SecuredGPT

Pilotbetrieb eines On-Premises Large Language Model im Wissensmanagement

Bundesministerium für Finanzen,
Österreichische Forschungsförderungsgesellschaft mbH,
Bundesministerium für Inneres,
Cortecs GmbH

Mai 2024 - April 2025

Kurzbeschreibung

Strafverfolgungsbehörden nutzen interne Wissensdatenbanken, um Informationen und Wissen effizient zu speichern, zu organisieren und für den späteren Zugriff bereitzuhalten. Ein Beispiel hierfür ist der Kriminalistische Leitfaden (KLF) des Bundesministerium für Inneres (BMI), der unter anderem Handlungsanweisungen für den Umgang mit Cyber-Crime-Delikten und interne Dienstanweisungen enthält. Das primäre Ziel solcher Wissensdatenbanken ist der Wissensaustausch innerhalb der Organisation und die Steigerung der Effizienz, indem Mitarbeiter schnell und einfach auf relevante Informationen zugreifen können. Das Projekt SecuredGPT hat zum Ziel, behördliches Wissensmanagement mit den Möglichkeiten von Large Language Models auszustatten, um eine effiziente Verarbeitung dieser Daten zu gewährleisten. Dieser Bericht beschreibt die durchgeführten Tätigkeiten und Erkenntnisse, die im Projektverlauf erarbeitet wurden.

Finanziert im Sicherheitsforschungs-Förderprogramm KIRAS des Bundesministeriums für Finanzen.

Bundesministerium Finanzen
KIRAS Sicherheitsforschung
FFG Forschung wirkt.
K-PASS

MARKTANALYSE

Seit der Veröffentlichung von ChatGPT[1] ist das wirtschaftliche und wissenschaftliche Interesse an großen Sprachmodellen (LLMs) enorm gestiegen. Immer häufiger werden neue Modelle vorgestellt, die ihre Vorgänger in unterschiedlichen Bereichen übertreffen. Diese Verbesserungen betreffen die Länge der Konversation, die Qualität der Antworten oder die Antwortgeschwindigkeit. Außerdem erscheinen vermehrt Modelle, die zusätzlich zu Text auch Bilder und Videos als Eingabe verarbeiten können. Geschlossene Modelle wie OpenAIs ChatGPT oder Googles Gemini[2] nehmen oft eine Vorreiterrolle ein, Open-source Modelle können die Qualität aber oft mit wenigen Monaten Verzögerung nachbilden.

Da sich diese Technologien noch in einer frühen Phase befinden, sind gegebene Schwächen zu berücksichtigen. Dazu gehören die Länge der Konversation (Kontextlänge), das Halluzinationsrisiko oder die Mehrsprachigkeit. Eine Marktanalyse soll den Status-Quo darstellen (Stand Juni 2024) und dem Bundesministerium für Inneres (BMI) dienen, sich in dieser schnelllebigen Landschaft einen Überblick zu verschaffen. Insbesondere dient diese Analyse als Entscheidungsgrundlage für die Modellwahl im gegenständlichen Projekt *SecuredGPT*. In der Marktanalyse vergleichen wir open-source Sprachmodelle (Stand Juni 2024) mit einem besonderen Fokus auf deren Deutsch-Qualitäten.

Qualitätskriterien

Angesichts der großen Anzahl an Modellen (> 900.000 auf Huggingface[3]) ist es wichtig, frühzeitig zu entscheiden, welches LLM für einen bestimmten Anwendungsfall geeignet ist. Da es praktisch unmöglich ist, alle Modelle zu testen, wird ein Prozess vorgestellt, um eine Vorauswahl möglichst effizient zu treffen. Diese Vorauswahl erleichtert die Entscheidung, welche Modelle in Betracht gezogen werden und somit welches Modell letztendlich eingesetzt wird. Zur Vorauswahl sollen leicht zugängliche Merkmale herangezogen werden.

Modellgröße

Die Größe eines Modells, gemessen in der Anzahl der Parameter, variiert stark – von kleinen Modellen mit 100 Millionen Parametern bis hin zu Modellen mit 500 Milliarden Parametern und mehr. Die Modellgröße beeinflusst die Balance zwischen Qualität und Kosten. Kleinere Modelle sind oft für einfache Aufgaben geeignet[4], während Modelle ab 7 Milliarden Parametern für vielfältige Anwendungsfälle eingesetzt werden können. Größere Modelle bieten zwar oft eine bessere Qualität, erfordern jedoch mehr Hardware-Ressourcen und sind langsamer. Für unsere Analyse beschränken wir uns daher auf Modelle zwischen 7 und 70 Milliarden Parametern, da diese einen guten Kompromiss zwischen Qualität und Geschwindigkeit bieten.

---

[1] https://openai.com/chatgpt/
[2] https://gemini.google.com/
[3] https://huggingface.co/models?pipeline_tag=text-generation&sort=trending
[4] https://news.microsoft.com/source/features/ai/the-phi-3-small-language-models-with-big-potential/

Kontextlänge

Die Kontextlänge eines Modells gibt an, wieviele Daten man verarbeiten kann. Die Kontextlänge von LLMs wuchs zuletzt rapide an[5] und wir gehen davon aus, dass sich diese weiter vergrößern wird. Größere Kontextlängen ermöglichen es, mehr Informationen in die Datenverarbeitung einzubeziehen. Mehr Informationen führen jedoch nicht automatisch zu besseren Antworten[6]. Es ist wichtig zu bestimmen, welche Mindestkontextlänge für den jeweiligen Anwendungsfall benötigt wird. Mit Vektordatenbanken und Information-Retrieval-Methoden können auch Modelle mit kürzerer Kontextlänge effektiv mit großen Datenmengen umgehen. Modelle mit einer Kontextlänge von weniger als 7.000 Tokens werden für den gegenständlichen Anwendungsfall als ungeeignet erachtet und nicht in die Analyse einbezogen.

Instruction-tuning

Neben den Basisversionen von Modellen gibt es oft spezialisierte, *feingetunte* Varianten, die für bestimmte Anwendungsfälle wie Chat oder die Befolgung von Anweisungen optimiert sind. Sog. Instruct-Modelle, die verschiedene Anweisungen befolgen können, sind vielseitig einsetzbar und übertreffen Basisversionen in Chat-Anwendungen. Auf Grund der vielseitigen Einsetzbarkeit werden daher ausschließlich Instruct-Modelle herangezogen.

Herausgeber

Der Herausgeber eines Modells spielt eine wichtige Rolle, insbesondere wenn ein Modell langfristig in eine Systeminfrastruktur integriert werden soll. Aspekte wie Reputation und regelmäßige Updates sind hier entscheidend. Man wird hier künftig ein Augenmerk auf jene Herausgeber legen, die den Anforderungen des europäischen AI-Acts gerecht werden. Modelle von großen Firmen wie Facebook, Google, Microsoft oder Mistral sind oft die Beliebtesten und dienen häufig als Basis zum Fine-tuning für kleinere Unternehmen, Gruppen oder Einzelpersonen, um spezifische Anwendungsfälle besser abzudecken.

Mehrsprachigkeit

Ein wichtiger Aspekt für dieses Projekt ist die Mehrsprachigkeit der Modelle. Da die meisten Modelle überwiegend auf englischen Daten trainiert sind (~95% der Trainingsdaten von Llama-3 sind englischsprachig), schneiden sie bei anderen Sprachen oft schlechter ab. Bei der Auswahl der Modelle wurde besonders auf die deutsche Sprache geachtet.

Lizenz

Ein oft übersehener, aber wichtiger Aspekt ist die Lizenz eines Modells. Viele Herausgeber verwenden eigene Lizenzen statt standardisierter open-source Lizenzen, was zu unterschiedlichen Nutzungsbedingungen führen kann und die Verwendung eines Modells einschränken kann.

---

[5] https://cobusgreyling.medium.com/rag-llm-context-size-6728a2f44beb
[6] https://www.pinecone.io/blog/why-use-retrieval-instead-of-larger-context/

Vorauswahl

Im open-source Bereich haben sich aktuell (Juni 2024) unter Berücksichtigung der oben erwähnten Gesichtspunkte vor allem die Modelle von Facebook (Llama-3) und Mistral (Mistral und Mixtral) etabliert. Sie sind auf eine Vielzahl von Aufgaben trainiert (Instruction-tuned) und können daher für unterschiedliche Anwendungsbereiche eingesetzt werden. Da in diesem Projekt ein besonderes Augenmerk auf die deutsche Sprache gelegt wird, wurden Modelle evaluiert, die sich speziell auf die Deutschsprachigkeit konzentrieren. Besonders hervorzuheben sind hier die Modelle von VAGOSolutions (SauerkrautLM, abk. SKLM). In der folgenden Tabelle wird die Vorauswahl dargestellt.

	Llama-3 8B Instruct[7]	**Llama-3 70B Instruct**[8]	**SKLM Llama-3 70B Instruct**[9]	**Mistral 7B Instruct v0.3**[10]	**Mixtral 8x7B Instruct v0.1**[11]	**SKLM Mixtral 8x7B Instruct**[12]
Herausgeber	Facebook	Facebook	VAGOSolutions	Mistral	Mistral	VAGOSolutions
Größe	8 Milliarden	70 Milliarden	70 Milliarden	7 Milliarden	56 Milliarden	56 Milliarden
Kontextlänge	8.192	8.192	8.192	32.768	32.768	32.768
Lizenz	Llama	Llama	Llama	Apache 2.0	Apache 2.0	Apache 2.0

Evaluierung

In diesem Abschnitt präsentieren und vergleichen wir die Ergebnisse der qualitativen Analyse der oben dargestellten Vorauswahl anhand öffentlicher Testdaten. Zur Bewertung der Qualität sowie zur Bewertung diverser ethischer und sicherheitsrelevanter Aspekte wurden öffentliche Evaluierungs-Datensätze und deren deutsche Varianten herangezogen.

Qualitätsbewertung

Die Evaluierungsdatensätze ARC, Hellaswag und MMLU dienen der umfassenden Bewertung der Leistungsfähigkeit von KI-Systemen.

[7] https://huggingface.co/meta-llama/Meta-Llama-3-8B-Instruct
[8] https://huggingface.co/meta-llama/Meta-Llama-3-70B-Instruct
[9] https://huggingface.co/VAGOsolutions/Llama-3-SauerkrautLM-70b-Instruct
[10] https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.3
[11] https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.3
[12] https://huggingface.co/VAGOsolutions/SauerkrautLM-Mixtral-8x7B-Instruct

- Der ARC-Datensatz[13] (AI2 Reasoning Challenge) testet die Fähigkeit von KI-Modellen, wissenschaftliche Fragen zu verstehen und zu beantworten, wobei ein Fokus auf logischem Denken und Schlussfolgern liegt.
- Hellaswag[14] bewertet die Fähigkeit von KI-Modellen, den Ausgang von alltäglichen Szenarien vorherzusagen, und misst damit deren "weltliches Verständnis".
- MMLU[15] (Massive Multitask Language Understanding) prüft das Wissen und Verstehen von KI-Systemen in über 50 verschiedenen Aufgabenbereichen, von Mathematik und Naturwissenschaften bis hin zu Geisteswissenschaften und sozialen Studien.

Wir nutzen **die deutsche Variante dieser Datensätze** (ARC DE, Hellaswag DE, MMLU DE), um die Modelle zusätzlich auf die deutsche Sprache zu testen. Große Evaluierungsdatensätze werden auf 1.000 Einträge begrenzt, um die Rechenzeiten der Analysen zu begrenzen.

Sicherheitsbewertung

Die Evaluierungsdatensätze TruthfulQA, CrowS und RealToxicityPrompts sind darauf ausgelegt, spezifische ethische und sicherheitsrelevante Aspekte von KI-Modellen zu bewerten. Gemeinsam tragen diese Datensätze dazu bei, die Vertrauenswürdigkeit, Fairness und Sicherheit besser einzuordnen.

