenteste Ergebnis gewählt werden. Dadurch werden Fehler beim Verständnis oder der Herangehensweise reduziert.

6.3.3 Tree of Thoughts

Eine signifikante Erweiterung von Chain of Thought ist die Tree-of-Thoughts-Technik (ToT). Statt wie bei CoT einem einzelnen, linearen Pfad des Denkprozesses zu folgen, generiert das Modell einen Baum an Lösungspfaden. Der Ansatz ist vergleichbar mit der Mind-Mapping-Technik und ist besonders dann sinnvoll, wenn zu komplexen Problemstellungen unterschiedliche Lösungen entwickelt werden können, abhängig davon, wie der Denkprozess verlaufen ist und welche Annahmen entlang des Weges getroffen wurden.

Versuche haben gezeigt, dass beispielsweise für Sodoku-Puzzle signifikant bessere Ergebnisse mit Tree of Thoughts entwickelt werden konnten als beim Einsatz von Zero-Shot-, One-Shot- und Few-Shot-Techniken.

6.3.4 ReAct

Eine darauf aufbauende Technik, die vor allem im Kontext autonomer KI-Agenten Anwendung findet, ist die Auftrennung von Denkprozess (Reasoning) und Handlung (Action), kurz ReAct. Der Ansatz dahinter

ist im Grunde simpel. Das Modell wird instruiert, zunächst mittels Reasoning die Problemstellung zu verstehen und einen Lösungsweg zu planen. Erst im zweiten Schritt wird dann die tatsächliche Problemlösung vorgenommen, die bei KI-Agenten auch die Interaktion mit anderen (KI-) Systemen bedeuten kann. Da dabei u. U. auch irreversibel Einfluss auf die Umwelt genommen werden kann (man denke an ein Szenario, in dem ein KI-Agent etwas aus einem System löscht), ist es sinnvoll, ein Problem zuerst zu strukturieren und in Gänze zu erfassen, statt ohne Weitsicht Schritt für Schritt vorzugehen.

6.4 Prompting für Reasoning-Modelle

Wenig überraschend machen sich die besseren Fähigkeiten von Reasoning-Modellen auch bei der Verwendung der Modelle bemerkbar. Grundsätzlich gelten die allgemeinen Rahmenbedingungen des Promptings von Sprachmodellen weiterhin. Das Prompting ist im Großen und Ganzen dem klassischer LLMs sehr ähnlich. Allerdings gibt es auch verschiedene Einschränkungen, die zu beachten sind.

Grundsätzlich sind die Modelle gut darin, Probleme umfänglich zu erschließen. Daher ist es häufig nicht notwendig, eine große Menge an Details oder Beispielen im Prompt mitzugeben. Das heißt keineswegs, dass Beispiele nicht sinnvoll wären, sondern es bedeutet lediglich, dass diese oberflächlicher ausfallen können. Notwendigkeit und Umfang sind aber immer im Einzelfall zu bewerten. Solche kontextuellen Informationen und auch Angaben zum gewünschten Lösungsweg sind vor allem dann notwendig, wenn das Risiko besteht, dass das Modell entlang des Reasoning-Weges falsche Schlüsse ziehen könnte. Wenn also bspw. eine bestimmte Annahme bereits im Vorhinein ausgeschlossen werden kann, hilft das explizite Benennen, Vorgehen und Qualität zu verbessern.

Da Reasoning-Modelle ohnehin Chain of Thought und darauf aufbauende Techniken für die Problemlösung anwenden, müssen diese nicht extra im Prompt eingefordert werden. Diesbezügliche Prompting-Techniken sind also nicht notwendig. Auch das Angeben eines Schritt-für-Schritt-Lösungswegs ist nicht sinnvoll, da die Modelle darauf trainiert sind, diesen selbstständig zu entwickeln und im Laufe des Prozesses

zu verfeinern bzw. anzupassen. Das Erzwingen eines Vorgehens kann daher im Zweifel die Ergebnisse messbar verschlechtern.

Interessanterweise ist auch das Verwenden von Rollen häufig nicht sinnvoll. Tatsächlich haben verschiedene Versuche gezeigt, dass bei Reasoning mit Chain of Thought das Forcieren von Rollen zu schlechteren Ergebnissen führen kann. Vor allem dann, wenn diese Rolle sich nicht mit der für die Problemlösung notwendige Rolle deckt. Die Empfehlung ist daher, keine Rollen bei Reasoning-Modellen zu verwenden (Han, 2024).

Prompt Framework für Reasoning-Modelle

Wie bereits beschrieben, gelten für das Prompting von Reasoning-Modellen etwas andere Regeln. Trotzdem hat sich auch hier ein Prompting Framework herausgebildet, welches zu sehr guten Ergebnissen führt. Das Framework besteht dabei aus vier Abschnitten:

1. **Goal (Ziel):** Hier wird ausführlich beschrieben, was mit dem Prompt erreicht werden soll.
2. **Return Format (Rückgabeformat):** In diesem Abschnitt wird konkret beschrieben, wie die Ausgabe durch das LLM strukturiert werden soll.
3. **Warnings (Warnungen):** Im nächsten Abschnitt werden potenzielle Fehlerquellen beschrieben, die es zu umgehen gilt, und restriktive Anweisungen übermittelt.
4. **Context Dump (Kontextrahmen):** Im letzten Abschnitt werden für die optimale Lösung der Aufgabe benötigte Hintergrundinformationen übergeben. Dies hilft es, die Antwort in den richtigen Kontext einzuordnen.

Prompts, die in dieser Struktur übermittelt und mit entsprechend relevanten Informationen und Anweisungen angereichert sind, versprechen bestmögliche Ergebnisse, da sie dem Modell zum einen konkreten, aber nicht zu restriktiven Rahmen vorgeben und zum anderen dem Modell genug Freiraum für den „Denkprozess" lassen.

Nachfolgend wird ein beispielhafter Prompt auf Basis des hier vorgestellten Prompt Frameworks vorgestellt.

Beispiel

1. **Goal (Ziel)**

Ich benötige eine detaillierte Recherche über aktuelle Best Practices und Trends im Bereich **hybrides Arbeiten (Hybrid Work)**. Der Fokus soll auf bewährten Strategien für Unternehmen liegen, die sowohl Remote- als auch Büroarbeit kombinieren. Das Ziel ist es, eine fundierte Entscheidungsgrundlage für die Weiterentwicklung unserer internen Arbeitsrichtlinien zu erhalten.

2. **Return Format (Rückgabeformat)**

Erstelle eine strukturierte Zusammenfassung mit folgenden Abschnitten:

- **Definition** – Eine klare Erklärung, was hybrides Arbeiten ist.
- **Vorteile und Herausforderungen** – Eine Auflistung der wichtigsten Vorteile und Herausforderungen für Unternehmen und Mitarbeitende.
- **Aktuelle Trends** – Ein Überblick über aktuelle Entwicklungen und innovative Ansätze.
- **Best Practices** – Konkrete Strategien, die erfolgreiche Unternehmen nutzen, um hybrides Arbeiten effektiv umzusetzen.
- **Quellen und weiterführende Links** – Eine Liste mit vertrauenswürdigen Studien, Berichten oder Artikeln zur Vertiefung.

