#8 TECHARTIKEL

RETRIEVAL AUGMENTED GENERATION

19.04.2024

Andreas Schmid & Dr. Matthias Sommer

Inhalt

Retrieval Augmented Generation - RAG

Generative künstliche Intelligenz-Lösungen mit Unternehmensdaten anreichern

> *„Cheaper, faster, better. […] RAG is like pointing a large language model in the right direction and giving it more specified guidance.“* [1]
>
> (c) Gabriel Skelton,
> Head of Artificial Intelligence Solutions & LinkedIn Top AI Voice

Generative künstliche Intelligenz (**GenKI**) ist eine Form der künstlichen Intelligenz (**KI**), die mithilfe von Algorithmen des tiefen Lernens neue Inhalte wie Texte, Bilder, Code oder Videos generieren kann [2]. Mit dieser Fähigkeit kann GenKI helfen, Empfehlungssysteme, Suchmaschinen oder Dialogsysteme zu optimieren. Zudem können GenKI-Systeme dazu beitragen, Routineaufgaben zu automatisieren oder Ideen für neue Produktdesigns zu generieren. Zu den momentan bekanntesten Vertretern gehören zum Beispiel ChatGPT und DALL-E von OpenAI, GitHub Copilot oder Gemini von Google [3].

Trotz der vielversprechenden Einsatzmöglichkeiten von GenKI existieren zahlreiche Herausforderungen zur erfolgreichen Implementierung eigener Lösungen. Dieser Artikel führt in die GenKI, ihre Problemstellungen und den Lösungsansatz Retrieval Augmented Generation (**RAG**) ein.

1. Einordnung in das Feld der künstlichen Intelligenz

GenKI-Modelle sind ein Teilbereich des tiefen Lernens. Zu den bekanntesten Modellen zählen große Sprachmodelle (**engl. large language models – LLMs**). LLMs sind eine moderne Form von Sprachmodellen, die aus aktuellen Forschungen in den Bereichen natürliche Sprachverarbeitung, tiefes Lernen und GenKI resultieren [4]. *Abbildung 1* veranschaulicht die Einordnung von GenKI und LLMs.

