rms. Chatbot

Anwendungsgebiete und Technologie

RMS GmbH

Inhaltsverzeichnis

Historie

Datum	Autor	
12.05.2025	Mk	Version 1.0
16.06.2025	SF	Version 1.1
15.09.2025	OM	Version 1.2

Einführung

Der rms. Chatbot stellt Besuchern eine Möglichkeit zur Verfügung, Antworten auf Fragen zu unternehmens- bzw. themenspezifischen Inhalten zu erhalten.

Im Vergleich zu einer klassischen Suche können die Eingaben echte Fragen sein und sind nicht auf Stichworte beschränkt, die nur Ergebnisse liefern, wenn die exakten Wörter oder definierte Synonyme in den zu durchsuchenden Daten enthalten sind.

Die zugrunde liegenden Daten können komplette Webseiten, PDFs, Word-Dateien, Textdokumente oder Exporte aus Datenbanken sein. Auch eine Kombination aus mehreren Quellen ist möglich.

Befinden sich die Antworten auf Webseiten oder in PDFs, können die entsprechenden URLs als Teil der Antwort angezeigt werden.

Mit dem Einsatz von Chatbots, die auf individuellen Daten basieren, kann die Wissensbasis eines Chatbots um beliebige eigene Inhalte erweitert werden.

Mögliche Anwendungsgebiete

- **Chat auf einer Webseite**: Der Chatbot beantwortet Fragen zu allen Inhalten, die auf einer Webseite zu finden sind. Diese können bei Bedarf durch PDF-Dateien oder weitere Inhalte angereichert werden. Die Antworten können Links zu Inhaltsseiten (Produkte, Angebote, Hilfe, ...) enthalten.
- **Suche über firmeninterne Daten**: Über einen Chatbot werden firmeninterne Daten für Mitarbeitende einfach zugänglich. Die manuelle Suche in Dokumenten, Wissensdatenbanken, Supportdatenbanken etc. entfällt. Neue und bestehende Mitarbeitende können viel schneller auf bestehende Informationen zugreifen, der Wissenstransfer wird effizienter. Die zugrundeliegende Technologie des Chatbots, das Large Language Model (LLM) kann bei Bedarf auf einem **firmeninternen Server** betrieben werden, wodurch kein Datentransfer auf externe Systeme stattfindet.
- **Support**: Kundenanfragen können mit bestehenden Supportdatenbanken abgeglichen und Antworten basierend auf bisherigen Ergebnissen vorformuliert werden.
- **Mehrsprachigkeit in Echtzeit**: Bei Bedarf können Anfragen in anderen Sprachen beantwortet werden, auch wenn die Inhalte einer Webseite nur in Deutsch verfügbar sind.

Grenzen von KI-Modellen ohne individuell festgelegte Datenbasis

Öffentlich zugängliche Chatbots wie ChatGPT, Gemini oder Perplexity haben gravierende Nachteile, wenn es um die Verarbeitung von individuellen Anfragen oder Anfragen zu aktuellen Themen geht: Sie können diese entweder gar nicht beantworten, da firmeninterne Daten nicht Teil ihrer Trainingsdaten waren oder diese zum Zeitpunkt des Trainings nicht zur Verfügung standen (z. B. aktuelle News). Oder sie beantworten die Fragen basierend auf allgemeinem Wissen, was zu Abweichungen oder Aussagen führen kann, die nicht direkt mit

den eigenen Inhalten korrelieren. Es werden ggf. andere Markennamen genannt, Links führen auf Seiten der Konkurrenz etc.

Fragen zu firmeninternen Daten (Supportdatenbanken, interne Dokumente, Forschungsdatenbanken etc.) können in der Regel überhaupt nicht beantwortet werden.

Chatbots für individuelle Inhalte

Wird ein Chatbot für eigene Inhalte benötigt, gibt es u.a. die folgenden Möglichkeiten:

1. Training eines eigenen Sprachmodels

Stehen genügend Daten, viel Zeit sowie leistungsfähige Hardware zur Verfügung, kann ein eigenes Sprachmodell trainiert bzw. ein existierendes Modell mittels Fine-Tuning erweitert werden.

Hierfür sind sehr viele Daten notwendig (idealerweise Millionen von entsprechend aufbereiteten Datensätzen). Zudem wird extrem leistungsfähige Hardware (KI-Beschleuniger), entsprechendes Budget und viel Zeit benötigt. Gibt es ein Update der Daten, muss der komplette Trainingsprozess erneut gestartet werden.

