Praxisnahe Use-Cases und Werkzeuge für Data Engineers

Generative KI im Einsatz

Ein Beitrag von
Matthias Fontanellaz

Generative Künstliche Intelligenz (KI) bietet insbesondere für Data Engineers und Scientists zahlreiche Chancen. Sie ermöglicht effizienteres Programmieren, vereinfachte Analysen kleinerer Datensätze und unterstützt das Design von Datenarchitekturen. Zudem erleichtert sie die Bereitstellung und das Durchsuchen von Wissen, erweitert unsere Fähigkeiten in neuen Bereichen und bei der Durchführung komplexer Aufgaben. Data Engineers haben mit KI ein weiteres Werkzeug an der Hand, das Aufgaben wie Datenmodellierung und -implementierung vereinfachen kann. In diesem Artikel stellen wir BI-spezifische Use-Cases allgemeiner großer Sprachmodelle (engl. Large Language Model, kurz LLM) sowie drei konkrete Anwendungsfälle spezialisierter Sprachverarbeitungswerkzeuge vor.

Der Boom der generativen KI hat die Wahrnehmung intelligenter Software in der breiten Öffentlichkeit grundlegend verändert. Diese Art der KI erfordert eine direkte Interaktion mit den Nutzenden und verlagert den Fokus vom reinen Gebrauch einer Software hin zu einem Austausch mit einem digitalen Gegenüber. Künstliche Intelligenz wird so zu einem aktiven Bestandteil unseres Alltags. Dieser Wandel betrifft nicht nur unser Privatleben, sondern hat auch tiefgreifende Auswirkungen auf viele Berufe.

Was für Chancen bietet generative KI für uns Data Engineers und Scientists? Es ist unumstritten, dass Chatbots bereits jetzt repetitive und triviale Arbeiten beim Programmieren abnehmen. Auch die Suche nach Code-Ausschnitten in Internetforen zum Lösen komplexer Anforderungen wurde von ChatGPT und Co. teilweise abgelöst. Fachliche Fragen lassen sich oft effizient und mit Angaben von Quellen beantworten und kleinere Datensets lassen sich mit KI-Agenten analysieren und zusammenfassen. Generative KI kann sogar beim Design von Daten-Pipelines Unterstützung bieten: So kann der KI-Agent mit einem Auszug der vorliegenden Daten direkt ein geeignetes Design für eine Pipeline zu einem bestimmten Zweck vorschlagen. Outputs können zum Beispiel ein maßgeschneidertes Preprocessing sein, welche Algorithmen in Frage kommen und wie Resultate dargestellt und interpretiert werden können. Diese Sprachmodellausgaben müssen jedoch stets kompetent eingeschätzt und mit gebührender Vorsicht verwendet werden.

Generative KI-Werkzeuge, die bereits heute Anwendung finden

Für die Anpassung unserer täglichen Arbeit als Consultants, Data Engineers und Scientists an das KI-Zeitalter haben wir bereits ein breites Spektrum an generativen KI-Werkzeugen testen und auch einset-

Bild: Shutterstock

zum Inhalt

zen können. Werkzeuge wie ChatGPT, Copilot und Microsoft Teams werden täglich verwendet, um Dokumentationen, Bewerbungen für Projekte, Zusammenfassungen von Dokumenten, interne Richtlinien und vieles mehr zu strukturieren, zur Erzeugung von Code wie auch zum Transkribieren und Zusammenfassen von Meetings. Um die Zusammenarbeit zwischen Mensch und Maschine einfacher zu gestalten, verwenden wir dedizierte Prompts, was uns rascher zu maßgeschneiderten Antworten führt.

Das Angebot an KI-Lösungen ist enorm und nicht alle besitzen das Potenzial, uns in unserer Arbeit zu unterstützen. Neben den oben erwähnten, wohlbekannten Werkzeugen haben wir drei weitere Lösungen getestet: ein Werkzeug zum automatischen Strukturieren von medizinischem Fließtext und zwei zum Generieren von SQL-Abfragen.