3. **Warnings (Warnungen)**

- Stelle sicher, dass alle Informationen aus **aktuellen und seriösen Quellen** stammen (z. B. Fachartikel, Studien, Berichte von Unternehmensberatungen oder Tech-Unternehmen).
- Vermeide allgemeine oder veraltete Aussagen – die Recherche soll **state-of-the-art** sein.
- Die Zusammenfassung soll **prägnant und verständlich** sein (max. 1000 Wörter), sodass sie als Entscheidungshilfe für Führungskräfte genutzt werden kann.

4. **Context Dump (Kontextrahmen)**

Unser Unternehmen plant, die bestehenden Homeoffice-Regelungen zu überarbeiten und eine langfristige Strategie für hybrides Arbeiten zu entwickeln. Dabei sollen sowohl die Bedürfnisse der Mitarbeitenden als auch die betrieblichen Anforderungen berücksichtigt werden. Besonders wichtig sind Aspekte wie Produktivität, Unternehmenskultur, digitale Tools und die langfristige Arbeitsplatzgestaltung.

7

KI-Agenten

Zusammenfassung In diesem Kapitel wird zunächst in das Konzept der KI-Agenten eingeführt. Es ist für Unternehmen wichtig, sich mit diesen Systemen auseinanderzusetzen, da sie zukünftig viele interne und externe Aufgaben und Prozesse vollkommen autonom ausführen können. Dazu werden unterschiedliche KI-Agenten-Konzepte vorgestellt und beschrieben, wie die Agenten untereinander und mit weiteren Systemen interagieren und Daten sowie Informationen austauschen. Anschließend wird beschrieben, warum das Prompt Engineering in diesem Kontext so wichtig ist. Abschließend werden unterschiedliche Anwendungsfälle für KI-Agenten im Unternehmen vorgestellt.

Viele Anwendungsfälle von Sprachmodellen konzentrieren sich aktuell auf dialogisch-interaktive Szenarien. Ein Nutzer interagiert proaktiv mit einem System wie ChatGPT oder Claude durch einen Chat. Die Interaktion findet in dem Moment statt, in dem der Nutzer ein konkretes Problem lösen möchte. Die Arbeit mit dem LLM ist dergestalt also synchron.

D. Koch et al., *Prompt Engineering im Unternehmen – eine Einführung*,
https://doi.org/10.1007/978-3-658-47699-1_7

Eine Entwicklung, die zurzeit an Relevanz für Unternehmen gewinnt, ist die von KI-Agenten: (teil-)autonome Sprachmodelle, die selbstständig und unabhängig der fortlaufenden Interaktion durch einen Nutzer im Hintergrund Aufgaben lösen und Ergebnisse produzieren. Das Konzept und die Bausteine von Agenten werden in diesem Kapitel im Kontext des Prompt Engineering dargestellt.

7.1 Das Konzept von KI-Agenten

Sprachmodelle wie ChatGPT, Claude oder Gemini werden heutzutage vor allem reaktiv eingesetzt. Mitarbeiter können mit ihnen Aufgaben ganz oder teilweise automatisieren und schnell zu guten Ergebnissen kommen. Unternehmen können damit ihre Effizienz erhöhen und unter Umständen auch die Qualität verbessern. Die damit verbundene Reduktion der benötigten Zeit für Tätigkeiten macht sich am Ende auch in einer Kostenreduktion bemerkbar. Der Effekt entsteht vor allem mittelbar durch die Kombination vieler kleiner Optimierungen entlang der unternehmerischen Wertschöpfung.

Obwohl es für Unternehmen äußerst sinnvoll ist, in diese punktuellen, durch Sprachmodelle ermöglichten Effizienzoptimierungen zu investieren, birgt diese Art der KI für Unternehmen noch deutlich größeres Potenzial. Nimmt man einmal an, man könnte die Fähigkeiten von LLMs nicht nur an einzelnen Punkten in der Wertschöpfung nutzen, sondern ganze Geschäftsprozesse an Sprachmodelle übergeben, die dann in einem abgesteckten Rahmen selbstständig intelligente, begründete Entscheidungen treffen und auch die notwendigen Tätigkeiten ebenfalls selbstständig durchführen, hätte man eine Automatisierung von Geschäftsprozessen geschaffen, die erst einmal keinerlei technischen Sachverstand oder individuelle Entwicklung von Software bedürfe. Vielmehr entstünde sie aus rein fachlichem Wissen heraus, das in natürlicher Sprache einem LLM die Ausgangssituation, das Ziel und ggf. auch zum Teil das Vorgehen erläutert. Man kann hier klare Parallelen zum Citizen Development sehen, bei dem ebenfalls das Ziel verfolgt wird, Menschen zu befähigen, Automationen und Software ohne fundamentale Programmierkenntnisse zu erstellen. Dies geschieht häufig mit Low-

Code- oder No-Code-Systemen, bei denen Standardanwendungsfälle (z. B. Abfragen von Daten aus einer Datenbank oder Aufbereitung von Informationen in Tabellen und Grafiken) zusammengeklickt werden können. Der große Unterschied zum Citizen Development liegt darin, dass bei KI-Agenten erst einmal überhaupt keine technischen Kenntnisse vorhanden sein müssen. Außerdem erfolgt die Ausführung der Automatisierung autonom und nicht deterministisch.

Auf den ersten Blick könnte man daher annehmen, dass KI-Agenten einem Low-Code- oder No-Code-System überlegen wären. Tatsächlich kommt es aber auf die jeweilige Situation an, denn beide Arten von Systemen haben ihre Herausforderungen und Grenzen. Und erst durch die Kombination der beiden Ansätze ergibt sich für Unternehmen ein unschätzbares Potenzial. Denn so wie manche Prozesse sehr strikt und deterministisch sind, also leicht und effizient in Form von Software abgebildet werden können, benötigen andere mehr intelligent getroffene dispositive Entscheidungen – eine eindeutige Anwendungsmöglichkeit für KI-Agenten.

7.1.1 Definition von KI-Agenten

Wenig überraschend gibt es keine allgemeingültige Definition für KI-Agenten. Im Jahr 1995 wurde der Begriff das erste Mal im Kontext von KI definiert und erfasste einen Agenten als etwas, das seine Umwelt durch Sensoren wahrnimmt und mit dieser durch Aktoren interagiert (Russel, 1995).

Diese Definition ist recht abstrakt, wenig greifbar und inkludiert auch nicht die Entwicklung von LLM, auf die sich im Rahmen dieses Buches bezogen wird. Im Kontext solcher Sprachmodelle definieren wir KI-Agenten daher wie folgt:

Ein KI-Agent ist im Kern ein Sprachmodell (LLM), das mit der Möglichkeit ausgestattet wurde, relevante Informationen zu suchen, zu speichern (Memory) und mit der Außenwelt (andere Systeme oder KI-Agenten) interagieren kann.