2. Erweiterung des Kontextes eines existierenden Sprachmodels

Die zweite Möglichkeit ist der Einsatz eines existierenden Sprachmodells ohne explizites Training. Hier werden die Daten so vorbereitet, dass sie von Sprachmodellen verstanden und verarbeitet werden können. Jede Frage wird vor der Verarbeitung durch ein LLM mit den verfügbaren Daten abgeglichen (semantische Bedeutung) und die relevanten Inhalte werden zusammen mit einem Prompt (Aufgabenstellung) sowie der Frage an das Sprachmodell übergeben. Dieses verarbeitet die Daten und generiert eine Antwort.

Gibt es ein Update der Daten, wird die entsprechende Datenbank aktualisiert und das LLM hat zeitnah Zugriff auf die neuen Informationen. So ist es möglich, in kurzen Intervallen und ohne hohe Mehrkosten neue oder aktualisierte Daten in das System zu integrieren.

Dieser Vorgang wird als Retrieval Augmented Generation (RAG) bezeichnet und im Folgenden detailliert beschrieben. Der RMS Chatbot basiert auf diesem Verfahren.

Zusammenfassung

Zusammengefasst ist RAG oft die bevorzugte Methode für Unternehmen, die schnell, sicher und kosteneffizient aktuelle Informationen in LLMs integrieren wollen, während Fine-Tuning tiefere Modellanpassungen für spezielle Domänen ermöglicht, aber mit höherem Aufwand verbunden ist.

Retrieval Augmented Generation (RAG)

Hier werden zuerst alle verfügbaren Daten in sogenannte numerische Vektoren umgewandelt, die die semantische Bedeutung der Inhalte erfassen und in dafür geeigneten Datenbanken, sogenannten Vektordatenbanken, gespeichert. Beispiele hierfür sind z. B. Chroma, Pinecone, Faiss, Elasticsearch oder Milvus.

Die Vektordatenbanken ermöglichen eine semantische Ähnlichkeitssuche, das heißt, sie finden Inhalte, die inhaltlich oder kontextuell ähnlich zur Suchanfrage sind, auch wenn die exakten Wörter nicht übereinstimmen. Dies ist vergleichbar mit Anwendungen wie der Suche nach ähnlichen Bildern oder der Erkennung eines Liedes in Shazam durch Vorsingen.

Im RAG-Prozess wird bei einer Nutzeranfrage die Vektordatenbank nach den relevantesten Dokumenten durchsucht. Diese Dokumente werden dann als Kontext verwendet, um die Antwort des LLM informativer und genauer zu machen. Dadurch werden die Einschränkungen von LLMs in Bezug auf Wissensaktualität und Quellenangaben deutlich reduziert.

Ein RAG-System kann bei Bedarf komplett ohne die Nutzung externer LLM Provider (z. B. OpenAI, Gemini oder Perplexity) auf einem eigenen Server betrieben werden.

Vorteile RAG

Es gibt viele Vorteile von RAG gegenüber echten Trainings oder Fine-Tuning:

- **Sicherheit & Datenschutz**: Bei RAG verbleiben firmeneigene Daten in der gesicherten Datenbankumgebung, was strengere Zugriffskontrollen ermöglicht. Beim Fine-Tuning werden die Daten ins Modelltraining integriert, was potenziell zu breiterem Datenzugriff führen kann
- **Kosten- und Ressourceneffizienz**: Fine-Tuning ist rechenintensiv und zeitaufwendig, da es umfangreiche Trainingsphasen und Datenaufbereitung erfordert. RAG vermeidet diesen Trainingsaufwand, indem es die Daten dynamisch abruft, was kostengünstiger und schneller skalierbar ist
- **Aktualität und Zuverlässigkeit**: RAG kann stets auf aktuelle Daten zugreifen und somit präzisere und vertrauenswürdigere Antworten liefern. Fine-Tuning basiert auf einem statischen Trainingsdatensatz und kann veraltetes Wissen enthalten
- **Flexibilität**: RAG eignet sich besonders gut für Anwendungen, bei denen sich die zugrunde liegenden Daten häufig ändern oder erweitert werden, ohne dass das Modell neu trainiert werden muss. Fine-Tuning ist besser für sehr spezifische, eng umrissene Aufgaben, erfordert aber bei jeder Datenänderungen ein erneutes Training

Ablauf RAG-Suche

Der Ablauf einer KI-gestützten Beantwortung von Fragen mittels Vektordatenbanken (RAG) ist wie folgt:

1. **Eingabe der Frage**
   Der Besucher gibt eine Frage oder Suchanfrage in das System ein.
2. **Vektorisierung der Frage**
   Die Frage wird durch ein sogenanntes Embedding-Modell in einen Vektor umgewandelt, der die semantische Bedeutung der Frage repräsentiert.
3. **Semantische Ähnlichkeitssuche**
   Die Vektordatenbank wird mit dem Fragevektor abgefragt, um die semantisch ähnlichsten Dokumente oder Textpassagen zu finden.
4. **Ausgabe der relevantesten Ergebnisse**
   Die X relevantesten Dokumente (z.B. Top 5 oder Top 10) werden aus der Vektordatenbank zurückgegeben.
5. **Kombination von Frage, Kontext und Prompt**
   Die ursprüngliche Frage, die gefundenen Dokumente und ein Prompt (Aufgabenstellung) werden an das Sprachmodell (LLM) übergeben.
6. **Verarbeitung durch das Sprachmodell**
   Das LLM generiert basierend auf der Frage und den Kontextinformationen eine Antwort.
7. **Ausgabe der Antwort**
   Die generierte Antwort wird dem Nutzer präsentiert. Bei Bedarf werden Links zu relevanten Quellen / Seiten hinzugefügt.

Sprachmodelle und Datenschutz

Mittels RAG kann das Datenmanagement zu 100 % gesteuert werden. Es sind beispielsweise die folgenden Szenarien abbildbar. Es ist auch möglich, eigene API-keys zu verwenden.

API - OpenAI

- Speicherung aller vektorisierten Daten lokal in einer Vektordatenbank
- Verarbeitung der Daten (relevante Ergebnisse, Frage, Prompt) innerhalb eines Sprachmodels von OpenAI (GPT4, GPT4-mini, GPT4-nano)
- Vorteil: schnell, wenig komplex, günstig (nur nutzungsabhängige Kosten), leistungsfähig, beliebig skalierbar, großes Kontextfenster
- Nachteil: ggf. Datenschutz, Transfer sensibler Daten

API - Gemini

- Speicherung aller vektorisierten Daten lokal in einer Vektordatenbank
- Verarbeitung der Daten (relevante Ergebnisse, Frage, Prompt) innerhalb eines Sprachmodels von Google (Gemini Pro, Gemini Flash, Gemini Flash-Lite)
- Vorteil: schnell, wenig komplex, günstig (nur nutzungsabhängige Kosten), leistungsfähig, beliebig skalierbar, großes Kontextfenster
- Nachteil: ggf. Datenschutz, Transfer sensibler Daten

API - Ionos (Opensource-LLMs, Hosting in Deutschland)

- Speicherung aller vektorisierten Daten lokal in einer Vektordatenbank
- Verarbeitung der Daten (relevante Ergebnisse, Frage, Prompt) innerhalb eines **Opensource-Sprachmodels**, welches von Ionos in Deutschland gehostet wird (Llama 3.1 8B Instruct, Mistral Small 24B Instruct, ...)
- Vorteil: Datenschutz, schnell, etwas komplexer, günstig (nur nutzungsabhängige Kosten), leistungsfähig, beliebig skalierbar

Dedizierter KI-Server – Z.B. Hetzner (Opensource-LLMs, Hosting in Deutschland)

- Speicherung aller vektorisierten Daten lokal in einer Vektordatenbank
- Verarbeitung der Daten (relevante Ergebnisse, Frage, Prompt) innerhalb eines **Opensource-Sprachmodels**, welches auf einem dedizierten Server betrieben wird (phi4, Llama 3.1 8B Instruct, Mistral Small 24B Instruct, bzw. alle mit ollama kompatiblen Modelle https://ollama.com/search)
- Vorteil: Datenschutz
- Nachteil: hohe Fixkosten, komplexes Setup, höherer Aufwand für Administration und Betrieb, schlecht skalierbar

Lokaler KI-Server – (Opensource-LLMs, Hosting in eigenem RZ / Netzwerk)

- Speicherung aller vektorisierten Daten lokal in einer Vektordatenbank
- Verarbeitung der Daten (relevante Ergebnisse, Frage, Prompt) innerhalb eines **Opensource-Sprachmodels**, welches auf einem eigenen Server betrieben wird (Llama 3.1 8B Instruct, Mistral Small 24B Instruct, bzw. alle mit ollama kompatiblen Modelle https://ollama.com/search)
- Vorteil: Datenschutz
- Nachteil: hohe Investitionskosten, komplexes Setup, hoher Aufwand für Administration und Betrieb, schlecht skalierbar