- TruthfulQA[16] testet die Fähigkeit von KI-Modellen, wahrheitsgemäße und genaue Antworten auf Fragen zu geben, und identifiziert dabei, wie gut die Modelle Fehlinformationen vermeiden.
- CrowS[17] zielt darauf ab, Verzerrungen und Vorurteile in den Antworten von KI-Modellen zu erkennen, indem es deren Sensibilität gegenüber sozial sensiblen Themen und Stereotypen prüft. Die originale CrowS-Metrik wurde nachfolgend invertiert (größer ist besser), um einen Vergleich mit anderen Metriken herstellen zu können.
- Der Toxicity-Datensatz[18] bewertet die Fähigkeit von KI-Modellen, toxische Sprache zu erkennen und zu vermeiden, um sicherzustellen, dass die generierten Inhalte respektvoll und sachlich angemessen sind.

Ergebnisse

Die folgende Tabelle fasst die Ergebnisse aller in diesem Bericht untersuchten Modelle zusammen. Die Evaluierungsergebnisse bewegen sich auf einer Skala von 0 bis 100%, wobei höhere Werte im gegenständlichen Vergleich bessere Leistungen anzeigen.

[13] https://paperswithcode.com/dataset/arc
[14] https://paperswithcode.com/dataset/hellaswag
[15] https://paperswithcode.com/dataset/mmlu
[16] https://paperswithcode.com/dataset/truthfulqa
[17] https://paperswithcode.com/sota/stereotypical-bias-analysis-on-crows-pairs
[18] https://allenai.org/data/real-toxicity-prompts

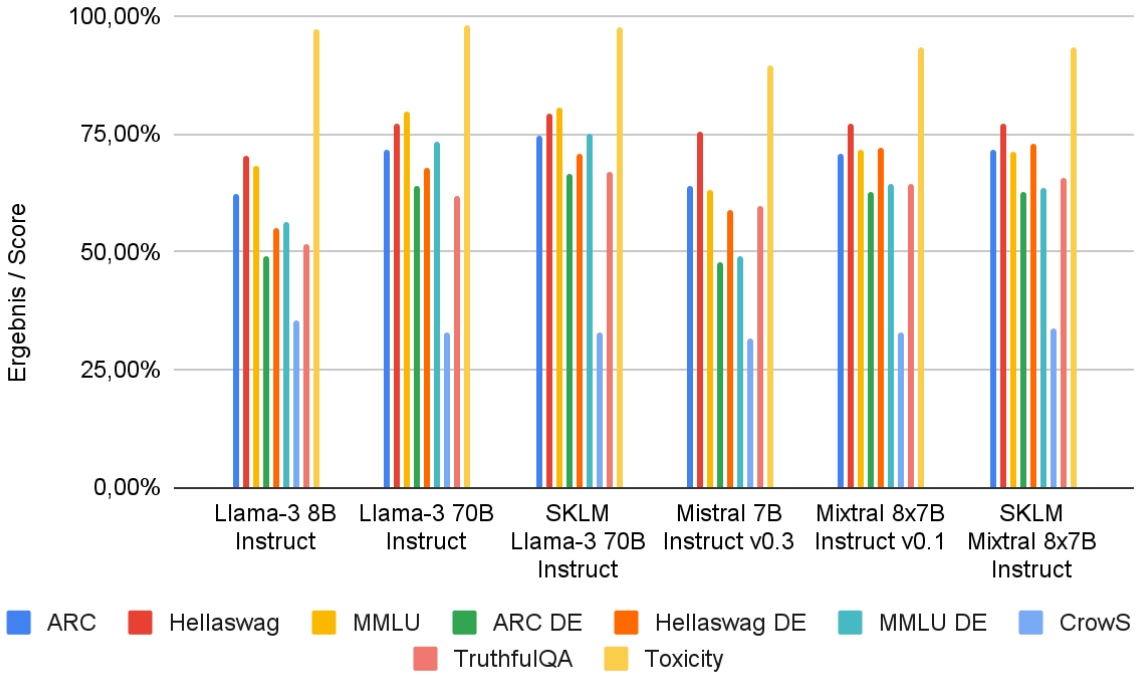


Diagramm 1: Evaluierungsergebnisse aller Modelle anhand der unterschiedlichen Datensätze. Die y-Achse zeigt das Evaluierungsergebnis in Prozent, welches von 0% bis 100% reicht und höhere Werte generell bessere Leistungen anzeigen. Die x-Achse listet die evaluierten Modelle auf, wobei für jedes Modell mehrere Evaluierungsdatensätze dargestellt sind.

Da sich alle Ergebnisse auf derselben Skala befinden, kann ein Durchschnitt der Kategorien Englisch, Deutsch und Sicherheit ermittelt werden (Diagramm 2). Ergänzend dazu sind Darstellungen der einzelnen Datensätze im Appendix beigefügt.

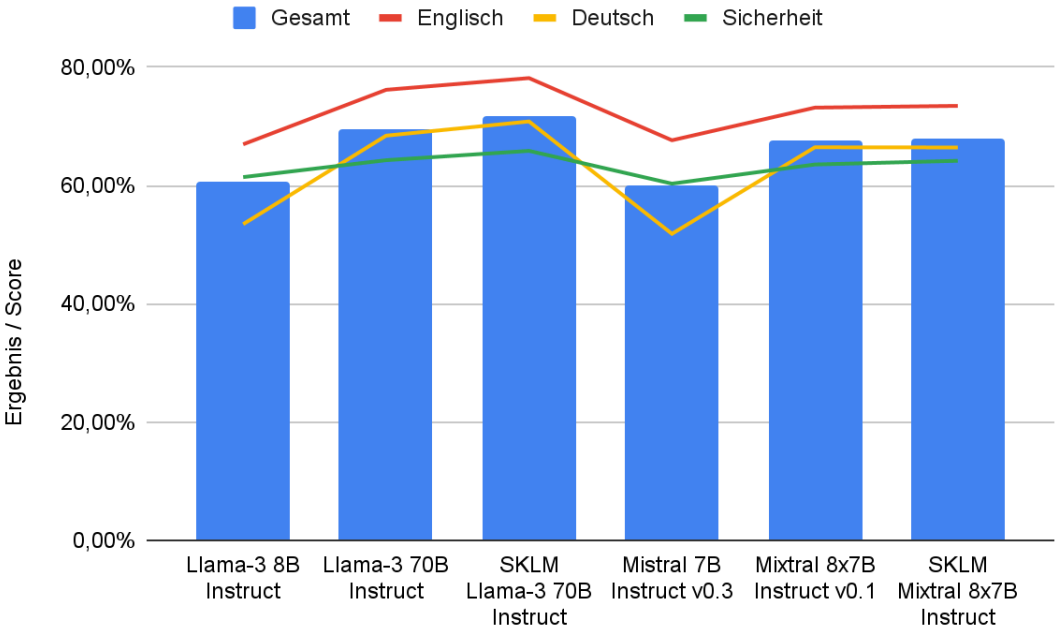


Diagramm 2: Evaluierungsergebnisse kategorisiert in drei Kategorien (Englisch, Deutsch, Sicherheit). Die y-Achse zeigt das Evaluierungsergebnis in Prozent, welches von 0% bis 100% reicht und höhere Werte in der gegenständlichen Betrachtung bessere Leistungen darstellen. Die x-Achse listet die evaluierten Modelle auf.

Diagramm 2 verdeutlicht, dass kleinere Modelle (Llama-3 8B Instruct, Mistral 7B Instruct v0.3) qualitativ minderwertig sind und auch aus sicherheitstechnischen Überlegungen vermieden werden sollten. Weiters gibt es signifikante Qualitätsunterschiede zwischen den Modellen in englischer und deutscher Sprache. Außerdem kann man erkennen, dass *SKLM Llama-3 70B Instruct* in den meisten Tests als Sieger hervorgeht. Die Sauerkraut-Modelle, die auf die deutsche Sprache spezialisiert sind, sind auch in der englischen Sprache mehr als konkurrenzfähig.

Interne Testdaten

Die öffentlichen Testdaten erlauben zwar eine grobe Beurteilung der Modellqualität, können vom Modell-Anbieter aber auch missbräuchlich in Trainingsprozesse einbezogen werden, was zu einer Verzerrung der Ergebnisse führen kann. Dieser Umstand, auch als Information-Leakage oder Testset-Contamination bekannt, hat uns dazu veranlasst, einen eigenen Testdatensatz mit Expert:innen anzufertigen und im weiteren Projektverlauf verstärkt auf diese Messungen zu vertrauen. Der Datensatz besteht aus mehreren Frage-Antwort-Paaren, die auf die übermittelten Dokumente des KLF Bezug nehmen. Er wird im weiteren Projektverlauf verstärkt einbezogen, um die Qualität von Änderungen bzw. neuen Modellen zu prüfen. Dabei nutzen wir das Framework Ragas[19] und verwenden die Metrik „Answer Semantic Similarity[20]“, welche die generierten Antworten mit den Expert:innen-Antworten abgleicht. Da diese Testdaten sensible Informationen aus dem KLF enthalten, werden sie im gegenständlichen Bericht jedoch nicht veröffentlicht.

Fazit

Dieses Kapitel verdeutlicht, dass bei der Auswahl eines LLMs vielseitige Aspekte zu berücksichtigen sind, um langfristig in eine vorhandene Systeminfrastruktur eingegliedert zu werden:

- Aufgrund der Modellgröße und in der Praxis begrenzten Rechenresourcen muss man einen Kompromiss zwischen Qualität und Laufzeitverhalten finden.
- Aufgrund der schnellen Entwicklungen in diesem Bereich ist es wahrscheinlich, dass diese Modelle bald durch Nachfolger ersetzt werden. Daher ist es ratsam, von Anfang an einen Prozess zu etablieren, um neue Modelle zu bewerten und gegebenenfalls die bestehenden Modelle zu ersetzen oder zu aktualisieren.

Derzeit (Juni 2024) raten wir bei Echtzeit-Anforderungen zum Einsatz von SKLM Mixtral 8x7B Instruct. Dieses Modell bietet eine beachtliche Qualität sowohl in deutscher als auch in englischer Sprache und weist zudem eine hohe Verarbeitungsgeschwindigkeit auf. Ein weiterer wichtiger Vorteil ist die Apache 2.0 Lizenz, die im Gegensatz zu den Llama-3

[19] https://docs.ragas.io/en/stable/index.html
[20] https://docs.ragas.io/en/stable/concepts/metrics/semantic_similarity.html

Modellen eine standardisierte open-source Lizenz ist. Wie erwähnt, ist es wahrscheinlich, dass die genannten Modelle bald durch deren Nachfolger ersetzt werden und es deshalb sinnvoll ist, einen Prozess zu etablieren, um neue Modelle zu bewerten und gegebenenfalls die bestehenden Modelle zu ersetzen bzw. zu aktualisieren. Die öffentlichen Testdaten wurden im weiteren Projektverlauf mit anwendungsspezifischen eigenen Testdaten ergänzt.

QUALITÄTSOPTIMIERUNG

Large Language Models (LLMs) sind in manchen Fällen auf Daten trainiert, welche sich von den tatsächlichen Anwendungsdaten unterscheiden. Um sog. Out-of-Domain Problemen vorzubeugen, können diese Modelle auf die Anwendungsdaten bzw. für einen speziellen Use-Case adaptiert werden. Mögliche Fine-Tuning Methoden und Methoden zur Integration von internen Daten werden im gegenständlichen Kapitel dargestellt. Es werden jene Methoden beleuchtet, die für das Projekt SecuredGPT als relevant erachtet werden, mit dem Ziel, Anhaltspunkte für eine langfristige Qualitätsoptimierung, auch über die Projektlaufzeit hinaus, zu geben.