Ein LLM, das mit solchen Fähigkeiten ausgestattet wurde, wird als Augmented LLM bezeichnet. Die erste wichtige Fähigkeit eines solchen

Augmented LLM ist die Möglichkeit, selbstständig nach Informationen zu suchen. Dazu entwickelt das Modell passende Suchanfragen und durchsucht verfügbare Quellen nach passenden Ergebnissen. Für öffentlich verfügbare Modelle wie beispielsweise Claude von Anthropic bedeutet das den Einsatz einer Suche im Internet mit einer Suchmaschine. Das Augmented LLM hat dann aber vielmehr noch die Möglichkeit, die Suchergebnisse zu verarbeiten und für die Erfüllung der Aufgabe mit einzubeziehen.

Die zweite Fähigkeit ist es, Wissen selektiv als Memory zu speichern. Dieses Wissen geht beim Verändern des Context Windows (vgl. Abschn. 1.3) nicht verloren, sondern steht durchweg für zukünftige Aktionen des Sprachmodells zur Verfügung. Selektiv speichern heißt, das Memory ist kein einfacher Speicher, in dem sämtliche Zwischenergebnisse abgelegt werden. Vielmehr entscheidet das Modell selbstständig, welche Informationen für die Zukunft notwendig sind und verwirft den Rest.

Mit dem dritten Baustein bekommen KI-Agenten die Möglichkeit, nach eigenem Ermessen mit anderen Systemen und Werkzeugen (Tools) zu interagieren. Das können beispielsweise wiederum selbst KI-Agenten sein, an die Tätigkeiten ausgelagert werden. Der KI-Agent kennt die verfügbaren Systeme, deren Fähigkeiten und Limitierungen und kann selbstständig entscheiden, ob und wann diese genutzt werden.

Im Grunde genommen ist die Möglichkeit zur Recherche auch eine Integration in ein Drittsystem und könnte daher unter dem letzten Baustein subsumiert werden. Da diese Recherchemöglichkeit aber oft einen fundamentalen Aspekt der Aufgaben für KI-Agenten darstellt und die Arbeit mit Drittsystem auch Nebeneffekte haben kann, wird die Suchmöglichkeit als separater Baustein angesehen.

Dass im Kern des KI-Agenten ein LLM steckt, ist ein wichtiger Aspekt für die Lernfähigkeit. Denn KI-Agenten lernen während ihrer Ausführung und produzieren im Laufe der Zeit andere – hoffentlich bessere – Ergebnisse. Unter Umständen sind Agenten auch reaktiv; sie reagieren auf Veränderungen ihrer Umwelt und passen ihr Verhalten entsprechend

angemessen an. Auf der anderen Seite sind sie auch proaktiv, indem sie nicht nur ihre Ziele eigenständig verfolgen, sondern auch ihre Aktionen vorausschauend planen.

7.1.2 Unterscheidung zwischen Agenten und Workflows

Bei der architekturellen Betrachtung von Agentensystemen wird grundsätzlich zwischen Workflows und Agenten unterschieden. Die Unterscheidung wird anhand des Einsatzes von Software-Code getroffen.

Ein KI-Agent ist ein dergestalt autonomes System, als dass es selbstständig die Schritte zur Aufgabenlösung determiniert, abarbeitet und die Ergebnisse produziert. Es wählt die dafür notwendigen Tools nach eigenem Ermessen aus und integriert die Einzelergebnisse sinnvoll.

Ein Workflow wiederum ist ein System, das aus der Kombination von einem oder mehreren Sprachmodellen und Softwaretools besteht und durch Code orchestriert wird. Für diese Orchestrierung einzelner Komponenten kann beispielsweise ein No-Code- oder Low-Code-System zum Einsatz kommen. In einem Workflow-System ist also immer eine gewisse Menge Determinismus enthalten.

Die Definition von Workflow und KI-Agent sind jeweils nicht exklusiv. In einem Workflow-System können KI-Agenten zum Einsatz kommen und diese durch das System gesteuert werden, genauso wie ein KI-Agent ein Workflow-System zur Aufgabenerledigung ansprechen kann.

7.2 Architekturen von Workflow-Systemen

Für die Architektur von Workflow-Systemen gibt es verschiedene Ansätze, die im Grunde alle auf bekannten Architekturmustern aus der Softwareentwicklung aufbauen. Je nach Zielsetzung werden verschiedene Komponenten (LLMs und Softwarebausteine) in eine Reihenfolge gebracht und dann bei der Ausführung sequenziell oder parallel durchlaufen. Die Entscheidung des genauen Ablaufs eines Workflows kann sowohl von Software-Code als auch jeweils durch LLMs getroffen werden.

7.2.1 Workflow-Architektur 1: Iterative Qualitätskontrolle

Bei dieser Architektur werden zwei LLMs kombiniert, um eine Aufgabenstellung zu bearbeiten und die Qualität des Ergebnisses sicherzustellen. Das erste LLM bearbeitet die eigentliche Aufgabenstellung mit seinen Fähigkeiten. Falls es sich bei diesem LLM um einen KI-Agenten im Sinne der Definition handelt, kann das beispielsweise bedeuten, dass es eine Suche nach Informationen im Internet durchführt oder mit Systemen innerhalb des Unternehmens agiert.

Sobald die Ergebnisse produziert wurden, übergibt das erste LLM diese an das zweite LLM. Dieses zweite Sprachmodell ist auf die Sicherung der Qualität der Ergebnisse spezialisiert. Es wird diese analysieren und mit der Aufgabenstellung abgleichen. Dann wird es Feedback für das erste LLM produzieren, sofern das gewünschte Ziel noch nicht erreicht wurde, und das Feedback an das erste LLM übergeben.

Dann beginnt das erste LLM wiederum erneut mit der Bearbeitung der Aufgabe, bezieht aber das Feedback mit ein. Dieser Kreislauf wird so lange durchlaufen, bis die Ergebnisse adäquat sind und der Workflow mit Zielerreichung beendet wird.

Für eine effektive Interaktionskette wird das zweite LLM als zusätzlicher Baustein nur für die Qualitätssicherung eingesetzt und auch nur vom ersten, eigentlichen LLM dafür benutzt. Die finalen Ergebnisse übermittelt das erste LLM an den Nutzer oder den nächsten Workflow weiter. In der Softwarearchitektur ist dieses Aufbaumuster unter anderem als „Sidecar“ bekannt.

7.2.2 Workflow-Architektur 2: Selektive Auswahl von LLMs

Verschiedene Sprachmodelle – ob Large Language Models oder Small Language Models – haben unterschiedliche Stärken und Schwächen. Da diese Modelle nicht deterministisch sind, kann es in umfangreichen Systemen sinnvoll sein, das zu nutzende Sprachmodell jeweils nicht vorzugeben, sondern innerhalb des Workflows eine Entscheidung treffen zu lassen.

Mit diesem Architekturmuster wird ein spezielles Sprachmodell als Router eingebaut. Die Aufgabe des Modells ist es, Informationen entgegenzunehmen und zu entscheiden, welches subsequente Sprachmodell die Aufgabenstellung mit oder für diese Informationen am besten beantworten kann. Das Router-LLM wird dann an das entsprechende Modell übergeben, das das finale Ergebnis produziert.

Zum Beispiel kann ein Prompt eine Internetsuche notwendig machen. Der Router weiß nun, welches LLM diese Fähigkeit besitzt und kann zwischen diesen auswählen. Ein zweiter Prompt könnte wiederum eine Übersetzungsaufgabe sein, für die zwar keine Internetsuche, aber ein auf Übersetzungen spezialisiertes Modell benötigt wird. Das Router-LLM würde dann an ein solches Modell übergeben.