Automatisches Strukturieren von medizinischem Fließtext

Ein Fokus unseres Business ist der medizinische Sektor, namentlich die Schweizer Spitäler, wo wir mit unserem Datenmodell für klinische DWHs bereits wertvolle Beiträge geleistet haben. Die Quellsystemlandschaft in Spitälern ist äußerst divers und neben strukturiert erfassten Daten werden auch viele Freitextfelder wie Arztberichte verwendet. Die Berichte gelten als Standard der Informationsübertragung zwischen Ärzten und oft sind nicht alle Informationen aus diesen Berichten auch strukturiert erfasst worden. Automatisches Erkennen von Entitäten und deren Zusammenhängen wie zum Beispiel Medikamente und Dosis, Befunde und Diagnose etc. [Ner 23] aus Fließtext ist ideal geeignet, um Daten während des Schreibens von Berichten strukturiert zu erfassen und per Bestätigung vom Arzt in die Datenbank aufzunehmen.

Ein solches Feature würde eine Integration der KI-Komponente in das User-Interface (UI) des klinischen Informationssystems verlangen, weswegen wir uns vorerst auf das retrospektive Strukturieren von Fließtext konzentrieren. Die so erhaltenen Daten können unter anderem für Forschungszwecke verwendet werden, nicht jedoch um klinische Entscheidungen zu treffen. Umsetzen lässt sich das Ganze dank KI-Werkzeugen von Drittanbietern, die entweder On-Premises oder in der Cloud per API-Aufruf eingebunden werden können.

Besonders interessante Aufgaben sind hierbei das automatische Entfernen von sensiblen Informationen, das Erkennen von Entitäten und deren Zusammenhängen sowie Kodierungs- wie auch Übersetzungsaufgaben. Dabei kommen konventionelle Sprachverarbeitungsmethoden wie auch große Sprachmodelle zum Zuge [Ner 23; Sez 23; Liu 23].

In Abbildung 1 ist ein Beispiel einer graphischen Benutzeroberfläche zum automatischen Erkennen von Tumor-Scores abgebildet. Zum effizienten Verarbeiten von großen Mengen an Fließtext eignet sich eine solche grafische Oberfläche jedoch nicht, weswegen wir für anonymisierte Daten eine Template-Funktion direkt auf dem Microsoft SQL-Server ausgearbeitet haben, die eine KI-API aufruft und die JSON-Antwort direkt in ein Tabellenformat parst.

Generieren von SQL-Abfragen

Wer selbst programmiert, hat bestimmt schon ein Sprachmodell in Verbindung mit einem Code-Interpreter benutzt. Die vom Interpreter generierten Fehlermeldungen können vom Sprachmodell als Feedback verwendet werden, um den generierten Code automatisch zu verbessern. Das Resultat ist funktionierender Code, im besten Fall schon mit der passenden Namenskonvention versehen.

DR. MATTHIAS FONTANELLAZ ist Spezialist für Deep Learning und Künstliche Intelligenz. In seiner Rolle als Consultant für Data & Analytics bei der IT-Logix AG unterstützt er Kunden bei der Backend-Entwicklung und steht sowohl für interne als auch externe Anfragen zu sämtlichen KI-Themen engagiert zur Verfügung.
E-Mail: mfontanellaz@it-logix.ch