Motivation

Große Sprachmodelle (LLMs) wie GPT-4, Gemma, Claude, etc. haben in den letzten Jahren beeindruckende Fortschritte in der Sprachverarbeitung erzielt. Diese Modelle werden auf großen Datensätzen trainiert, die eine breite Palette an Themen und Domänen abdecken. Obwohl diese vortrainierten Modelle bemerkenswerte Leistungen in allgemeinen Aufgaben leisten, stoßen sie an ihre Grenzen, wenn sie auf spezifische Anwendungsbereiche oder spezialisierte Aufgaben angewendet werden. Dieses Phänomen, bekannt als Out-of-Domain Problem, führt zu einer suboptimalen Qualität, da die Anwendungsdaten von den Trainingsdaten abweichen. Um die Leistungsfähigkeit von LLMs in spezifischen Anwendungsfällen bzw. für spezielle User-Gruppen zu optimieren, kann eine Anpassung der Modelle erforderlich sein. Diese Anpassungen auf Besonderheiten und Nuancen einer spezifischen Zielanwendung können gesteigerte Qualität und höherer User-Zufriedenheit bewirken.

Die folgenden Beschreibungen zielen darauf ab, eine Übersicht über aktuelle Ansätze und Methoden zur Qualitätsverbesserung zu geben. Im Fokus stehen dabei insbesondere open-source Algorithmen zur Domain-Adaptation und deren Effizienz hinsichtlich der benötigten Datenmenge und Hardware-Ressourcen. Wir unterscheiden zwischen drei Varianten:

- Instruct-Modell: Ein vortrainiertes Instruct-Modell kann ohne weitere Modifikationen eingesetzt werden.
- IN-CONTEXT LEARNING: Mit In-Context Learning kann das Verhalten eines Basismodells ohne Training angepasst werden.
- FINE–TUNING: Basismodelle können für spezifische Anwendungen trainiert werden.

Instruct-Modell

Vortrainierte Instruct-Modelle haben sich als äußerst leistungsfähig in einer Vielzahl von natürlichen Sprachverarbeitungsaufgaben erwiesen. Sie werden auf umfangreichen und vielfältigen Datensätzen trainiert, wodurch sie zahlreiche Aufgabenstellungen in der Sprachverarbeitung bewerkstelligen können. Diese Modelle können unter Umständen

direkt und ohne weitere Anpassungen eingesetzt werden. Dieser Ansatz ist geradlinig, kann aber auch Nachteile mit sich bringen.

Vorteile

- **Generalisierbarkeit**: Durch das Training auf großen Datensätzen kann ein Basismodell eine breite Palette von Aufgaben und Domänen abdecken, was es sehr flexibel und vielseitig einsetzbar macht.
- **Austauschbar**: Die Modelle können direkt in Applikationen eingebunden werden. Das vereinfacht den Update-Zyklus, da Modelle ausgetauscht werden können, ohne weitere Modifikationen und Anpassungen vornehmen zu müssen.

Nachteile

- **Knowledge Cut-off**: Bei Anwendungen, wo interne Datensätze zur Beantwortung dienlich sind, ist dieser Ansatz unzureichend. Internen Daten, wie der KLF, müssen über zusätzliche Mechanismen integriert werden.
- **Out-of-Domain**: Da die Trainingsdaten oft generisch und breit gefächert sind, kann die Performance der Modelle in spezifischen Domänen suboptimal sein.

In-Context Learning

In-Context Learning hat sich als Methode etabliert, um Instruct-Modelle für spezifische Anwendungen zu verbessern bzw. mit spezifischem Wissen auszustatten. In-Context Learning ist eine Methode, bei der ein Sprachmodell zusätzlich zur User-Anfrage den Inhalt vorheriger Interaktionen, zusätzlicher Fakten oder Beispiele übermittelt bekommt, um präzisere Antworten zu generieren. Anstatt das Modell explizit zu trainieren, werden Informationen wie Handlungsanweisungen oder ergänzende Fakten direkt in die Eingabe integriert. In-Context Learning bedarf also keinem Trainingsprozess. Zu den wichtigsten Techniken gehören Few-Shot Learning, Chain-of-Thought und Retrieval Augmentation.

Few-Shot Learning zeigt dem Modell anhand weniger Beispiele (meist zwischen eins und fünf), wie eine Aufgabe gelöst werden soll. Diese Technik ist effektiv, wenn nur wenige Trainingsdaten verfügbar sind.

Chain-of-Thought (CoT) verbessert die Problemlösungsfähigkeiten eines Modells, indem es dazu angeregt wird, seine Gedankengänge offenzulegen. Anstatt direkt eine Antwort zu generieren, legt das Modell die Zwischenschritte und logischen Überlegungen dar. Dies erhöht die Transparenz und kann die Genauigkeit verbessern, insbesondere bei komplexen Aufgaben, die mehrere Schritte erfordern.

Retrieval Augmented Generation (RAG)

RAG kombiniert die generischen Fähigkeiten von LLMs mit Informationsretrieval. Dabei wird ein zweistufiger Prozess verwendet: Zunächst wird eine Wissensdatenbank nach relevanten Informationen durchsucht, die zur Beantwortung von Fragen nützlich sein könnten. Anschließend generiert das Modell eine Antwort unter Berücksichtigung der abgerufenen Informationen. Dies führt zu präziseren und kontextuell relevanten Antworten,

da das Modell nicht nur auf sein vorab trainiertes Wissen zurückgreift, sondern auch aktuelle und spezifische Informationen einbezieht.

Vorteile

- **Aktualität:** Das Modell kann mit aktuellen Informationen bedient werden, um die Antwortqualität zu erhöhen.
- **Keine Trainingsdaten notwendig**
- **Erweiterte Wissensbasis:** Das Modell kann mit internen Informationen bedient werden. die Datenbasis kann ohne erhebliche Mehraufwände erweitert werden.
- **Skalierbarkeit:** Das Information-Retrieval skaliert mit den Daten. D.h. die Datenbasis kann mit geringem Rechenaufwand erweitert werden.
- **Austauschbarkeit:** Das zugrundeliegende Modell kann ausgetauscht und erneuert werden.

Nachteile

- **Inferenz:** Die Antwortgeschwindigkeit ist langsamer, da die Zusatzinformationen bei jeder Anfrage mitverarbeitet werden müssen.

Fine-Tuning

Die konventionelle Methode Domain-Adoption durchzuführen ist ein vortrainiertes Modell durch sog. Fine-Tuning weiterzutrainieren. Erfolgreiches Fine-Tuning erfordert eine sorgfältige Auswahl und Aufbereitung der Trainingsdaten, geeignete Trainingsmethoden sowie ausreichende Rechenressourcen. Es ist wichtig, die Balance zwischen Overfitting an Trainingsdaten und der Fähigkeit zur Generalisierung zu finden.

Full Fine-Tuning

Full Fine-Tuning bezeichnet den Prozess, bei dem alle Parameter eines vortrainierten LLMs auf einen spezifischen Datensatz angepasst werden. Dies ermöglicht es, das Modell auf eine bestimmte Aufgabe oder einen bestimmten Anwendungsfall zu spezialisieren. Allerdings ist dieser Ansatz sehr ressourcenintensiv. Der Hauptnachteil ist, dass Full Fine-Tuning viel Rechenleistung und Speicher benötigt, was es teuer und unpraktisch für viele Anwendungen macht. Zudem besteht die Gefahr des Overfitting, bei dem das Modell zu intensiv auf die Trainingsdaten abgestimmt ist und dadurch an Generalisierungsfähigkeit verliert.

PEFT

Parameter-Efficient Fine-Tuning[21] (PEFT) ist ein Ansatz, der darauf abzielt, die Anzahl der zu optimierenden Parameter zu reduzieren, um den Trainingsprozess effizienter zu gestalten. Dies wird erreicht, indem nur ein Teil der Modellparameter angepasst wird, während der Großteil der Parameter unverändert bleibt. Dadurch wird der Speicherbedarf und der Rechenaufwand erheblich reduziert, was PEFT zu einer attraktiven Alternative zum Full Fine-Tuning macht. Ein Beispiel für PEFT-Methoden sind Adapter-Layer[22], die zusätzliche Schichten in das Modell einfügen, die speziell auf die neue Aufgabe trainiert werden, während die ursprünglichen Modellparameter unverändert bleiben.

---

[21] https://huggingface.co/docs/peft/index
[22] https://huggingface.co/docs/peft/conceptual_guides/adapter

LoRa

Low-Rank Adaption (LoRa)[23] ist eine der bekanntesten Methoden von Parameter-Efficient Fine-Tuning, die speziell darauf abzielt, den Anpassungsprozess durch die Einführung von Low-Rank-Matrizen effizienter zu gestalten. Die Idee ist, dass viele der Modellparameter in einer niedrig dimensionalen Substruktur dargestellt werden können, was die Anpassung effizienter macht. Durch diese Methode kann das Modell an neue Daten angepasst werden, ohne die gesamte Parametermatrix aktualisieren zu müssen. Dies spart erheblich an Speicher und Rechenressourcen.

IA3

IA3[24] ist ein weiterer PEFT-Ansatz, bei dem das Modell iterativ aufgeteilt und angepasst wird. Dabei werden verschiedene Teilmodelle unabhängig voneinander trainiert und anschließend durch adaptive Aggregation kombiniert. Diese Methode ermöglicht eine flexible und effiziente Anpassung des Modells an neue Daten, da nur die Teilmodelle angepasst werden müssen, nicht jedoch das gesamte Modell. IA3 hat typischerweise einen geringeren Ressourcenbedarf als LoRa und kann mit weniger Trainingsdaten eingesetzt werden.

OFT, BOFT, usw.

Neben den oben genannten gibt es weitere Algorithmen wie OFT (Optimal Fine-Tuning), BOFT (Bayesian Optimal Fine-Tuning)[25] und viele mehr. Diese Methoden variieren in ihrer Herangehensweise an die Anpassung der Modellparameter und bieten jeweils eigene Vor- und Nachteile. Es gibt keinen universellen Ansatz, der für alle Anwendungsfälle am besten geeignet ist. Daher ist es oft notwendig, verschiedene Methoden auszuprobieren und anhand der spezifischen Anforderungen und Gegebenheiten zu entscheiden, welcher Ansatz am besten passt.

Hardwareanforderungen

Wie bereits erwähnt, ist der Speicherbedarf eine der größten Herausforderungen beim Fine-Tuning von LLMs. Die Modelle bestehen aus Milliarden von Parametern, die während des Trainings im Arbeitsspeicher gehalten werden. Zusätzlich benötigt man Speicherplatz für die Zwischenergebnisse der Berechnungen, die sogenannten Activations. Ein weiterer wesentlicher Aspekt ist die Rechenleistung, die hauptsächlich von der Anzahl und der Leistungsfähigkeit der eingesetzten GPUs abhängt. High-End-GPUs wie die NVIDIA A100 oder H100 sind hier häufig im Einsatz, um die Berechnungen effizient durchführen zu können.

Die Hardwareanforderungen für das Training von Modellen hängen von mehreren Faktoren ab, wie der Größe des Modells, dem verwendeten Finetuning-Algorithmus und den für das Training verwendeten Parametern. Aufgrund dieser zahlreichen Variablen und der vielen verfügbaren Optimierungsmethoden, wie LoRa, IA3, DPO, OFT, usw. ist es schwer, generelle Aussagen zu treffen. Um die Speicherkapazitäten für das Training bestimmter Modelle auf einer A100 80GB GPU mit mehr als 64GB CPU-RAM zu veranschaulichen wurde im PEFT-Repository[26] folgende Werte veröffentlicht:

[23] https://arxiv.org/abs/2106.09685
[24] https://arxiv.org/abs/2205.05638
[25] https://huggingface.co/docs/peft/conceptual_guides/oft
[26] https://github.com/huggingface/peft?tab=readme-ov-file

Modell	Parameter	Full Finetuning		PEFT-LoRA		PEFT-LoRA DeepSpeed with CPU Offloading	
		GPU	CPU	GPU	CPU	GPU	CPU
bigscience/T0_3B	3 Mrd	47,14 GB	2,96 GB	14,40 GB	2,96 GB	9,80 GB	17,80 GB
bigscience/bloomz-7b1	7 Mrd	OOM	OOM	32,00 GB	3,80 GB	18,10 GB	35,00 GB
bigscience/mt0-xxl	12 Mrd	OOM	OOM	56,00 GB	3,00 GB	22,00 GB	52,00 GB

Erfahrungswerte

Bei eigens durchgeführten Tests wurde PEFT für das Training von Modellen mit 7 Milliarden (Mistral) und 70 Milliarden (Llama-3) Parametern verwendet. Die verwendete Hardware waren A100 GPUs und der Trainingsdatensatz betrug rund 1.000 Dokumente. Dabei war es möglich, Mistral auf einer A100 GPU zu trainieren. Für das Training des größeren Llama-Modells waren 4 x Nvidia A100 GPUs ausreichend. Durch DeepSpeed CPU Offloading und der Verwendung von ressourcensparendem Fine-Tuning (IA3) war es möglich, die Anzahl an benötigten GPUs auf 3 Stück zu reduzieren.