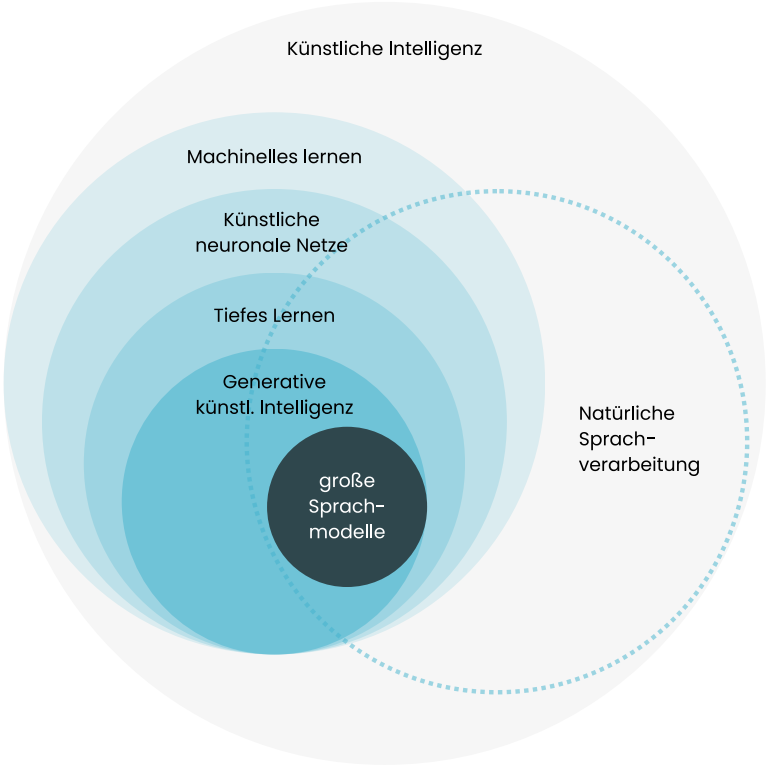

Abbildung 1: Einordnung GenKI und LLMs

2. Funktionsweise

Im Wesentlichen basieren GenKI-Modelle wie LLMs auf statistischen Zusammenhängen, die aus dem Trainingsmaterial erhoben werden. Dabei werden die Eingaben des Benutzers analysiert, in computerlesbare Form gebracht, mithilfe eines künstlichen neuronalen Netzes weiterverarbeitet und das Ergebnis schlussendlich wieder als Text ausgegeben [4]. Die Funktionsweise eines LLMs ist in *Abbildung 2* dargestellt.

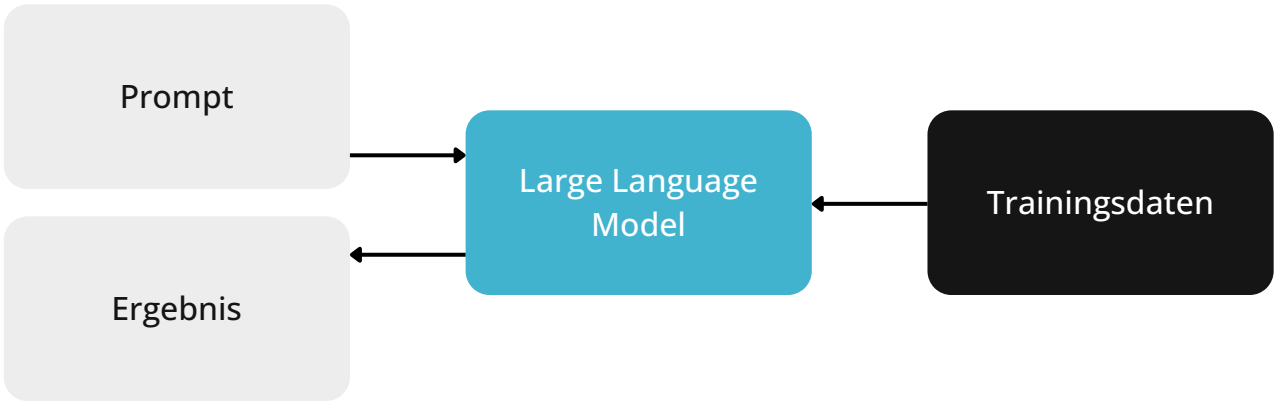

Abbildung 2: Funktionsweise eines LLMs

LLMs werden auf riesigen Datenmengen trainiert und verwenden Milliarden an Parametern, um Aufgaben wie die Beantwortung von Fragen oder die Generierung von Bildern und Videos zu erledigen. Hierfür nimmt das LLM die natürlichsprachliche Benutzereingabe in Form eines Prompts entgegen und erstellt auf Basis seiner Architektur sowie der vortrainierten Wissensbasis eine Ausgabe, welche dem Benutzer als Ergebnis angezeigt werden kann [5].

3. Herausforderungen

Trotz der beeindruckenden Fähigkeiten stoßen GenKI und LLMs an ihre Grenzen. So fehlt es ihnen an detailliertem Domänenwissen, da diese auf einem breiten Spektrum an Daten, meist ohne gezielte Fokussierung auf spezifische Domänen, trainiert werden. Darüber hinaus verfügen sie nur über Wissen bis zu einem bestimmten Stichtag und können keine Aussagen über aktuelle Daten liefern. Diese Einschränkung ist mit dem einmaligen Trainingsprozess verbunden, der äußerst zeit-, kosten- und rechenintensiv ist. Des Weiteren besitzen LLMs kein Know-how über Unternehmensinterna, wobei das hierfür notwendige Training für die meisten Unternehmen viel zu kostspielig ist. Zusätzlich neigen LLMs in ihren Ausgaben zu Halluzinationen. Unter Halluzinationen versteht man die Generierung von Ausgaben, die nicht der Realität entsprechen. Diese resultieren aus der Tatsache, dass LLMs ihre Ausgaben auf Basis der Wahrscheinlichkeiten für das nächste Wort erstellen, ohne dabei den Wahrheitsgehalt zu berücksichtigen [6]. So kann eine Aussage sehr wahrscheinlich, jedoch trotzdem falsch sein.

