



Verarbeitung multimodaler medizinischer PDF-Dokumente



Masterarbeit

Carla Kalaf

Matrikel-Nr.: 219202923

Abgabedatum: 03.08.2026

Erstgutachter:

Dr.-Ing. Hannes Grunert

Zweitgutachter:

Dr.-Ing. Robin Nicolay

Kurzfassung

Medizinische Dokumente verbinden Fließtext, Tabellen und Abbildungen zu einer für die maschinelle Auswertung schwer zugänglichen Form und liegen häufig als PDF vor. Diese Arbeit konzipiert und erprobt einen lokal ausführbaren ETL-Prozess, der solche multimodalen medizinischen PDF-Dokumente in ein strukturiertes, abfragbares Datenmodell überführt. Der Prozess arbeitet vollständig ohne externe Dienste, sodass die verarbeiteten Dokumente die lokale Umgebung nicht verlassen und auch datenschutzsensible Inhalte verarbeitet werden können.

Der Prozess gliedert sich in drei Phasen. In der Extraktionsphase überführt ein Dokumentenwerkzeug den Text- und Tabelleninhalt in eine einheitliche Markdown-Repräsentation, während ein lokales Vision-Language-Modell die enthaltenen Abbildungen in textuelle Beschreibungen umsetzt. In der Transformationsphase gewinnt ein lokal ausgeführtes Sprachmodell aus dieser Repräsentation in zwei aufeinander aufbauenden Schritten zunächst typisierte Entitäten und anschließend die kausalen Beziehungen zwischen ihnen. Die Ladephase überführt diese Ergebnisse in ein relationales Datenmodell, dessen Schema gemeinsam mit der im Sprachmodell vorgegebenen Ausgabestruktur aus einer zentralen Definition hervorgeht.

Die beispielhafte Anwendung auf drei medizinische Publikationen mit unterschiedlich strukturierten Bestandteilen zeigt, dass der Prozess durchgängig eine befüllte, abfragbare Repräsentation erzeugt. Die textuelle Verarbeitung trägt am weitesten und erfasst auch implizite kausale Beziehungen; die Grenzen liegen bei stark verschachtelten Tabellen, bei Diagrammen mit langen, an anderen Verbindungen ansetzenden Pfeilen, bei der auf einzelne Sätze beschränkten Kausalitätsextraktion und bei der größenbasierten Filterung von Grafikelementen. Die Arbeit ordnet diese Beobachtungen ein und leitet Ansätze für weiterführende Arbeiten ab.

Schlagwörter: ETL-Prozess, Informationsextraktion, Kausalitätsextraktion, Large Language Models (LLM), lokale Verarbeitung, medizinische Dokumentverarbeitung, multimodale PDF-Verarbeitung

Abstract

Medical documents combine running text, tables, and figures into a form that is difficult to process automatically, and they are frequently available as PDF. This thesis designs and demonstrates a locally executable ETL process that converts such multimodal medical PDF documents into a structured, queryable data model. The process operates entirely without external services, so that the processed documents never leave the local environment and privacy-sensitive content can be handled as well.

The process consists of three phases. In the extraction phase, a document tool converts the textual and tabular content into a uniform Markdown representation, while a local vision-language model turns the contained figures into textual descriptions. In the transformation phase, a locally executed language model derives typed entities from this representation and, in a second step building on the first, the causal relations between them. The loading phase transfers these results into a relational data model whose schema is derived, together with the output structure prescribed for the language model, from a single central definition.

The exemplary application to three medical publications with differently structured components shows that the process consistently produces a populated, queryable representation. Textual processing carries furthest and captures implicit causal relations as well; the limitations lie in heavily nested tables, in diagrams with long arrows branching off other connections, in the sentence-level scope of causal extraction, and in the size-based filtering of graphical elements. The thesis situates these observations and derives directions for further work.

Keywords: ETL process, information extraction, causal extraction, large language models (LLM), local processing, medical document processing, multimodal PDF processing

Inhaltsverzeichnis

Kurzfassung	II
Abstract	III
Inhaltsverzeichnis	IV
1 Einleitung	1
1.1 Zielsetzung und Forschungsfragen	2
1.2 Aufbau der Arbeit	2
2 Theoretische Grundlagen	5
2.1 ETL-Prozess	5
2.2 PDF-Dokumente	8
2.2.1 Multimodale PDF-Dokumente	8
2.2.2 Medizinische PDF-Dokumente als Datenquelle	10
2.3 Extraktion strukturierter Informationen	12
2.3.1 Informationsextraktion	12
2.3.2 Kausalitätsextraktion	13
2.4 Large Language Models (LLMs)	15
2.5 Evaluationsprinzipien	16
3 Stand der Technik und Forschung	19
3.1 Vorgehen bei der Suche	19
3.2 Werkzeuge zur Verarbeitung multimodaler PDF-Dokumente	20
3.2.1 Regel- und Heuristik-basierte Werkzeuge	20
3.2.2 Deep-Learning-basierte Werkzeuge	27
3.2.3 KI-gestützte Werkzeuge	32
3.2.4 Vergleich der Werkzeuge	44
3.3 Lokale LLMs für Informations- und Kausalitätsextraktion	46
3.3.1 Llama	47
3.3.2 DeepSeek	48
3.3.3 gpt-oss	49
3.3.4 Qwen	50
3.3.5 Mistral	51
3.3.6 Vision-Language-Modelle für die Bildbeschreibung	53
3.3.7 Vergleich der LLMs	53
4 Vergleichende Analyse der KI-gestützten Werkzeuge	57
4.1 Technischer Vergleich der JSON-Ausgaben	57
4.2 Vergleich der Markdown-Ausgaben	64
4.3 Vergleich der Bild- und Abbildungsextraktion	66

5	Konzeption des ETL-Prozesses	71
5.1	Systemanforderungen.....	71
5.2	Gesamtarchitektur des ETL-Prozesses.....	72
5.3	Auswahl geeigneter Werkzeuge und Modelle	75
5.3.1	Werkzeug für die Extraktionsphase.....	75
5.3.2	Modell für die Transformationsphase	76
5.4	Extraktionsphase: Verarbeitung multimodaler PDF-Dokumente.....	77
5.5	Transformationsphase: LLM-basierte Informations- und Kausalitätsextraktion.....	79
5.6	Ladephase: Überführung in ein strukturiertes Datenmodell	82
6	Prototypische Implementierung	85
6.1	Technische Rahmenbedingungen.....	85
6.1.1	Hardware und Ausführungsumgebung.....	85
6.1.2	Verwendete Modelle	86
6.1.3	Eingabedokumente	87
6.2	Extraktionsphase	87
6.2.1	Dokumentkonvertierung mit Docling	87
6.2.2	Bildbeschreibung mit dem Vision-Language-Modell.....	88
6.2.3	Robustheit gegenüber realen Dokumenteigenschaften.....	89
6.2.4	Ergebnis der Extraktionsphase	89
6.3	Transformationsphase	92
6.3.1	Zentrale Schemadefinition und Erzwingung gültiger Ausgaben.....	92
6.3.2	Erster Schritt: Extraktion der Entitäten	95
6.3.3	Zweiter Schritt: Extraktion der kausalen Beziehungen	96
6.3.4	Verarbeitung im gemeinsamen Kontext	96
6.3.5	Ergebnis der Transformationsphase	97
6.4	Ladephase.....	97
6.4.1	Ableitung des Zielschemas aus der zentralen Schemadefinition	97
6.4.2	Dreistufiges Laden entlang der Fremdschlüssel.....	98
6.4.3	Vergabe eindeutiger Bezeichner	99
6.4.4	Ergebnis der Ladephase	100
6.5	Gesamtablauf des Prototyps.....	100
7	Beispielhafte Anwendung	103
7.1	Vorgehen bei der Anwendung.....	103
7.2	Verarbeitung textueller Bestandteile.....	104
7.2.1	Extraktion der Entitäten.....	104
7.2.2	Extraktion der kausalen Beziehungen	105
7.2.3	Grenzen der Kausalitätsextraktion	106

7.3	Verarbeitung tabellarischer Bestandteile	106
7.4	Verarbeitung bildlicher Bestandteile	108
7.4.1	Klar strukturierte Flussdiagramme	108
7.4.2	Komplexes Diagramm mit abzweigenden Beziehungen.....	110
7.4.3	Grenze der vorgelagerten Filterung	111
7.5	Zusammenfassende Betrachtung	111
8	Fazit und Ausblick	115
8.1	Zusammenfassung	115
8.2	Beantwortung der Forschungsfragen	116
8.3	Grenzen	117
8.4	Ausblick	118
	Literaturverzeichnis	VII
	Abbildungsverzeichnis.....	XIII
	Tabellenverzeichnis.....	XIV
	Listing-Verzeichnis	XV
	Selbständigkeitserklärung	XVII

1 Einleitung

Mehr als die Hälfte der medizinischen Information in elektronischen Patientenakten liegt in unstrukturierter Form vor [1]. Klinisch relevante Inhalte wie Befunde, Symptome oder Verlaufsbeschreibungen liegen in Freitexten oder PDF-Dokumenten vor und sind dadurch nicht unmittelbar als strukturierte Daten für nachgelagerte Analyse- oder Entscheidungsprozesse verfügbar [2,3]. Medizinische PDF-Dokumente sind dabei typischerweise multimodal aufgebaut: Sie verbinden Fließtext mit Tabellen, Layoutstrukturen und weiteren visuellen Bestandteilen, deren logische Struktur nicht explizit hinterlegt ist und erst rekonstruiert werden muss [4,5]. Die Überführung unstrukturierter klinischer Inhalte in strukturierte, maschinenlesbare Repräsentationen ist daher eine grundlegende Voraussetzung, um medizinische Informationen automatisiert nutzbar zu machen [6].

Eine zentrale Herausforderung ergibt sich aus der besonderen Schutzbedürftigkeit medizinischer Daten. Gesundheitsdaten unterliegen hohen Anforderungen an Datenschutz und Datensouveränität, was die Verarbeitung über externe, cloudbasierte Dienste einschränkt [7]. Vor diesem Hintergrund gewinnen lokal ausführbare Verarbeitungsansätze an Bedeutung, da sie eine Verarbeitung sensibler Dokumente ohne Übertragung an externe Dienste ermöglichen [8]. Insbesondere kleinere, lokal betreibbare Sprachmodelle sind relevant für datenschutzsensible und ressourcenbeschränkte klinische Umgebungen [7].

Über die Extraktion einzelner Fakten hinaus sind im medizinischen Bereich kausale Zusammenhänge von besonderer Bedeutung, da medizinische Erklärungen, Diagnosen und Therapieentscheidungen wesentlich auf Ursache-Wirkungs-Beziehungen beruhen. Die Erkennung kausaler Beziehungen ist eine grundlegende Aufgabe der biomedizinischen Informationsextraktion [9]. Zugleich ist die Extraktion kausaler Zusammenhänge aus medizinischen Texten anspruchsvoll und bislang nicht vollständig gelöst, da Kausalität häufig implizit oder satzübergreifend auftritt und große Sprachmodelle bei der Kausalitätsextraktion hinter spezialisierten Verfahren zurückbleiben [10]. Große Sprachmodelle eignen sich grundsätzlich, um kausale Relationen aus medizinischen Texten zu extrahieren [11]. Insgesamt ergibt sich damit ein Zusammenspiel mehrerer Anforderungen: Medizinische Dokumente sollen trotz ihrer multimodalen und unstrukturierten Form verarbeitet, aufgrund ihrer Sensibilität lokal gehalten und über die reine Faktenextraktion hinaus auch hinsichtlich kausaler Zusammenhänge erschlossen werden.

1.1 Zielsetzung und Forschungsfragen

Ziel dieser Arbeit ist die Konzeption und prototypische Umsetzung eines lokal ausführbaren ETL-Prozesses, der multimodale medizinische PDF-Dokumente verarbeitet und mithilfe lokaler Large Language Models (LLMs) sowohl strukturierte Informationen als auch kausale Beziehungen extrahiert und in ein strukturiertes Datenmodell überführt. Die Verarbeitung soll dabei vollständig lokal erfolgen, um den besonderen Anforderungen an den Schutz medizinischer Daten gerecht zu werden. Der wissenschaftliche Beitrag liegt in der Verbindung von multimodaler Dokumentverarbeitung und LLM-gestützter Kausalitätsextraktion innerhalb einer klar definierten ETL-Architektur.

Zur Erreichung dieser Zielsetzung werden drei Forschungsfragen betrachtet:

1. Welche Werkzeuge eignen sich für die Verarbeitung multimodaler medizinischer PDF-Dokumente?
2. Welche lokal ausführbaren LLMs eignen sich für die Informations- und Kausalitätsextraktion aus medizinischen Dokumentinhalten?
3. Wie lassen sich die ausgewählten Werkzeuge und Modelle in einem lokal ausführbaren ETL-Prozess zusammenführen, und wie verhält sich dieser bei der Anwendung auf medizinische PDF-Dokumente mit unterschiedlichen Bestandteilen?

Die erste Forschungsfrage betrifft die Extraktionsphase und zielt auf die Auswahl geeigneter Werkzeuge zur Überführung multimodaler PDF-Inhalte in eine strukturierte, maschinenlesbare Repräsentation. Die zweite Forschungsfrage betrifft die Transformationsphase und untersucht, welche lokal ausführbaren Modelle sich für die Extraktion von Informationen und kausalen Zusammenhängen eignen. Die dritte Forschungsfrage führt beide Aspekte zusammen und betrachtet die Integration in einen durchgängigen ETL-Prozess sowie dessen prototypische Umsetzung und exemplarische Anwendung auf medizinische PDF-Dokumente mit unterschiedlichen Bestandteilen.

1.2 Aufbau der Arbeit

Die vorliegende Arbeit gliedert sich in acht Kapitel. Im Anschluss an diese Einleitung legt Kapitel [2](#) die theoretischen Grundlagen zu ETL-Prozessen, multimodalen und medizinischen PDF-Dokumenten, der Informations- und Kausalitätsextraktion, Large Language Models sowie Evaluationsprinzipien. Kapitel [3](#) stellt den Stand der Technik

dar und vergleicht Werkzeuge zur Verarbeitung multimodaler PDF-Dokumente sowie lokal ausführbare Large Language Models für die Informations- und Kausalitätsextraktion. Kapitel 4 analysiert die betrachteten Werkzeuge vergleichend anhand ihrer Ausgaben an einer gemeinsamen Beispielseite. Kapitel 5 überführt diese Erkenntnisse in die Konzeption des ETL-Prozesses. Kapitel 6 beschreibt die prototypische Implementierung und die Verarbeitung eines medizinischen Beispieldokuments. Kapitel 7 wendet den ETL-Prozess exemplarisch auf medizinische PDF-Dokumente mit unterschiedlichen Bestandteilen an, und Kapitel 8 fasst die Arbeit zusammen und gibt einen Ausblick.

2 Theoretische Grundlagen

Dieses Kapitel beschreibt die theoretischen Grundlagen, die für die Konzeption eines lokalen ETL-Prozesses zur Verarbeitung multimodaler medizinischer PDF-Dokumente erforderlich sind. Es schafft die fachliche und technische Basis für die nachfolgende Betrachtung des Stands der Technik und die spätere Entwicklung des ETL-Prozesses. Zunächst wird in Abschnitt [2.1](#) der ETL-Prozess als grundlegendes Verarbeitungskonzept erläutert. Anschließend werden in Abschnitt [2.2](#) multimodale und medizinische PDF-Dokumente als Datenquelle eingeordnet. Darauf aufbauend wird in Abschnitt [2.3](#) die Informationsextraktion einschließlich der Kausalitätsextraktion beschrieben. Abschließend werden in Abschnitt [2.4](#) Large Language Models sowie in Abschnitt [2.5](#) grundlegende Evaluationsprinzipien betrachtet.

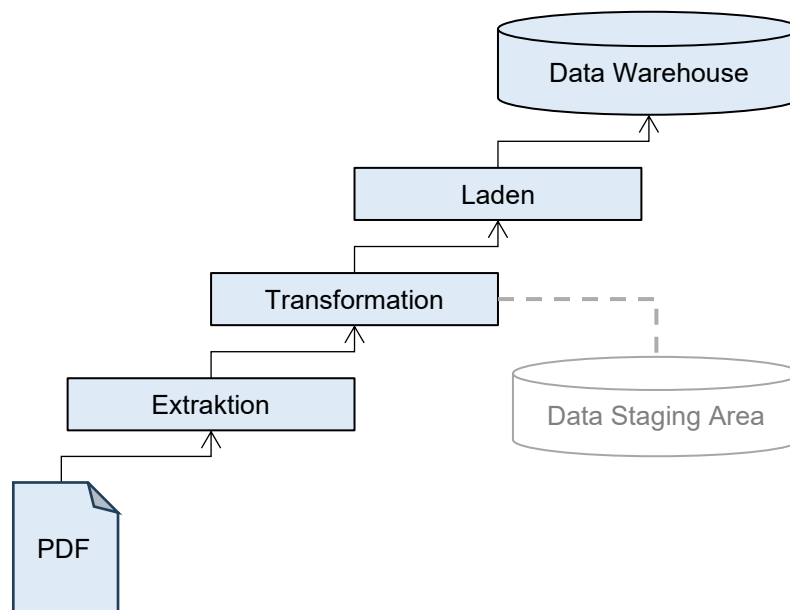
2.1 ETL-Prozess

Der Begriff ETL steht für Extract, Transform und Load und beschreibt einen mehrstufigen Prozess zur Überführung von Daten aus unterschiedlichen Quellen in ein Zielsystem. In klassischen Data-Warehouse-Architekturen werden Daten zunächst aus operativen Quellsystemen extrahiert, anschließend in einer Data Staging Area bereinigt, vereinheitlicht und transformiert und schließlich in ein Data Warehouse geladen [12,13]. Die Data Staging Area dient dabei als Arbeitsbereich zwischen Quell- und Zielsystem, in dem Daten zwischengespeichert, vereinheitlicht und für die weitere Verarbeitung vorbereitet werden [13]. ETL-Prozesse bilden damit einen zentralen Bestandteil der Datenintegration, da sie Daten aus heterogenen Quellen zusammenführen und in eine konsistente, auswertbare Struktur überführen [14]. Über die drei Schritte Extract, Transform und Load hinaus umfassen ETL-Systeme auch deren technische Koordination, die Abbildung zwischen Quell- und Zielstrukturen sowie die Dokumentation von Transformationsregeln und Zielschemata über Metadaten [13,15].