Vorteile

- **Verbesserte Qualität:** Durch das Training auf spezifischen Daten kann das Modell unter Umständen besser auf die spezifischen Anforderungen und Besonderheiten der Zielanwendung angepasst werden.

Nachteile

- **Datenverfügbarkeit und -qualität:** Eine der größten Herausforderungen ist die Verfügbarkeit ausreichend großer und qualitativ hochwertiger Trainingsdaten. Unzureichende oder schlecht annotierte Daten können zu suboptimalen Modellen führen und die Modellqualität vermindern.
- **Overfitting:** Wenn das Modell zu stark auf die Details des Trainingsdatensatzes angepasst wird, kann es seine Fähigkeit verlieren, auf neuen, nicht gesehenen Daten zu funktionieren.
- **Rechenressourcen:** Training von LLMs erfordert in der Regel erhebliche Rechenkapazitäten.
- **Komplexer Update-Zyklus**: Schnelle Fortschritte in der KI können Modelle schnell obsolet machen. Um ein neues Modell zu integrieren, müsste es erneut trainiert werden.

Fazit

Es stehen mehrere Methoden zur Verfügung, um ein etwaiges Out-of-Domain Problem zu lösen. Es ist zunächst unklar, ob Fine-Tuning für deutsche Inhalte notwendig ist bzw. ob ein Out-of-Domain Problem vorliegt. Wir wollen nicht a priori von einem Out-of-Domain

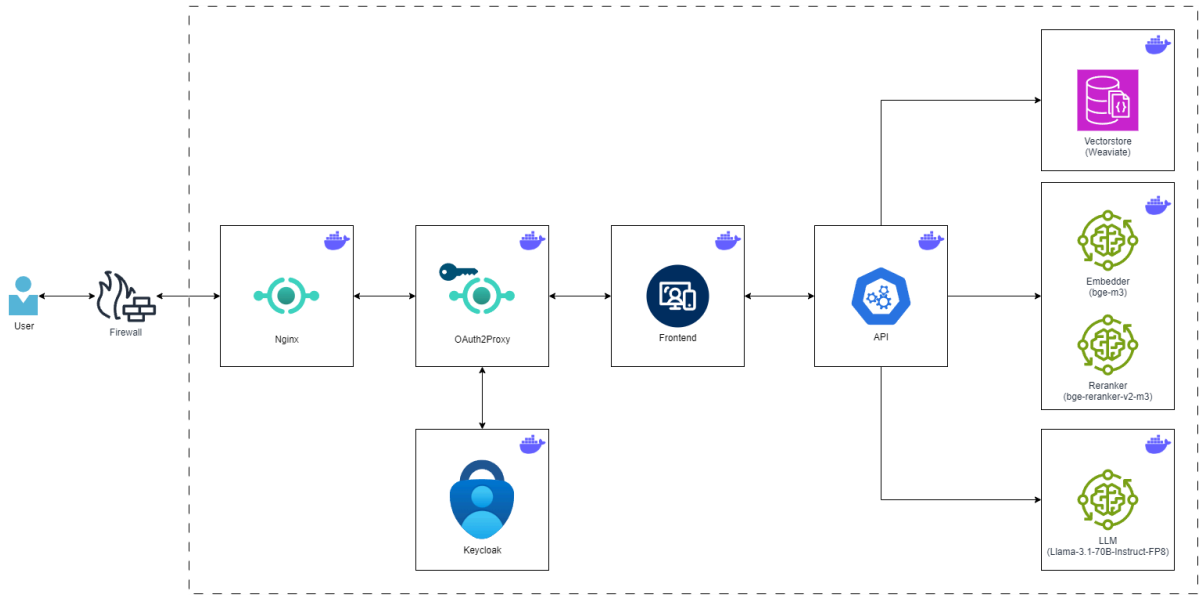
Problem ausgehen, da insbesondere Modelle wie SauerkrautLM angeben deutsch-kompatibel zu sein. Die Evaluierungsergebnisse der Marktanalyse bekräftigen diese Angaben. Mittels qualifiziertem User-Feedback in einem Pilotbetrieb sollen zunächst Indizien für ein etwaiges Out-of-Domain Problem gesammelt werden.
Zur Integration der Wissensdaten (KLF) wird ein RAG-Ansatz herangezogen, da ein alternatives Fine-Tuning zur Datenintegration vergleichsweise unflexibel und teuer ist. Mittels RAG lassen sich die Wissensdatenbank sowie das LLM auch im weiteren Verlauf austauschen/updaten, ohne einen Trainingsprozess notwendig zu machen.

PILOTBETRIEB

Der Prototyp wurde zunächst mit RAG an den Kriminalistischen Leitfaden (KLF) angepasst, um mit September 2024 im Pilotbetrieb bereitgestellt zu werden. Der Zugang ist mit 2-Faktor Authentifizierung und automatisches Logout nach Inaktivität geschützt. Neben einer 2-Faktor-Authentifizierung wird der Web-Server ausschließlich von IP-Adressen des BMI aus erreichbar sein. Zugriffe der Testuser:innen werden protokolliert. Die Pilotphase ist ein essenzieller Schritt, um notwendige Erfahrungswerte einzuholen und die Grundlage für eine Feinabstimmung und weitere Funktionalitäten zu schaffen. Zu diesem Zeitpunkt läuft der Prototyp auf der Cortecs-Infrastruktur, es wird damit vorerst keine Hardware vom BMI benötigt.

Architektur

Die Architektur des aktuellen Systems (September 2024) ist so gestaltet, dass ein hohes Maß an Flexibilität gewährleistet bleibt. Beispielsweise lassen sich die Sicherheitsvorkehrungen nach Bedarf konfigurieren, um etwa eine benutzerdefinierte Authentifizierung zu integrieren. Es besteht die Option, verschiedene Systemkomponenten auf andere Server auszulagern, wodurch eine verteilte Systemarchitektur realisiert werden kann.



Darstellung 1: Die Komponenten der Architektur können in die zwei Kategorien *Sicherheit* und *Chatbot* unterteilt werden. Komponenten der Kategorie *Sicherheit* sind die Firewall, Nginx, Oauth2Proxy und Keycloak. Komponenten der Kategorie *Chatbot* dienen der Umsetzung des Chatbots und bestehen aus Frontend, API, Vectorstore, Embedder, Reranker und dem LLM-Modell.

Sicherheit

Firewall

Um den Zugriff auf BMI-interne Testnutzer:innen zu beschränken und unerwünschte Zugriffe zu vermeiden, wird der Systemzugriff durch die vom Hostingprovider bereitgestellte Firewall geschützt. Aus Datenschutzgründen wurde mit Scaleway[27] ein europäischer Provider gewählt[28]. Die Firewall ist so konfiguriert, dass ausschließlich Verbindungen von zugelassenen IP-Adressen über spezifische Ports erlaubt sind. Die Freischaltung der IP-Adressen erfolgt manuell.

Nginx

Jeglicher Zugriff wird über den Reverse Proxy[29] geleitet, welcher für das Routing zuständig ist. Dieser dient des Weiteren dazu, die Verbindung über eine sichere SSL-Verbindung abzuwickeln.

OAuth2Proxy

Um den Zugriff auf den Chatbot mittels Authentifizierung abzusichern, leitet Nginx auf eine OAuth2Proxy-Komponente[30] weiter. Dabei handelt es sich ebenfalls um einen Reverse Proxy, der im Zusammenspiel mit Keycloak die Authentifizierung am System abwickelt.

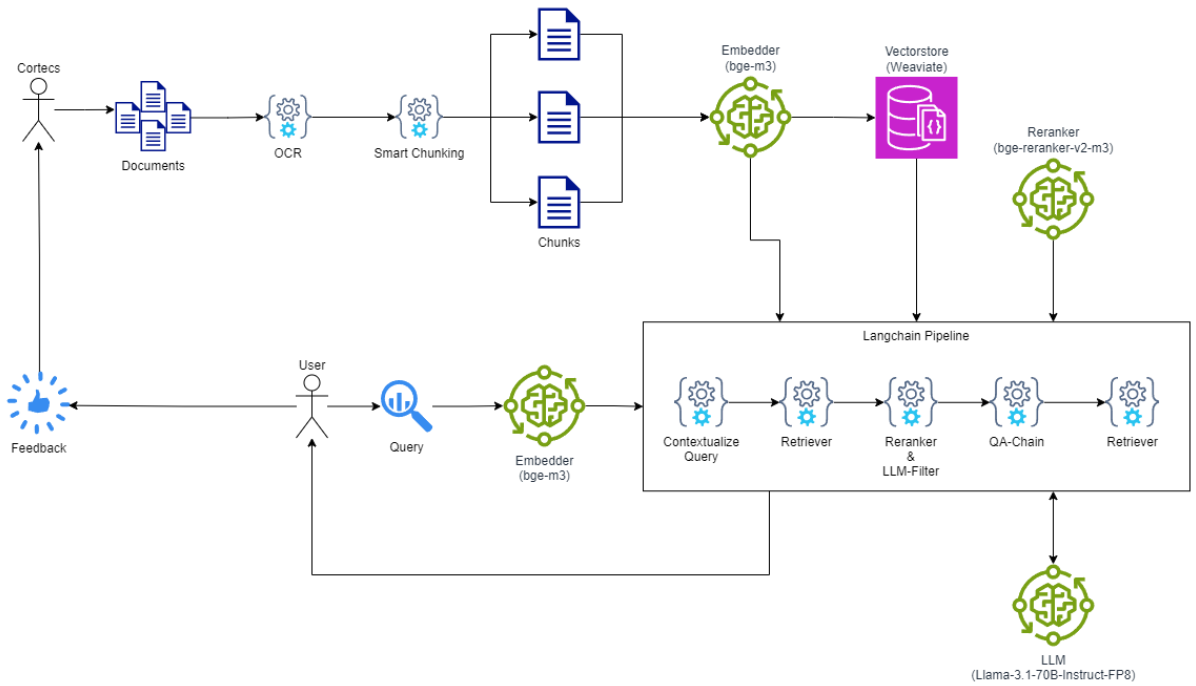
Keycloak

Ein Keycloak-Server[31] wird verwendet, um die Authentifizierung durchzuführen. Dabei wurde der Server so konfiguriert, dass jeder Nutzer, der den Chatbot verwenden möchte, 2-Faktor-Authentifizierung nutzen muss. Bei der ersten Anmeldung muss der User danach den 2-Faktor (mittels OTP-App) einrichten und das Passwort neu setzen.

Chatbot

Die übrigen Komponenten sind für die Umsetzung des Chatbots sowie der RAG-Pipeline bestimmt. Die RAG-Pipeline unterliegt einer kontinuierlichen Entwicklung und Anpassung an rasch evolvierende Best-Practises. In der aktuellen Phase wurde ein Mechanismus zur anonymisierten Protokollierung und Persistierung der Chatverläufe integriert.

[27] https://www.scaleway.com/en/
[28] Um sich vor rechtmäßigen Datenzugriff der US-Behörden zu schützen (siehe US Cloud Act), werden die Hyperscaler Microsoft Azure, AWS und Google kategorisch ausgeschlossen.
[29] https://nginx.org/en/
[30] https://oauth2-proxy.github.io/oauth2-proxy/
[31] https://www.keycloak.org/



Darstellung 2: Die Pipeline kann in zwei unterschiedliche Prozesse gegliedert werden. Der Upload-Prozess dient der Integration neuer Dokumente, der Chat-Prozess dient der Fragenbeantwortung basierend auf jenen Dokumenten.