4. Retrieval Augmented Generation als Lösungsansatz

Um die genannten Herausforderungen zu überwinden, beschäftigen sich neue Ansätze mit der Kombination von LLMs mit aktuellen, proprietären und domänenspezifischen Daten. In Bezug auf Geschäftsprozesse können das etwa Dokumente, Handbücher, Webseiteninhalte oder andere Informationsquellen sein. Ein Architekturkonzept, dass die Integration der eigenen Daten in GenKI-Lösungen erlaubt, ist RAG. RAG ermöglicht dies durch das Abrufen von relevanten Informationen aus zusätzlichen Datenquellen, die anschließend dem LLM zur Verfügung gestellt werden. Auf diese Weise optimiert RAG die Ausgabe, ohne das LLM neu trainieren zu müssen [2].

RAG ist ein kostengünstiger Ansatz und macht LLMs im geschäftlichen Umfeld nutzbarer. Die von RAG genutzten Daten können aus APIs, Datenbanken oder Dokumenten stammen. Dabei muss jedoch darauf geachtet werden, dass der Datenschutz beachtet wird und keine personenbezogenen Daten oder geistiges Eigentum mit einbezogen werden.

5. Architektur und Prozess

Um zu verstehen, wie RAG im Detail funktioniert, führt *Abbildung 3* in den RAG-Prozess und die zugrunde liegende Architektur ein.