Wie in Abbildung [1](#) dargestellt, durchläuft ein ETL-Prozess die drei Phasen Extraktion, Transformation und Laden. Die Extraktionsphase umfasst die Gewinnung der relevanten Daten aus den Quellsystemen, die von relationalen Datenbanken über semi-strukturierte Formate bis hin zu Dokumenten und Webquellen reichen können. In der Transformationsphase werden die extrahierten Daten bereinigt, harmonisiert und an das Zielschema angepasst; dazu gehören die Vereinheitlichung von Formaten, die Behandlung fehlerhafter oder fehlender Werte sowie die Auflösung von Konflikten auf

Schema-, Datensatz- und Wertebene [12]. Gerade bei heterogenen Datenquellen ist dieser Schritt wesentlich, da Unterschiede nicht nur auf technischer, sondern auch auf struktureller und semantischer Ebene bestehen können [12,16]. In der Ladephase werden die transformierten Daten schließlich aus der Data Staging Area in ein Zielsystem wie ein Data Warehouse oder eine relationale Datenbank überführt, wo sie für nachgelagerte Anwendungen wie Reporting, Analyse oder strukturierte Abfragen bereitstehen [12,17].

Abbildung 1: Grundlegende Architektur eines ETL-Prozesses (in Anlehnung an [20])



Die einzelnen Phasen können dabei unterschiedlich ausgestaltet sein. Bei einer initialen Extraktion werden Daten erstmals aus den operativen Quellsystemen übernommen, um ein Data Warehouse zu befüllen; anschließend verarbeitet eine inkrementelle Extraktion nur noch neu hinzugekommene oder geänderte Daten seit der letzten Extraktion, wofür unter anderem Audit-Spalten, Datenbank-Logs, Zeitstempel oder Delta-Verfahren genutzt werden [13]. Hinsichtlich des Auslösezeitpunkts kann ETL periodisch in festen Zeitintervallen, ereignis- beziehungsweise triggerbasiert oder kontinuierlich in Real-Time-Data-Warehouse-Szenarien angestoßen werden. Zusätzlich wird zwischen Push- und Pull-Ansätzen unterschieden: Beim Push-Ansatz lösen die Datenquellen die Übertragung aus und senden Daten an das Data Warehouse, während beim Pull-Ansatz die ETL-Komponente die Daten aktiv aus den Quellen abfragt [18].

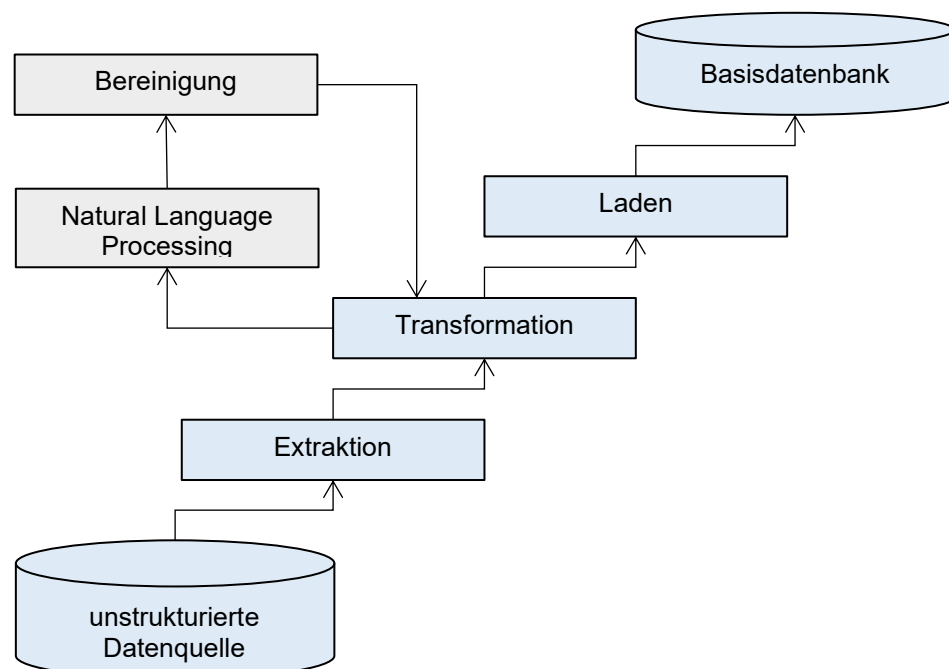
Auch Transformation und Laden besitzen unterschiedliche Ausprägungen. In der Transformationsphase werden die Daten nicht nur syntaktisch angepasst, sondern bereinigt, integriert und an das Zielmodell angepasst, sodass korrekte, vollständige, konsistente und eindeutige Daten entstehen [13]; in medizinischen Data-Warehouse-

Szenarien kann dies beispielsweise bedeuten, lokale Datenmodelle und Codes in ein standardisiertes Zielmodell mit standardisierten Codes zu überführen. Die Ladephase kann als vollständiges Neuladen oder als inkrementelle Aktualisierung umgesetzt werden: Beim vollständigen Neuladen wird ein Zielbestand vollständig ersetzt, während inkrementelle Verfahren neue, geänderte oder gelöschte Datensätze gezielt aktualisieren und dafür beispielsweise Audit- oder Markierungsspalten verwenden [15].

Während ETL ursprünglich stark mit strukturierten Daten verbunden war, hat sich der Anwendungsbereich durch die zunehmende Verfügbarkeit unstrukturierter Daten erweitert. Unstrukturierte Daten besitzen kein festes Schema, liegen in unterschiedlichen Formaten vor und sind nicht standardisiert, wodurch ihre relevanten Informationen nicht bereits in klar definierten Feldern vorliegen [19]. Für solche Daten muss der ETL-Prozess daher um Schritte ergänzt werden, die Informationen zunächst aus den Quellen identifizieren und in eine maschinenlesbare Struktur überführen. In diesem Zusammenhang gewinnt die Informationsextraktion innerhalb von ETL-Prozessen an Bedeutung: Aus Rohdaten werden Strukturen wie Entitäten und relevante Informationseinheiten extrahiert, integriert und in einer strukturierten Datenbank nutzbar gemacht [17].

Für unstrukturierte Daten erweitert ein spezialisierter ETL-Prozess diese Grundarchitektur um einen Vorverarbeitungsschritt: Eine NLP-gestützte Analyse und Bereinigung überführt die unstrukturierten Inhalte während der Transformation in eine strukturierte Repräsentation, bevor das Laden in die Basisdatenbank erfolgt, wie in Abbildung 2 dargestellt [20].

Abbildung 2: Spezialisierter ETL-Prozess für unstrukturierte Daten (in Anlehnung an [20])



Besonders anspruchsvoll ist diese Verarbeitung im medizinischen Bereich, wo relevante Informationen häufig in strukturierten, semi-strukturierten und unstrukturierten Quellen zugleich vorliegen. Klinisch relevante Informationen liegen oft nur in Freitexten oder PDF-Dokumenten vor und können daher nicht unmittelbar durch Standard-ETL-Werkzeuge verarbeitet werden; solche Inhalte lassen sich über zusätzliche Vorverarbeitungsschritte wie NLP, Bildverarbeitung oder OCR in strukturierte Repräsentationen überführen und anschließend in ein Clinical Data Warehouse laden [2]. Entsprechend extrahieren NLP-Methoden in der Transformationsphase Begriffe und Werte aus unstrukturierten oder semi-strukturierten Krankenhausinformationssystemen [21]. LLMs erweitern ETL-Prozesse über klassische NLP-Verfahren hinaus: Lokal eingesetzte offene Modelle überführen unstrukturierte Dokumente in ein relationales Datenbankschema. Der Einsatz von LLMs ist dabei weniger ein Ersatz des gesamten ETL-Prozesses als eine Komponente innerhalb der Extraktions- und Transformationsphase, die semantische Informationen aus unstrukturierten Textquellen ableitet [22].

2.2 PDF-Dokumente

Dieser Abschnitt ordnet PDF-Dokumente als multimodale Datenquelle ein und beschreibt die daraus entstehenden Anforderungen an die technische Verarbeitung. Zunächst werden in Abschnitt [2.2.1](#) multimodale PDF-Dokumente hinsichtlich ihrer textuellen, visuellen und layoutbezogenen Bestandteile betrachtet. Anschließend wird in Abschnitt [2.2.2](#) erläutert, welche Besonderheiten medizinische PDF-Dokumente als Datenquelle aufweisen.

2.2.1 Multimodale PDF-Dokumente

Multimodale Dokumente kombinieren mehrere Kommunikationsformen innerhalb eines Dokuments. Dazu gehören insbesondere geschriebene Sprache, Bilder, Layout und weitere visuelle Gestaltungselemente. Die Bedeutung eines Dokuments entsteht dabei nicht ausschließlich durch den Text, sondern durch das Zusammenspiel textueller, visueller und räumlicher Elemente. Übertragen auf PDF-Dokumente bedeutet dies, dass neben Fließtext auch Tabellen, Abbildungen, Diagramme, Formeln, Beschriftungen und räumliche Anordnungen relevante Bestandteile des Dokuments sein können [4].

Für die technische Verarbeitung unterscheiden sich digital erzeugte und gescannte PDF-Dokumente. Digital erzeugte PDF-Dokumente entstehen aus elektronischen Quelldateien wie Textverarbeitungs-, Satz- oder Layoutprogrammen. In solchen

Dokumenten können Zeicheninformationen wie Zeichencode, Schriftart und Koordinaten auf der Seite eingebettet sein. Gescannte PDF-Dokumente bestehen dagegen aus Bildrepräsentationen der Seiten und enthalten nicht zwingend eine direkt nutzbare Textebene. Bei gescannten Dokumenten erkennt daher Optical Character Recognition den Text aus dem Seitenbild [23].

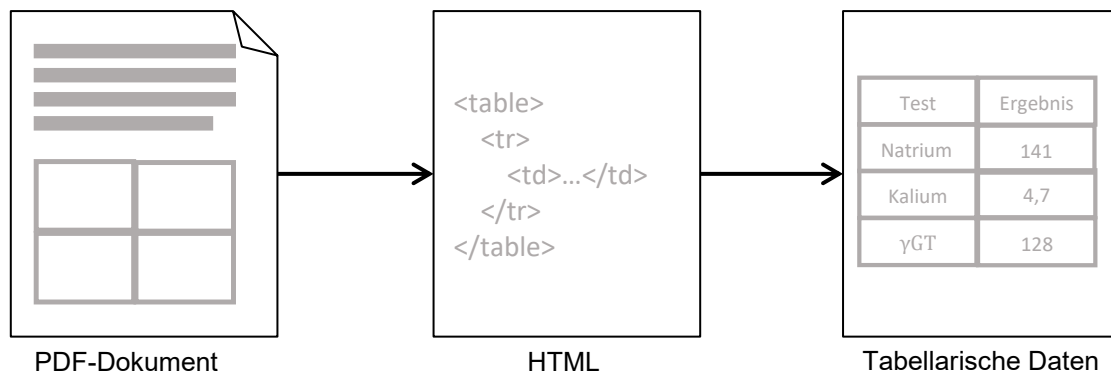
PDF-Dokumente können Inhalte als einzelne, positionsbezogene Objekte speichern. Dabei können Zeichen, grafische Elemente und weitere Bestandteile zwar visuell auf der Seite angeordnet sein, ohne dass die hierarchische Struktur des Dokuments vollständig explizit hinterlegt ist. Im Unterschied zu Formaten wie XML oder JSON, die hierarchischen Strukturen direkt abbilden können, muss die Dokumentstruktur bei PDF-Renderings häufig erst rekonstruiert werden. Dies betrifft insbesondere Strukturelemente wie Abschnitte, Tabellen, Abbildungen, Tabellenzellen und deren Beziehungen [5].

Für die Analyse von Dokumenten unterscheiden sich physische und logische Layoutstruktur. Die physische Layoutanalyse beschreibt die visuelle Gliederung einer Seite und unterscheidet Bereiche wie Text, Bild oder Tabelle. Die logische Layoutanalyse ordnet diesen Bereichen semantische Rollen zu, beispielsweise Kapitelüberschrift, Abschnittsüberschrift, Absatz oder Bildnotiz. Tabellen, Abbildungen und Diagramme bilden dabei eigene Dokumentelemente; Tabellen lassen sich über ihre Zeilen-, Spalten- und Zellstruktur erfassen, Abbildungen und grafische Elemente als Layoutbereiche mit semantischen Kategorien. Für wissenschaftliche und andere komplexe Dokumente bilden neben Text auch Tabellen, Figuren, Überschriften, Beschriftungen und weitere Layoutbestandteile relevante Einheiten [24].

Die Lesereihenfolge ist eine eigene Aufgabe innerhalb der Verarbeitung visuell reichhaltiger Dokumente. Sie beschreibt die Reihenfolge, in der Wörter oder Textbereiche für menschliche Leser sinnvoll gelesen werden können. Bei einfachen Dokumenten können OCR-Systeme Textzeilen häufig anhand einfacher Regeln von oben nach unten und von links nach rechts anordnen. Bei mehrspaltigen Dokumenten, Formularen, Rechnungen und anderen komplexen Layouts können solche Heuristiken jedoch unzureichend sein [25].

Für die maschinenlesbare Weiterverarbeitung überführen entsprechende Werkzeuge Dokumente in strukturierte Repräsentationen. Solche Repräsentationen können Dokumentbestandteile wie Text, Tabellen, Bilder, Beschriftungen, Listen, Dokumenthierarchien, Layoutinformationen und Herkunftsinformationen enthalten. Als Ausgabeformate dienen unter anderem JSON, Markdown und HTML. Sie überführen Inhalte aus PDF-, Office-, Bild- oder HTML-Dokumenten in eine einheitliche Dokumentrepräsentation und bilden damit die Grundlage für nachgelagerte Verarbeitungsschritte [26]. Abbildung 3 veranschaulicht diese Verarbeitung an einer einfachen Pipeline: Ein Werkzeug überführt ein PDF-Dokument zunächst in eine maschinenlesbare Zwischenrepräsentation wie HTML und daraus in strukturierte, tabellarische Daten.

Abbildung 3: Beispiel: Verarbeitung eines PDF-Dokuments zu strukturierten Daten (eigene Darstellung)



2.2.2 Medizinische PDF-Dokumente als Datenquelle

Medizinische Dokumente können strukturierte und unstrukturierte Informationen enthalten. Zu den strukturierten Informationen zählen beispielsweise physiologische Messwerte, Laborergebnisse, Diagnose- und Medikamentencodes sowie Bewertungsskalen. Zu den unstrukturierten Informationen zählen textuelle medizinische Dokumente wie Operationsberichte, Bildgebungsberichte, Pathologieberichte und Entlassungsberichte [27].

Medizinische PDF-Dokumente können sowohl digital erzeugt als auch gescannt vorliegen. Zu den klinisch relevanten gescannten Dokumenten in elektronischen Patientenakten zählen unter anderem Verschreibungen, Radiologieberichte, Laborergebnisse und Screening-Dokumente. Gescannte Dokumente enthalten keine diskreten Datenelemente und sind dadurch nicht unmittelbar durchsuchbar, sodass die Suche nach einzelnen Informationen erschwert sein kann; für solche Dokumente überführt Optical Character Recognition die Seiten in maschinenlesbaren Text [28].

Ein häufig tabellarisch aufgebauter und damit multimodaler Dokumenttyp sind medizinische Laborberichte, deren Inhalte Testnamen, Testergebnisse, Maßeinheiten und Referenzbereiche umfassen können. Ihre automatische Verarbeitung kombiniert Komponenten wie Bildvorverarbeitung, Layoutanalyse, OCR, Korrektur der OCR-Ergebnisse und Dokumentverständnis; die extrahierten Daten lassen sich anschließend in einem strukturierten Ausgabeformat speichern. Zur Unterstützung der Extraktion ordnen Wissensbasen erkannte Inhalte normalisierten Testbezeichnungen und Maßeinheiten zu [29]. Abbildung 4 veranschaulicht diese Struktur an einem schematischen Beispiel: Auf einen Kopfbereich mit Freitextangaben folgt ein tabellarischer Ergebnisteil, in dem jedem Test ein Ergebnis, eine Einheit, ein Referenzbereich und eine Bewertung zugeordnet sind, ergänzt durch eine abschließende Gesamtbeurteilung in Freitextform.

Über ihre formale Vielfalt hinaus weisen klinische Texte domänenspezifische sprachliche Eigenschaften auf. Typische Herausforderungen sind Abkürzungen, Akronyme, Kurzformen, Schreibfehler, ungrammatische Formulierungen, eingebettete Laborwerte und pseudo-tabellarische Strukturen [30]. Insbesondere medizinische Abkürzungen erscheinen in klinischen Notizen häufig ohne ausgeschriebene Langform und können je nach Kontext unterschiedliche Bedeutungen besitzen, was ihre maschinelle Verarbeitung zusätzlich erschwert [31].

Abbildung 4: Schematischer Aufbau eines medizinischen Laborberichts (eigene Darstellung)

Endbefund

Patient: Mustermann, Max · geb. 01.01.1980				
Fragestellung: Routinekontrolle · Material: Serum				
Klinische Chemie				
Test	Ergebnis	Einheit	Referenzbereich	Bewertung
Harnsäure	6,5	mg/dL	3,4 – 7,0	Normal
GPT (ALAT)	66	U/L	<= 50	Erhöht
γGT	126	U/L	<= 60	Erhöht
Natrium	141	mmol/L	133 – 150	Normal
Kalium	4,7	mmol/L	3,4 – 5,1	Normal
Gesamtbeurteilung				
Leberwerte erhöht, Verlaufskontrolle empfohlen.				

2.3 Extraktion strukturierter Informationen

Dieser Abschnitt beschreibt, wie aus unstrukturierten und semi-strukturierten Texten strukturierte Informationen gewonnen werden können. Zunächst werden in Abschnitt [2.3.1](#) grundlegende Aufgaben, Informationseinheiten und methodische Ansätze der Informationsextraktion erläutert. Anschließend wird in Abschnitt [2.3.2](#) die Kausalitätsextraktion als spezielle Form der Relationsextraktion betrachtet.