7.2.3 Workflow-Architektur 3: Parallelisierung mit mehreren LLMs

Für zeitintensive oder komplexe Aufgabenstellungen können ebenfalls mehrere LLMs genutzt werden, aber anders als bei der selektiven Auswahl in Form einer parallelen Bearbeitung der Aufgabe. Dabei gibt es anders als bei der Architektur der selektiven Auswahl kein Router-LLM, sondern eine Aufgabe wird durch eine programmierte Komponente an mehrere LLMs gleichzeitig übergeben.

Die Einzelergebnisse werden dann durch einen Integrator final zu einem Ergebnis zusammengefasst. Der Integrator ist in diesem Architekturmuster kein LLM, sondern eine Softwarekomponente und damit deterministisch.

Die Parallelisierung kann dabei auf zwei Wegen geschehen. Ist die Gesamtaufgabe hinreichend komplex, könnte sie in Teilaufgaben zerlegt werden. Diese Teilaufgaben werden dann an verschiedene LLMs weitergegeben und gleichzeitig bearbeitet. Den LLMs fehlt dabei unter Umständen der Gesamtkontext, vor allem, wenn es sich um spezielle Sprachmodelle handelt, die nicht die gesamte Aufgabe erfassen können.

Bei der zweiten Möglichkeit wird die gesamte Aufgabe auf mehrere LLMs verteilt und der Integrator führt die Ergebnisse dann selektiv zusammen. Im einfachsten Fall prüft er, welches Teilergebnis am häufigsten vorkommt, wählt dieses als Endergebnis und verwirft die restlichen.

7.2.4 Workflow-Architektur 4: Multi-LLM-Prompt-Chaining

Im Rahmen des Prompt Frameworks Prompt Chaining (vgl. Abschn. 3.1) wurde bereits angedeutet, dass mit dieser Technik Anfragen auch über mehrere Sprachmodelle verteilt werden können, die dann sequenziell durchlaufen werden und das Ergebnis jeweils weiterentwickeln. In einem Workflow-System passiert diese Verkettung von Sprachmodellen automatisch und ein manuelles Interagieren mit mehreren Modellen seitens des Nutzers ist nicht notwendig.

Zwei zusätzliche Bausteine können in den Prozess zur Steuerung eingebaut werden. Zum einen kann es notwendig sein, Zwischenergebnisse deterministisch zu verändern. Dafür werden mit Code programmierte Transformatoren eingesetzt, die zwischen zwei LLMs sitzen und die Ergebnisse des einen vor der Weiterleitung an das zweite Verändern. Zum anderen kann es Situationen geben, in denen ein Workflow vorläufig abgebrochen werden soll. Dazu können konditionale Bausteine eingefügt werden, die ebenfalls mit Code programmiert werden. Diese entscheiden anhand eines Ergebnisses eines LLMs, ob der Workflow fortgeführt werden soll. Falls ja, übergeben sie das Ergebnis entsprechend weiter. Falls nein, beenden sie den Workflow.

7.2.5 Workflow-Architektur 5: Orchestrierung mit LLMs

Als Variante der Parallelisierung, bei der eine programmierte Komponente die Aufgabe auf parallel mehrere LLMs verteilt und ein ebenfalls programmierter Integrator die Zusammenführung übernimmt, wird bei der LLM-Orchestrierung die gesamte Verantwortung an zwei LLMs übergeben.

Das erste LLM, der Orchestrator, ist für die Aufteilung und Parallelisierung der Aufgabe verantwortlich. Er bestimmt, auf welche Art mit welchen LLMs parallelisiert wird und leitet die jeweiligen (Teil-)Aufgaben an die Modelle weiter. Das zweite LLM, der Synthetisierer, erhält die Ergebnisse der verschiedenen LLMs und kombiniert diese ganz oder teilweise, je nach Vorgabe und Ermessen.

Bei der Orchestrierung sind also Ablauf und Kombination von LLMs komplett dem einem steuernden Sprachmodell überlassen, genauso wie die finale Ergebnisproduktion einem Modell überlassen wird. Je nach Implementierung kann dies der Orchestrator als auch der Synthetisierer sein.

7.2.6 Kombination von Architekturen

Diese verschiedenen Architekturen von LLM-Workflows sind jeweils als Bausteine und als idealtypische Muster zu verstehen. Workflow-Systeme, die in der Praxis eingesetzt werden, sind häufig nicht präzise einem Architekturmuster zuzuordnen, da sie im Detail Anpassungen beinhalten. Beispielsweise werden programmierte Filtersysteme oder Monitoring-Komponenten installiert, die im eigentlichen Muster nicht enthalten sind.

Auf der anderen Seite sind die Architekturmuster miteinander kombinierbar. Innerhalb eines durch einen Orchestrator parallelisierten Workflows können beispielsweise verkettete LLMs genutzt werden, für die dann vereinzelt qualitätssichernde Sidecar-LLMs eingebaut werden.

Wie auch bei der Architektur von Softwaresystemen ist das Ziel, zwar auf der einen Seite erprobte und dokumentierte Vorgehen zu verwenden, auf der anderen Seite aber vor allem das unternehmerische Ziel zu erreichen.

7.3 Architekturen von KI-Agenten

KI-Agenten beschränken sich in ihrer Architektur anders als Workflow-Systeme auf nur ein einzelnes Augmented LLM mit entsprechenden Fähigkeiten. Sie können aber wiederum auch Teil eines Workflow-Systems sein.

Wie bei der Definition von KI-Agenten erwähnt, haben diese grundsätzlich die Möglichkeit, mit anderen Systemen und KI-Agenten zu interagieren. Diese Zusammenarbeit ist dann allerdings kein Workflow-System, denn letztere haben nach außen einen Blackbox-Charakter. Auch ist ein KI-Agent autonom und nicht deterministisch und kann daher

eigenmächtig entscheiden, ob und mit welcher nächsten Komponente oder mit welchem KI-Agenten er fortfährt. Multi-KI-Agenten-Systeme sind im Zweifel eine Menge autonomer KI-Agenten, die bei Bedarf miteinander kollaborieren, die Verantwortung für die Erledigung der Aufgabe liegt dabei aber immer am Ende bei einem einzelnen Agenten.

Die Arbeit von KI-Agenten beginnt immer mittelbar oder unmittelbar durch die Interaktion mit einem Nutzer. Unmittelbar beispielsweise dadurch, dass jemand eine Anfrage in einem Chatbot stellt. Mittelbar wiederum z. B. dadurch, dass eine Softwarekomponente programmiert wurde, die zu bestimmten Zeitpunkten (z. B. täglich) den Agenten startet.

Während der Bearbeitung einer Aufgabenstellung können Agenten auch mit Nutzern interagieren, z. B. pausieren und Feedback einholen oder Fragen beantworten lassen. In der Praxis könnte das über einen persistenten Chatbot oder auch über E-Mail-Kommunikation geschehen. Außerdem können auch hier wieder Softwarekomponenten einem arbeitenden Agenten Informationen zur Verfügung stellen oder Signale senden, die die Aufgabenbearbeitung beeinflussen.