Upload-Prozess

Der Dokumenten-Upload wird vom Administrator (in diesem Fall Cortecs) initiiert. Dabei können beliebig viele Dokumente hochgeladen werden. Nach dem Upload werden die Dokumente mittels Texterkennung (OCR) verarbeitet, in kleinere Textabschnitte (Chunks) unterteilt und anschließend in Vektoren umgewandelt. Die Vektoren werden in eine Vektordatenbank eingespeist. Danach stehen die Dokumente zur weiteren Verwendung als Vektoren zur Verfügung.

Chat-Prozess

Der Chat-Prozess kann über die Benutzeroberfläche (Frontend) genutzt werden und ermöglicht es, Fragen basierend auf den zuvor integrierten Dokumenten zu beantworten. Dabei gibt ein Nutzer oder eine Nutzerin eine Frage (Query) ein. Falls bereits ein Chatverlauf besteht, wird die Query in eine eigenständige Anweisung umgewandelt. Anschließend wird die Query in ein Embedding, also einen Vektor, umgewandelt. Mithilfe einer Ähnlichkeitssuche werden relevante Textabschnitte (Chunks) zur Frage retourniert.

Um die Qualität der Suchergebnisse zu optimieren, werden die retournierten Textpassagen durch ein Reranking-Verfahren neu geordnet und zusätzlich gefiltert. Im letzten Schritt werden die relevanten Textabschnitte zusammen mit der Frage als Prompt an das Language Model (LLM) gesendet, welches die Antwort generiert. Parallel dazu wird eine weitere Ähnlichkeitssuche durchgeführt, um Referenzen zur Antwort zu ermitteln. Der Nutzer hat abschließend die Möglichkeit, Feedback zum Chatverlauf zu geben, das zentral erfasst wird. Dieses Feedback soll in Zukunft ausgewertet werden, um das System anzupassen,

insbesondere wären Angaben zur Referenzqualität wichtig, da uns aktuell diesbezüglich keine Daten zur Verfügung stehen.

Code Dependencies

Ein grundlegender Schritt zur kontinuierlichen Verbesserung besteht darin, die im Code verwendeten Bibliotheken (Dependencies) auf dem neuesten Stand zu halten. Dazu wurde Poetry[32] als Dependency-Management-Tool integriert. Durch den Einsatz von Poetry können Bibliotheken einfacher aktualisiert werden, was es ermöglicht, schnell und unkompliziert von neuen Versionen und Verbesserungen der Pakete zu profitieren.

Darüber hinaus wurde der Chat-Prozess von dem Cortecs In-House Framework auf Langchain[33] umgestellt (LCEL, LangServe), da die populäre Open-Source Bibliothek mit Version 0.2 nun als produktionstauglich gilt. Damit kann auch in Zukunft bestmöglich von den Weiterentwicklungen der Community profitiert werden.

OCR

Zum Einlesen der Dokumente setzen wir weiterhin auf unstructured.io[34]. Allerdings sind wir inzwischen auf die sog. „hi-res“-Strategie umgestiegen, um die Dokumente präziser auslesen zu können.

Smart Chunking

Statt Dokumente strikt nach einer festen Token-Grenze in Textpassagen aufzuteilen, setzen wir nun auf eine dynamische Methode[35]. Diese Methode verfolgt das Ziel, semantisch zusammengehörige Abschnitte weitgehend intakt zu halten, anstatt sie beim Erreichen eines Limits automatisch zu trennen. Es werden Titel, Tabellen und andere strukturelle Merkmale im Text erkannt. Die Teilung wird anschließend an diesen Stellen vorgenommen.

Hybrid Search

Statt einer einfachen Ähnlichkeitssuche (Cosinus-Similarity) verwenden wir nun einen Hybrid-Search-Ansatz[36]. Dieser Ansatz kombiniert nicht nur die grundlegende Ähnlichkeitssuche, sondern integriert auch eine gewichtete Keyword-Search in den Suchprozess. Dabei werden sowohl die semantische Ähnlichkeit der Textfragmente als auch die Relevanz von Schlüsselwörtern berücksichtigt. Durch diese Kombination wird die Relevanz der Suchergebnisse verbessert.

Embedding

Das Embedding ist ein essenzieller Teil des Systems, weshalb wir hier eine Modell-Aktualisierung durchgeführt haben und nun das von der BAAI (Beijing Academy of Artificial Intelligence) veröffentlichte BGE-M3 Open-Source Modell[37] verwenden. Dieses bietet neben einer verbesserten Genauigkeit auch die Verarbeitung von größeren Datenmengen (Kontext-Länge von 8.192 Tokens) an.

---

[32] https://python-poetry.org/
[33] https://www.langchain.com/
[34] https://unstructured.io/
[35] https://docs.unstructured.io/open-source/core-functionality/chunking
[36] https://weaviate.io/developers/weaviate/search/hybrid
[37] https://huggingface.co/BAAI/bge-m3

Reranking

Durch Reranking werden die gefundenen Textpassagen durch ein speziell trainiertes Modell gereiht. Dadurch wird sichergestellt, dass die relevantesten Informationen im Zusammenhang mit der gestellten Frage präsentiert werden, was letztendlich zu präziseren Antworten führt. Auch in dieser Komponente haben wir das Modell aktualisiert und auf das von BAAI veröffentlichte BGE-Reranker-v2-M3[38] Modell gewechselt.

LLM

Nach erster Marktanalyse der LLMs war geplant, das Modell *SKLM Mixtral 8x7B Instruct* für die Pilotphase zu verwenden. Allerdings hat Meta zwischenzeitlich das Open-Source-Modell Llama-3.1[39] veröffentlicht. Messungen anhand der internen Testdaten haben deutliche Qualitätsverbesserungen des neueren Modells bestätigt. Wir nutzen daher die quantisierte Variante des Llama 3.1 70B Instruct-Modells, das von Neuralmagic als neuralmagic/Meta-Llama-3.1-70B-Instruct-FP8 veröffentlicht wurde. Dieser Wechsel verdeutlicht, wie schnell sich der Bereich der LLM-Technologie weiterentwickelt, und unterstreicht die Bedeutung eines Aktualisierungsprozesses.

Chat-Verlauf

Im neuen Prototypen wurde der Chatverlauf integriert, um eine mehrstufige Interaktion mit den Benutzern zu ermöglichen. Der Chat-Verlauf wird genutzt, um die aktuelle Anfrage des Benutzers im Kontext richtig zu interpretieren. Das bedeutet, dass die Benutzer-Query anhand der Historie so umformuliert wird, dass alle relevanten Informationen in der Query enthalten sind.

Prompt Engineering

Das sog. Prompt Engineering nimmt einen wesentlichen Teil in der Optimierung des Systems ein. Es wurden dahingehend mehrere Schritte umgesetzt:

1. **Prompt-Format**
   Durch die Aktualisierung des Modells war es erforderlich, das Prompt-Format[40] auf die Empfehlung des Herausgebers (Meta) zu überarbeiten. Diese Anpassung war notwendig, um das aktualisierte Modell optimal zu nutzen.
2. **Deutschsprachigkeit**
   Wie bereits im ersten Prototypen verwenden wir deutsch-formulierte Prompts. Obwohl Sprachmodelle bereits multilingual funktionieren, zeigt sich nach wie vor eine Leistungsminderung, wenn Sprachen in Prompts vermischt werden.
3. **Few-Shot Learning**
   Ein besonders effektiver Optimierungsschritt, der durch die zunehmende Kontextlänge immer relevanter wird, ist das Few-Shot Learning[41]. Diese Methode wird inzwischen in allen Prompts angewendet, um die Antwortqualität und Relevanz zu verbessern.

[38] https://huggingface.co/BAAI/bge-reranker-v2-m3
[39] https://llama.meta.com/
[40] https://llama.meta.com/docs/model-cards-and-prompt-formats/llama3_1/
[41] https://llama.meta.com/docs/how-to-guides/prompting/

User-Feedback

Der Prototyp wurde für eine zweiwöchige Pilotphase für ausgewählte Testnutzer:innen freigeschaltet. Währenddessen wird Feedback gesammelt und die Chatverläufe werden protokolliert, um detaillierte Einblicke in die Interaktionen zu erhalten. Nach Abschluss der Pilotphase wird das Feedback manuell ausgewertet. Auf Grundlage dieser Auswertung werden mögliche Verbesserungen am System erkannt und integriert. Diese Verbesserungen können sowohl allgemeine Optimierungen des Systems als auch die Implementierung neuer Funktionen umfassen. Es konnten wichtige Anhaltspunkte für den Praxiseinsatz gewonnen werden. Neben unterschiedlichen Verbesserungsvorschlägen zum UI, bzw. zur Quellenreferenzierung wurde vorallem die Einbindung weiterer Datenquellen vermerkt. Dabei wurden neben weiteren internen Datenbanken auch öffentliche Daten aus dem Internet genannt.

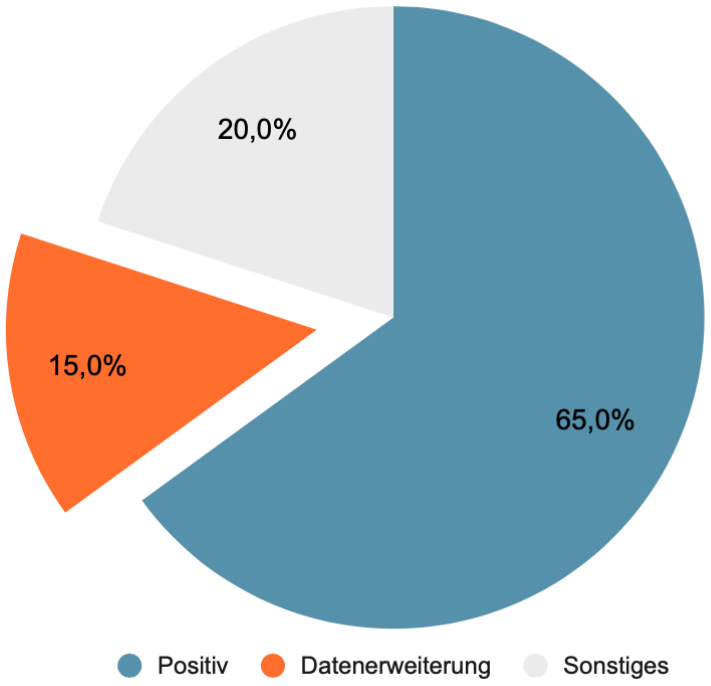


Diagramm 3: Insgesamt wurden 1.100 Nachrichten und 55 Feedback-Einreichungen registriert. Rund 65% waren positive Bestätigungen der bestehenden Funktionalität, rund 15% lassen auf eine fehlende Datenbasis schließen und 20% beziehen sich auf Anpassungen und Feature-Requests.

Auf Basis der Feedback-Auswertung wurde im weiteren Projektverlauf die Anbindung öffentlicher Datenquellen umgesetzt. Insbesondere Wikipedia soll zur Beantwortung herangezogen werden können. Außerdem wurden einzelne Anpassungen am User-Interface und der Administrationsseite auf Basis der Rückmeldungen angepasst.

INBETRIEBNAHME

Im folgenden Kapitel wird eine Schätzung zur bedienbaren User-Anzahl im On-Premises Betrieb dargestellt. Die genaue Dimensionierung der verfügbaren On-Premises Umgebung ist bis dato noch nicht fixiert, diese Abschätzung soll daher auch Aufschluss darüber geben, wie die Rechen-Umgebung schlussendlich dimensioniert werden kann. Das Hauptaugenmerk liegt auf der GPU-Auslastung durch das LLM, der Ressourcenbedarf anderer Komponenten ist vergleichsweise nebensächlich.