2.3.1 Informationsextraktion

Informationsextraktion beschreibt die Gewinnung strukturierter Informationen aus unstrukturiertem oder semi-strukturiertem Text. Dabei identifiziert sie relevante Inhalte aus Textdokumenten und überführt sie in strukturierte Formen wie Datenbankfelder und Datensätze. Sie unterscheidet sich damit von reiner Textextraktion, da nicht nur Zeichen oder Textpassagen übernommen, sondern bestimmte Informationseinheiten erkannt, klassifiziert und strukturiert werden. In diesem Sinne ist Informationsextraktion eine Vorstufe des Data Mining: Sie befüllt eine Datenbank aus unstrukturiertem Text, während das Data Mining anschließend Muster in dieser Datenbank entdeckt [32].

Zu den typischen Informationseinheiten zählen Entitäten und Relationen sowie, in weiter gefassten Darstellungen, Fakten und Ereignisse [33,34]. Entitäten können beispielsweise Personen, Organisationen, Orte oder fachliche Begriffe wie Krankheiten, Symptome und Medikamente sein; Relationen beschreiben semantische Beziehungen zwischen solchen Entitäten. Named Entity Recognition und Relation Extraction bilden zwei grundlegende Teilaufgaben der Informationsextraktion: Named Entity Recognition dient der Erkennung und Klassifikation relevanter Entitäten im Text, Relation Extraction der Erkennung von Beziehungen zwischen zuvor identifizierten Entitäten [33].

Informationsextraktion nutzt unterschiedliche methodische Ansätze. Regelbasierte Verfahren und Wörterbuchansätze verwenden vordefinierte Muster, Regeln oder Terminologien zur Erkennung relevanter Informationen. Klassische Machine-Learning-Verfahren nutzen gelabelte Trainingsdaten und Merkmale, während Deep-Learning-Verfahren neuronale Modelle verwenden, die Merkmale durchgängig aus den Daten lernen. Die Entwicklung der Informationsextraktion verläuft in drei Stufen: von regel- und wörterbuchbasierten Ansätzen über statistische Machine-Learning-Verfahren bis hin zu Deep-Learning-Verfahren [35]. Darüber hinaus verknüpft die Extraktion die gewonnenen Inhalte mit semantischen Ressourcen wie Ontologien oder Wissensbasen, indem sie Erwähnungen eindeutigen, disambiguierten Bezeichnern zuordnet [36].

Im medizinischen Kontext betrifft Informationsextraktion insbesondere klinische Freitexte. Patientensymptome liegen beispielsweise typischerweise in unstrukturierten Notizen vor und sind dadurch nicht unmittelbar als strukturierte Daten für Analyse- oder Entscheidungsmodelle verfügbar. Für solche Texte erkennen NLP-Verfahren die relevanten Informationen aus dem Freitext und machen sie in strukturierter Form nutzbar [3]. Die Eigenschaften unstrukturierter Daten erschweren dies: Sie besitzen kein festes Schema, sind uneinheitlich standardisiert und bringen Herausforderungen wie Rauschen, Heterogenität sowie semantische und kontextuelle Mehrdeutigkeit mit sich [19].

Für medizinische PDF-Dokumente schließt die Informationsextraktion an die vorgelagerten Schritte der Dokumentverarbeitung an. Nachdem ein Dokument durch Texterkennung, Layoutanalyse oder Konvertierung in eine strukturierte Zwischenrepräsentation aufbereitet wurde, lassen sich relevante medizinische Inhalte aus Texten, Tabellen oder Formularfeldern identifizieren und strukturieren; diese Reihenfolge ist eine Designentscheidung der vorliegenden Arbeit und wird in Kapitel 5 begründet. Die extrahierten Informationen können anschließend als Grundlage für Datenbanken, Auswertungen, Suche oder weitere Verarbeitungsschritte dienen [3,32,36]. Einen Spezialfall der Relationsextraktion bildet die Extraktion kausaler Zusammenhänge zwischen Entitäten oder Ereignissen, die im folgenden Abschnitt betrachtet wird.

2.3.2 Kausalitätsextraktion

Kausalitätsextraktion ist eine spezielle Form der Relationsextraktion. Sie befasst sich mit der Erkennung von Ursache-Wirkungs-Beziehungen zwischen Ereignissen, Zuständen oder Entitäten in Texten. Eine kausale Relation beschreibt dabei eine Beziehung, bei der eine Ursache zu einer Wirkung beiträgt oder dieser zeitlich vorausgeht. Im biomedizinischen Kontext können solche Relationen beispielsweise zwischen Krankheiten, Symptomen, Medikamenten, Behandlungen oder biologischen Prozessen bestehen [9].

Kausale Relationen besitzen eine gerichtete Struktur, die Ursache und Wirkung unterscheidet. Ursache und Wirkung können dabei als einzelne Entitäten, als Ereignisse oder als längere Textabschnitte auftreten. Die Erkennung kausaler Beziehungen ist eine grundlegende Aufgabe der biomedizinischen Informationsextraktion und unterstützt unter anderem den Aufbau wissenschaftlicher Wissensbasen und die Entscheidungsunterstützung [9].

Kausalität erscheint in Texten explizit oder implizit. Explizite Kausalität liegt vor, wenn sprachliche Marker wie „cause“, „lead to“ oder „because“ die Ursache-Wirkungs-Beziehung direkt anzeigen; implizite Kausalität liegt vor, wenn keine eindeutigen Marker vorhanden sind und die Beziehung aus dem Kontext oder semantischen Zusammenhang hervorgeht. Kausale Relationen treten zusätzlich satzintern oder satzübergreifend auf, je nachdem, ob sich Ursache und Wirkung innerhalb eines Satzes oder über mehrere Sätze verteilt befinden [37]. Diese Unterscheidungen gelten auch für klinische Texte, in denen kausale Beziehungen zwischen biomedizinischen Konzepten ebenfalls implizit oder explizit sowie satzintern oder satzübergreifend auftreten können [38].

Für die Kausalitätsextraktion ist die Festlegung der kausalen Einheiten relevant. Ursache und Wirkung können als vollständige Textspannen annotiert werden, die einzelne Wörter, Nominalphrasen, Teilsätze oder längere Aussagen umfassen [39]. Dies ist insbesondere im biomedizinischen Bereich bedeutsam, da Ursache und Wirkung dort häufig nicht aus einzelnen Begriffen, sondern aus mehrwörtigen Phrasen oder abhängigen Teilsätzen bestehen; unvollständig erfasste Ursache- oder Wirkungsausdrücke können die intendierte kausale Aussage verändern [40].

Methodisch nutzt die Kausalitätsextraktion dieselben grundlegenden Ansätze wie die allgemeine Informationsextraktion. Regel- und musterbasierte Verfahren nutzen kausale Marker und syntaktische Muster, maschinelle Lernverfahren verwenden annotierte Daten und Merkmale, und neuere Verfahren setzen neuronale sowie transformerbasierte Modelle ein [9,37]. Eine besondere Herausforderung im medizinischen Bereich besteht darin, dass kausale Zusammenhänge oft mit Unsicherheit einhergehen: Ein Symptom, das nach einer Medikamentengabe auftritt, kann eine vermutete Nebenwirkung darstellen, ohne dass die Kausalität eindeutig feststeht [41].

Zentrale Herausforderungen der Kausalitätsextraktion sind implizite Kausalität, satzübergreifende Relationen, mehrdeutige kausale Marker sowie lange und komplexe Ursache- und Wirkungsspannen [9,37,40]. Eine zusätzliche Schwierigkeit besteht darin, dass kausale Relationen in biomedizinischen Datensätzen mit allgemeinen Assoziationen verschwimmen können; für eine verlässliche Kausalitätsextraktion sind daher Ressourcen erforderlich, die kausale von nicht-kausalen Textstellen unterscheiden und Ursache-Wirkungs-Spannen explizit annotieren [38,39].

2.4 Large Language Models (LLMs)

LLMs sind große neuronale Sprachmodelle. Sie entstehen durch Vortraining auf umfangreichen Textkorpora und verarbeiten, erzeugen und nutzen natürliche Sprache für verschiedene NLP-Aufgaben [42]. Viele dieser Modelle basieren auf der Transformer-Architektur [42,43]. Transformer verwenden Attention-Mechanismen, um Beziehungen zwischen Tokens innerhalb einer Sequenz zu modellieren [43]. Je nach Modelltyp nutzen die Modelle encoder-only-, decoder-only- oder encoder-decoder-basierte Architekturen [42].

Ein wesentliches Merkmal von LLMs ist das Vortraining auf großen Datenmengen, bei dem die Modelle sprachliche Muster und statistische Zusammenhänge aus umfangreichen Korpora lernen. In autoregressiven Sprachmodellen besteht eine typische Trainingsaufgabe darin, das nächste Token in einer Sequenz vorherzusagen, während andere Verfahren wie Masked Language Modeling verdeckte Tokens aus dem Kontext rekonstruieren. Nach dem Vortraining passen Fine-Tuning, Instruction Tuning oder Reinforcement Learning from Human Feedback die Modelle weiter an bestimmte Aufgaben oder Interaktionsformen an [42].

LLMs können Aufgaben in Zero-shot- und Few-shot-Szenarien bearbeiten. In Zero-shot-Anwendungen erhält das Modell nur eine Aufgabenbeschreibung ohne Beispiele, während Few-shot-Anwendungen wenige Eingabe-Ausgabe-Beispiele in den Kontext aufnehmen, damit das Modell Aufgabe und gewünschtes Ausgabeformat aus dem Prompt ableiten kann [44]. Diese Fähigkeit heißt In-Context Learning, da das Modell während der Inferenz aus den bereitgestellten Kontextbeispielen lernt, ohne seine Parameter zu verändern [42,44]. Die gezielte Gestaltung solcher Eingaben heißt Prompt Engineering; ein Prompt kann dabei eine Aufgabenbeschreibung, Kontextinformationen, Eingabedaten, Beispiele und Hinweise zum gewünschten Ausgabeformat enthalten, und neben den grundlegenden Formen Zero-shot und Few-shot werden komplexere Verfahren wie Chain-of-Thought Prompting beschrieben, das die Ausgabe durch schrittweises Schließen verbessern soll [44].

Für die Informationsextraktion erkennen und strukturieren LLMs Informationen aus Text. Sie übernehmen unter anderem Named Entity Recognition, Relation Extraction und Event Extraction und überführen Entitäten, Beziehungen oder Ereignisse aus unstrukturierten Texten in strukturierte Ausgaben wie Schlüssel-Wert-Paare oder JSON-ähnliche Strukturen [44,45]. Im klinischen und biomedizinischen Kontext erkennen LLMs entsprechend medizinische Entitäten – etwa Krankheiten, Medikamente, Behandlungen und Symptome – aus klinischen Texten [46].

Speziell für die Kausalitätsextraktion sind LLMs jedoch nur eingeschränkt zuverlässig. Generative Modelle bleiben bei der Erkennung kausaler Relationen häufig hinter spezialisierten, auf biomedizinischen Daten trainierten Encoder-Modellen zurück und erfassen insbesondere implizite sowie satzübergreifende kausale Relationen nur schwer [10].

Medizinische Anwendungen nutzen sowohl allgemeine als auch domänenspezifische Sprachmodelle. Allgemeine Modelle können bei spezialisierten biomedizinischen oder klinischen Aufgaben durch fehlendes Domänenwissen eingeschränkt sein, weshalb domänenspezifische Modelle ein Vortraining oder eine Feinabstimmung auf biomedizinischen oder klinischen Korpora erhalten [47]. Neben großen, cloudbasierten Modellen gewinnen im Gesundheitswesen zunehmend kleinere, lokal ausführbare Sprachmodelle an Bedeutung, die Anforderungen wie Datenschutz, begrenzte Rechenressourcen und eine Verarbeitung in lokalen Umgebungen adressieren. Diese Modellklasse ist relevant für ressourcenbeschränkte und datenschutzsensible klinische Systeme [7].

Gleichzeitig bestehen bei LLMs mehrere grundlegende Einschränkungen. Die Evaluation von LLMs im Gesundheitswesen ist eine eigenständige Herausforderung: Reale Patientendaten dienen nur selten zur Bewertung, überwiegend dient Genauigkeit als Maß, während Aspekte wie Bias, Kalibrierung und Einsatzbedingungen seltener im Fokus stehen [48]. Eine weitere zentrale Einschränkung sind Halluzinationen, also plausibel wirkende, aber faktisch falsche oder nicht kontexttreue Ausgaben; diese umfassen Abweichungen von überprüfbaren Fakten sowie Abweichungen von Anweisung, Kontext oder logischer Konsistenz [49]. Solche Eigenschaften sind besonders im medizinischen Kontext relevant, in dem die Korrektheit und Nachvollziehbarkeit extrahierter Informationen entscheidend ist.

2.5 Evaluationsprinzipien

Die Evaluation LLM-basierter Informationsextraktion bewertet, ob relevante Informationen korrekt erkannt, vollständig erfasst, strukturiert ausgegeben und zuverlässig weiterverarbeitet werden können [50,51]. Für klinische Extraktionsaufgaben reicht eine einzelne Genauigkeitskennzahl nicht aus, da neben der inhaltlichen Richtigkeit auch strukturierte Ausgabequalität, menschlicher Prüfaufwand, Stabilität und Compliance eine Rolle spielen [50]. Eine reproduzierbare Evaluation benötigt zudem standardisierte Eingabedaten, einheitliche Prompt-Konfigurationen, validierbare Ausgabeformate und einen Vergleich mit Referenzdaten [51].

Ein zentrales Evaluationsprinzip ist die Extraktionsqualität, also wie zuverlässig ein Verfahren Entitäten, Attribute, Relationen oder Ereignisse aus einem Text erkennt [50,51]. Für Named Entity Recognition sowie Relation- und Event-Extraction dienen Precision, Recall und F1-Score als zentrale Metriken: Precision gibt an, welcher Anteil der extrahierten Treffer korrekt ist, Recall, welcher Anteil der tatsächlich vorhandenen relevanten Treffer erkannt wurde, und der F1-Score verbindet beide zu einer gemeinsamen Kennzahl [50].

Für Extraktionsaufgaben sind außerdem die Bewertungsregeln der Treffer relevant. Strict beziehungsweise exact Matching zählt einen Treffer nur dann als korrekt, wenn die extrahierte Information vollständig mit der Referenz übereinstimmt, während lenient beziehungsweise partial Matching auch teilweise Übereinstimmungen erlaubt. Diese Unterscheidung ist besonders bei Textspannen und Entitätsgrenzen relevant, da sich automatisch extrahierte Informationen in Länge oder Formulierung von der Referenz unterscheiden können [50]. Sie gewinnt bei der Kausalitätsextraktion zusätzlich an Bedeutung, da Ursache- und Wirkungsspannen häufig lange, mehrwörtige Ausdrücke sind; eine ausschließlich auf exakter Übereinstimmung beruhende Bewertung ist hier zu streng, weshalb ergänzend token- oder einbettungsbasierte Ähnlichkeitsmaße zum Einsatz kommen [10].

Ein weiteres Evaluationsprinzip ist die Strukturqualität der Ausgabe, da LLM-basierte Extraktion Informationen nicht nur als freien Text, sondern in maschinenlesbarer Form bereitstellen soll [52]. Strukturqualität umfasst syntaktische Gültigkeit, Schemaeinhaltung, vollständige Felder, korrekte Datentypen und fachlich richtige Feldwerte [51,52]. Dabei erfordern strukturelle Validität und semantische Korrektheit eine getrennte Betrachtung, da eine Ausgabe syntaktisch gültig sein und dennoch semantisch falsche Werte enthalten kann [52]. Für strukturierte Extraktion sind insbesondere JSON- und schema-basierte Ausgabeformate relevant, bei denen ein Schema erwartete Felder, Datentypen und erlaubte Werte definiert und so prüfbar macht, ob die Ausgabe vollständig, parsebar und für nachgelagerte Verarbeitungsschritte geeignet ist [51,53]. Typische Fehler sind dabei fehlende Schlüssel, fehlende oder falsche Werte und Typfehler, die die Nutzbarkeit der Ausgabe auch dann beeinträchtigen, wenn einzelne Textinformationen korrekt erkannt wurden [52].

Ein weiteres Evaluationsprinzip ist die Nachvollziehbarkeit der extrahierten Informationen. Extraktionsergebnisse müssen mit Referenzdaten, Ground-Truth-Annotationen oder manuell geprüften Ergebnissen vergleichbar sein, sodass sich korrekte, fehlende, falsche und zusätzlich erzeugte Informationen systematisch

unterscheiden lassen. Die menschliche Prüfung ist besonders relevant, wenn klinische Informationen fachlich interpretiert werden müssen oder wenn kein vollständig automatisierter Goldstandard verfügbar ist [50,53].

Schließlich erfordert LLM-basierte Extraktion eine Bewertung der praktischen Einsetzbarkeit der Modelle. Neben der Extraktionsleistung sind Laufzeit, Kosten, Modellgröße, Ressourcenbedarf und lokale Ausführbarkeit relevante Eigenschaften [8,51]. Lokale Modelle ermöglichen eine Verarbeitung ohne Übertragung sensibler Daten an externe Cloud-Dienste, können dabei aber eine geringere Extraktionsleistung als größere proprietäre Modelle erreichen [8]. Qualität und praktische Ausführbarkeit erfordern daher eine gemeinsame Betrachtung [8,51].

3 Stand der Technik und Forschung

Dieses Kapitel gibt einen Überblick über den Stand der Technik in den für diese Arbeit relevanten Bereichen. Im Fokus stehen Werkzeuge zur Verarbeitung multimodaler PDF-Dokumente sowie lokal ausführbare LLMs für die Informations- und Kausalitätsextraktion. Das Kapitel dient dazu, bestehende Ansätze systematisch zu erfassen, einzuordnen und anhand geeigneter Kriterien vergleichbar zu machen. Zunächst stellt Abschnitt [3.1](#) das Vorgehen bei der Recherche dar. Anschließend stellt Abschnitt [3.2](#) Werkzeuge für die multimodale Dokumentverarbeitung vor und untersucht sie hinsichtlich ihrer Eignung für diese Arbeit. Abschnitt [3.3](#) betrachtet schließlich lokal ausführbare LLMs für die Informations- und Kausalitätsextraktion.