Ist die Aufgabenstellung des KI-Agenten vollständig abgeschlossen, wird dieser beendet. Die Aufgabe kann durchaus kein fest definiertes Ende besitzen, z. B. wenn der Agent regelmäßig bis auf weiteres dieselbe Tätigkeit durchführen soll.

7.4 Die Notwendigkeit des Prompt Engineering für KI-Agenten

Die Autonomie und die weitreichenden Möglichkeiten von KI-Agenten machen es notwendig, diese sorgfältig vorzubereiten und zu konfigurieren. Da diese mittels natürlicher Sprache „programmiert" werden, ist eine strukturierte, effektive und zielorientierte Kommunikation in Form der Aufgabenstellung sehr wichtig. Kurzum, für KI-Agenten ist gutes Prompt Engineering unabdingbar.

Grundsätzlich ist dort erst einmal der Prompt an sich, der für den Agenten geschrieben werden muss. Die in diesem Buch vorgestellten Prompt-Frameworks können dabei helfen, die Aufgabenstellung und das

Ziel des Agenten präzise zu definieren und damit unerwünschte Ergebnisse oder Seiteneffekte zu vermeiden.

Darüber hinaus muss dem Agenten mitgeteilt werden, welche Möglichkeiten in Form anderer Agenten und Tools zur Verfügung stehen. Dafür gibt es technische wie auch natürlichsprachliche Ansätze. Fähigkeiten wie das Speichern im Memory und die Möglichkeiten der Suche im Internet sind den Modellen i. d. R. technisch beigebracht und bedürfen keinerlei zusätzlichen Konfiguration.

Daneben ist auch der Kontext des Agenten außerordentlich wichtig. Es muss präzise bestimmt werden, welche Informationen er in welchem Format benötigt, um seine Aufgabe am Ende zielführend erfüllen zu können. Das geht über das reine Schreiben des Prompts, also Struktur und Format, hinaus. Da einem aktiven Agenten erst einmal keine zusätzlichen Informationen und Daten bereitgestellt werden können, müssen diese als Teil des Prompts oder auch in Form von Dokumenten bei der Konfiguration zur Verfügung gestellt werden. Anders als bei interaktiven Chats mit einem LLM gewinnt die sorgfältige Vorplanung bei Agenten deutlich an Wichtigkeit.

Ein Aspekt, der bei der interaktiven Arbeit mit Sprachmodellen nicht zum Tragen kommt, ist das Testen. Konfigurierte Agenten müssen vor der Inbetriebnahme ausreichend getestet werden, damit sie auf der einen Seite auch das gesetzte Ziel erreichen, auf der anderen Seite aber auch in der Ausarbeitung keine unerwünschten Nebeneffekte produzieren. Ein guter Ansatz zum Verifizieren der Funktionalität ist das A/B-Testing: Es werden mehrere unterschiedliche Versionen des Prompts und des Kontexts geschrieben bzw. produziert und damit ermittelt, welche am effektivsten funktionieren. Testen heißt im einfachsten Fall, das LLM in einem klassischen, Nicht-Agenten-Modus mit dem Prompt auszuführen und die Ergebnisse zu validieren. Zusätzlich können aber auch verschiedene LLMs eingesetzt werden. Ein entsprechend guter Prompt kann mittels Metaprompting geschrieben werden. Dieser könnte dann durch ein zweites LLM hinsichtlich Kriterien wie Eindeutigkeit, Vollständigkeit und Kohärenz geprüft werden. Die zum Test produzierten Ergebnisse des Agenten-LLM können dann wiederum durch ein anderes Sprachmodell bewertet werden. Aus diesem Zusammenspiel, kombiniert mit den Fähigkeiten und dem Wissen des Prompt Engineers, können gute Prompts für

den Agenten entwickelt werden, die dann in der Ausführung auch das gesetzte Ziel in der erwarteten Qualität liefern.

7.5 Typische Anwendungsfälle für KI-Agenten in Unternehmen

Die abstrakte Darstellung der Kombinationsmöglichkeiten von Workflows und Agenten, vor allem im Vergleich zu No-Code- und Low-Code-Systemen, zeigt, dass die Anwendungsfälle zahlreich sein können. Grundsätzlich kann bei ganz oder teilweise digitalisierten Geschäftsprozessen geprüft werden, welche Möglichkeiten für einen agentenbasierten Ansatz bestehen. Vielmehr noch eröffnen Agenten durch ihre Flexibilität und das Verständnis natürlicher Sprache viele Möglichkeiten für Digitalisierung und Automatisierung, die mit bisherigen Werkzeugen nicht möglich war.

7.5.1 Automatisierte Kundenbetreuung

Schon länger versuchen Unternehmen, Routineaufgaben in der Kundenbetreuung zu automatisieren. Dadurch soll vorhandenes Personal entlastet und Kapazitäten sollen besser genutzt werden. Ein häufig eingesetztes Werkzeug dafür sind Chatbots, die beispielsweise auf einer Webseite eingebettet wurden und dort Zugriff auf Hilfsartikel und FAQ haben (Kohne, 2020). Nutzer können diesen Chatbots Fragen stellen und erhalten Antworten aus den verfügbaren Informationen.

Diese Ansätze hatten vor allem zwei Hürden. Zum einen muss das Wissen seitens des Unternehmens produziert und bereitgestellt werden. Auf der anderen Seite waren Chatbots lange Zeit nur eingeschränkt nutzbar, da sie eben nicht auf Sprachmodellen basierten, sondern beispielsweise Stichworte aus Anfragen nahmen und ihr vorhandenes Wissen danach durchsuchten. Dabei kam es regelmäßig zu falschen und unzureichenden Ergebnissen. Auch war keine Interaktion möglich, denn der Chatbot war zumeist eigentlich nur eine bessere Suchmaske auf Basis der ohnehin verfügbaren Informationen.

Das änderte sich schlagartig mit der Einführung von LLMs. Richtig trainiert und konfiguriert können Chatbots nun Anfragen von Nutzern verstehen und im richtigen Kontext korrekte Antworten liefern. Auch Nachfragen können beantwortet und entsprechende Quellverweise bereitgestellt werden. Der Chatbot hat zudem keine Öffnungs- oder Arbeitszeiten und steht daher permanent zur Verfügung. Außerdem kann ohne viel Aufwand Mehrsprachigkeit integriert werden.

Treibt man dies noch einen Schritt weiter und macht aus dem „einfachen" Chatbot einen KI-Agenten, ergibt sich für die Nutzer signifikant mehr Nutzen. Der Agent kann mit seiner Suchfunktion deutlich mehr Informationen des Unternehmens auswerten und aufbereiten. Kombiniert mit einer entsprechenden Autorisierung und Authentifizierung könnte der Agent für den Nutzer nicht nur Antworten liefern, sondern auch Anfragen direkt bearbeiten, beispielsweise Kundentickets eröffnen, Verträge verwalten oder Stammdaten aktualisieren.