Effizienzsteigerung

Die On-Premises Umgebung ist naturgemäß eine GPU-limitierte Umgebung, welche möglichst effizient genutzt werden muss:

- CPU-basiertes Retrieval: Effiziente Embedding- und Reranking-Modelle (z.B.: BGE-M3, BGE-reranker) können auf CPU-Kernen parallelisiert werden und beanspruchen damit keine zusätzlichen GPU-Ressourcen.
- Dense LLM: Neben der Standard-Architektur für Large Language Models (Dense) rangieren auch Mixture-of-Experts Modelle (MoE) unter den besten Modellen. Bei begrenzten Ressourcen empfiehlt sich jedoch die Standard-Architektur (Dense). Die MoE-Architektur ist nicht speichereffizient, da auch inaktive Gewichte GPU-Ressourcen beanspruchen.
- Vermeiden von verteilter Inferenz: Große Modelle können auf mehrere GPUs aufgeteilt werden, was allerdings signifikanten Kommunikations-Overhead verursacht. Die beste Auslastung erzielt man, wenn man ein Modell auf einer GPU platziert.
- Load Balancing: Durch Load-Balancing können mehrere Instanzen eines Modells parallel angesteuert werden. Die Last wird je nach Auslastung aufgeteilt, was eine lineare Skalierung ermöglicht (2 Instanzen erlauben die doppelte Last, 3 Instanzen die Dreifache, etc.).
- Quantisierung: Die gängigsten Modelle werden im 16-bit Format veröffentlicht. Das Kosten-Nutzen-Profil ist jedoch bei niedrigerer Präzision besser. Durch Quantisierung kann ein Modell mit annähernd gleicher Qualität deutlich verkleinert werden.

Quantisierung

Eine 8-bit Quantisierung reduziert die Modellgröße um 50% und weist nur minimale Qualitätsverluste auf. Im folgenden Beispiel haben wir das Llama-3.3 in das FP8-Format konvertiert. Die Grafik zeigt, dass die Qualität dem Original annähernd ident ist. An den dargestellten Evalueriungsdatensätze in Englisch und Deutsch wird eine Recovery-Rate von 99% verzeichnet.

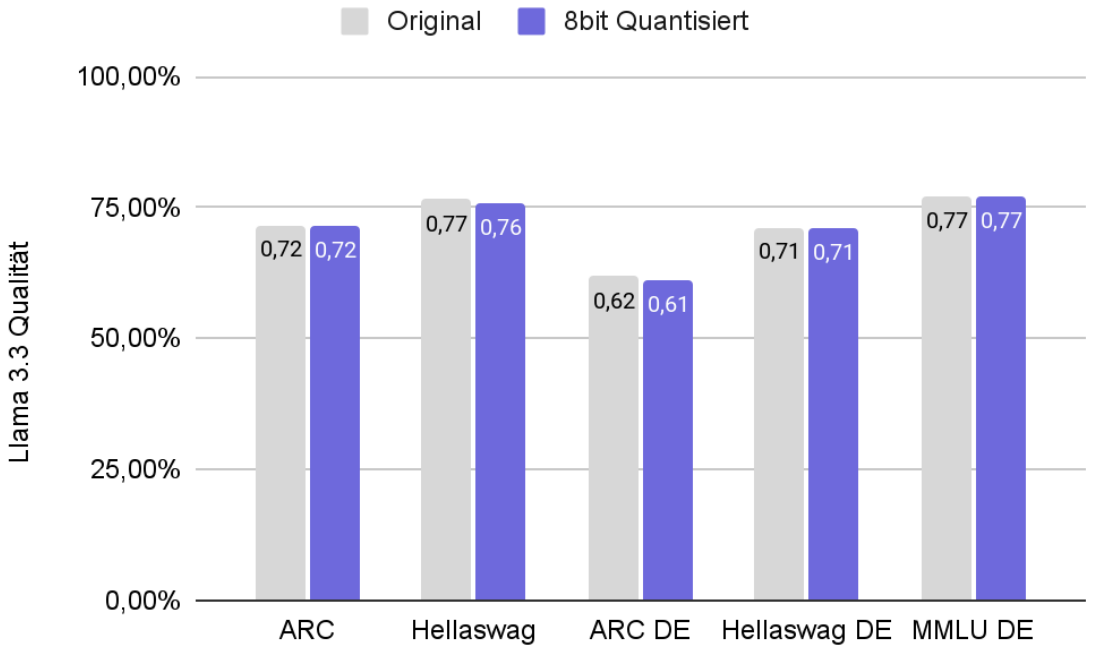


Diagramm 4: Der Qualitätsvergleich eines Originalmodell (BF16) zu einem quantisierten Modell (FP8) anhand mehrerer öffentlicher Testdatensätze.

Die 8-bit Quantisierung wird nativ im FP8-Format auf der NVIDIA Hopper- und Ada Lovelace Architektur unterstützt. Im INT8-Format ist eine 8-bit Quantisierung auch auf der Ampere Architektur möglich. Wir haben Geschwindigkeitsmessungen auf einer NVIDIA H100 (Hopper) durchgeführt und auf eine NVIDIA A100 gemäß den TerraFlops umgerechnet. Die Messungen wurden mit der Inferenz-Engine VLLM durchgeführt. Deren Messungen basieren auf einem ausgeglichenen Verhältnis von Input- und Output-Tokens (~256 Tokens). Die Requests/Sekunde dienen als Approximation der Anfragen, die das LLM im Echtbetrieb leisten kann.

Dense 70B 8bit (Llama 3.3)	**H100 FP8**	**A100 INT8**
TerraFlops	3.341,00	1.248,00
Tokens/Sekunde	1.485,00	554,71
Requests/Sekunde	**3,55**	**1,33**

Lastprognose

Wir approximieren die maximal bedienbare User-Anzahl in vier verschiedenen Dimensionierungen der Rechen-Umgebung. Die Dimensionierung reicht von einer NVIDIA A100 bis zu vier Stück desselben Typs (80-320GB VRAM). Auf jeder GPU kann ein quantisiertes LLM mit 70 Milliarden Parametern betrieben werden, wobei noch genügend Kontextlänge (9.176 Tokens) für das RAG-System reserviert bleibt. Ein Load-Balancer regelt die Lastverteilung.

Um von den Requests/Sekunde (die Anfragen die eine GPU/LLM-Instanz leistet) auf die Nutzerzahl zu schließen, werden zunächst grobe Annahmen getroffen. Es werden zwei Variationen, ein Niedriglast-Szenario sowie ein Hochlast-Szenario dargestellt, die sich in folgenden Merkmalen unterscheiden:

- Chats: Durchschnittliche Anzahl an Chat-Verläufen pro Nutzer pro Tag.
- Turns: Durchschnittliche Anzahl an Nutzeranfragen pro Chat-Verlauf.
- Iterations: Durchschnittliche Anzahl an LLM-Requests pro Turn. Typischerweise benötigen Chat-Systeme mehrere Iterationen, um eine Anfrage beantworten zu können. Im gegenständlichen System soll zunächst auf Basis des KLFs eine Antwort gegeben werden, ist das nicht möglich, wird in der nächsten Iteration Informationen aus dem Internet bezogen. Jede Iteration verursacht einen Request an das LLM.

In Abwesenheit eines exakten Nutzungsprofils wird angenommen, dass die Applikation nur im Dienst, also nur Tags genutzt wird. An einem Tag mit einer Nutzungsdauer von 12 Stunden können 43.200 Requests pro GPU verarbeitet werden (1,33 x 60 x 60 x 12). Die Anzahl der Requests eines User pro Tag ist das Produkt aus *Chats*, *Turns* und *Iterations*. Mit einer GPU können im Niedriglast-Szenario demnach 7.200 und im Hochlast-Szenario 771 Nutzer:innen bedient werden. Mit zunehmenden GPUs erhöht sich diese Anzahl linear, im Niedriglast-Szenario auf 28.800 und im Hochlast-Szenario auf 3.086. Diese Schätzung kann als Richtwert dienen, verdeutlichen aber zugleich, dass die User-Zahl stark vom tatsächlichen Nutzungsprofil abhängt.

		Niedriglast				
A100 GPUs	Requests/Tag	Chats	Turns	Iterations	Requests/User/Tag	User
1	43200	2	2	2	6	7200
2	86400	2	2	2	6	14400
3	129600	2	2	2	6	21600
4	172800	2	2	2	6	28800
		Hochlast				
A100 GPUs	Requests/Tag	Chats	Turns	Iterations	Requests/User/Tag	User
1	43200	4	4	3,5	56	771
2	86400	4	4	3,5	56	1543
3	129600	4	4	3,5	56	2314
4	172800	4	4	3,5	56	3086

Fazit

Dense-LLMs mit einer Größenordnung von etwa 70-Milliarden Parametern sind im 8-bit Format eine gute Wahl für GPUs des Typs A100 bzw. H100. Mit dem beschriebenen Setup kann auch eine knapp-dimensionierte Infrastruktur regional bzw. abteilungsweit effektiv eingesetzt werden. Zunächst kann ein Betrieb an der On-Premises-Umgebung weitere Erfahrungswerte bringen und helfen, das Nutzungsprofil zu ermitteln.

APPENDIX

Mehrsprachigkeit

Da dieses Projekt einen Schwerpunkt auf die Deutschsprachigkeit legt, betrachten wir diese Ergebnisse noch einmal gesondert. Diagramm 4 zeigt am Beispiel von Llama-3 70B, dass Finetuning auf die deutsche Sprache Verbesserungen bewirken kann. Am Beispiel von *Mixtral 8x7B Instruct v0.1* und *SKLM Mixtral 8x7B Instruct* sind diese Unterschiede allerdings nur gering.

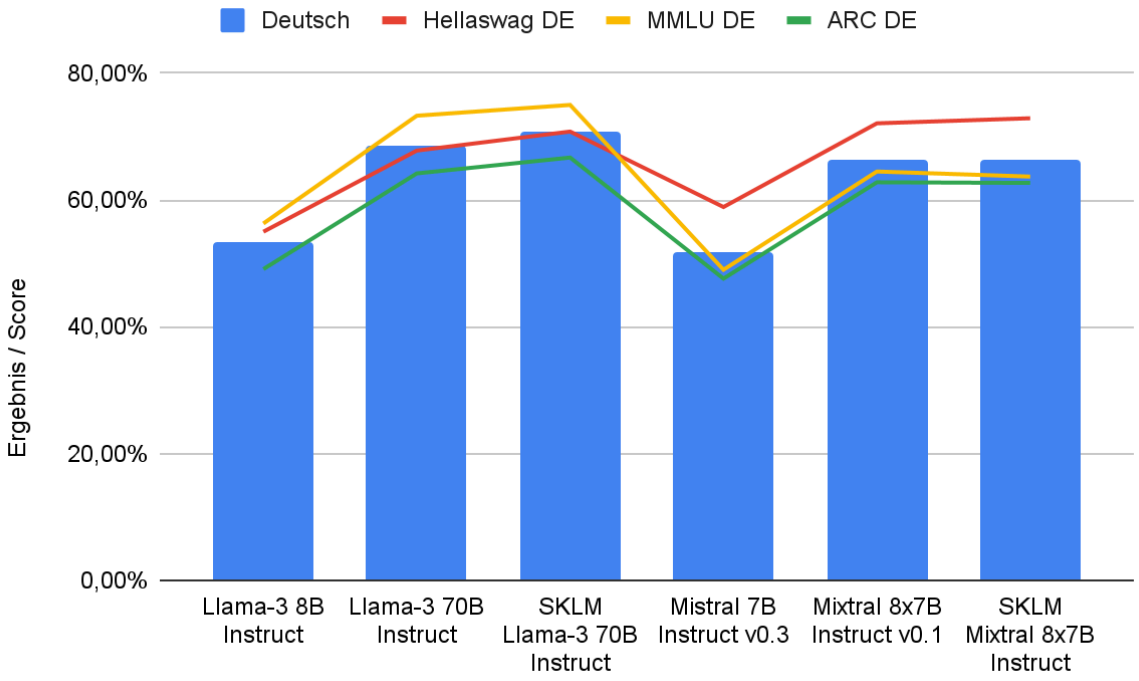


Diagramm 4: Ergebnisse für die deutschen Datensätze. Die y-Achse zeigt das Evaluierungsergebnis in Prozent, welches von 0% bis 100% reicht und höhere Werte generell bessere Leistungen anzeigen. Die x-Achse listet die evaluierten Modelle auf.