3.1 Vorgehen bei der Suche

Für die Recherche zum Stand der Technik wurden verschiedene wissenschaftliche Suchplattformen in deutscher und englischer Sprache genutzt. Dazu gehörten unter anderem:

- Google Scholar¹
- PubMed²
- arXiv³
- medRxiv⁴

Ergänzend wurden offizielle Projektseiten, technische Dokumentationen und Repository-Beschreibungen herangezogen, sofern diese über die Recherche identifiziert wurden und für die Einordnung einzelner Werkzeuge oder Modelle relevante technische Informationen enthielten. Die verwendeten Suchbegriffe umfassten sowohl allgemeine als auch spezifische Suchanfragen. Dazu gehörten unter anderem:

- multimodal PDF processing
- table extraction PDF
- document understanding LLM
- medical NLP LLM
- OCR framework deep learning
- LLM information extraction
- document layout analysis tools

Die Recherche berücksichtigte insbesondere Arbeiten und Systeme, die einen Bezug zu Text-, Tabellen-, Diagramm- oder Layoutverarbeitung aufweisen und sich grundsätzlich in einen Python-basierten Verarbeitungsprozess integrieren lassen. Bei den LLMs lag der Fokus auf lokal ausführbaren Ansätzen, da diese für datenschutzsensible

¹ <https://scholar.google.com/>

² <https://pubmed.ncbi.nlm.nih.gov/>

³ <https://arxiv.org/>

⁴ <https://www.medrxiv.org/>

Anwendungsszenarien besonders relevant sind. Die folgenden Abschnitte stellen die identifizierten Werkzeuge und Modelle systematisch dar und vergleichen sie anhand fachlicher und technischer Kriterien. Umgekehrt blieben Werkzeuge außerhalb der Betrachtung, die sich nicht aus einem Python-Prozess ansprechen lassen, sowie Modelle ohne offengelegte Gewichte, die ausschließlich über einen Anbieterdienst nutzbar sind. Die cloudgebundene Nutzung allein führte dagegen nicht zum Ausschluss: LlamaParse bleibt als Vergleichspunkt enthalten und scheidet erst bei der Auswahl aus (siehe Abschnitt [5.3.1](#)).

3.2 Werkzeuge zur Verarbeitung multimodaler PDF-Dokumente

Dieser Abschnitt betrachtet Werkzeuge zur Verarbeitung multimodaler PDF-Dokumente. Der Fokus liegt auf Ansätzen, die unterschiedliche Dokumentbestandteile wie Fließtext, Tabellen, Diagramme und Layoutinformationen erfassen und für eine weitere Verarbeitung verfügbar machen. Zur besseren Einordnung unterteilt dieser Abschnitt die Werkzeuge in regel- und heuristikbasierte, Deep-Learning-basierte sowie KI-gestützte Werkzeuge.

3.2.1 Regel- und Heuristik-basierte Werkzeuge

Dieser Abschnitt betrachtet Werkzeuge, die PDF-Inhalte ohne den Einsatz großer lernbasierter Modelle extrahieren und strukturieren. Solche Ansätze lassen sich im PDF-Parsing grob in regelbasierte und lernbasierte Verfahren einteilen. Regelbasierte Methoden arbeiten mit festen Regeln und zeichnen sich insbesondere durch Interpretierbarkeit, geringe Rechenanforderungen sowie einen Einsatz ohne zusätzliches Modelltraining aus [54]. Für die vorliegende Arbeit sind sie vor allem deshalb relevant, weil sie neben dem Text auch positions- und formatbezogene Informationen aus PDF-Dokumenten zugänglich machen, die als Grundlage weiterer Extraktions- und Strukturierungsschritte dienen können [55]. Darüber hinaus liegt bei PDF-Dokumenten Fließtext häufig mit Metadaten und semi-strukturierten Inhalten wie Tabellen vermischt vor, sodass eine vorgelagerte strukturierende Verarbeitung sinnvoll ist [56]. Die folgenden vier Werkzeuge – PyMuPDF, Pdfminer.six, Pdfplumber und Tabula-py – decken unterschiedliche Hauptaufgaben innerhalb der regel- und heuristikbasierten PDF-Verarbeitung ab: PyMuPDF und Pdfminer.six für die Extraktion von Text- und Strukturinformationen, Pdfplumber für den objekt- und layoutbezogenen Zugriff auf PDF-Inhalte sowie Tabula-py für die spezialisierte Tabellenextraktion [54].

3.2.1.1 PyMuPDF

PyMuPDF ist eine Python-Bibliothek zur Verarbeitung von PDF-Dokumenten [57]. Das Werkzeug unterstützt unter anderem Datenextraktion, Analyse, Konvertierung und Bearbeitung von Dokumenten⁵. Für die vorliegende Arbeit sind insbesondere die Extraktion von Text, Tabellen und Bildern sowie die Bereitstellung strukturbezogener Informationen relevant^{5,6}. Für die Textextraktion stellt PyMuPDF unter anderem Fließtext, Textblöcke und Wörter einschließlich Bounding-Box-Informationen bereit. Das Werkzeug gibt Textinhalte damit nicht nur als fortlaufenden Text, sondern auch in segmentierter Form mit Positionsbezug aus⁶. Zusätzlich umfasst der Funktionsumfang eine grundlegende Seitenanalyse zur Bereitstellung von Dokumentstrukturinformationen⁵. In regelbasierten Verarbeitungspipelines analysiert PyMuPDF zudem visuelle Textblöcke anhand von Koordinaten, Schriftgrößen und räumlicher Positionierung, etwa in einer hybriden Pipeline zur Tag-Extraktion aus spaltenbasierten PDF-Dokumenten⁶ [58]. Abbildung 5 und Abbildung 6 zeigen eine Beispielseite aus einem PDF-Dokument [59] sowie die mit PyMuPDF extrahierten Textblöcke mit Bounding Boxes. Die erkannten Textsegmente erscheinen dabei mit Positionsbezug auf der gerenderten PDF-Seite.

3.2.1.2 Pdfminer.six

pdfminer.six ist eine Python-Bibliothek mit Schwerpunkt auf der Extraktion und Analyse von Text aus PDF-Dokumenten [57]. Die Textgewinnung erfolgt über einen Layout-Analyse-Algorithmus, der Zeichen zunächst zu Wörtern und Zeilen gruppiert, diese anschließend in Textboxen anordnet und hierarchisch zu Layoutobjekten zusammenfasst⁷ [57]. Für die vorliegende Arbeit ist insbesondere relevant, dass pdfminer.six nicht nur fortlaufenden Text, sondern auch textuelle Einheiten mit Layoutbezug verarbeitet⁷ [60]. Darüber hinaus liefert pdfminer.six Textblöcke in korrekter Lesereihenfolge und stellt Positions- sowie Schriftinformationen einzelner Zeichen bereit. Gleichzeitig bleibt die Extraktion auf der Ebene struktureller Analyse, während die logische Rolle der Textblöcke nicht automatisch bestimmt wird [60]. Abbildung 7 zeigt die mit pdfminer.six erkannten textuellen Layoutbereiche mit Bounding Boxes. Dargestellt ist die layoutbezogene Gruppierung von Texteinheiten auf der bereits in Abbildung 5 gezeigten PDF-Seite.

⁵ <https://pymupdf.readthedocs.io/en/latest/about.html> [Zugriff am 13.04.2026].

⁶ <https://pymupdf.readthedocs.io/en/latest/app1.html> [Zugriff am 13.04.2026].

⁷ https://pdfminersix.readthedocs.io/en/latest/topic/converting_pdf_to_text.html [Zugriff am 13.04.2026].

Abbildung 5: Beispielseite eines PDF-Dokuments

Clinical Review & Education

JAMA | US Preventive Services Task Force | **RECOMMENDATION STATEMENT**

Screening for Gestational Diabetes

US Preventive Services Task Force Recommendation Statement

US Preventive Services Task Force

IMPORTANCE Gestational diabetes is diabetes that develops during pregnancy. Prevalence of gestational diabetes in the US has been estimated at 5.8% to 9.2%, based on traditional diagnostic criteria, although it may be higher if more inclusive criteria are used. Pregnant persons with gestational diabetes are at increased risk for maternal and fetal complications, including preeclampsia, fetal macrosomia (which can cause shoulder dystocia and birth injury), and neonatal hypoglycemia. Gestational diabetes has also been associated with an increased risk of several long-term health outcomes in pregnant persons and intermediate outcomes in their offspring.

OBJECTIVE The USPSTF commissioned a systematic review to evaluate the accuracy, benefits, and harms of screening for gestational diabetes and the benefits and harms of treatment for the pregnant person and infant.

POPULATION Pregnant persons who have not been previously diagnosed with type 1 or type 2 diabetes.

EVIDENCE ASSESSMENT The USPSTF concludes with moderate certainty that there is a moderate net benefit to screening for gestational diabetes at 24 weeks of gestation or after to improve maternal and fetal outcomes. The USPSTF concludes that the evidence on screening for gestational diabetes before 24 weeks of gestation is insufficient, and the balance of benefits and harms of screening cannot be determined.

RECOMMENDATION The USPSTF recommends screening for gestational diabetes in asymptomatic pregnant persons at 24 weeks of gestation or after. (B recommendation) The USPSTF concludes that the current evidence is insufficient to assess the balance of benefits and harms of screening for gestational diabetes in asymptomatic pregnant persons before 24 weeks of gestation. (I statement)

JAMA. 2021;326(6):531-538. doi:10.1001/jama.2021.11922

Corrected on October 5, 2021.

← Editorial page 487

+ Multimedia

← Related article page 539 and JAMA Patient Page page 577

+ Supplemental content

+ CME Quiz at jamacmelookup.com and CME Questions page 565

Author/Group Information: The US Preventive Services Task Force (USPSTF) members are listed at the end of this article.

Corresponding Author: Karina W. Davidson, PhD, MASC, Feinstein Institutes for Medical Research, 130 E 59th St, Ste 14C, New York, NY 10032 (chair@uspstf.net).

Summary of Recommendations

The USPSTF recommends screening for gestational diabetes in asymptomatic pregnant persons at 24 weeks of gestation or after.	B
The USPSTF concludes that the current evidence is insufficient to assess the balance of benefits and harms of screening for gestational diabetes in asymptomatic pregnant persons before 24 weeks of gestation.	I

See the Figure for a more detailed summary of the recommendations for clinicians. USPSTF indicates US Preventive Services Task Force.

See the Summary of Recommendations figure.

Importance

Gestational diabetes is diabetes that develops during pregnancy.¹⁻³ Prevalence of gestational diabetes in the US has been estimated at 5.8% to 9.2%, based on traditional diagnostic criteria, although it may be higher if more inclusive criteria are used.⁴⁻⁸ Pregnant

persons with gestational diabetes are at increased risk for maternal and fetal complications, including preeclampsia, fetal macrosomia (which can cause shoulder dystocia and birth injury), and neonatal hypoglycemia.^{3,9-11} Gestational diabetes has also been associated with an increased risk of several long-term health outcomes in pregnant persons and intermediate outcomes in their offspring.¹²⁻¹⁶

Abbildung 6: Mit PyMuPDF extrahierte Textblöcke auf der Beispielseite aus Abbildung 5

Clinical Review & Education

JAMA | US Preventive Services Task Force | RECOMMENDATION STATEMENT

Screening for Gestational Diabetes

US Preventive Services Task Force Recommendation Statement

US Preventive Services Task Force

IMPORTANCE Gestational diabetes is diabetes that develops during pregnancy. Prevalence of gestational diabetes in the US has been estimated at 5.8% to 9.2%, based on traditional diagnostic criteria, although it may be higher if more inclusive criteria are used. Pregnant persons with gestational diabetes are at increased risk for maternal and fetal complications, including preeclampsia, fetal macrosomia (which can cause shoulder dystocia and birth injury), and neonatal hypoglycemia. Gestational diabetes has also been associated with an increased risk of several long-term health outcomes in pregnant persons and intermediate outcomes in their offspring.

OBJECTIVE The USPSTF commissioned a systematic review to evaluate the accuracy, benefits, and harms of screening for gestational diabetes and the benefits and harms of treatment for the pregnant person and infant.

POPULATION Pregnant persons who have not been previously diagnosed with type 1 or type 2 diabetes.

EVIDENCE ASSESSMENT The USPSTF concludes with moderate certainty that there is a moderate net benefit to screening for gestational diabetes at 24 weeks of gestation or after to improve maternal and fetal outcomes. The USPSTF concludes that the evidence on screening for gestational diabetes before 24 weeks of gestation is insufficient, and the balance of benefits and harms of screening cannot be determined.

RECOMMENDATION The USPSTF recommends screening for gestational diabetes in asymptomatic pregnant persons at 24 weeks of gestation or after. (B recommendation) The USPSTF concludes that the current evidence is insufficient to assess the balance of benefits and harms of screening for gestational diabetes in asymptomatic pregnant persons before 24 weeks of gestation. (I statement)

JAMA. 2021;326(6):531-538. doi:10.1001/jama.2021.11922
Corrected on October 5, 2021.

Editorial page 487

Multimedia

Related article page 539 and JAMA Patient Page page 577

Supplemental content

CME Quiz at jamacmelookup.com and CME Questions page 565

Author/Group Information: The US Preventive Services Task Force (USPSTF) members are listed at the end of this article.

Corresponding Author: Karina W. Davidson, PhD, MASc, Feinstein Institutes for Medical Research, 130 E 59th St, Ste 14C, New York, NY 10032 (chair@uspstf.net).

Summary of Recommendations

The USPSTF recommends screening for gestational diabetes in asymptomatic pregnant persons at 24 weeks of gestation or after.	B
The USPSTF concludes that the current evidence is insufficient to assess the balance of benefits and harms of screening for gestational diabetes in asymptomatic pregnant persons before 24 weeks of gestation.	I

See the Figure for a more detailed summary of the recommendations for clinicians. USPSTF indicates US Preventive Services Task Force.

See the Summary of Recommendations figure.

Importance

Gestational diabetes is diabetes that develops during pregnancy.¹⁻³ Prevalence of gestational diabetes in the US has been estimated at 5.8% to 9.2%, based on traditional diagnostic criteria, although it may be higher if more inclusive criteria are used.⁴⁻⁸ Pregnant

persons with gestational diabetes are at increased risk for maternal and fetal complications, including preeclampsia, fetal macrosomia (which can cause shoulder dystocia and birth injury), and neonatal hypoglycemia.^{3,9-11} Gestational diabetes has also been associated with an increased risk of several long-term health outcomes in pregnant persons and intermediate outcomes in their offspring.¹²⁻¹⁶

Abbildung 7: Mit pdfminer.six erkannte textuelle Layoutbereiche auf der Beispielseite aus Abbildung 5

Clinical Review & Education

JAMA | US Preventive Services Task Force | RECOMMENDATION STATEMENT

Screening for Gestational Diabetes

US Preventive Services Task Force Recommendation Statement

US Preventive Services Task Force

IMPORTANCE Gestational diabetes is diabetes that develops during pregnancy. Prevalence of gestational diabetes in the US has been estimated at 5.8% to 9.2%, based on traditional diagnostic criteria, although it may be higher if more inclusive criteria are used. Pregnant persons with gestational diabetes are at increased risk for maternal and fetal complications, including preeclampsia, fetal macrosomia (which can cause shoulder dystocia and birth injury), and neonatal hypoglycemia. Gestational diabetes has also been associated with an increased risk of several long-term health outcomes in pregnant persons and intermediate outcomes in their offspring.

OBJECTIVE The USPSTF commissioned a systematic review to evaluate the accuracy, benefits, and harms of screening for gestational diabetes and the benefits and harms of treatment for the pregnant person and infant.

POPULATION Pregnant persons who have not been previously diagnosed with type 1 or type 2 diabetes.

EVIDENCE ASSESSMENT The USPSTF concludes with moderate certainty that there is a moderate net benefit to screening for gestational diabetes at 24 weeks of gestation or after to improve maternal and fetal outcomes. The USPSTF concludes that the evidence on screening for gestational diabetes before 24 weeks of gestation is insufficient, and the balance of benefits and harms of screening cannot be determined.

RECOMMENDATION The USPSTF recommends screening for gestational diabetes in asymptomatic pregnant persons at 24 weeks of gestation or after. (B recommendation) The USPSTF concludes that the current evidence is insufficient to assess the balance of benefits and harms of screening for gestational diabetes in asymptomatic pregnant persons before 24 weeks of gestation. (I statement)

JAMA. 2021;326(6):531-538. doi:10.1001/jama.2021.11922
Corrected on October 5, 2021.

← Editorial page 487

+ Multimedia

← Related article page 539 and JAMA Patient Page page 577

+ Supplemental content

+ CME Quiz at amacmelookup.com and CME Questions page 565

Author/Group Information: The US Preventive Services Task Force (USPSTF) members are listed at the end of this article.

Corresponding Author: Karina W. Davidson, PhD, MASc, Feinstein Institutes for Medical Research, 130 E 59th St, Ste 14C, New York, NY 10032 (chair@uspstf.net).

Summary of Recommendations

The USPSTF recommends screening for gestational diabetes in asymptomatic pregnant persons at 24 weeks of gestation or after.	B	See the Figure for a more detailed summary of the recommendations for clinicians. USPSTF indicates US Preventive Services Task Force.
The USPSTF concludes that the current evidence is insufficient to assess the balance of benefits and harms of screening for gestational diabetes in asymptomatic pregnant persons before 24 weeks of gestation.	I	

See the Summary of Recommendations figure.

Importance

Gestational diabetes is diabetes that develops during pregnancy.¹⁻³ Prevalence of gestational diabetes in the US has been estimated at 5.8% to 9.2%, based on traditional diagnostic criteria, although it may be higher if more inclusive criteria are used.⁴⁻⁸ Pregnant

persons with gestational diabetes are at increased risk for maternal and fetal complications, including preeclampsia, fetal macrosomia (which can cause shoulder dystocia and birth injury), and neonatal hypoglycemia.^{3,9-11} Gestational diabetes has also been associated with an increased risk of several long-term health outcomes in pregnant persons and intermediate outcomes in their offspring.¹²⁻¹⁶

3.2.1.3 Pdfplumber

Pdfplumber ist ein in Python entwickeltes Werkzeug zur Verarbeitung von PDF-Dokumenten, das auf pdfminer aufbaut. Für die vorliegende Arbeit ist insbesondere relevant, dass pdfplumber spezifische Informationen zu Zeichen, Rechtecken, Linien und weiteren Seitenelementen zugänglich macht und damit einen Zugriff auf einzelne PDF-Objekte und Seitenelemente ermöglicht. Darüber hinaus unterstützt das Werkzeug die Extraktion von Text- und Tabellendaten. Für die Tabellenextraktion verwendet pdfplumber standardmäßig vertikale und horizontale Linien beziehungsweise Kanten von Rechtecken als Zellgrenzen, wobei sich dieses Verhalten über Tabellenparameter anpassen lässt [61]. In regelbasierten Verarbeitungspipelines verfeinert pdfplumber die Geometrie, prüft Koordinaten und korrigiert extrahierte Informationen bei komplexeren Layoutstrukturen, während PyMuPDF und Pdfminer.six andere Schwerpunkte abdecken [58]. Abbildung 8 zeigt die mit pdfplumber extrahierten Texteinheiten mit Bounding Boxes auf der bereits in Abbildung 5 dargestellten PDF-Seite.