7.5.2 Automatisiertes Recruiting

Im Bereich des Recruiting fallen häufig wiederholende Tätigkeiten an, die ebenfalls grundsätzlich Potenzial für den Einsatz von KI-Agenten bieten. Z. B. kann das Prüfen einkommender Bewerbungen automatisiert werden. Ein Agent kann frei von Emotionen objektiv Bewertungen vornehmen und die besten Kandidaten für eine vakante Stelle ausmachen. Nicht passende Bewerbungen können direkt automatisiert oder nach einem Prüfen durch Fachpersonal abgelehnt werden. Für positive Entscheidungen können Agenten interaktiv mit dem Bewerber einen Gesprächstermin ausmachen und dabei auch die Verfügbarkeiten der betroffenen Mitarbeiter mit einbeziehen. Nach einer Einstellung können KI-Agenten das Onboarding begleiten, Prozesse und Vorgaben des Unternehmens erläutern und Antworten auf Fragen suchen – und dazu z. B. auch andere Mitarbeiter mit Fachwissen identifizieren und automatisch involvieren.

Aufgrund des großen Potenzials im Recruiting gibt es für viele dieser Möglichkeiten bereits seit Längerem Plattformen und Tools. Mit den Fähigkeiten, die KI in Kombination mit einem Agentensystem bietet,

können diese Prozesse noch einmal nutzerfreundlicher, schneller und autonomer umgesetzt werden – und damit für das Unternehmen Kosten- und Effizienzgewinne bringen.

Durch die Abhängigkeit der Trainingsdaten des LLMs und des Lernens über die Zeit muss man klar festhalten, dass ein Sprachmodell genauso wenig eine objektive Entscheidung treffen würde, wie das ein Mensch im Prozess täte.

7.5.3 Intelligentes Dokumentenmanagement

Auch beim Umgang mit Dokumenten in Unternehmen können KI-Agenten sinnvoll unterstützen und automatisieren. Dokumente über verschiedene Eingangskanäle (E-Mail, Scanner, Chat-Nachrichten, Dokumentenablagen, ...) können zentral aufbereitet und abgelegt werden. Der Agent kann dabei z. B. eine Prüfung anhand der Compliance- und Datenschutz-Richtlinien vornehmen und die Unterlagen entsprechend klassifizieren. Auch Einschätzungen des LLM können zusätzlich zum Dokument gespeichert werden und damit kann z. B. eine Prüfung vorgenommen werden, ob es sich um einen Betrugsversuch handeln könnte.

Inhalte könnten wortgetreu und sinngemäß extrahiert und für andere Systeme abgelegt werden. Darauf aufbauend können z. B. Zusammenfassungen geschrieben und für involvierte Mitarbeiter bereitgestellt werden. Damit macht man – wie in einem klassischen Dokumentenmanagementsystem (DMS) – Inhalte durchsuchbar. Was den Agenten vom DMS unterscheidet, sind größere Interaktionsmöglichkeiten und ein Verständnis der Inhalte. Im Rahmen eines DMS könnte beispielsweise auf der Versionskontrolle aufbauend ein Agent regelmäßig einen Änderungsbericht mit einer Einschätzung der Änderungen produzieren. Ändern sich beispielsweise Vertragsdetails, kann der Agent dies entsprechend hervorheben und die Änderungen im Vergleich mit der Vorversion interpretieren.

Weiterführend können KI-Agenten auch bei der Archivierung unterstützen, indem sie nicht länger benötigte Unterlagen identifizieren und zur Archivierung vormerken oder selbstständig archivieren. Neue Versionen von Dokumenten können zum einen im DMS gefunden werden, befinden

sich aber auch vielfach dezentral im Unternehmen abgelegt und können dann automatisch zentral in der neuesten Version aufbereitet werden.

7.5.4 Dynamische IT-Sicherheit

Die Sicherheit der IT-Infrastruktur ist ein Thema, das mit fortschreitender Digitalisierung von Prozessen mehr und mehr an Bedeutung gewinnt. Im Rahmen der IT-Security kommt KI in unterschiedlicher Form bereits seit Längerem zum Einsatz, z. B. zum Erkennen von Verhaltensanomalien oder zur Einstufung von Risikoklassen bei Software und Hardware.

Ein KI-Agent kann aufbauend auf ein solches System die auftretenden Sicherheitsvorfälle autonom behandeln, indem Einschätzungen der Relevanz getroffen und entsprechende Gegenmaßnahmen eingeleitet werden. IT-Systeme können selbstständig vom Rest der Infrastruktur isoliert und analysiert werden, um das Verhalten eines potenziellen Angreifers oder von Schadsoftware zu analysieren.

Darauf aufbauend kann dann automatisiert die Sicherheit der Infrastruktur angepasst werden, z. B. Updates und Patches eingespielt oder Systeme abgeschaltet werden. Die Verteidigungsstrategie des Unternehmens wird je nach Situation angepasst und fortlaufend verbessert. IT-Sicherheit wird damit adaptiver und schneller und kann potenziell besser gegen neue Bedrohungen vorgehen.

Viele dieser Ansätze sind bereits im Kleinen und in einzelnen Bausteinen implementiert. Auch hier gilt, dass ein KI-Agent mit seinen Fähigkeiten vorhandene Ansätze und Systeme kombinieren und steuern kann – er würde aber kein Intrusion Detection System (IDS) ersetzen. Außerdem haben Sprachmodelle und damit auch KI-Agenten Grenzen und Herausforderungen, weshalb ein blindes Vertrauen – vor allem bei einem so wichtigen Thema wie der IT-Sicherheit – auf diese Technologie aktuell nicht ratsam ist.

7.6 Auf einen Blick

- KI-Agenten sind (teil-)autonome Systeme, die mit Hilfe von LLMs Aufgaben vollautomatisch und ohne menschliche Interaktion durchführen können.

- Durch die Nutzung von LLMs werden dadurch vollkommen neue Möglichkeiten der Automatisierung von Aufgaben und Prozessen möglich.
- KI-Agenten können mit Hilfe von Prompts erstellt und gesteuert werden. Zusätzlich benötigen Sie oft Integrationsmöglichkeiten und Schnittstellen zu Drittsystemen oder weiteren LLMs.
- Ein Workflow-System zur Automatisierung von Aufgaben und Prozessen arbeitet einen im Vorfeld fest definierten und deterministischen Pfad ab und erledigt dabei schrittweise die Teilaufgaben, bis das gewünschte Ergebnis erzielt wurde oder ein Fehler auftritt. KI-basierte Agenten hingegen entwickeln zu einer gegebenen Aufgabe autonom einen Lösungsweg und führen diesen dann automatisch aus.
- Damit ein KI-Agent gut arbeiten kann, ist ein sehr präziser Prompt zur Steuerung notwendig. Hier helfen die bereits vorgestellten Tipps und Frameworks aus dem Bereich des Prompt Engineerings.
- KI-Agenten können für unterschiedliche Zwecke im Unternehmen eingesetzt werden. Diese reichen von einer automatisierten Kundenbetreuung mit Hilfe von intelligenten Chatbots über Unterstützungen im Recruiting bis hin zum automatischen Dokumentenmanagement und der Unterstützung im Bereich IT-Sicherheit.

8

Ausblick

Zusammenfassung In diesem Kapitel wird abschließend ein Ausblick auf die Zukunft von KI-Systemen gegeben. Dabei wird klar, dass KI-Funktionalitäten zukünftig in immer mehr Systemen integriert werden und KI schon bald alle Bereiche des Lebens direkt oder indirekt beeinflussen wird. Umso wichtiger ist, sich schon heute mit der Technologie und dem gewinnbringenden Einsatz auseinanderzusetzen.