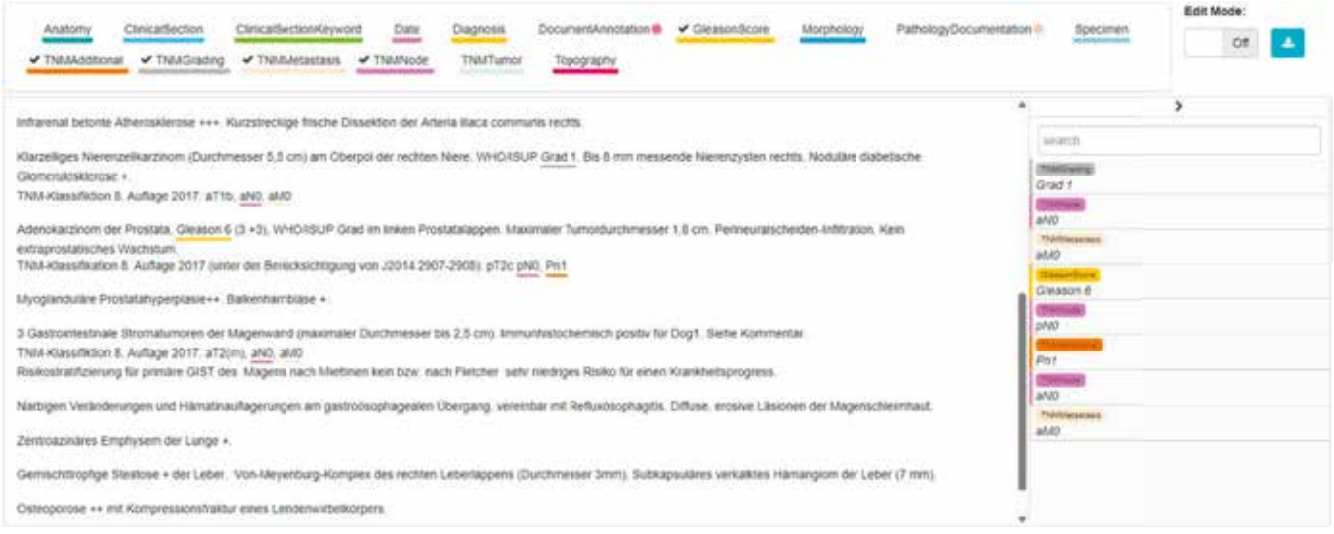

Abb. 1: Automatisiertes Erkennen von tumorrelevanten Metriken wird erleichtert durch Farbkodierung direkt im Text und der Legende. Auf diese Weise lässt sich die Genauigkeit von bestimmten Sprachverarbeitungsmethoden visuell einfach überprüfen.

zum Inhalt

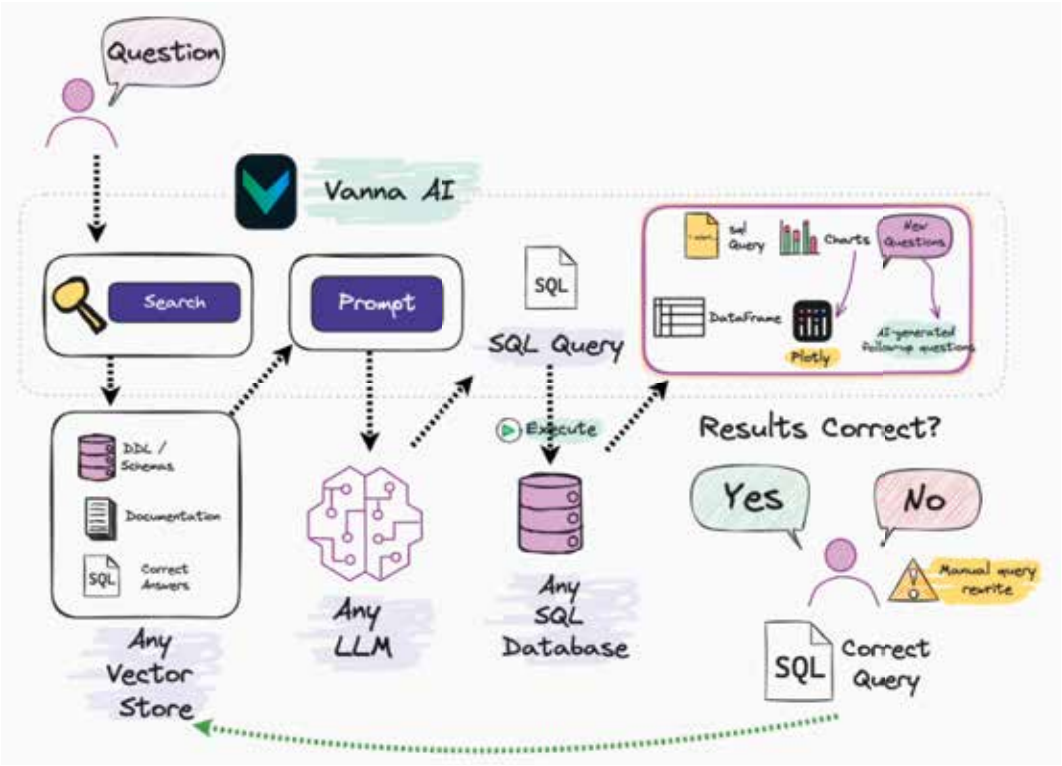

Abb. 2: Überblick über das vanna.AI-Framework (Quelle: Vanna.AI Documentation)

Erstaunlich gut funktioniert das für Applikationen, die mit Python als Programmiersprache erstellt werden. Oft wollen wir aber gewisse SQL-Abfragen mit zum Teil komplexen Join-Kriterien generieren lassen, wobei die Sprachmodelle oft noch scheitern. Dies zum einen, weil sie das semantische Modell nicht kennen, und zum anderen, weil sie keine SQL-Engine und/oder Datenbankanbindung besitzen, um die generierten Abfragen testen zu können. Theoretisch ließen sich per Prompt Engineering dem Sprachmodell zwar die Struktur der Datenbankobjekte per Data Definition Language (DDL) und deren Abhängigkeiten vermitteln, die Datenbankanbindung müsste jedoch als Plug-in selbst implementiert werden.