3.2.1.4 Tabula-py

Tabula-py ist ein Python-Wrapper für die Extraktion von Tabellen aus PDF-Dokumenten. Für die vorliegende Arbeit ist insbesondere relevant, dass das Werkzeug auf tabellarische Inhalte spezialisiert ist und damit einen engeren Funktionsschwerpunkt als die zuvor betrachteten allgemeineren PDF-Werkzeuge aufweist⁸ [62]. Die Extraktion kann seitenbezogen erfolgen und unterstützt auch eine koordinatenbasierte Eingrenzung von Tabellenbereichen [62]. Darüber hinaus lassen sich die extrahierten Tabellen in strukturierter Form weiterverarbeiten, insbesondere als DataFrames. Diese Arbeit berücksichtigt tabula-py daher als spezialisiertes Werkzeug zur Tabellenextraktion⁸ [62].

⁸ <https://tabula-py.readthedocs.io/en/latest/> [Zugriff am 13.04.2026].

Abbildung 8: Mit pdfplumber extrahierte Texteinheiten auf der Beispielseite aus Abbildung 5

Clinical Review & Education

JAMA | US Preventive Services Task Force | RECOMMENDATION STATEMENT

Screening for Gestational Diabetes

US Preventive Services Task Force Recommendation Statement

US Preventive Services Task Force

IMPORTANCE Gestational diabetes is diabetes that develops during pregnancy. Prevalence of gestational diabetes in the US has been estimated at 5.8% to 9.2%, based on traditional diagnostic criteria, although it may be higher if more inclusive criteria are used. Pregnant persons with gestational diabetes are at increased risk for maternal and fetal complications, including preeclampsia, fetal macrosomia (which can cause shoulder dystocia and birth injury), and neonatal hypoglycemia. Gestational diabetes has also been associated with an increased risk of several long-term health outcomes in pregnant persons and intermediate outcomes in their offspring.

OBJECTIVE The USPSTF commissioned a systematic review to evaluate the accuracy, benefits, and harms of screening for gestational diabetes and the benefits and harms of treatment for the pregnant person and infant.

POPULATION Pregnant persons who have not been previously diagnosed with type 1 or type 2 diabetes.

EVIDENCE ASSESSMENT The USPSTF concludes with moderate certainty that there is a moderate net benefit to screening for gestational diabetes at 24 weeks of gestation or after to improve maternal and fetal outcomes. The USPSTF concludes that the evidence on screening for gestational diabetes before 24 weeks of gestation is insufficient, and the balance of benefits and harms of screening cannot be determined.

RECOMMENDATION The USPSTF recommends screening for gestational diabetes in asymptomatic pregnant persons at 24 weeks of gestation or after. (B recommendation) The USPSTF concludes that the current evidence is insufficient to assess the balance of benefits and harms of screening for gestational diabetes in asymptomatic pregnant persons before 24 weeks of gestation. (I statement)

JAMA. 2021;326(6):531-538. doi:10.1001/jama.2021.11922
Corrected on October 5, 2021

← Editorial page 487

+ Multimedia

← Related article page 539 and JAMA Patient Page page 577

+ Supplemental content

+ CME Quiz at [jamacmelookup.com](#) and CME Questions page 565

Author/Group Information: The US Preventive Services Task Force (USPSTF) members are listed at the end of this article.

Corresponding Author: Karina W. Davidson, PhD, MASC, Feinstein Institutes for Medical Research, 130 E 59th St, Ste 14C, New York, NY 10032 ([chair@uspstf.net](#)).

Summary of Recommendations

The USPSTF recommends screening for gestational diabetes in asymptomatic pregnant persons at 24 weeks of gestation or after.	B
The USPSTF concludes that the current evidence is insufficient to assess the balance of benefits and harms of screening for gestational diabetes in asymptomatic pregnant persons before 24 weeks of gestation.	I

See the Figure for a more detailed summary of the recommendations for clinicians. USPSTF indicates US Preventive Services Task Force.

See the Summary of Recommendations figure.

Importance

Gestational diabetes is diabetes that develops during pregnancy.¹⁻³ Prevalence of gestational diabetes in the US has been estimated at 5.8% to 9.2%, based on traditional diagnostic criteria, although it may be higher if more inclusive criteria are used.⁴⁻⁸ Pregnant

persons with gestational diabetes are at increased risk for maternal and fetal complications, including preeclampsia, fetal macrosomia (which can cause shoulder dystocia and birth injury), and neonatal hypoglycemia.^{3,9-11} Gestational diabetes has also been associated with an increased risk of several long-term health outcomes in pregnant persons and intermediate outcomes in their offspring.¹²⁻¹⁶

3.2.2 Deep-Learning-basierte Werkzeuge

Deep-Learning-basierte Werkzeuge gewinnen in der Dokumentverarbeitung insbesondere dort an Bedeutung, wo komplexe Layouts, variierende Schriftbilder oder mehrstufige Verarbeitungsaufgaben mit klassischen Verfahren nur eingeschränkt erfasst werden können [63,64]. Im Unterschied zu regel- und heuristikbasierten Ansätzen beruhen sie auf trainierten Modellen, die visuelle und teils auch strukturelle Merkmale aus Dokumenten lernen und für Aufgaben wie Texterkennung, Layoutanalyse oder strukturierte Informationsextraktion nutzen [63,64,65]. Für die vorliegende Arbeit sind solche Werkzeuge vor allem deshalb relevant, weil sie unterschiedliche Schwerpunkte innerhalb der multimodalen Dokumentverarbeitung abdecken, darunter OCR, Layoutanalyse und die Verarbeitung komplexer Dokumentstrukturen [63,64]. Die folgenden vier Deep-Learning-basierten Werkzeuge – docTR, PaddleOCR, Surya und LayoutParser – adressieren unterschiedliche Aufgaben in der OCR- und Dokumentanalyse⁹ [63,64,66].

3.2.2.1 docTR

docTR ist eine Deep-Learning-basierte Open-Source-Bibliothek für Optical Character Recognition mit Fokus auf die Erkennung und Extraktion von Dokumenttext. Das Werkzeug verarbeitet OCR als zweistufige Aufgabe aus Textdetektion und Texterkennung und stellt dafür kombinierbare Modelle sowie einen OCR-Predictor bereit. docTR kann Dokumente aus Bildern und PDF-Dateien verarbeiten und liefert erkannte Textbereiche mit zugehörigem Textinhalt, sodass docTR Dokumentinhalte nicht nur als fortlaufenden Text, sondern mit räumlichem Bezug zur Dokumentseite ausgibt. Darüber hinaus unterstützt docTR mit dem KIE-Predictor die Erkennung mehrerer Klassen von Dokumentelementen innerhalb einer gemeinsamen Pipeline. docTR verbindet damit OCR mit layoutbezogener Dokumentverarbeitung und geht über die reine Texterkennung im engeren Sinn hinaus. Der Schwerpunkt liegt auf der Deep-Learning-basierten Kombination von Textlokalisierung, Texterkennung und dokumentbezogener OCR-Verarbeitung^{9,10}. Abbildung 9 zeigt die mit docTR erkannten Textbereiche mit Bounding Boxes auf der bereits in Abbildung 5 dargestellten PDF-Seite. Sichtbar ist die segmentierte Lokalisierung von Textinhalten auf der Dokumentseite.

⁹ <https://mindee.github.io/doctr/> [Zugriff am 13.04.2026].

¹⁰ <https://github.com/mindee/doctr> [Zugriff am 13.04.2026].

Abbildung 9: Mit docTR erkannte Textbereiche auf der Beispielseite aus Abbildung 5

Clinical Review & Education

JAMA | US Preventive Services Task Force | **RECOMMENDATION STATEMENT**

Screening for Gestational Diabetes

US Preventive Services Task Force Recommendation Statement

US Preventive Services Task Force

IMPORTANCE Gestational diabetes is diabetes that develops during pregnancy. Prevalence of gestational diabetes in the US has been estimated at 5.8% to 9.2%, based on traditional diagnostic criteria, although it may be higher if more inclusive criteria are used. Pregnant persons with gestational diabetes are at increased risk for maternal and fetal complications, including preeclampsia, fetal macrosomia (which can cause shoulder dystocia and birth injury), and neonatal hypoglycemia. Gestational diabetes has also been associated with an increased risk of several long-term health outcomes in pregnant persons and intermediate outcomes in their offspring.

OBJECTIVE The USPSTF commissioned a systematic review to evaluate the accuracy, benefits, and harms of screening for gestational diabetes and the benefits and harms of treatment for the pregnant person and infant.

POPULATION Pregnant persons who have not been previously diagnosed with type 1 or type 2 diabetes.

EVIDENCE ASSESSMENT The USPSTF concludes with moderate certainty that there is a moderate net benefit to screening for gestational diabetes at 24 weeks of gestation or after to improve maternal and fetal outcomes. The USPSTF concludes that the evidence on screening for gestational diabetes before 24 weeks of gestation is insufficient, and the balance of benefits and harms of screening cannot be determined.

RECOMMENDATION The USPSTF recommends screening for gestational diabetes in asymptomatic pregnant persons at 24 weeks of gestation or after. (B recommendation) The USPSTF concludes that the current evidence is insufficient to assess the balance of benefits and harms of screening for gestational diabetes in asymptomatic pregnant persons before 24 weeks of gestation. (I statement)

JAMA. 2021;326(6):531-538. doi:10.1001/jama.2021.11922
Corrected on October 5, 2021.

Editorial page 487

Multimedia

Related article page 539 and JAMA Patient Page page 577

Supplemental content

CME Quiz at jamacmelookup.com and CME Questions page 565

Author/Group Information: The US Preventive Services Task Force (USPSTF) members are listed at the end of this article.

Corresponding Author: Karina W. Davidson, PhD, MASc, Feinstein Institutes for Medical Research, 130 E 59th St, Ste 14C, New York, NY 10032 (chair@uspstf.net).

Summary of Recommendations

The USPSTF recommends screening for gestational diabetes in asymptomatic pregnant persons at 24 weeks of gestation or after.	B
The USPSTF concludes that the current evidence is insufficient to assess the balance of benefits and harms of screening for gestational diabetes in asymptomatic pregnant persons before 24 weeks of gestation.	I

See the Figure for a more detailed summary of the recommendations for clinicians. USPSTF indicates US Preventive Services Task Force.

See the Summary of Recommendations figure.

Importance

Gestational diabetes is diabetes that develops during pregnancy.¹⁻³ Prevalence of gestational diabetes in the US has been estimated at 5.8% to 9.2%, based on traditional diagnostic criteria, although it may be higher if more inclusive criteria are used.⁴⁻⁸ Pregnant

persons with gestational diabetes are at increased risk for maternal and fetal complications, including preeclampsia, fetal macrosomia (which can cause shoulder dystocia and birth injury), and neonatal hypoglycemia.^{3,9-11} Gestational diabetes has also been associated with an increased risk of several long-term health outcomes in pregnant persons and intermediate outcomes in their offspring.¹²⁻¹⁶

3.2.2.2 PaddleOCR

PaddleOCR ist ein Deep-Learning-basiertes Open-Source-Werkzeug für OCR und Dokumentverarbeitung. Das System umfasst Modelle für Texterkennung, Dokumentparsen und Informationsextraktion. Dazu gehören insbesondere PP-OCRv5 für mehrsprachige Texterkennung, PP-StructureV3 für hierarchisches Dokumentparsen und PP-ChatOCRv4 für die Extraktion von Schlüsselinformationen [63]. Darüber hinaus werden Layoutanalyse, Tabellenverarbeitung und Layoutrekonstruktion als Bestandteile der Dokumentanalyse unterstützt [63,67]. PaddleOCR verbindet damit OCR mit weiterführenden Schritten der Layout- und Strukturverarbeitung und deckt Aufgaben der Dokumentanalyse über die reine Texterkennung hinaus ab. Der Schwerpunkt liegt damit auf einer modularen Deep-Learning-basierten Verarbeitung von Text, Layout und strukturierten Dokumentinhalten [63]. Abbildung 10 zeigt die mit PaddleOCR erkannten Textbereiche mit Bounding Boxes auf der bereits in Abbildung 5 dargestellten PDF-Seite. Sichtbar ist die segmentierte Lokalisierung von Textinhalten mit Positionsbezug auf der Dokumentseite.

3.2.2.3 Surya

Surya ist ein Deep-Learning-basiertes OCR-Werkzeug für die Verarbeitung komplexer Dokumente [66,68]. Das Werkzeug unterstützt die Erkennung und Analyse von Layout-Elementen wie Titeln, Beschriftungen, Tabellen, Abbildungen und Textblöcken sowie die Bestimmung von Spalten- und Lesereihenfolge. Surya beschränkt sich damit nicht auf reine Texterkennung, sondern verarbeitet auch layoutbezogene Strukturen von Dokumentinhalten [68]. Surya kann zudem als eigenständiges OCR-Modell für verschiedene Dokument- und Texteingaben eingesetzt werden [66]. In PDF-Verarbeitungspipelines wird Surya außerdem für OCR-Extraktion und die Identifikation dokumentbezogener Strukturelemente verwendet. Damit werden Textinhalte sowohl erkannt als auch in ihren strukturellen Dokumentzusammenhang eingeordnet [68]. Surya deckt damit Aufgaben an der Schnittstelle von OCR, Layoutanalyse und dokumentbezogener Strukturverarbeitung ab. Der Schwerpunkt liegt auf der Deep-Learning-basierten Erkennung von Text und Dokumentelementen sowie auf der Bestimmung ihrer strukturellen Beziehungen in komplexen Dokumenten [66,68]. Abbildung 11 zeigt die mit Surya erkannten Textbereiche mit Bounding Boxes auf der bereits in Abbildung 5 dargestellten PDF-Seite.

Abbildung 10: Mit PaddleOCR erkannte Textbereiche auf der Beispielseite aus Abbildung 5

Clinical Review & Education

JAMA | US Preventive Services Task Force | RECOMMENDATION STATEMENT

Screening for Gestational Diabetes

US Preventive Services Task Force Recommendation Statement

US Preventive Services Task Force

IMPORTANCE Gestational diabetes is diabetes that develops during pregnancy. Prevalence of gestational diabetes in the US has been estimated at 5.8% to 9.2%, based on traditional diagnostic criteria, although it may be higher if more inclusive criteria are used. Pregnant persons with gestational diabetes are at increased risk for maternal and fetal complications, including preeclampsia, fetal macrosomia (which can cause shoulder dystocia and birth injury), and neonatal hypoglycemia. Gestational diabetes has also been associated with an increased risk of several long-term health outcomes in pregnant persons and intermediate outcomes in their offspring.

OBJECTIVE The USPSTF commissioned a systematic review to evaluate the accuracy, benefits, and harms of screening for gestational diabetes and the benefits and harms of treatment for the pregnant person and infant.

POPULATION Pregnant persons who have not been previously diagnosed with type 1 or type 2 diabetes.

EVIDENCE ASSESSMENT The USPSTF concludes with moderate certainty that there is a moderate net benefit to screening for gestational diabetes at 24 weeks of gestation or after to improve maternal and fetal outcomes. The USPSTF concludes that the evidence on screening for gestational diabetes before 24 weeks of gestation is insufficient, and the balance of benefits and harms of screening cannot be determined.

RECOMMENDATION The USPSTF recommends screening for gestational diabetes in asymptomatic pregnant persons at 24 weeks of gestation or after. (B recommendation) The USPSTF concludes that the current evidence is insufficient to assess the balance of benefits and harms of screening for gestational diabetes in asymptomatic pregnant persons before 24 weeks of gestation. (I statement)

JAMA. 2021;326(6):531-538. doi:10.1001/jama.2021.11922
Corrected on October 5, 2021.

Editorial page 487

Multimedia

Related article page 539 and JAMA Patient Page page 577

Supplemental content

CME Quiz at jamacmelookup.com and CME Questions page 565

Author/Group Information: The US Preventive Services Task Force (USPSTF) members are listed at the end of this article.

Corresponding Author: Karina W Davidson, PhD, MASc, Feinstein Institutes for Medical Research, 130 E 59th St, Ste 14C, New York, NY 10032 (chair@uspstf.net)

Summary of Recommendations

The USPSTF recommends screening for gestational diabetes in asymptomatic pregnant persons at 24 weeks of gestation or after.	B	
The USPSTF concludes that the current evidence is insufficient to assess the balance of benefits and harms of screening for gestational diabetes in asymptomatic pregnant persons before 24 weeks of gestation.	I	See the Figure for a more detailed summary of the recommendations for clinicians. USPSTF indicates US Preventive Services Task Force.

See the Summary of Recommendations figure.

Importance	persons with gestational diabetes are at increased risk for maternal and fetal complications, including preeclampsia, fetal macrosomia (which can cause shoulder dystocia and birth injury), and neonatal hypoglycemia. ^{3,9-11} Gestational diabetes has also been associated with an increased risk of several long-term health outcomes in pregnant persons and intermediate outcomes in their offspring. ¹²⁻¹⁶
Gestational diabetes is diabetes that develops during pregnancy. ¹⁻³	
Prevalence of gestational diabetes in the US has been estimated at 5.8% to 9.2%, based on traditional diagnostic criteria, although it may be higher if more inclusive criteria are used. ⁴⁻⁸ Pregnant	

Abbildung 11: Mit Surya erkannte Textbereiche auf der Beispielseite aus Abbildung 5

Clinical Review & Education

JAMA | US Preventive Services Task Force | RECOMMENDATION STATEMENT

Screening for Gestational Diabetes

US Preventive Services Task Force Recommendation Statement

US Preventive Services Task Force

IMPORTANCE Gestational diabetes is diabetes that develops during pregnancy. Prevalence of gestational diabetes in the US has been estimated at 5.8% to 9.2%, based on traditional diagnostic criteria, although it may be higher if more inclusive criteria are used. Pregnant persons with gestational diabetes are at increased risk for maternal and fetal complications, including preeclampsia, fetal macrosomia (which can cause shoulder dystocia and birth injury), and neonatal hypoglycemia. Gestational diabetes has also been associated with an increased risk of several long-term health outcomes in pregnant persons and intermediate outcomes in their offspring.