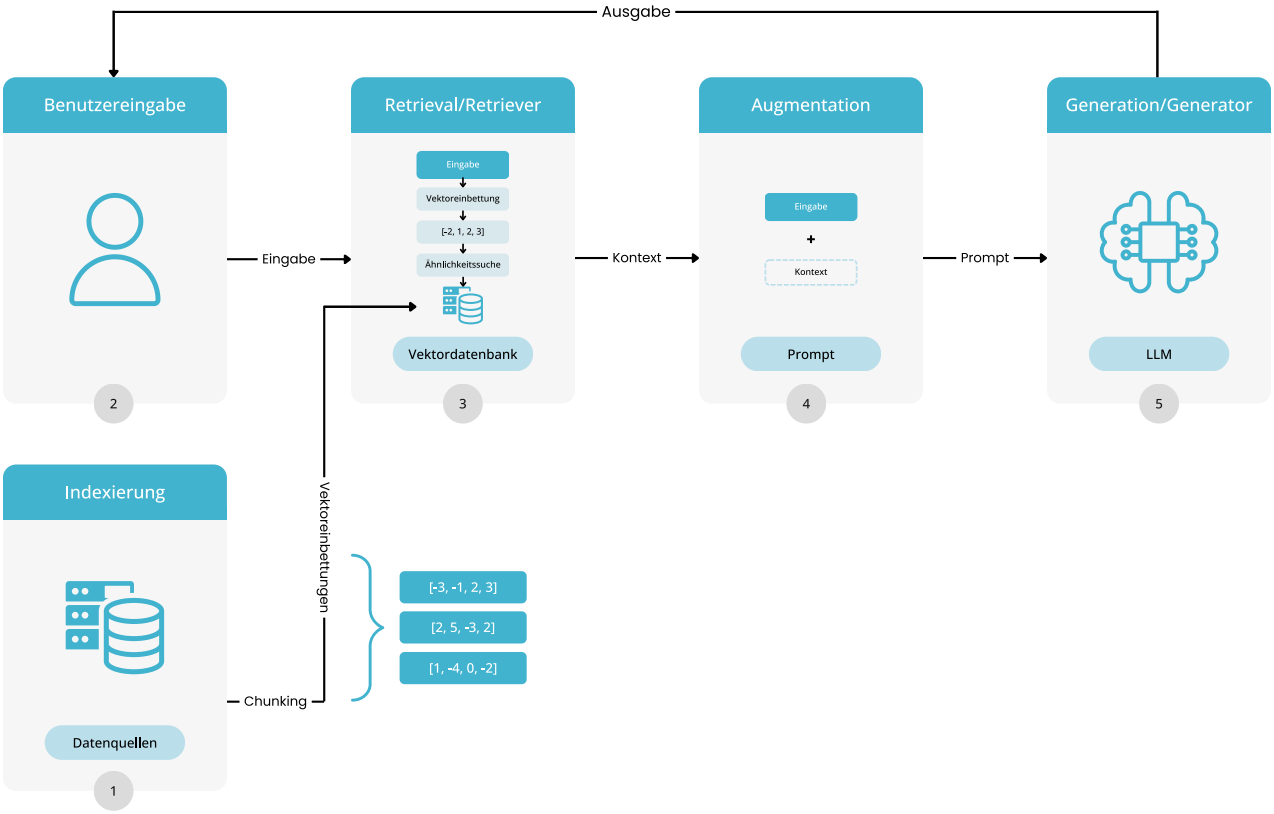

Abbildung 3: RAG-Architektur und Prozess

RAG setzt sich aus fünf Schritten zusammen, welche im Folgenden sukzessive erläutert werden.

Schritt 1: Indexierung

Im ersten Schritt ist eine Indexierung notwendig, um die Daten für GenKI-Modelle nutzbar zu machen. Dazu werden die Daten in kürzere Abschnitte, sog. Chunks, aufgeteilt. Die Chunks werden anschließend in eine numerische Darstellung konvertiert und schließlich in einer Vektordatenbank als Vektoreinbettungen abgelegt. Dabei können Webseiteninhalte (zum Beispiel HTML), Textdokumente (zum Beispiel PDF) oder Mediendateien (zum Beispiel MP3) indiziert werden. Der hochdimensionalen Suchraum der Vektordatenbank eignet sich nun hervorragend für Ähnlichkeitssuchen, da vektorisierte Dokumente mit ähnlicher Thematik näher beieinander liegen als inhaltlich weniger verwandte.

Schritt 2: Benutzereingabe

Der Benutzer gibt eine Aufgabe, Frage oder Beschreibung in natürlicher Sprache in Form eines Prompts ein, um relevante Informationen aus den zuvor gespeicherten Dokumenten zu erhalten. Üblicherweise erfolgt dies über eine Chat-UI wie zum Beispiel bei ChatGPT.

Schritt 3: Retrieval

Nach Erhalt einer Benutzeranfrage wandelt der Retriever die Eingabe des Benutzers ebenfalls in eine Vektoreinbettung um. Optional kann die Eingabe noch erweitert oder umgeschrieben werden, zum Beispiel durch Hinzufügen von Metadaten. Anschließend werden die Ähnlichkeitswerte zwischen der Vektoreinbettung der Eingabe und den vektorisierten Chunks in der Vektordatenbank mithilfe des Verfahrens des nächsten Nachbars berechnet und priorisiert. Die k-ähnlichsten Chunks werden daraufhin geladen und als Kontext weitergereicht. Auch hier können optional weitere Optimierungsschritte erfolgen, zum Beispiel das Zusammenfassen der erhaltenen Dokumente oder Hervorhebungen der wichtigsten Aspekte.

Schritt 4: Augmentation

Der generierte Kontext wird mit der Eingabe des Benutzers zu einem Prompt kombiniert, welcher anschließend an einen Generator, in der Regel ein LLM, gesendet werden kann.