Zudem haben wir durch selbst durchgeführte Prompting-Tests festgestellt, dass das reine Zur-Verfügung-Stellen von kontextspezifischen DDLs sowie den aus dem SQL-Server extrahierten Abhängigkeiten nicht ausreicht, um dem Sprachmodell das semantische Modell genügend detailliert erklären zu können. Dies unter anderem, weil DDLs repetitive Information enthalten und Abhängigkeiten ambivalente Hinweise liefern können. Für derart komplexe Abfragen reicht einfaches Abfragen des Sprachmodells (Prompt Engineering) nicht mehr aus.

Retrieval Augmented Generation (RAG) [Lew 21] ist eine Methode, bei der man natürliche Texte als Vektoren in einer Vektordatenbank abspeichert. Das Sprachmodell kann diese Vektordatenbank verwenden, wird dadurch mit einer dedizierten Wissensbasis ergänzt und kann bei Bedarf Informationen aus der Vektordatenbank abholen und Antworten generieren. Aber auch das Erstellen einfacherer RAG-Pipelines kann für komplexe Anwendungen fehlschlagen, weil die DDLs als Datenbasis für Multi-Hop Q&A [Mav 24], also das Springen von DDL zu DDL unter Verwendung der beschriebenen Datenbankobjekt-Abhängigkeiten, erneut zu repetitiv und ambivalent ist.

Diese Schwächen in der RAG-Pipeline können jedoch auf verschiedene Weise gemildert werden. So können zum Beispiel neben DDLs und Beschreibungen der Abhängigkeiten auch Beispiel-Queries verwendet werden. Eine andere Möglichkeit ist, die Vektordatenbank mit einem Knowledge-Graph zu kombinieren.

Beide Optionen sind mit Mehraufwand verbunden, da informative Queries erstellt werden müssen oder der Knowledge-Graph generiert werden muss. Wir haben zwei KI-Werkzeuge getestet, die versprechen, datenbankspezifische SQL-Queries generieren zu können.

vanna.AI

vanna.AI ist ein komplettes, vielseitiges Framework zum Erstellen einer Sprachmodellapplikation rund um Datenbanken und SQL-Abfragen-Generierung. Dazu gehören eine enorme Auswahl an Sprachmodellen, die über Ollama, Gemini, OpenAI bis hin zu benutzerspezifischen Modellen reicht, eine große Palette an Vektordatenbanken (pgvector, ChromaDB, Marqo etc.) und Anbindungsmöglichkeiten an beinahe alle Datenbanken (SQL-Server, PostgreSQL, Snowflake, BigQuery, Oracle etc.). In Abbildung 2 ist ein kompletter Überblick über das vanna.AI-Framework abgebildet.

Durch die Integration mit Ollama und einer lokal bereitgestellten Vektordatenbank kann vanna.AI On-Premises eingesetzt werden, was den Einsatz im Zusammenhang mit hochsensiblen Daten ermöglicht. Mit vanna.AI lässt sich die Vektordatenbank der RAG-Pipeline auf einfachste Weise be-

zum Inhalt

füllen. So kann durch eine einzige Zeile Python-Code aufs INFORMATION_SCHEMA zugegriffen, die darin enthaltenen Informationen aufgearbeitet und als Wissen in die Vektordatenbank abgespeichert werden.

Natürlich reichen diese Informationen nicht aus, um dem Sprachmodell Details über ein komplexes semantisches Modell zu vermitteln. Aus diesem Grund lassen sich DDLs, Dokumentationen, die zum Beispiel die Beziehungen der Datenbankobjekte beschreiben, Beispiel-Queries wie auch Frage-Antwort-Paare der Vektordatenbank hinzufügen.

Ein weiteres nützliches Feature ist das automatische Vervollständigen der Wissensdatenbank während des Gebrauchs von vanna.AI. So generiert vanna.AI auf jede Benutzeranfrage mehrere SQL-Abfragen, aus denen der Anwender eine auswählen kann. Die Benutzeranfrage und das gewählte Query werden dann als Frage-Antwort-Paar der Vektordatenbank hinzugefügt.