OBJECTIVE The USPSTF commissioned a systematic review to evaluate the accuracy, benefits, and harms of screening for gestational diabetes and the benefits and harms of treatment for the pregnant person and infant.

POPULATION Pregnant persons who have not been previously diagnosed with type 1 or type 2 diabetes.

EVIDENCE ASSESSMENT The USPSTF concludes with moderate certainty that there is a moderate net benefit to screening for gestational diabetes at 24 weeks of gestation or after to improve maternal and fetal outcomes. The USPSTF concludes that the evidence on screening for gestational diabetes before 24 weeks of gestation is insufficient, and the balance of benefits and harms of screening cannot be determined.

RECOMMENDATION The USPSTF recommends screening for gestational diabetes in asymptomatic pregnant persons at 24 weeks of gestation or after. (B recommendation) The USPSTF concludes that the current evidence is insufficient to assess the balance of benefits and harms of screening for gestational diabetes in asymptomatic pregnant persons before 24 weeks of gestation. (I statement)

JAMA. 2021;326(6):531-538. doi:10.1001/jama.2021.11922
Corrected on October 5, 2021.

← Editorial page 487

+ Multimedia

← Related article page 539 and JAMA Patient Page page 577

+ Supplemental content

+ CME Quiz at jamacmelookup.com and CME Questions page 565

Author/Group Information: The US Preventive Services Task Force. (USPSTF) members are listed at the end of this article.

Corresponding Author: Karina W. Davidson, PhD, MASc, Feinstein Institutes for Medical Research, 130 E 59th St, Ste 14C, New York, NY 10032 (chair@uspstf.net).

Summary of Recommendations

The USPSTF recommends screening for gestational diabetes in asymptomatic pregnant persons at 24 weeks of gestation or after.	B	See the Figure for a more detailed summary of the recommendations for clinicians. USPSTF indicates US Preventive Services Task Force.
The USPSTF concludes that the current evidence is insufficient to assess the balance of benefits and harms of screening for gestational diabetes in asymptomatic pregnant persons before 24 weeks of gestation.	I	

See the Summary of Recommendations figure.

Importance

Gestational diabetes is diabetes that develops during pregnancy.¹⁻³ Prevalence of gestational diabetes in the US has been estimated at 5.8% to 9.2%, based on traditional diagnostic criteria, although it may be higher if more inclusive criteria are used.⁴⁻⁸ Pregnant persons with gestational diabetes are at increased risk for maternal and fetal complications, including preeclampsia, fetal macrosomia (which can cause shoulder dystocia and birth injury), and neonatal hypoglycemia.^{3,9-11} Gestational diabetes has also been associated with an increased risk of several long-term health outcomes in pregnant persons and intermediate outcomes in their offspring.¹²⁻¹⁶

3.2.2.4 LayoutParser

LayoutParser ist eine Open-Source-Bibliothek für Deep-Learning-basierte Dokumentbildanalyse. Das Werkzeug stellt ein einheitliches Toolkit für die Anwendung und Anpassung von Modellen zur Layoutdetektion, Zeichenerkennung und weiteren Aufgaben der Dokumentverarbeitung bereit. Darüber hinaus umfasst LayoutParser eine Sammlung vortrainierter Modelle, Werkzeuge zur Annotation von Layoutdaten und Funktionen zur Modellanpassung für unterschiedliche Dokumenttypen. Die Bibliothek implementiert Python-Schnittstellen und kann in bestehende Verarbeitungspipelines integriert werden [64]. Im Bereich der Dokumentlayoutanalyse werden mit LayoutParser unter anderem Detectron2- und Mask-RCNN-Modelle für historische Dokumente bereitgestellt [64,69]. Damit ist das Werkzeug auf die lernbasierte Segmentierung und Organisation von Seiteninhalten ausgerichtet und unterstützt die Zerlegung von Dokumenten in analysierbare Layoutbereiche. Der Schwerpunkt liegt auf der Deep-Learning-basierten Layoutanalyse und der modularen Einbindung entsprechender Modelle in Dokumentverarbeitungsumgebungen [64]. Abbildung 12 zeigt die mit LayoutParser erkannten Layoutbereiche auf der bereits in Abbildung 5 dargestellten PDF-Seite. Sichtbar ist die Segmentierung der Seite in unterschiedliche Dokumentbereiche. Die farbliche Kennzeichnung weist die jeweils zugeordneten Layoutklassen aus.

3.2.3 KI-gestützte Werkzeuge

Werkzeuge zur KI-gestützten Dokumentverarbeitung sind auf die Verarbeitung komplexer Dokumente mit Text, Layout, Tabellen, Formeln und weiteren visuellen Dokumentbestandteilen ausgerichtet. Sie überführen Dokumentinhalte in strukturierte und weiterverarbeitbare Ausgaben und stellen diese in maschinenlesbaren Formaten für nachgelagerte Verarbeitungsschritte bereit [26,70]. Die folgenden fünf KI-gestützten Werkzeuge – LlamaParse, LiteParse, Docling, MinerU und Marker – stellen hierfür unterschiedliche technische Ansätze bereit.

3.2.3.1 LlamaParse

LlamaParse ist eine Dokumentverarbeitungslösung aus dem LlamaIndex-beziehungsweise LlamaCloud-Umfeld, die Dokumente in weiterverarbeitbare Ausgaben für KI-Pipelines überführt^{11,12}. LlamaParse ist Teil einer Plattform mit gemeinsamer API und gemeinsamem SDK, in der Parse zusammen mit Extract (Extraktion strukturierter

¹¹ <https://www.llamaindex.ai/llamaparse> [Zugriff am 18.04.2026].

¹² <https://developers.llamaindex.ai/llamaparse/> [Zugriff am 18.04.2026].

Daten nach vorgegebenem Schema), Classify (Kategorisierung von Dokumenten anhand natürlichsprachlicher Regeln), Split (Aufteilung zusammengefügtter Dokumente in logische Teile), Sheets (Verarbeitung tabellenähnlicher Daten mit zeilenbezogener Analyse) und Index (Aufbau einer gehosteten Vektorsuchpipeline für RAG) eingesetzt wird¹².

Abbildung 12: Mit LayoutParser erkannte Layoutbereiche auf der Beispielseite aus Abbildung 5

Clinical Review & Education

JAMA | US Preventive Services Task Force | RECOMMENDATION STATEMENT

Screening for Gestational Diabetes

US Preventive Services Task Force Recommendation Statement

US Preventive Services Task Force

IMPORTANCE Gestational diabetes is diabetes that develops during pregnancy. Prevalence of gestational diabetes in the US has been estimated at 5.8% to 9.2%, based on traditional diagnostic criteria, although it may be higher if more inclusive criteria are used. Pregnant persons with gestational diabetes are at increased risk for maternal and fetal complications, including preeclampsia, fetal macrosomia (which can cause shoulder dystocia and birth injury), and neonatal hypoglycemia. Gestational diabetes has also been associated with an increased risk of several long-term health outcomes in pregnant persons and intermediate outcomes in their offspring.

OBJECTIVE The USPSTF commissioned a systematic review to evaluate the accuracy, benefits, and harms of screening for gestational diabetes and the benefits and harms of treatment for the pregnant person and infant.

POPULATION Pregnant persons who have not been previously diagnosed with type 1 or type 2 diabetes.

EVIDENCE ASSESSMENT The USPSTF concludes with moderate certainty that there is a moderate net benefit to screening for gestational diabetes at 24 weeks of gestation or after to improve maternal and fetal outcomes. The USPSTF concludes that the evidence on screening for gestational diabetes before 24 weeks of gestation is insufficient, and the balance of benefits and harms of screening cannot be determined.

RECOMMENDATION The USPSTF recommends screening for gestational diabetes in asymptomatic pregnant persons at 24 weeks of gestation or after. (B recommendation) The USPSTF concludes that the current evidence is insufficient to assess the balance of benefits and harms of screening for gestational diabetes in asymptomatic pregnant persons before 24 weeks of gestation. (I statement)

JAMA. 2021;326(6):531-538. doi:10.1001/jama.2021.11922
Corrected on October 5, 2021.

- [← Editorial page 487](#)
- [+ Multimedia](#)
- [← Related article page 539 and JAMA Patient Page page 577](#)
- [+ Supplemental content](#)
- [+ CME Quiz at jamacmelookup.com and CME Questions page 565](#)

Author/Group Information: The US Preventive Services Task Force (USPSTF) members are listed at the end of this article.

Corresponding Author: Karina W. Davidson, PhD, MASC, Feinstein Institutes for Medical Research, 130 E 59th St, Ste 14C, New York, NY 10032 (chair@uspstf.net).

Summary of Recommendations

The USPSTF recommends screening for gestational diabetes in asymptomatic pregnant persons at 24 weeks of gestation or after.	B
The USPSTF concludes that the current evidence is insufficient to assess the balance of benefits and harms of screening for gestational diabetes in asymptomatic pregnant persons before 24 weeks of gestation.	I

See the Figure for a more detailed summary of the recommendations for clinicians. USPSTF indicates US Preventive Services Task Force.

See the Summary of Recommendations figure.

Importance

Gestational diabetes is diabetes that develops during pregnancy.¹⁻³ Prevalence of gestational diabetes in the US has been estimated at 5.8% to 9.2%, based on traditional diagnostic criteria, although it may be higher if more inclusive criteria are used.⁴⁻⁸ Pregnant

persons with gestational diabetes are at increased risk for maternal and fetal complications, including preeclampsia, fetal macrosomia (which can cause shoulder dystocia and birth injury), and neonatal hypoglycemia.^{3,9-11} Gestational diabetes has also been associated with an increased risk of several long-term health outcomes in pregnant persons and intermediate outcomes in their offspring.¹²⁻¹⁶

LlamaParse ist als API von LlamaIndex eingebunden¹³. LlamaParse ist auf die Verarbeitung komplexer Dokumente ausgerichtet. Als layoutbewusster OCR-Parser kann LlamaParse PDFs, Scans, komplexe Layouts, Tabellen, Diagramme, Handschrift, Auswahlkästchen und Bilder in weiterverarbeitbare Ausgaben wie Markdown, Text oder JSON überführen^{11,14}. In der Premium-Variante wird zusätzlich die Umwandlung von Gleichungen in LaTeX unterstützt, wodurch auch mathematische Formeln als verarbeitbare Dokumentelemente berücksichtigt werden können¹⁵. Layoutbewusste Verarbeitung, multimodales Parsing und verschiedene Parsing-Modi werden unterstützt, um Kosten und Genauigkeit an den jeweiligen Anwendungsfall anzupassen. LlamaParse unterstützt eine große Zahl an Dokumentformaten. Auf der Produktseite werden 90+ Formate und 130+ Formate in der Entwicklerdokumentation angegeben. LlamaParse ist nicht auf reinen Fließtext beschränkt, sondern berücksichtigt auch struktur- und layoutbezogene Dokumentbestandteile gezielt^{11,12}.

Die Ausgabe kann in Formaten erzeugt werden, die sich für nachgelagerte ETL-Prozesse und dokumentgestützte Frage-Antwort-Systeme mit Retrieval-Augmented Generation eignen. LlamaParse gibt Ergebnisse als Markdown, Text oder Raw-JSON aus. LlamaParse überführt Dokumentinhalte damit vor der weiteren Analyse in eine einheitliche, maschinenlesbare Repräsentation^{12,16}. Die Nutzung erfolgt über LlamaCloud mit API-Schlüssel und ist an die cloudbasierte Einbindung des Dienstes gekoppelt^{12,13}. Die cloudbasierte Nutzung bringt Abhängigkeit von Internetverbindung mit sich und kann bei sensiblen Dokumenten Datenschutzfragen aufwerfen. Zusätzlich bleibt das zugrunde liegende Modell für Nutzende intransparent [71].

LlamaParse Agentic erreicht im herstellereigenen Benchmark ParseBench den höchsten Gesamtscore über die fünf Bewertungsdimensionen Tabellen (unter anderem zusammengeführte Zellen, hierarchische Header und tabellenübergreifende Kontinuität), Charts (Extraktion exakter Datenpunkte mit korrekten Serien- und Achsenbezeichnungen), Content Faithfulness (Auslassungen, Halluzinationen und Lesereihenfolge), Semantic Formatting (semantisch relevante Formatierungen wie Durchstreichen, Hoch- und Tiefstellung, Fettdruck und Links) und Visual Grounding (Zuordnung extrahierter Inhalte zu ihrer Position im Dokument). Damit weist LlamaParse eine hohe Leistungsfähigkeit in mehreren für die Dokumentverarbeitung relevanten Bewertungsdimensionen auf [72]. LlamaParse ist für diese Arbeit vor allem deshalb

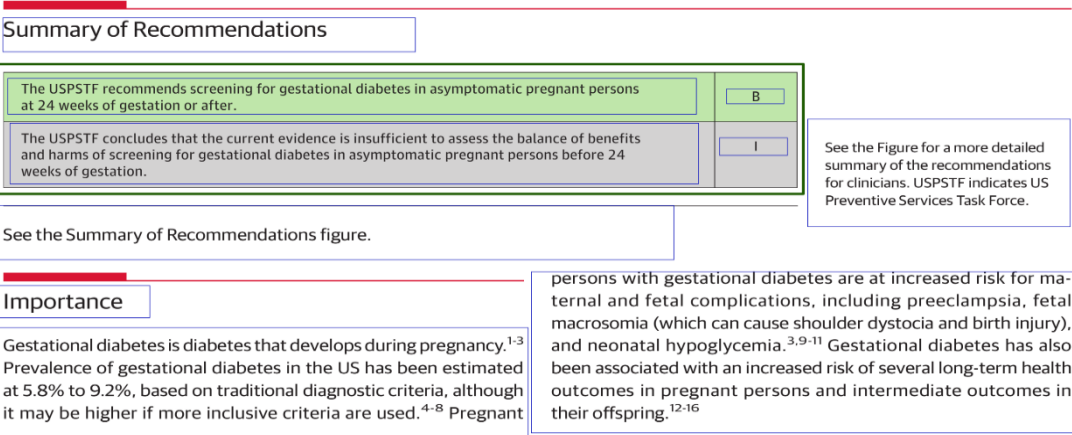
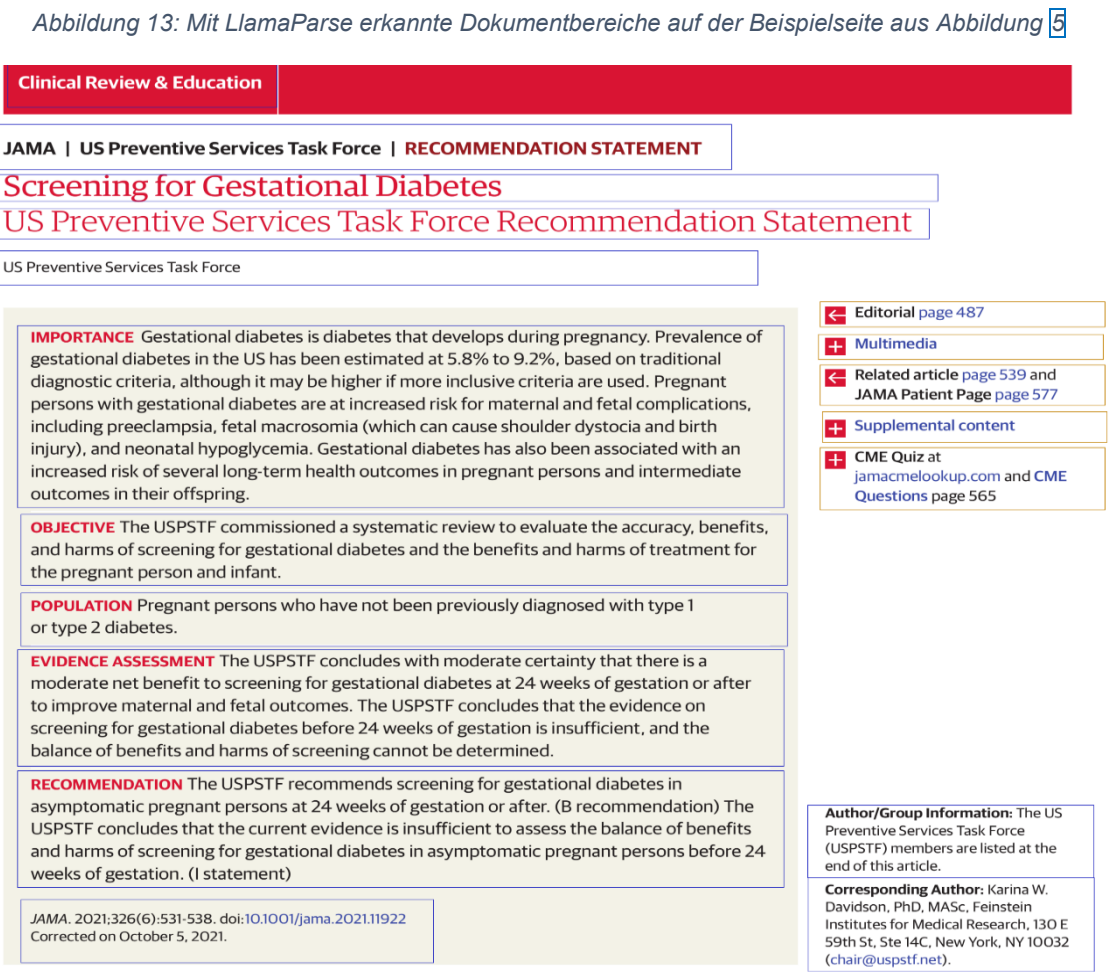
¹³ https://developers.llamaindex.ai/typescript/framework/modules/data/readers/llama_parse/ [Zugriff am 18.04.2026].

¹⁴ <https://developers.llamaindex.ai/llamaparse/parse/> [Zugriff am 18.04.2026].

¹⁵ <https://www.llamaindex.ai/blog/introducing-llamaparse-premium> [Zugriff am 18.04.2026].