Schritt 5: Generation

Der Prompt wird an den Generator gesendet, welcher die mittels Retrieval erhaltenen Informationen sowie die im Vortraining gelernten Fähigkeiten nutzt und kombiniert, um eine Antwort zu formulieren und diese dem Benutzer zur Verfügung zu stellen.

6. Vorteile

Die Integration von RAG in GenKI-Lösungen ist mit einer Reihe von Vorteilen verbunden:

Vorteil	Beschreibung
Aktualisierung der Wissensbasis	Die Vektordatenbank kann jederzeit um neue Inhalte erweitert werden, wodurch der Zugriff zu aktuellen Informationen gewährleistet wird.
Transparenz	Antworten können zu den Datenquellen zurückverfolgt werden, wodurch eine hohe Transparenz gegeben ist.
Vermeidung von Halluzinationen	Da Antworten ausschließlich aus den eigenen Dokumenten generiert werden, sind Halluzinationen seltener. Zudem kann über vordefinierte Prompts sichergestellt werden, dass das LLM standardisierte Antworten liefert, wenn keine Informationen gefunden werden.
Kostengünstige Alternative	Da das zugrundeliegende LLM nicht neu trainiert werden muss, sobald neue Daten hinzugefügt werden, können erhebliche Kosten eingespart werden.

7. Fazit und Ausblick

RAG ist ein vielversprechender Ansatz zur Integration von Unternehmensdaten in GenKI-Lösungen. Zur Implementierung von RAG bieten sich zahlreiche Möglichkeiten an, darunter beispielsweise standardisierte Cloud-Lösungen von Microsoft Azure oder Amazon Web Services, Individuallösungen mithilfe von Open-Source-Frameworks wie LangChain und Semantic Kernel oder eine Kombination aus den genannten Technologien [7,8].

Die aktuelle Forschung im Bereich der GenKI geht momentan wieder in Richtung kleinere und auf spezifische Anwendungsfälle spezialisierte Sprachmodelle, beispielsweise Code Llama für Quellcodegenerierung [9]. Diese sogenannten kleinen Sprachmodelle (**engl. small language models, SLMs**) weisen dank Innovationen in Bereichen wie verbesserter Datenaufbereitung und Modellskalierung ähnlich gute Resultate wie deutlich größere LLMs auf [10]. Hier zeigt sich erneut, dass die Qualität der einbezogenen Daten eine entscheidende Rolle spielt.

Abschließend bleibt somit festzuhalten:

1. Die Integration von Unternehmensdaten in GenKI-Lösungen kann mit RAG kostengünstig umgesetzt werden.
2. RAG kann bestehende Geschäftsprozesse optimieren und zu Effizienzsteigerung führen.
3. Der Erfolg von RAG-Lösungen hängt maßgeblich von der Qualität der Daten ab.
4. Die Qualität der mittels RAG erzeugten Ausgaben sollte überprüft und gemessen werden.

Quellen

1. **What is RAG and how will it impact your adoption of AI?** - abgerufen am 04.04.2024
2. Zhao P., Zhang H., Yu Q., Wang Z., Geng Y., Fu F., Yang L., Zhang W., Cui B. (2024): Retrieval-Augmented Generation for AI-Generated Content: A Survey.
3. **Top 10 Generative AI Tools You Should Know for 2024** - abgerufen am 04.04.2024
4. Amaratunga, T. (2023): Understanding Large Language Models
5. **Large language models: The basics and their applications** - abgerufen am 04.04.2024
6. **Was ist Retrieval Augmented Generation?** - abgerufen am 04.04.2024
7. **Retrieval Augmented Generation (RAG) in Azure KI Search** - abgerufen am 04.04.2024
8. **Harnessing the power of Large Language Models: A comparative overview of LangChain, Semantic Kernel, AutoGen and more** - abgerufen am 04.04.2024
9. **Introducing Code Llama, a state-of-the-art large language model for coding** - abgerufen am 03.04.2024
10. **Phi-2: The surprising power of small language models** - abgerufen am 01.04.2024