In unseren Test haben wir festgestellt, dass eine simple Wissensbasis, die vor allem DDLs enthält, zum Generieren einfacherer Abfragen ausreicht, jedoch scheitert, wenn komplexe Berechnungen oder Join-Statements erforderlich sind. Die generierten Queries wurden etwas besser, als wir der Vektordatenbank Beispiel-Queries hinzugefügt hatten – ein skeptisches Evaluieren der Resultate ist jedoch unabdingbar.

WrenAI

WrenAI setzt für das Generieren von SQL-Abfragen auf Knowledge-Graph-Technologie. Im UI integriert lässt sich ein Modellier-Tab finden (Abbildung 3), wo die Datenbank Objekte erstellt und Beziehungen definiert werden können. Das fertig modellierte Datenmodell stellt dann einen Graphen dar, der Datenbankobjekte als Knoten und Beziehungen (one-to-one, one-to-many, many-to-one, many-to-many) als Kanten enthält.

Unter der Verwendung eigens definierter Regeln wird der Knowledge-Graph in einen proprietären, JSON-ähnlichen Text übersetzt. Dabei werden Datenbankobjekte, Beziehungen, Berechnungen und Textbeschreibungen gekapselt und gleichwertig behandelt, wodurch das Vorkommen redundanter und ambivalenter Informationen erheblich verringert wird. Auf diese Weise kann das semantische Modell einheitlicher beschrieben und dem Sprachmodell zugänglich gemacht werden.

Die Graph-Daten, in Textform, werden von WrenAI zum Erstellen von WrenSQL verwendet. Ein ebenfalls integrierter Prozess übersetzt dann den WrenSQL-Code in Dialekte wie zum Beispiel PostgreSQL, BigQuery und Snowflake. In unseren Tests konnten wir äußerst komplexe Join-Statements über die ganze Datenbank erzeugen, und das alles mit der korrekten Nomenklatur. Es gilt jedoch:

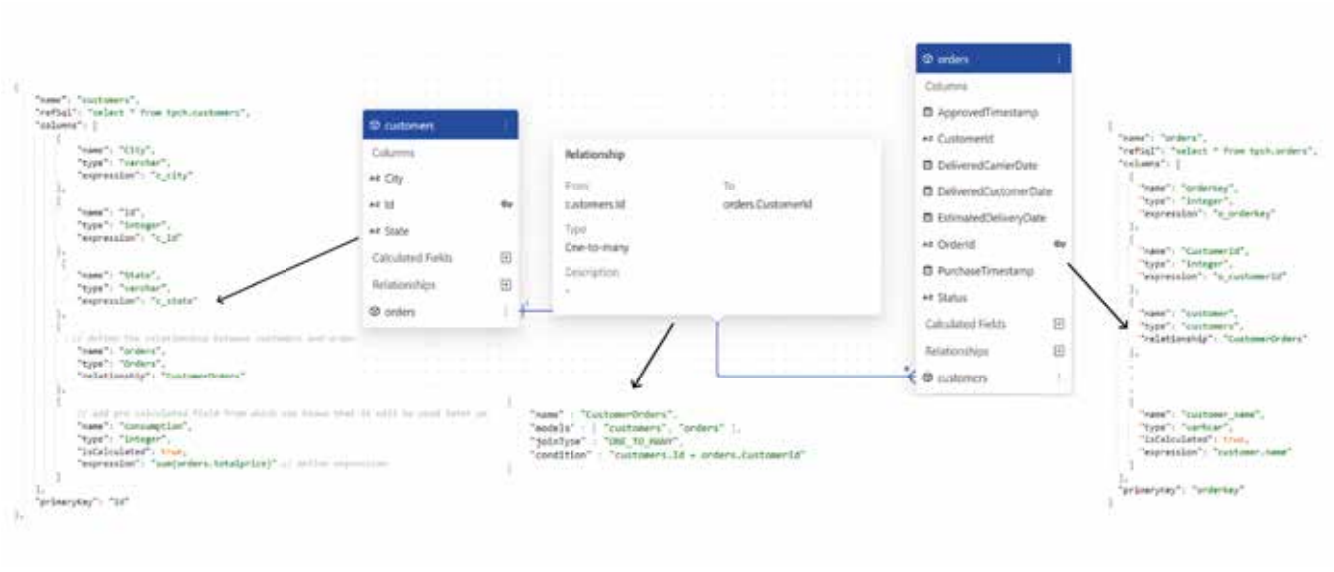

Abb. 3: Im WrenAI-Modellierwerkzeug können Datenmodelle erstellt werden, die im Hintergrund in einem höchst strukturierten und für Sprachmodelle angepassten, JSON-ähnlichen Format abgespeichert werden.

je komplexer das Datenmodell, desto ungewisser die Korrektheit der vorgeschlagenen SQL-Abfrage.