¹⁶ <https://pypi.org/project/llama-parse/> [Zugriff am 18.04.2026].

relevant, weil das Werkzeug komplexe Dokumente in strukturierte, weiterverarbeitbare Ausgaben überführt und dabei neben Text auch Layout- und multimodale Dokumentelemente berücksichtigt^{11,12} [72]. Gleichzeitig stellt die cloudbasierte Anbindung für lokal ausgerichtete und datensensible Verarbeitungsszenarien eine Einschränkung dar¹² [71]. Abbildung 13 zeigt die mit LlamaParse erkannten Dokumentbereiche auf der bereits in Abbildung 5 dargestellten PDF-Seite. Sichtbar ist die struktur- und layoutbezogene Segmentierung der Seite in analysierbare Dokumentelemente.



3.2.3.2 LiteParse

LiteParse ist eine Open-Source-Bibliothek zur lokalen Dokumentverarbeitung im LlamaIndex-Umfeld. Das Werkzeug verarbeitet Dokumente vollständig auf der eigenen Maschine und ist nicht an Cloud-Abhängigkeiten, LLM-Aufrufe oder API-Schlüssel gebunden. LiteParse ist auf schnelle lokale Dokumentverarbeitung mit räumlicher Textextraktion, OCR und Bounding Boxes ausgerichtet. LiteParse unterstützt PDF-Dokumente, Office-Dateien und Bilder. Dazu zählen unter anderem DOCX, XLSX, PPTX, PNG und JPG. Text wird mit Positionsbezug zur Seite zurückgegeben, und für Textzeilen können Bounding Boxes extrahiert werden¹⁷.

LiteParse kann über die CLI (die Kommandozeile), aus TypeScript oder aus Python genutzt werden. Die Python-Nutzung erfolgt über ein Paket, das die LiteParse-CLI kapselt^{17,18}. Die CLI unterstützt die Befehle **parse**, **batch-parse** und **screenshot** und gibt PDF-Seiten optional auch als Bilder aus. Als Ausgabeformate stehen Text und JSON zur Verfügung, wobei die JSON-Ausgabe Positionsdaten und Bounding Boxes enthält. LiteParse überführt Dokumentinhalte damit vor der weiteren Analyse in eine einheitliche, maschinenlesbare Repräsentation¹⁹.

LiteParse unterstützt OCR für gescannte Dokumente über eine integrierte Tesseract.js-basierte Verarbeitung oder über einen angebundenen OCR-Server¹⁷. OCR kann deaktiviert werden, wenn native Textinhalte ausreichen. Für höhere Genauigkeit oder GPU-beschleunigte OCR kann ein HTTP-OCR-Server angebunden werden²⁰. LiteParse ist für diese Arbeit vor allem deshalb relevant, weil das Werkzeug Dokumente lokal in strukturierte, weiterverarbeitbare Ausgaben mit Positionsbezug überführt und damit für lokal ausgerichtete ETL-Prozesse geeignet erscheint. Die lokale Ausführung ist insbesondere für datensensible Verarbeitungsszenarien von Bedeutung, da keine cloudbasierte Dokumentübertragung erforderlich ist^{17,19}.

Abbildung 14 zeigt die mit LiteParse extrahierten Textbereiche auf der bereits in Abbildung 5 dargestellten PDF-Seite. Sichtbar ist die positionsbezogene Erfassung der Dokumentinhalte über Bounding Boxes.

¹⁷ <https://developers.llamaindex.ai/liteparse/> [Zugriff am 18.04.2026].

¹⁸ <https://developers.llamaindex.ai/liteparse/guides/library-usage/> [Zugriff am 18.04.2026].

¹⁹ <https://developers.llamaindex.ai/liteparse/cli-reference/> [Zugriff am 18.04.2026].

²⁰ <https://developers.llamaindex.ai/liteparse/guides/ocr/> [Zugriff am 18.04.2026].

Abbildung 14: Mit LiteParse extrahierte Textbereiche auf der Beispielseite aus Abbildung 5

Clinical Review & Education

JAMA | US Preventive Services Task Force | RECOMMENDATION STATEMENT

Screening for Gestational Diabetes

US Preventive Services Task Force Recommendation Statement

US Preventive Services Task Force

IMPORTANCE

Gestational diabetes is diabetes that develops during pregnancy. Prevalence of gestational diabetes in the US has been estimated at 5.8% to 9.2%, based on traditional diagnostic criteria, although it may be higher if more inclusive criteria are used. Pregnant persons with gestational diabetes are at increased risk for maternal and fetal complications, including preeclampsia, fetal macrosomia (which can cause shoulder dystocia and birth injury), and neonatal hypoglycemia. Gestational diabetes has also been associated with an increased risk of several long-term health outcomes in pregnant persons and intermediate outcomes in their offspring.

OBJECTIVE

The USPSTF commissioned a systematic review to evaluate the accuracy, benefits, and harms of screening for gestational diabetes and the benefits and harms of treatment for the pregnant person and infant.

POPULATION

Pregnant persons who have not been previously diagnosed with type 1 or type 2 diabetes.

EVIDENCE ASSESSMENT

The USPSTF concludes with moderate certainty that there is a moderate net benefit to screening for gestational diabetes at 24 weeks of gestation or after to improve maternal and fetal outcomes. The USPSTF concludes that the evidence on screening for gestational diabetes before 24 weeks of gestation is insufficient, and the balance of benefits and harms of screening cannot be determined.

RECOMMENDATION

The USPSTF recommends screening for gestational diabetes in asymptomatic pregnant persons at 24 weeks of gestation or after. (B recommendation) The USPSTF concludes that the current evidence is insufficient to assess the balance of benefits and harms of screening for gestational diabetes in asymptomatic pregnant persons before 24 weeks of gestation. (I statement)

JAMA. 2021;326(6):531-538. doi:10.1001/jama.2021.11922

Corrected on October 5, 2021.

← Editorial page 487

+ Multimedia

← Related article page 539 and JAMA Patient Page page 577

+ Supplemental content

+ CME Quiz at [jamacmelookup.com](#) and CME Questions page 565

Author/Group Information: The US Preventive Services Task Force (USPSTF) members are listed at the end of this article.

Corresponding Author: Karina W. Davidson, PhD, MASc, Feinstein Institutes for Medical Research, 130 E 59th St, Ste 14C, New York, NY 10032 ([chair@uspstf.net](#))

Summary of Recommendations

The USPSTF recommends screening for gestational diabetes in asymptomatic pregnant persons at 24 weeks of gestation or after.	B
The USPSTF concludes that the current evidence is insufficient to assess the balance of benefits and harms of screening for gestational diabetes in asymptomatic pregnant persons before 24 weeks of gestation.	I

See the Figure for a more detailed summary of the recommendations for clinicians. USPSTF indicates US Preventive Services Task Force.

See the Summary of Recommendations figure.

Importance

Gestational diabetes is diabetes that develops during pregnancy.¹⁻³ Prevalence of gestational diabetes in the US has been estimated at 5.8% to 9.2%, based on traditional diagnostic criteria, although it may be higher if more inclusive criteria are used.⁴⁻⁸ Pregnant

persons with gestational diabetes are at increased risk for maternal and fetal complications, including preeclampsia, fetal macrosomia (which can cause shoulder dystocia and birth injury), and neonatal hypoglycemia.⁹⁻¹¹ Gestational diabetes has also been associated with an increased risk of several long-term health outcomes in pregnant persons and intermediate outcomes in their offspring.¹²⁻¹⁶

3.2.3.3 Docling

Docling ist ein einfach nutzbares, selbstständiges und MIT-lizenziertes Open-Source-Toolkit zur Dokumentkonvertierung. Das Werkzeug ist als Python-Paket verfügbar, kann als Python-API oder als CLI genutzt werden und läuft vollständig lokal auf Standardhardware. Docling verarbeitet mehrere verbreitete Dokumentformate. Docling unterstützt PDF-Dokumente, Bilder, MS-Office-Formate und HTML. Die Dokumentverarbeitung umfasst Layoutanalyse, OCR, Bestimmung der Lesereihenfolge, Abbildungsextraktion und Tabellenstrukturekonstruktion. Docling ist damit nicht auf reinen Fließtext beschränkt, sondern verarbeitet auch strukturelle und visuelle Dokumentbestandteile gezielt²¹ [26].

Die Verarbeitung erfolgt über Parser-Backends und Pipelines, die Dokumentinhalte in eine einheitliche und maschinenlesbare Repräsentation überführen. Für PDF- und Bilddokumente werden Seiten geparkt, Layout- und Tabelleninformationen extrahiert und die Ergebnisse anschließend zusammengeführt. Für markupbasierte Formate wie Office-Dokumente oder HTML werden semantische Strukturen direkt in das gemeinsame Dokumentmodell überführt. Die Ausgabe kann als JSON, Markdown und HTML erzeugt werden. Docling überführt Dokumentinhalte damit vor der weiteren Analyse in eine einheitliche, maschinenlesbare Repräsentation. JSON dient dabei als verlustfreie Serialisierung, während Markdown und HTML nicht alle Metadaten vollständig erhalten [26].

Für PDF-Dokumente nutzt Docling spezialisierte Modelle für Layoutanalyse und Tabellenstrukturekonstruktion. OCR verarbeitet gescannte PDFs und Bildinhalte. Dafür integriert Docling EasyOCR und Tesseract, OCR ist der rechenintensivste Teil der Pipeline [26]. Zusätzlich stellt Docling mehrere PDF-Backends bereit, darunter **docling-parse** als Standard-Backend und pypdfium als alternative Option [26,73]. Docling ist für diese Arbeit vor allem deshalb relevant, weil das Werkzeug Dokumente lokal in strukturierte, weiterverarbeitbare Ausgaben überführt und dabei Layout, Lesereihenfolge, Tabellenstruktur und Metadaten gezielt berücksichtigt. Die lokale Ausführung ist insbesondere für datensensible und abgeschirmte Verarbeitungsszenarien von Bedeutung [26]. Abbildung 15 zeigt die mit Docling extrahierten Dokumentbereiche auf der bereits in Abbildung 5 dargestellten PDF-Seite. Sichtbar ist die positionsbezogene Erfassung von Text- und Strukturbereichen.

²¹ <https://www.docling.ai/> [Zugriff am 19.04.2026].

Abbildung 15: Mit Docling extrahierte Text- und Strukturbereiche auf der Beispielseite aus Abbildung 9

Clinical Review & Education

JAMA | US Preventive Services Task Force | RECOMMENDATION STATEMENT

Screening for Gestational Diabetes

US Preventive Services Task Force Recommendation Statement

US Preventive Services Task Force

IMPORTANCE Gestational diabetes is diabetes that develops during pregnancy. Prevalence of gestational diabetes in the US has been estimated at 5.8% to 9.2%, based on traditional diagnostic criteria, although it may be higher if more inclusive criteria are used. Pregnant persons with gestational diabetes are at increased risk for maternal and fetal complications, including preeclampsia, fetal macrosomia (which can cause shoulder dystocia and birth injury), and neonatal hypoglycemia. Gestational diabetes has also been associated with an increased risk of several long-term health outcomes in pregnant persons and intermediate outcomes in their offspring.

OBJECTIVE The USPSTF commissioned a systematic review to evaluate the accuracy, benefits, and harms of screening for gestational diabetes and the benefits and harms of treatment for the pregnant person and infant.

POPULATION Pregnant persons who have not been previously diagnosed with type 1 or type 2 diabetes.

EVIDENCE ASSESSMENT The USPSTF concludes with moderate certainty that there is a moderate net benefit to screening for gestational diabetes at 24 weeks of gestation or after to improve maternal and fetal outcomes. The USPSTF concludes that the evidence on screening for gestational diabetes before 24 weeks of gestation is insufficient, and the balance of benefits and harms of screening cannot be determined.

RECOMMENDATION The USPSTF recommends screening for gestational diabetes in asymptomatic pregnant persons at 24 weeks of gestation or after. (B recommendation) The USPSTF concludes that the current evidence is insufficient to assess the balance of benefits and harms of screening for gestational diabetes in asymptomatic pregnant persons before 24 weeks of gestation. (I statement)

JAMA. 2021;326(6):531-538. doi:10.1001/jama.2021.11922
Corrected on October 5, 2021.

Editorial page 487

Multimedia

Related article page 539 and JAMA Patient Page page 577

Supplemental content

CME Quiz at jamacmelookup.com and CME Questions page 565

Author/Group Information: The US Preventive Services Task Force (USPSTF) members are listed at the end of this article.

Corresponding Author: Karina W. Davidson, PhD, MASc, Feinstein Institutes for Medical Research, 130 E 59th St, Ste 14C, New York, NY 10032 (chair@uspstf.net).

Summary of Recommendations

The USPSTF recommends screening for gestational diabetes in asymptomatic pregnant persons at 24 weeks of gestation or after.	B
The USPSTF concludes that the current evidence is insufficient to assess the balance of benefits and harms of screening for gestational diabetes in asymptomatic pregnant persons before 24 weeks of gestation.	I

See the Figure for a more detailed summary of the recommendations for clinicians. USPSTF indicates US Preventive Services Task Force.

See the Summary of Recommendations figure.

Importance

Gestational diabetes is diabetes that develops during pregnancy.¹⁻³ Prevalence of gestational diabetes in the US has been estimated at 5.8% to 9.2%, based on traditional diagnostic criteria, although it may be higher if more inclusive criteria are used.⁴⁻⁸ Pregnant

persons with gestational diabetes are at increased risk for maternal and fetal complications, including preeclampsia, fetal macrosomia (which can cause shoulder dystocia and birth injury), and neonatal hypoglycemia.^{3,9-11} Gestational diabetes has also been associated with an increased risk of several long-term health outcomes in pregnant persons and intermediate outcomes in their offspring.¹²⁻¹⁶

3.2.3.4 MinerU

MinerU ist eine Open-Source-Lösung zur präzisen Dokumentinhalteextraktion, die Dokumente in strukturierte und weiterverarbeitbare Ausgaben überführt [70]. Das Werkzeug ist für LLM-, RAG- und Agent-Workflows ausgelegt und kann unter anderem über Python-SDK, CLI und REST API eingebunden werden. MinerU kann auch in privaten beziehungsweise offline ausgeführten Einsatzumgebungen betrieben werden. MinerU verarbeitet verschiedene Dokument- und Eingabeformate. MinerU unterstützt PDF-Dokumente, DOCX-, PPTX- und XLSX-Dateien, Bilder und Web-Seiten²². MinerU ist nicht nur auf wissenschaftliche Artikel ausgerichtet, sondern kann auch für Lehrbücher, Prüfungen, Forschungsberichte, Bücher, historische Dokumente, Notizen und Präsentationsfolien eingesetzt werden [70]. MinerU ist damit nicht auf reinen Fließtext beschränkt, sondern verarbeitet auch struktur- und layoutbezogene Dokumentbestandteile gezielt²² [70].

Die Dokumentverarbeitung kombiniert Vorverarbeitung, Inhaltsparsen, Nachverarbeitung und Formatkonvertierung. Im Parsing werden Layoutanalyse, Formel- und Tabellenerkennung sowie OCR kombiniert [70]. Formeln können nach LaTeX und Tabellen nach HTML ausgegeben werden. Zudem werden gescannte Dokumente, Handschrift, mehrspaltige Layouts und seitenübergreifende Tabellenzusammenführung unterstützt. Die Ausgabe folgt dabei der menschlichen Leserichtung, und Kopf- und Fußzeilen können automatisch entfernt werden. Die Ausgabe kann als Markdown und JSON erzeugt werden²² [70]. Zusätzlich werden Dateien für Weiterverarbeitung und Qualitätsprüfung erzeugt, darunter Zwischenrepräsentationen, Inhaltslisten in Lesereihenfolge und Visualisierungen. MinerU überführt Dokumentinhalte damit vor der weiteren Analyse in eine einheitliche, maschinenlesbare Repräsentation²³ [70].

Für die Texterkennung nutzt MinerU OCR nur auf zuvor erkannten Textregionen, um Fehler in der Leserichtung bei mehrspaltigen Dokumenten zu vermeiden. Für Textblöcke mit Inline-Formeln werden die Formeln zunächst maskiert, anschließend per OCR verarbeitet und danach wieder in das Ergebnis eingefügt. Für Tabellen, Formelerkennung und OCR werden spezialisierte Modelle eingesetzt [70]. Zusätzlich wird eine mehrsprachige OCR-Unterstützung angegeben. MinerU ist für diese Arbeit vor allem deshalb relevant, weil das Werkzeug Dokumente in strukturierte, weiterverarbeitbare Ausgaben überführt und dabei Layout, Tabellen, Formeln, OCR-Inhalte und Lesereihenfolge gezielt berücksichtigt. Für lokal ausgeführte Verarbeitungsszenarien ist zudem relevant, dass private und vollständig offline

²² <https://github.com/pendatolab/MinerU/blob/master/README.md> [Zugriff am 19.04.2026].

²³ https://pendatolab.github.io/MinerU/reference/output_files/ [Zugriff am 19.04.2026].

betriebene Einsatzumgebungen unterstützt werden. Für die Vorverarbeitung sind Chinesisch und Englisch explizit unterstützt; die Qualität weiterer Sprachen ist nicht garantiert²² [70]. Abbildung 16 zeigt die mit MinerU extrahierten Dokumentbereiche auf der bereits in Abbildung 5 dargestellten PDF-Seite. Dargestellt ist die positionsbezogene Zerlegung der Seite in analysierbare Inhaltsbereiche als Grundlage der weiteren strukturierten Verarbeitung.

3.2.3.5 Marker

Marker ist ein Open-Source-Werkzeug zur Dokumentkonvertierung, das Dokumente schnell und mit hoher Genauigkeit in Markdown, JSON, Chunks und HTML überführt. Das Werkzeug ist als Python-Paket verfügbar, kann lokal auf CPU- und GPU-basierten Systemen betrieben werden und lässt sich über die Kommandozeile nutzen^{24,25}. Marker verarbeitet verschiedene Dokument- und Eingabeformate. Marker unterstützt PDF-Dokumente, Bilder, PPTX-, DOCX-, XLSX-, HTML- und EPUB-Dateien; Für PDF- und Bilddateien kann Marker bei Bedarf OCR einsetzen. Marker ist damit nicht auf reinen Fließtext beschränkt, sondern verarbeitet auch struktur- und layoutbezogene Dokumentbestandteile gezielt²⁴. Die Dokumentverarbeitung erfolgt pipeline-basiert. Marker kombiniert Dokumentlayoutanalyse, Bestimmung der Lesereihenfolge sowie spezialisierte Verarbeitungsschritte für Formeln und Tabellen und führt die Ergebnisse anschließend in der Ausgabe zusammen [74]. Tabellen, Formulare, Gleichungen, Inline-Mathematik, Links, Referenzen und Codeblöcke werden formatiert, Bilder werden extrahiert und Kopf- und Fußzeilen sowie andere Artefakte entfernt²⁴.