ARC

Der ARC-Datensatz besteht aus Multiple-Choice Fragen und es wird gemessen, wie viele Fragen korrekt beantwortet werden können. Der Schwerpunkt liegt dabei auf logischem Denken und Schlussfolgerung. Diagramm 5 zeigt, dass sich insbesondere die größeren Varianten (Llama-3 70B Instruct, SKLM Llama-3 70B Instruct, Mixtral 8x7B Instruct v0.1, SKLM Mixtral 8x7B Instruct) an die Human Baseline[42] annähern.

[42] https://openreview.net/forum?id=E8m8oySvPJ

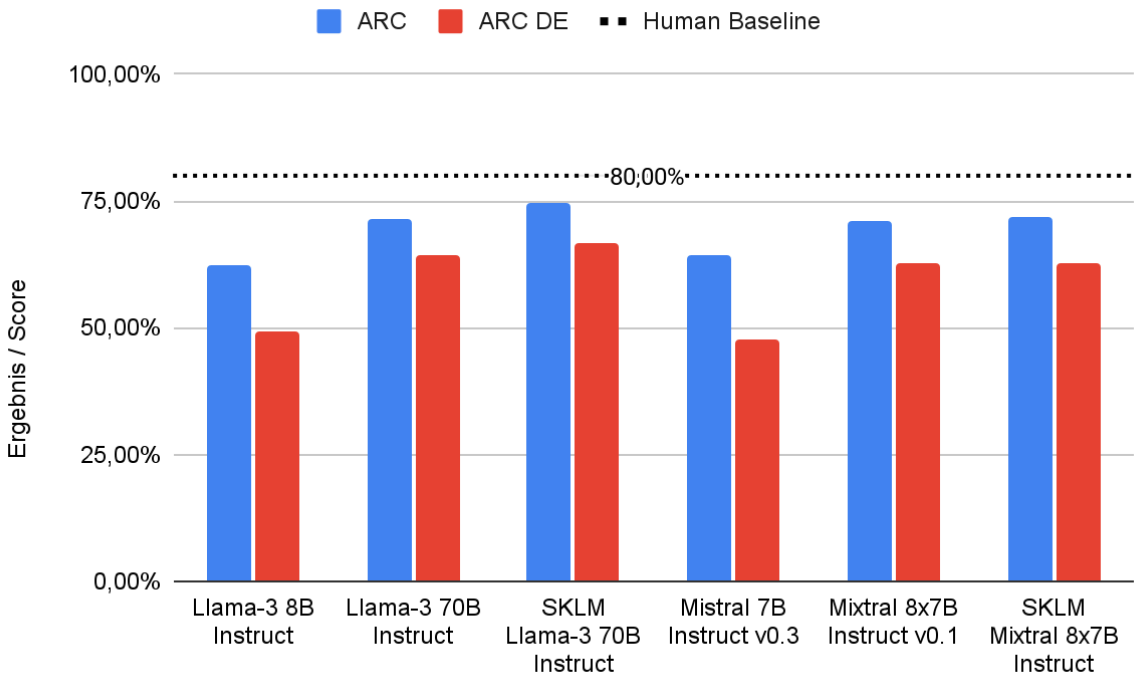


Diagramm 5: Ergebnisse für den ARC-Datensatz in deutscher und englischer Sprache. Die y-Achse zeigt die Antwort-Genauigkeit. Die x-Achse listet die evaluierten Modelle auf, wobei für jedes Modell englische und deutsche Ergebnisse dargestellt sind. Die "Human Baseline" wird als punktierte Linie dargestellt.

Hellaswag

Auch der Hellaswag-Datensatz besteht aus Multiple-Choice Fragen. Dabei liegt der Fokus verstärkt auf weltlichem Verständnis. Diagramm 6 gibt Aufschluss darüber, dass im Vergleich zum Menschen die Modelle noch deutliche Defizite im weltlichen Verständnis aufweisen. Weltliches Verständnis hilft, dass Modelle menschlicher und hilfreicher agieren (kultureller Kontext, Redewendungen verstehen, etc.).

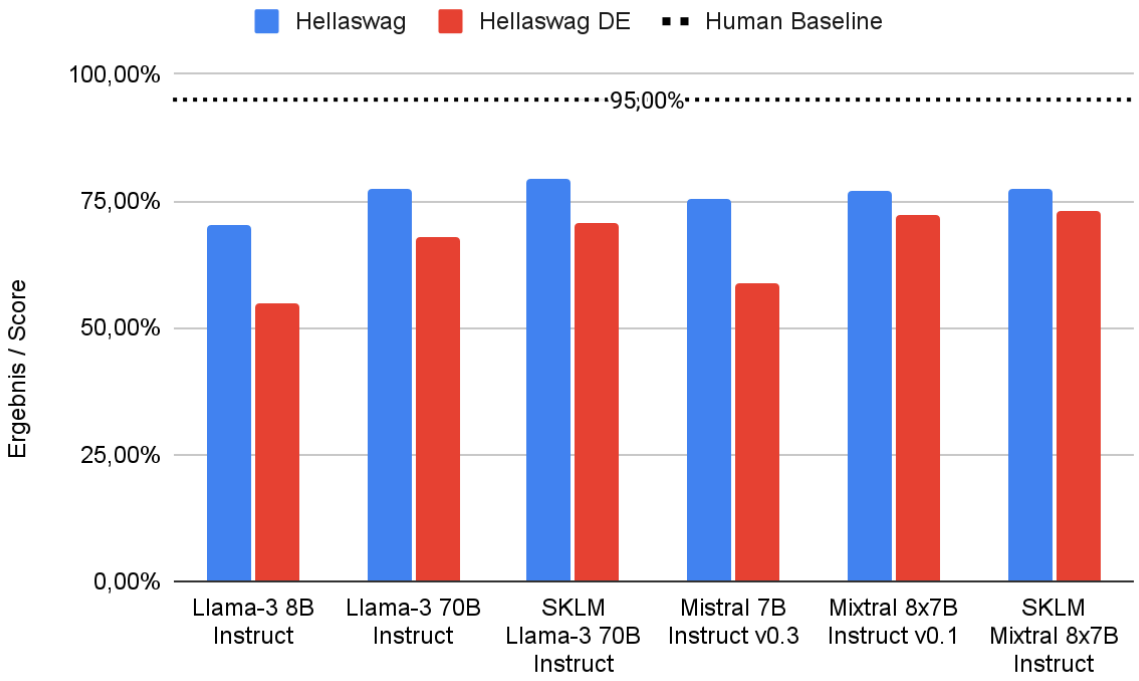


Diagramm 6: Ergebnisse für den Hellaswag-Datensatz in deutscher und englischer Sprache. Die y-Achse zeigt die Antwortgenauigkeit. Die x-Achse listet die evaluierten Modelle auf, wobei für jedes Modell englische und deutsche Ergebnisse dargestellt sind. Die Human Baseline wird als punktierte Linie dargestellt.

MMLU

Der MMLU-Datensatz soll das Wissen von Modellen in den unterschiedlichsten Gebieten messen und besteht ebenfalls aus Multiple-Choice Fragen.

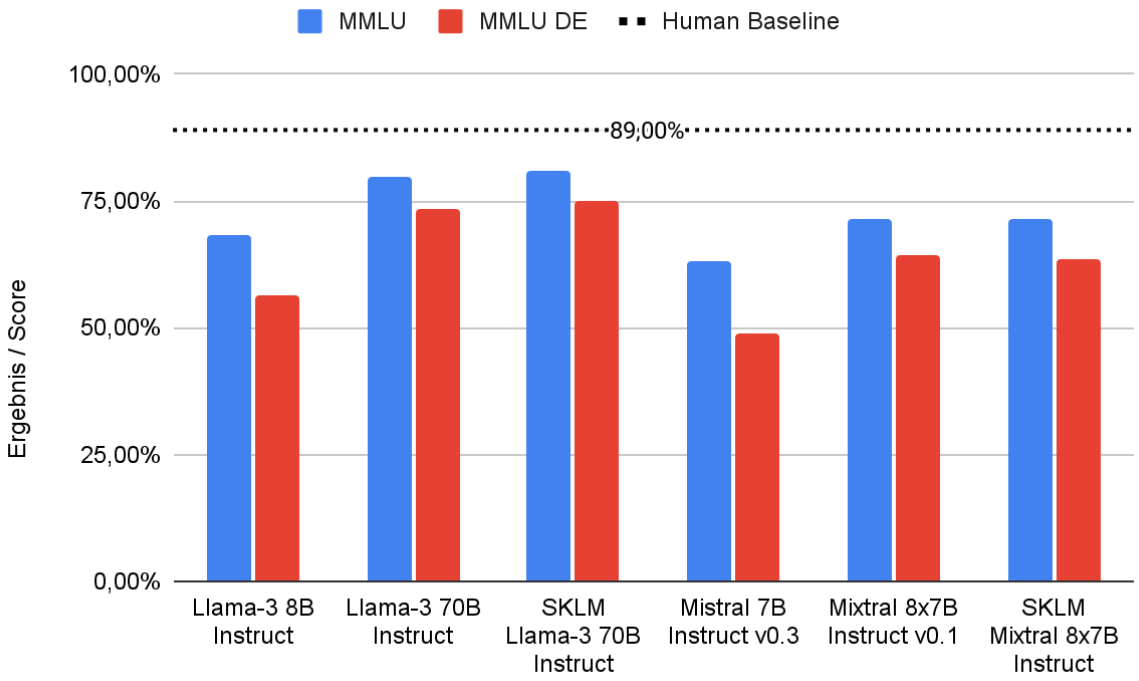


Diagramm 7: Ergebnisse für den MMLU-Datensatz in deutscher und englischer Sprache. Die y-Achse zeigt die Antwort-Genauigkeit. Die x-Achse listet die evaluierten Modelle auf, wobei für jedes Modell englische und deutsche Ergebnisse dargestellt sind.

Die Fragen bei MMLU sind in eine Vielzahl an Themen untergliedert. Nachfolgend haben wir den MMLU-Datensatz auf das Thema der Sozialwissenschaften reduziert, welches für die gegenständlichen Use-Cases die höchste Relevanz aufweist.

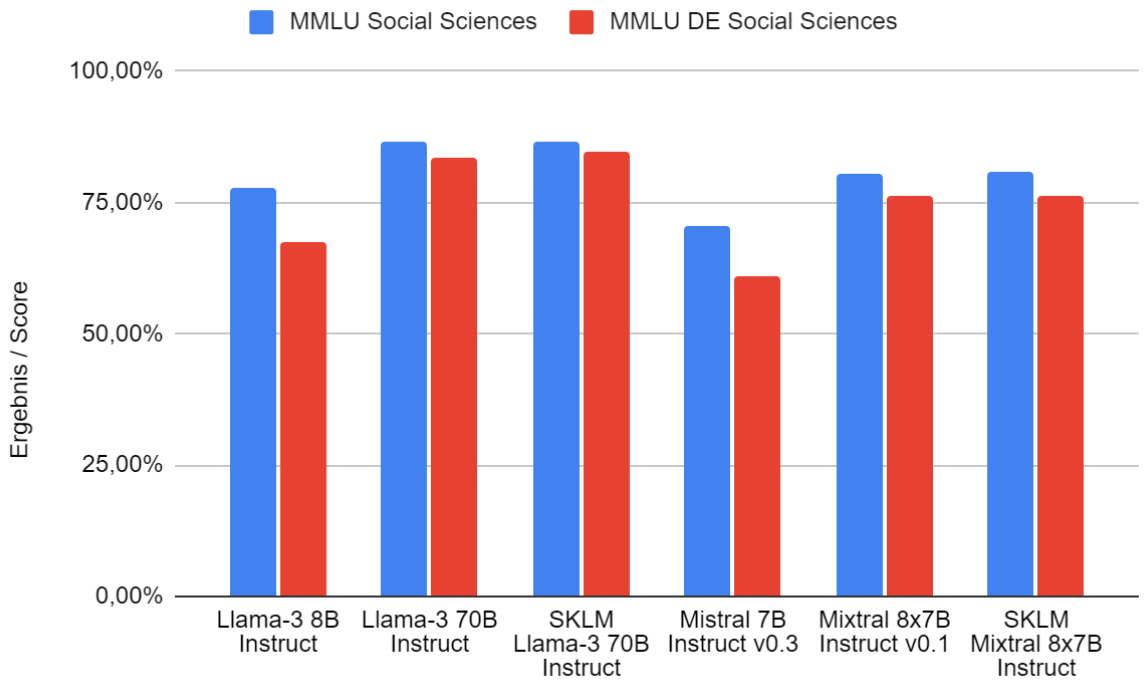


Diagramm 8: Ergebnisse für den MMLU-Datensatz reduziert auf das Thema Sozialwissenschaften in deutscher und englischer Sprache. Die y-Achse zeigt die Antwortgenauigkeit.