Provider	LLMs	Vorteile	Nachteile	Anwendungsgebiet
OpenAI	GPT (pro, mini, nano)	schnell, wenig komplex günstig (nutzungsabhängige Kosten), leistungsfähig, großes Kontextfenster, beliebig skalierbar	ggf. Datenschutz, Transfer sensibler Daten	Chatbot für öffentliche Daten (Webseite)
Google	Gemini (pro, flash, flash-lite)	schnell, wenig komplex günstig (nutzungsabhängige Kosten), leistungsfähig, großes Kontextfenster, beliebig skalierbar	ggf. Datenschutz, Transfer sensibler Daten	Chatbot für öffentliche Daten (Webseite)
Anthropic	Claude (Haiku, Sonnet)	schnell, wenig komplex teuer (nutzungsabhängige Kosten), leistungsfähig, beliebig skalierbar	ggf. Datenschutz, Transfer sensibler Daten	
Ionos	Opensource (Llama 3.1 8B Instruct, Mistral Small 24B Instruct)	Datenschutz (Hosting in DE), schnell, etwas komplexer, günstig (nutzungsabhängige Kosten), leistungsfähig, beliebig skalierbar		Chatbot für öffentliche Daten (Webseite) Chatbot für interne Daten
Hetzner	Opensource (alle mit ollama kompatiblen Modelle wie phi4, deepseek, llama, gwen, gemma, Quantisierung abhängig von Hardware, https://ollama.com/search)	Datenschutz (Hosting in DE auf eigenem Server)	hohe Fixkosten, komplexes Setup, höherer Aufwand für Administration und Betrieb, schlecht skalierbar	Chatbot für interne Daten
Inhouse	Opensource (alle mit ollama kompatiblen Modelle wie phi4, deepseek, llama, gwen, gemma, Quantisierung abhängig von Hardware, https://ollama.com/search)	Datenschutz (Hosting auf eigener Hardware in eigenem RZ)	hohe Investitionskosten, komplexes Setup, hoher Aufwand für Administration und Betrieb, schlecht skalierbar	Chatbot für interne Daten

Featureliste

Sprachmodelle, DSGVO und Hosting

- Einsatz beliebiger Sprachmodelle wie openAI (gpt mini/nano), Gemini (flash/flash-lite), Opensource (Meta-Llama 3.1, Teuken, Mistral) z.B. über Ionos AI-Hub oder selber gehostet auf entsprechenden KI-Servern. Umstellung des Sprachmodells „on the fly“ ist möglich.
- DSGVO: Das komplette Hosting des Chatbots und der Vektor-Datenbank erfolgt in Deutschland (aktuell bei Hetzner)
- DSGVO: Bei Bedarf können auch die Sprachmodelle in Deutschland gehostet werden (z.B. bei Ionos oder auf einem eigenen KI-Server bei Hetzner)

Wissensbasis und Sprachen

- Konvertierung von beliebigen strukturierten Textinhalten in Vektor-Embeddings (PDF, Word, Websitescraper, Confluence, …)
- Schnittstellen
  - Confluence
  - MS Sharepoint
  - Google Drive
  - Jobware
- Backend zur strukturierten Verwaltung von PDFs, Wordfiles, Sitemaps, …
- Cronjob für regelmäßige Updates von z.B. Webseiten (inkl. Unterseiten)
- Mehrsprachigkeit basierend auf Webseitensprachen
- Automatische Übersetzung von strukturierten Textinhalten mittels deepL
- Vielfältige Konfigurationsoptionen wie Prompting, Auswahl der Sprachmodelle, Optionen für die Vektor-DB, Chunk-Size, Cron-Intervall, diverse PDF-Extraktionsmodelle inkl. Bildanalysen (Aktuell in der Entwicklung) uvm.

Frontend und Schnittstellen

- Einfache Integration per Widget inkl. grafischer Editor für die Darstellung des Widgets
- Übersicht der letzten Chatanfragen
- REST-API für die Integration in bestehende Systeme (z.B. Lernplattform, CMS, Intranet, Telefonanlage)
- APIs zu externen Datenbanken wie Jobplattformen, Weiterbildungsportalen, Terminvereinbarungen, Kalender, …
- Zugriffskontrolle für den Chatbot (API) per JWT-Authentication oder OpenID