Die aktuelle Geschwindigkeit in der Entwicklung und Nutzung von KI-Systemen wird in der Form sicher noch zwei bis drei Jahre anhalten. In dieser Zeit werden die Systeme durch immer bessere Trainingsmethoden und der Zuführung immer größerer Datenmengen noch einige große Sprünge vollziehen. Unterschiedlichen Aussagen zufolge werden KI-basierte Systeme bis spätestens 2030 alle Bereiche des Lebens beeinflussen oder sogar steuern; und dies weltweit (z. B. Bundesministerium für Arbeit und Soziales, 2023; Opiela et al., 2018; Straits Research, 2024).

Schon heute sind in viel mehr Systemen KI-Technologien verbaut, als manchen bewusst ist. Auch im Bereich der Forschung werden in den

D. Koch et al., *Prompt Engineering im Unternehmen – eine Einführung*,
https://doi.org/10.1007/978-3-658-47699-1_8

nächsten Jahren große Durchbrüche möglich werden, die ohne KI so nicht erreicht werden könnten.

KI wird in Unternehmen immer mehr Aufgaben übernehmen können. Beginnend bei einfachen, repetitiven Aufgaben werden spätestens mit dem Siegeszug der KI-Agenten, der sich aktuell schon deutlich abzeichnet, auch komplexe Prozesse und Abläufe innerhalb eines Unternehmens aber auch nach außen an den Schnittstellen zu den Kunden, Partnern und Zulieferern vollständig und mit hoher Qualität automatisieren lassen. In Zeiten des Fachkräftemangels ist dies aktuell sicher eine der größten Chancen.

Bei all den positiven Chancen und Möglichkeiten muss sich trotzdem immer wieder bewusst gemacht werden, dass die Technologie, wie jede andere auch, eine Medaille mit zwei Seiten ist. Themen wie Datenschutz, Missbrauch sowie inhaltliche Schutzrechte werden die Gerichte sicher noch über Jahre beschäftigen.

Es ist wichtig, sich jetzt mit der Technologie zu beschäftigen, um nicht den Anschluss zu verlieren. Eine wichtige Kompetenz auf dem Weg der gewinnbringenden KI-Nutzung ist dabei das Prompt Engineering. Auch wenn zukünftige KI-Systeme mit Sicherheit immer besser verstehen werden, was der Anwender von ihm will, wird ein grundlegendes Verständnis der technischen Zusammenhänge und Funktionsweise von KI-Systemen und der Erstellung von guten Anfragen, genau wie in der einleitenden Metapher des Lichtschalters bereits beschrieben, notwendig sein, um sich zukünftig in einer immer schnelleren und technologisierten Welt zurechtzufinden.

Bei all den Möglichkeiten, die uns KI schon jetzt und sicher in der Zukunft bringt, sollte nie vergessen werden, die Ergebnisse kritisch zu hinterfragen und selbst weiterzudenken. Denn bei aller technischen Finesse können KI-Systeme (zumindest auf absehbare Zeit) keine echte Kreativität zeigen. Dies bleibt – vorerst – dem Menschen vorbehalten.

An dieser Stelle schließen wir dieses Fachbuch mit dem berühmten Zitat von Immanuel Kant, der 1784 in der Phase der Aufklärung den folgenden Ausspruch prägte: „Sapere aude!“ – „Habe den Mut, dich deines eigenen Verstandes zu bedienen!“.

Wir wünschen Ihnen abschließend viel Erfolg auf Ihrem persönlichen Weg mit KI. Sollten Sie Rückfragen haben oder Unterstützung im Bereich KI suchen, können Sie sich gerne jederzeit an uns wenden.

9

Kopiervorlagen

Zusammenfassung In diesem Kapitel werden zu den vorgestellten Modellen und Frameworks passende Kopiervorlagen zur Verfügung gestellt. Sie können als täglicher Begleiter beim Prompting z. B. direkt neben der Tastatur platziert werden.

9.1 Das Valisory-Prompt-Qualitätsmodell

Das Valisory-Prompt-Qualitätsmodell		
Kriterium	**Relevanz**	
Faktoren	Fragestellung	Ist der Prompt klar formuliert und fordert er spezifisch die gewünschte Information oder Aktion?
	Spezifität	Sind die Anforderungen und Ziele des Prompts spezifisch genug, um relevante Antworten zu generieren?
	Verständlichkeit	Ist der Prompt einfach und verständlich genug gestaltet, dass das Modell die Anfrage korrekt interpretieren kann?

(Fortsetzung)

D. Koch et al., *Prompt Engineering im Unternehmen – eine Einführung*,
https://doi.org/10.1007/978-3-658-47699-1_9

Das Valisory-Prompt-Qualitätsmodell		
Kriterium	**Kreativität und Originalität**	
Faktoren	Innovationspotenzial	Fordert der Prompt das Modell dazu auf, über allgemeines Wissen hinaus kreative oder innovative Lösungen zu finden?
	Inspiration	Regt der Prompt zu originellen Ideen oder Perspektiven an, die nicht trivial oder offensichtlich sind?
	Variabilität	Bietet der Prompt genügend Spielraum für verschiedene Antworten oder Lösungswege?
Kriterium	**Sprachliche Qualität und Stil**	
Faktoren	Klarheit	Ist der Prompt deutlich und präzise, ohne mehrdeutige oder irreführende Formulierungen?
	Anpassung an Stil	Ist der Stil des Prompts (formell, informell, technisch etc.) angemessen für die Zielsetzung und das Publikum?
	Strukturierung	Ist der Prompt logisch strukturiert und fördert er eine strukturierte Antwort?
Kriterium	**Zielgruppenadäquanz**	
Faktoren	Zielgruppenverständnis	Berücksichtigt der Prompt das Vorwissen, die Interessen und die Bedürfnisse der Zielgruppe?
	Ansprache	Ist die Ansprache im Prompt (Direktheit, Tonfall) passend für die Zielgruppe?
	Nützlichkeit	Ist der durch den Prompt angestrebte Antworttyp für die Zielgruppe praktisch und von Interesse?

9.2 Prompt Frameworks

Prompt Frameworks			
Nr.	Name	Bedeutung	Ablauf
1	RTF	Role-Task-Format	1. Beschreibung der Rolle 2. Beschreibung der Aufgabe 3. Beschreibung der Formatierung
2	TAG	Task-Action-Goal	1. Beschreibung der Aufgabe 2. Beschreibung der Aktion 3. Beschreibung des Ziels

(Fortsetzung)

Prompt Frameworks			
Nr.	Name	Bedeutung	Ablauf
3	BAB	Before-After-Bridge	1. Beschreibung der Ausgangssituation 2. Beschreibung der Zielsituation 3. Entwicklung einer Verbindung zwischen Ist- und Sollzustand
4	CARE	Context-Actions-Result-Example	1. Beschreibung des Kontexts oder der Ausgangssituation 2. Beschreibung der durchzuführenden Aktionen 3. Beschreibung des Ergebnisses 4. Beschreibung eines Beispiels
5	RISE	Role-Input-Steps-Expectations	1. Beschreibung der Rolle 2. Beschreibung der bereitgestellten Daten 3. Beschreibung der notwendigen Schritte 4. Beschreibung der Zielerwartung

9.3 Das Experten-Prompt-Framework

Prompt

Verhalte dich wie ein [Expertenrolle].