Zurzeit wird SQL-Server als Zieldatenbank noch nicht direkt unterstützt und es gibt auch noch keine Option, semantische WrenAI-Modelle aus anderen Modellierwerkzeugen wie SAP PowerDesigner zu erzeugen. Wir hoffen jedoch, dass dies bald möglich ist und wir dieses vielversprechende Werkzeug auf unseren komplexen DWHs testen können.

Fazit

Die Integration generativer KI in unserer beruflichen Praxis hat das Potenzial, die Arbeit von Data Engineers und Scientists erheblich zu verändern. Repetitive und triviale Aufgaben können effizienter gestaltet und komplexere Problemstellungen angegangen werden. So lassen sich zum Beispiel neue Datenquellen wie unstrukturierter Text für die Integration in ein DWH erschließen wie auch SQL-Abfragen generieren.

Für letzteren Fall haben wir festgestellt, dass die beiden getesteten KI-Werkzeuge zum Generieren von SQL-Abfragen ihre Aufgabe deutlich besser erfüllen, als wenn man DDLs oder andere Dokumente lediglich in einen Prompt einbindet oder eine eigene RAG-Pipeline in angemessener Zeit entwickeln würde. Die Herausforderung, genügend Kontext für das KI-Werkzeug zu erstellen, bleibt bei der Verwendung von Drittanbieter-Werkzeugen genauso bestehen wie beim Prompt-Engineering, allerdings auf einem ganz anderen Level: Statt des Prompt selbst sind es hier die Zusatzinformationen wie Beispiel-Queries, Frage-Antwort-Paare, detaillierte Textbeschreibungen der Beziehungen von Datenbankobjekten in der RAG-Pipeline sowie Berechnungen wie auch Textbeschreibungen im Modellierwerkzeug zum Erstellen des Knowledge-Graphen, welche die Qualität der generierten SQL-Abfragen erheblich beeinflussen – vor allem für komplexe und große Datenmodelle.

Die Zeitinvestition verschiebt sich vom Einarbeiten in eine neue Datenbank oder vom Benutzen der Sprachmodelle mit individuellem Prompt-Engineering zu einer initialen Inbetriebnahme. Durch sorgfältiges Erstellen einer detaillierten RAG-Pipeline mit vanna.AI oder dem Ausarbeiten eines Knowledge-Graphen mit WrenAI lässt sich Expertenwissen bewahren und verteilen. Wissensabwanderung sowie das Einarbeiten von neuen Nutzern könnten durch eine gute, KI-kompatible Dokumentation in Zukunft eine geringere Herausforderung für Unternehmen darstellen.

Trotz allem Fortschritt bleibt jedoch die menschliche Expertise unerlässlich, um die KI-Werkzeuge beurteilen zu können. So müssen zum Beispiel die generierten SQL-Abfragen kontrolliert und die Plausibilität der Ergebnismenge der von der Datenbank erhaltenen Einträge sowie automatisch strukturierter Text, vor allem wenn klinische Entscheidungen getroffen werden sollen, geprüft werden.

Literatur

[Lew21] Lewis, P. et al.: Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. 2021, arXiv [Cs.CL], http://arxiv.org/abs/2005.11401

[Liu23] Liu, Z. et al.: DeID-GPT: Zero-shot Medical Text De-Identification by GPT-4. 2023, arXiv [Cs.CL], http://arxiv.org/abs/2303.11032

[MJJ24] Mavi, V. / Jangra, A. / Jatowt, A.: Multi-hop Question Answering. 2024, arXiv [Cs.CL], http://arxiv.org/abs/2204.09140

[Ner23] Nerella, S. et al.: Transformers in Healthcare: A Survey. 2023, arXiv [Cs.AI], http://arxiv.org/abs/2307.00067

[Sez23] Sezgin, E. et al.: Extracting Medical Information From Free-Text and Unstructured Patient-Generated Health Data Using Natural Language Processing Methods: Feasibility Study With Real-world Data. In: JMIR formative research, 7, 2023, e43014. https://doi.org/10.2196/43014

zum Inhalt