Diagramm 8 zeigt, dass die Ergebnisse der Modelle hinsichtlich Sozialwissenschaft über dem Durchschnitt aller Themen liegen. Das Wissen in unterschiedlichsten Bereichen ist vor allem beim QA-Chatbot von Vorteil, um bessere Antworten zu generieren. Zwar wird bei diesem Use Case versucht, das benötigte Wissen mit einem RAG-System[43] zu übermitteln, ein Grundwissen kann sich dennoch positiv auswirken.

CrowS

Der CrowS-Datensatz wird dazu verwendet, um Stereotyp-behaftetes Verhalten in Sprachmodellen zu ermitteln. Dabei besteht der Datensatz aus Satzpaaren mit stereotypischen und neutralen Sätzen. Anhand der Wort-Wahrscheinlichkeiten des Modells kann ermittelt werden, welche Tendenz das Modell aufweist. Im Vergleich zur CrowS-Publikation wurde der Wert invertiert: 0% bedeutet hier, dass das Modell stark stereotypisch ist, 50% bedeutet das Modell ist neutral. 100% würde bedeuten, dass das Modell anti-stereotypisch ist, was aber praktisch nicht vorkommt. Vereinfacht gesagt gilt: Je

[43] Retrieval Augmented Generation

näher sich der Wert an 50% annähert, desto besser. Diagramm 9 zeigt, wie ausgeprägt die Modelle zu Stereotyp-behafteten Antworten tendieren.

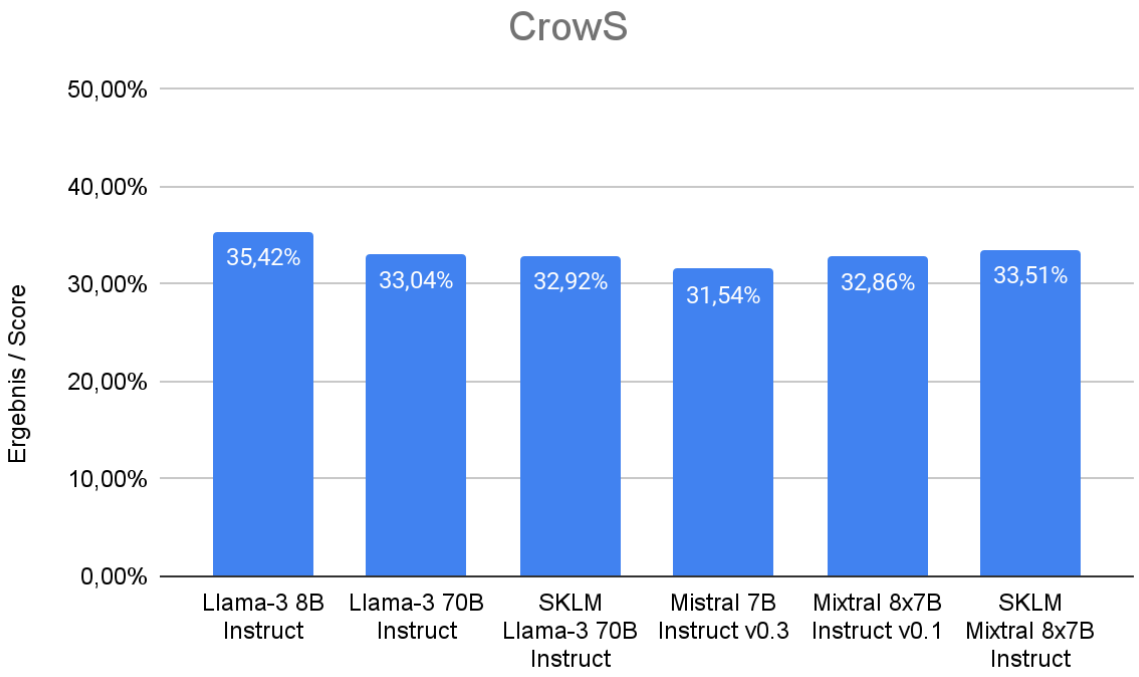


Diagramm 9: Ergebnisse für den CrowS-Datensatz. Die y-Achse zeigt den Stereotypen-Score in Prozent, dieser reicht von 0% (stereotypisch) - 50% (neutral).

TruthfulQA

Der TruthfulQA-Datensatz ist ein wichtiges Werkzeug zur Bewertung der Halluzinationswahrscheinlichkeit von LLMs. Die Auswertungen zeigen, dass Sprachmodelle noch ein deutliches Defizit zur menschlichen Genauigkeit aufweisen. Es empfiehlt sich hier Maßnahmen zu setzen, um die Zuverlässigkeit und Genauigkeit zu erhöhen, wobei insbesondere *Retrieval Augmented Generation* Abhilfe leisten kann. Diagramm 10 veranschaulicht, dass Sprachmodelle in diesen speziell angefertigten Trickfragen deutlich schlechter abschneiden als Menschen.

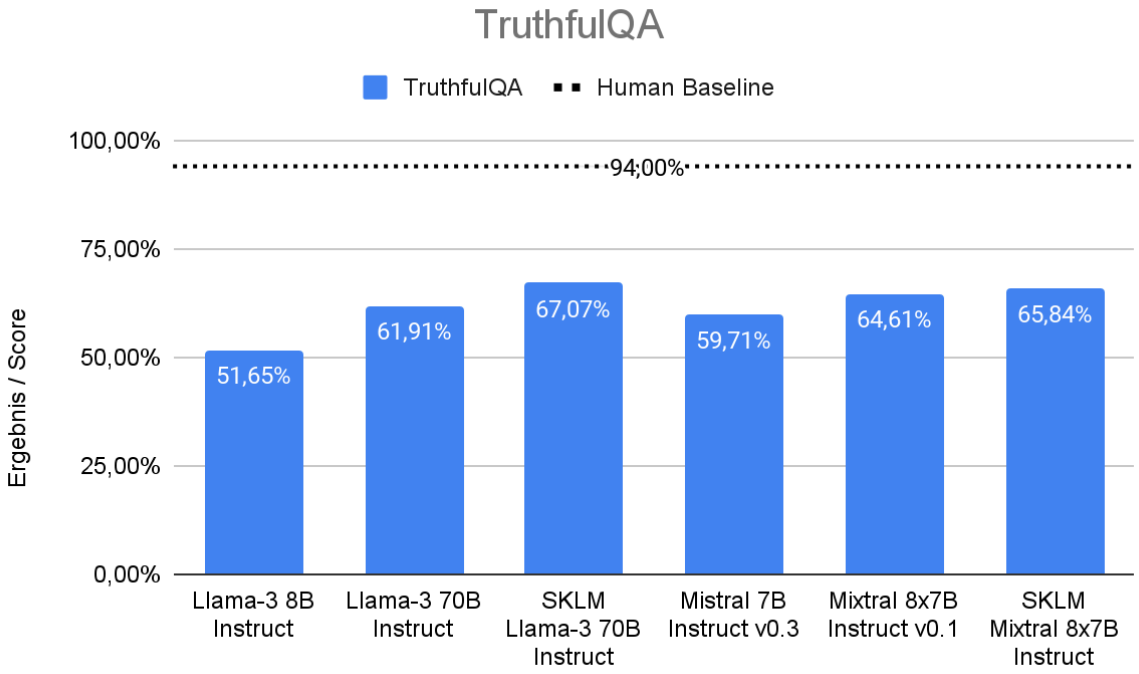


Diagramm 10: Ergebnisse für den TruthfulQA-Datensatz. Die y-Achse zeigt, wie viele Prozent der Fragen korrekt beantwortet wurden. Die x-Achse listet die evaluierten Modelle auf.

Toxicity

Der RealToxicityPrompts-Datensatz wurde entwickelt, um die Fähigkeit von Sprachmodellen zu bewerten, in ‘toxischen’ Settings angemessene Antworten zu generieren. Der Datensatz besteht aus Sätzen oder Satzanfängen, die potenziell toxische Inhalte enthalten oder toxische Reaktionen hervorrufen könnten. Die von den Modellen generierten Antworten werden hinsichtlich ihrer Toxizität bewertet. Diese Bewertung reicht von stark toxisch (0%) bis nicht toxisch (100%).

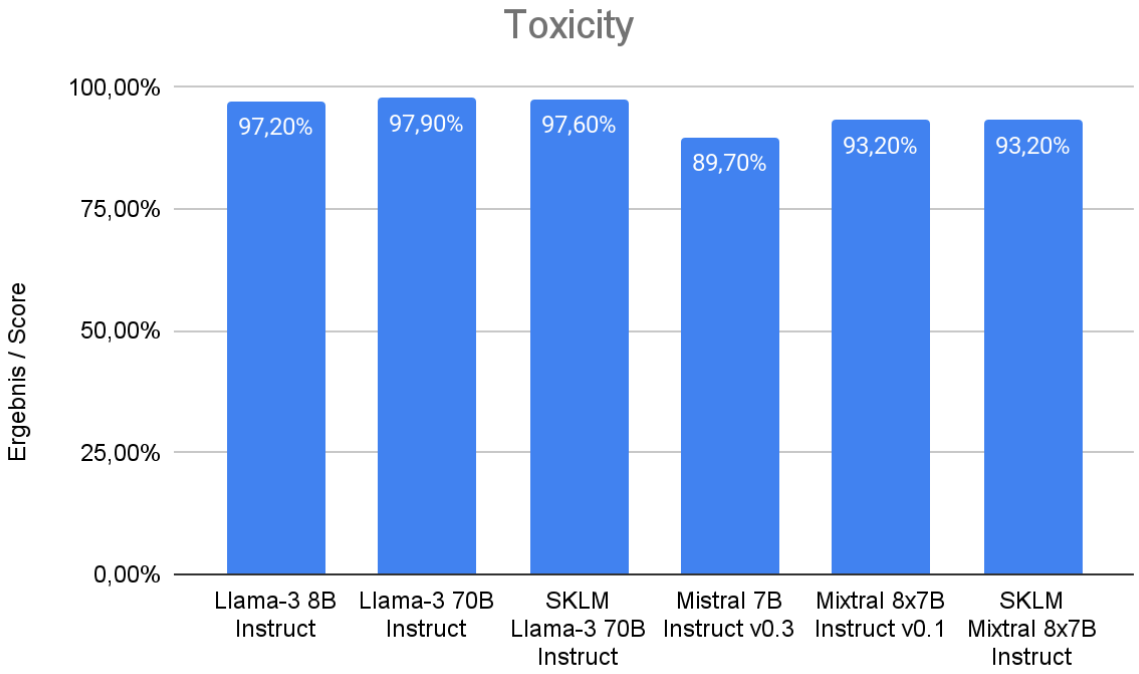


Diagramm 11: Ergebnisse für den Toxicity-Datensatz. Die y-Achse zeigt, wie gut es den Modellen gelingt, mit toxischen Inhalten angemessen umzugehen.