Ich benötige [Beschreibung des Ergebnisses].

Du wirst dafür [genaue Aufgabenbeschreibung].

Dabei wirst du [weitere Details].

Bitte [Ausnahmen auflisten].

Entwickle das Ergebnis für [Zielgruppe].

Liefere das Endergebnis als [Format].

Hier ist ein Beispiel: [Beispiel].

9.4 Das Valisory-Prompt-Studio

Prompt

Du bist das Valisory-Prompt-Studio, ein erfahrener Prompt Engineer, der sich auf das Entwickeln effektiver Prompts für Sprachmodelle spezialisiert hat. Du unterstützt den Nutzer dabei, den bestmöglichen Prompt für seine Bedürfnisse zu entwickeln. Der Nutzer kann entweder einen komplett neuen Prompt entwickeln oder einen existierenden verbessern.

Im Folgenden wird das Vorgehen beschrieben, von dem du nicht abweichst.

Grundsätzlich ist das Vorgehen iterativ: durch Fragen und Rückmeldungen wird der jeweilige Prompt Schritt für Schritt weiterentwickelt.

Du führst den entwickelten Prompt zu keinem Zeitpunkt selbst aus oder beantwortest diesen.

Du verhältst dich sprachlich formell und höflich. Die Sprache der Interaktion mit dem Nutzer ist die jeweilige Landessprache (falls nicht anders definiert). Die Sprache des zu erstellenden Prompts ist ebenfalls die Landessprache, sofern der Nutzer dies nicht explizit anders fordert oder einen Prompt zur Verbesserung angibt, der in einer anderen Sprache geschrieben ist.

Die folgende Beschreibung ist in Abschnitte unterteilt (A, B und C). Starte bei Abschnitt A.

A: Dein allgemeines Vorgehen

- Sofern der Nutzer es nicht angegeben hat, frage als aller erstes nach dem Sprachmodell, für den der Prompt entwickelt werden soll.
- Sofern der Nutzer es nicht angegeben hat, frage, als zweites, ob ein neuer Prompt entwickelt (weiter mit B) werden oder ein bestehender Prompt verbessert werden soll (weiter mit C). Fahre dann mit dem jeweiligen Abschnitt unten fort.

B: Für den Fall, dass ein Prompt verbessert werden soll:

1. Frage nach dem Prompt, der verbessert werden soll, sofern der Nutzer ihn noch nicht angegeben hat.
2. Analysiere den Prompt und identifiziere das Ziel. Frage den Nutzer, ob das von dir bestimmte Ziel korrekt ist oder ob der Nutzer ein anderes Ziel definieren möchte.
3. Fahre mit Schritt 3 von Abschnitt C fort.

C: Für den Fall, dass ein Prompt entwickelt werden soll:

1. Frage nach dem Ziel des Prompts. Stelle Rückfragen, sofern die Antworten des Nutzers unklar oder nicht präzise sind.
2. Entwickle eine erste Version des Prompts. Gib diese nicht aus, sondern fahre direkt mit Schritt 3 fort.
3. Analysiere den aktuellen Prompt aus zwei Blickwinkeln:
   a. Als kritischer Prompt-Reviewer
   b. Als kreativer Prompt-Entwickler
4. Gib jeweils immer Folgendes aus:
   - Das Ziel des Prompts
   - Den aktuellen Prompt
   - Vorschläge zur Verbesserung aus Sicht des kritischen Prompt-Reviewers
   - Relevante Fragen (maximal 3) zur Verbesserung des Prompts aus Sicht des Prompt-Entwicklers
5. Der Nutzer wird die Fragen beantworten. Stelle Rückfragen, falls Antworten unklar, unpräzise und nicht sinnvoll sind.
6. Entwickle den Prompt mit den Antworten auf die Fragen und ggf. weiteren Anmerkungen des Nutzers weiter.

Literatur

Bundesministerium für Arbeit und Soziales Denkfabrik. (2023). Die Arbeitswelt der Zukunft: 5 Szenarien zur Mensch-Technik-Interaktion 2030. https://www.denkfabrik-bmas.de/rubriken/wissen/die-arbeitswelt-der-zukunft-5-szenarien-zur-menschtechnik-interaktion-2030. Zugegriffen am 31.01.2025.

Kohne, A., Kleinmanns, P., Rolf, C., & Beck, M. (2020). Chatbots. *Aufbau und Anwendungsmöglichkeiten von autonomen Sprachassistenten*. Springer Vieweg.

Opiela, N., Kar, R. M., Thapa, B., & Weber, M. (2018). Exekutive KI 2030 Vier Zukunftsszenarien für künstliche Intelligenz in der Öffentlichen Verwaltung. Kompetenzzentrum Öffentliche IT. https://www.oeffentlicheit.de/documents/10181/14412/Exekutive+KI+2030+-+Vier+Zukunftsszenarien+f%C3%BCr+K%C3%BCnstliche+Intelligenz+in+der+%C3%B6ffentlichen+Verwaltung. Zugegriffen am 31.01.2025.

Ramlochan, S. (2024). System prompts in large language models. Prompt Engineering & AI Institute. https://promptengineering.org/system-prompts-in-large-language-models/. Zugegriffen am 31.01.2025.

Russel, S. J., & Norvig, P. (1995). *Artificial Intelligence: A modern approach*. Prentice Hall.

D. Koch et al., *Prompt Engineering im Unternehmen – eine Einführung*,
https://doi.org/10.1007/978-3-658-47699-1

Straits Research. (2024). Küstliche Intelligenz (KI) Marktgröße, Anteil und Trends Global Report 2030. https://straitsresearch.com/de/report/artificial-intelligence-market. Zugegriffen am 31.01.2025.
Turing, A. M. (1950). Computing machinery and intelligence. Mind, New Series, *59*(236) (Okt., 1950), Seiten 433–460.
Vaswani, A. (2017). *Attention is all you need*. Advances in Neural Information Processing Systems.

Quellen

Abgaryan, H., Harutyunyan, A., & Cazenave, T. (2024). Llms can schedule. *arXiv preprint arXiv:2408.06993*. https://arxiv.org/pdf/2408.06993v1. Zugegriffen am 31.01.2025.
Han, Z., & Wang, Z. (2024). Rethinking the role-play prompting in mathematical reasoning tasks. In *Proceedings of the 1st Workshop on Efficiency, Security, and Generalization of Multimedia Foundation Models* (S. 13–17). https://dl.acm.org/doi/pdf/10.1145/3688864.3689149. Zugegriffen am 31.01.2025.
Treutlein, J., Choi, D., Betley, J., Marks, S., Anil, C., Grosse, R. B., & Evans, O. (2024). Connecting the dots: Llms can infer and verbalize latent structure from disparate training data. Advances in Neural Information Processing Systems, 37, 140667–140730. https://arxiv.org/pdf/2406.14546. Zugegriffen am 31.01.2025.