Die Ausgabe kann als Markdown, JSON, HTML und Chunks erzeugt werden. Marker überführt Dokumentinhalte damit vor der weiteren Analyse in eine einheitliche, maschinenlesbare Repräsentation. Zusätzlich wird eine strukturierte Extraktion auf Basis eines JSON-Schemas im Beta-Status unterstützt. Marker kann über das Python-Paket **marker-pdf** installiert werden^{24,25}.

Für die Nutzung über die Kommandozeile sind unter anderem Aufrufe zur Konvertierung einzelner Dateien und zur Wahl des Ausgabeformats dokumentiert. Optional erhöht ein zusätzlicher LLM-Modus die Genauigkeit, etwa für seitenübergreifende Tabellen, Inline-Mathematik, Tabellenformatierung und Formularwerte²⁴. Für komplexe Layouts geben pipeline-basierte Verfahren Fehler aus der Layoutanalyse an nachgelagerte Verarbeitungsschritte weiter. Bei komplexen Layouts kann Marker Inline-Mathematik mit benachbartem Text falsch zusammenführen und Seitenzahlen als Fließtext klassifizieren

²⁴ <https://github.com/datalab-to/marker/blob/master/README.md> [Zugriff am 20.04.2026].

²⁵ <https://pypi.org/project/marker-pdf/> [Zugriff am 20.04.2026].

[74]. Marker ist für diese Arbeit vor allem deshalb relevant, weil das Werkzeug Dokumente in strukturierte, weiterverarbeitbare Ausgaben überführt und dabei neben Text auch Layout, Tabellen, Formeln, Bilder und Lesereihenfolge berücksichtigt²⁴ [74]. Gleichzeitig ist bei komplexen Layouts zu berücksichtigen, dass Fehler in einer pipeline-basierten Verarbeitung in spätere Verarbeitungsschritte fortgeschrieben werden können [74]. Die nachfolgende Abbildung 17 zeigt die mit Marker extrahierten Dokumentbereiche auf der bereits in Abbildung 5 dargestellten PDF-Seite. Sichtbar ist die struktur- und layoutbezogene Segmentierung der Seite in weiterverarbeitbare Inhaltsbereiche.

Abbildung 16: Mit MinerU extrahierte Dokumentbereiche auf der Beispielseite aus Abbildung 5

Clinical Review & Education

JAMA | US Preventive Services Task Force | RECOMMENDATION STATEMENT

Screening for Gestational Diabetes

US Preventive Services Task Force Recommendation Statement

US Preventive Services Task Force

IMPORTANCE

Gestational diabetes is diabetes that develops during pregnancy. Prevalence of gestational diabetes in the US has been estimated at 5.8% to 9.2%, based on traditional diagnostic criteria, although it may be higher if more inclusive criteria are used. Pregnant persons with gestational diabetes are at increased risk for maternal and fetal complications, including preeclampsia, fetal macrosomia (which can cause shoulder dystocia and birth injury), and neonatal hypoglycemia. Gestational diabetes has also been associated with an increased risk of several long-term health outcomes in pregnant persons and intermediate outcomes in their offspring.

OBJECTIVE

The USPSTF commissioned a systematic review to evaluate the accuracy, benefits, and harms of screening for gestational diabetes and the benefits and harms of treatment for the pregnant person and infant.

POPULATION

Pregnant persons who have not been previously diagnosed with type 1 or type 2 diabetes.

EVIDENCE ASSESSMENT

The USPSTF concludes with moderate certainty that there is a moderate net benefit to screening for gestational diabetes at 24 weeks of gestation or after to improve maternal and fetal outcomes. The USPSTF concludes that the evidence on screening for gestational diabetes before 24 weeks of gestation is insufficient, and the balance of benefits and harms of screening cannot be determined.

RECOMMENDATION

The USPSTF recommends screening for gestational diabetes in asymptomatic pregnant persons at 24 weeks of gestation or after. (B recommendation) The USPSTF concludes that the current evidence is insufficient to assess the balance of benefits and harms of screening for gestational diabetes in asymptomatic pregnant persons before 24 weeks of gestation. (I statement)

JAMA. 2021;326(6):531-538. doi:10.1001/jama.2021.11922

Corrected on October 5, 2021

Editorial page 487

Multimedia

Related article page 539 and JAMA Patient Page page 577

Supplemental content

CME Quiz at jamacmelookup.com and CME Questions page 565

Author/Group Information: The US Preventive Services Task Force (USPSTF) members are listed at the end of this article.

Corresponding Author: Karina W. Davidson, PhD, MASc, Feinstein Institutes for Medical Research, 130 E 59th St, Ste 14C, New York, NY 10032 (chair@uspstf.net).

Summary of Recommendations

The USPSTF recommends screening for gestational diabetes in asymptomatic pregnant persons at 24 weeks of gestation or after.	B
The USPSTF concludes that the current evidence is insufficient to assess the balance of benefits and harms of screening for gestational diabetes in asymptomatic pregnant persons before 24 weeks of gestation.	I

See the Figure for a more detailed summary of the recommendations for clinicians. USPSTF indicates US Preventive Services Task Force.

See the Summary of Recommendations figure.

Importance

Gestational diabetes is diabetes that develops during pregnancy.¹⁻³ Prevalence of gestational diabetes in the US has been estimated at 5.8% to 9.2%, based on traditional diagnostic criteria, although it may be higher if more inclusive criteria are used.⁴⁻⁸ Pregnant persons with gestational diabetes are at increased risk for maternal and fetal complications, including preeclampsia, fetal macrosomia (which can cause shoulder dystocia and birth injury), and neonatal hypoglycemia.^{3,9-11} Gestational diabetes has also been associated with an increased risk of several long-term health outcomes in pregnant persons and intermediate outcomes in their offspring.¹²⁻¹⁶

Abbildung 17: Mit Marker extrahierte Dokumentbereiche auf der Beispielseite aus Abbildung 5

Clinical Review & Education

JAMA | US Preventive Services Task Force | **RECOMMENDATION STATEMENT**

Screening for Gestational Diabetes

US Preventive Services Task Force Recommendation Statement

US Preventive Services Task Force

IMPORTANCE Gestational diabetes is diabetes that develops during pregnancy. Prevalence of gestational diabetes in the US has been estimated at 5.8% to 9.2%, based on traditional diagnostic criteria, although it may be higher if more inclusive criteria are used. Pregnant persons with gestational diabetes are at increased risk for maternal and fetal complications, including preeclampsia, fetal macrosomia (which can cause shoulder dystocia and birth injury), and neonatal hypoglycemia. Gestational diabetes has also been associated with an increased risk of several long-term health outcomes in pregnant persons and intermediate outcomes in their offspring.

OBJECTIVE The USPSTF commissioned a systematic review to evaluate the accuracy, benefits, and harms of screening for gestational diabetes and the benefits and harms of treatment for the pregnant person and infant.

POPULATION Pregnant persons who have not been previously diagnosed with type 1 or type 2 diabetes.

EVIDENCE ASSESSMENT The USPSTF concludes with moderate certainty that there is a moderate net benefit to screening for gestational diabetes at 24 weeks of gestation or after to improve maternal and fetal outcomes. The USPSTF concludes that the evidence on screening for gestational diabetes before 24 weeks of gestation is insufficient, and the balance of benefits and harms of screening cannot be determined.

RECOMMENDATION The USPSTF recommends screening for gestational diabetes in asymptomatic pregnant persons at 24 weeks of gestation or after. (B recommendation) The USPSTF concludes that the current evidence is insufficient to assess the balance of benefits and harms of screening for gestational diabetes in asymptomatic pregnant persons before 24 weeks of gestation. (I statement)

JAMA. 2021;326(6):531-538. doi:10.1001/jama.2021.11922
Corrected on October 5, 2021.

← Editorial page 487

+ Multimedia

← Related article page 539 and JAMA Patient Page page 577

+ Supplemental content

+ CME Quiz at jamacmelookup.com and CME Questions page 565

Author/Group Information: The US Preventive Services Task Force (USPSTF) members are listed at the end of this article.

Corresponding Author: Karina W. Davidson, PhD, MASC, Feinstein Institutes for Medical Research, 130 E 59th St, Ste 14C, New York, NY 10032 (chair@uspstf.net).

Summary of Recommendations

The USPSTF recommends screening for gestational diabetes in asymptomatic pregnant persons at 24 weeks of gestation or after.	B
The USPSTF concludes that the current evidence is insufficient to assess the balance of benefits and harms of screening for gestational diabetes in asymptomatic pregnant persons before 24 weeks of gestation.	I

See the Figure for a more detailed summary of the recommendations for clinicians. USPSTF indicates US Preventive Services Task Force.

See the Summary of Recommendations figure.

Importance

Gestational diabetes is diabetes that develops during pregnancy.¹⁻³ Prevalence of gestational diabetes in the US has been estimated at 5.8% to 9.2%, based on traditional diagnostic criteria, although it may be higher if more inclusive criteria are used.⁴⁻⁸ Pregnant

persons with gestational diabetes are at increased risk for maternal and fetal complications, including preeclampsia, fetal macrosomia (which can cause shoulder dystocia and birth injury), and neonatal hypoglycemia.^{3,9-11} Gestational diabetes has also been associated with an increased risk of several long-term health outcomes in pregnant persons and intermediate outcomes in their offspring.¹²⁻¹⁶

3.2.4 Vergleich der Werkzeuge

Für den weiteren Vergleich werden die KI-gestützten Werkzeuge LlamaParse, LiteParse, Docling, MinerU und Marker betrachtet. Die zuvor beschriebenen regel-, heuristik- und Deep-Learning-basierten Werkzeuge bilden wichtige technische Grundlagen der PDF-Verarbeitung, adressieren jedoch überwiegend einzelne Verarbeitungsschritte wie Textextraktion, OCR, Tabellenextraktion oder Layoutdetektion. Der in dieser Arbeit entwickelte ETL-Prozess benötigt dagegen eine möglichst zusammenhängende Vorverarbeitung, bei der medizinische PDF-Inhalte mit Text, Tabellen, Layoutstrukturen und strukturierten Abbildungen in eine einheitliche und maschinenlesbare Repräsentation überführt werden. Für diese Zielsetzung sind KI-gestützte Dokumentverarbeitungswerkzeuge besonders geeignet, da sie mehrere dieser Verarbeitungsschritte innerhalb einer Dokumentkonvertierung bündeln und Ausgaben bereitstellen, die unmittelbar für nachgelagerte LLM-basierte Extraktionsschritte genutzt werden können. Der Vergleich konzentriert sich daher auf diese Werkzeuggruppe. Der allgemeine Werkzeugvergleich und der Vergleich unterstützter Dokumentelemente basieren auf den in den Abschnitten [3.2.3.1](#) bis [3.2.3.5](#) dargestellten Werkzeugeigenschaften. Eine vertiefte vergleichende Analyse der Ausgaben dieser Werkzeuge an einer gemeinsamen Beispielseite erfolgt in Kapitel [4](#).

3.2.4.1 Allgemeiner Werkzeugvergleich

Die Tabelle [1](#) fasst die zuvor beschriebenen KI-gestützten Werkzeuge anhand zentraler Merkmale zusammen. Der Vergleich berücksichtigt Betriebsform, unterstützte Eingabeformate, Integrationsmöglichkeiten, Ausgabeformate sowie die in den Einzelabschnitten genannten Einschränkungen.

Die Tabelle [1](#) zeigt, dass alle betrachteten Werkzeuge Dokumentinhalte in weiterverarbeitbare und maschinenlesbare Ausgaben überführen können. Unterschiede bestehen vor allem in der Betriebsform, den unterstützten Eingabeformaten, den Integrationsmöglichkeiten und den Ausgabeformaten. LlamaParse ist in eine cloudbasierte LlamaCloud- und LlamaIndex-Umgebung eingebunden, während LiteParse, Docling, MinerU und Marker lokal einsetzbare Alternativen darstellen. Für datensensible Verarbeitungsszenarien sind daher insbesondere die lokal ausführbaren Werkzeuge relevant.

Tabelle 1: Allgemeiner Vergleich der KI-gestützten Werkzeuge

Werk- zeug	Betriebs- -form	Eingabe- -formate	Integration	Ausgabe- -formate	Genannte Einschränkungen
Llama- Parse	Cloud- basiert	90+ Formate, u. a. PDF, Office- Bild- und Tabellenformate	LlamaIndex -API, CLI	Markdown, Text, Raw- JSON	Cloud-Abhängigkeit, Datenschutzfragen bei sensiblen Dokumenten, Intransparenz des zugrunde liegenden Modells
Lite- Parse	Lokal	PDF, Office- Formate und Bildformate	CLI, TypeScript, Python	Text, JSON	Für höhere Genauigkeit oder GPU- beschleunigte OCR kann ein HTTP-OCR- Server angebunden werden
Docling	Lokal	PDF, Office- Formate, HTML und Bildformate	Python- Paket, Python-API, CLI	JSON, Markdown, HTML	OCR wird als rechenintensivster Teil der Pipeline beschrieben; Markdown und HTML erhalten nicht alle Metadaten vollständig
MinerU	Lokal	PDF, Office- Formate, Bildformate und Web-Seiten	Python- SDK, CLI, REST API	Markdown, JSON; Tabellen nach HTML, Formeln nach LaTeX	Für die Vorverarbeitung werden im Paper direkt nur Chinesisch und Englisch als unterstützte Sprachen genannt; Qualität anderer Sprachen wird dort nicht garantiert
Marker	Lokal	PDF, Office- Formate, HTML, EPUB und Bildformate	Python- Paket, CLI	Markdown, JSON, HTML, Chunks	Bei pipeline-basierten Verfahren können Fehler aus der Layoutanalyse in nachgelagerte Schritte weitergegeben werden; optionaler LLM-Modus kann die Genauigkeit erhöhen

3.2.4.2 Vergleich unterstützter Dokumentelemente

Tabelle 2 fasst zusammen, welche Dokumentelemente von den betrachteten KI-gestützten Werkzeugen laut Dokumentation beziehungsweise anhand der zuvor beschriebenen Werkzeugeigenschaften grundsätzlich unterstützt oder verarbeitet werden können. Ein Pluszeichen (+) bedeutet, dass eine entsprechende Unterstützung belegt ist; ein leeres Feld bedeutet, dass diese Unterstützung in den herangezogenen Quellen nicht eindeutig nachgewiesen wurde. Die Spalte Diagramme meint dabei datentragende Diagramme, deren Inhalt in Datenpunkte überführt wird, und nicht das bloße Erfassen der Abbildung als Bildbereich. Die Tabelle bewertet nicht die Extraktionsqualität und sagt nicht aus, dass jedes Element in jeder konkreten JSON-Ausgabe als eigener strukturierter Typ erscheint.

Tabelle 2: Unterstützte Dokumentelemente der KI-gestützten Werkzeuge

Werkzeug	Text	Layout	Bilder/ Abbildungen	Tabellen	OCR	Formeln	Diagramme
LlamaParse	+	+	+	+	+	+	+
LiteParse	+	+	+		+		
Docling	+	+	+	+	+	+	
MinerU	+	+	+	+	+	+	
Marker	+	+	+	+	+	+	

Die Übersicht zeigt, dass alle betrachteten Werkzeuge Text, Layoutinformationen, Bilder beziehungsweise Abbildungen sowie OCR verarbeiten können. Unterschiede bestehen vor allem bei Tabellen, Formeln und Diagrammen. Während die Verarbeitung von Diagrammen ausschließlich von LlamaParse unterstützt wird, decken die übrigen Merkmale jeweils nur einen Teil der betrachteten Werkzeuge ab. Diese Merkmale sind für die strukturierte Vorverarbeitung multimodaler PDF-Dokumente relevant, da sie bestimmen, welche Dokumentbestandteile durch das jeweilige Werkzeug adressiert werden können.

3.3 Lokale LLMs für Informations- und Kausalitätsextraktion

Nach der Dokumentextraktion liegen Inhalte multimodaler PDF-Dokumente in weiterverarbeitbaren Repräsentationen wie Markdown, JSON oder HTML vor. Dokumentparsen dient der Überführung unstrukturierter oder semi-strukturierter Dokumente in strukturierte, maschinenlesbare Formate und bildet damit eine Grundlage für nachgelagerte Aufgaben wie Informationsextraktion, Retrieval und dokumentbezogene Analyse [76].

Für die weitere Verarbeitung medizinischer Dokumentinhalte sind LLMs relevant, da medizinische Informationen häufig in unstrukturierten Texten vorliegen und für analytische Zwecke in strukturierte Merkmale überführt werden müssen [6]. Neben der allgemeinen Informationsextraktion ist für diese Arbeit die Kausalitätsextraktion zentral. Die Identifikation kausaler Beziehungen zwischen Ereignissen oder Entitäten gilt als grundlegende NLP-Aufgabe und bildet eine Teilaufgabe der biomedizinischen Informationsextraktion [9]. Gerade im medizinischen Bereich kommt kausalen Zusammenhängen eine besondere Rolle zu, da medizinische Erklärungen wesentlich auf Ursache-Wirkungs-Beziehungen beruhen. LLMs extrahieren kausale Relationen aus medizinischen Texten wie klinischen Leitlinien sowie aus Titeln und Abstracts wissenschaftlicher Publikationen [11,77]. Für LLM-basierte Informationsextraktion aus medizinischen Dokumenten sind strukturierte Ausgabeformate wie JSON, ausreichende Kontextgröße, Open-Source-Verfügbarkeit und lokale Nutzbarkeit wichtige Kriterien [8]. Vor diesem Hintergrund werden im Folgenden lokal ausführbare Modellfamilien anhand ausgewählter Kriterien, darunter lokale Nutzbarkeit, strukturierte Ausgabeformate und Modellgröße beziehungsweise Kontextumfang, für die Informations- und Kausalitätsextraktion medizinischer Dokumentinhalte eingeordnet.

3.3.1 Llama

Llama ist eine von Meta entwickelte Modellfamilie offengewichteter LLMs. Die Llama-3.1-Generation umfasst textbasierte Modelle mit 8B, 70B und 405B Parametern sowie vortrainierte und instruction-tuned Varianten. Die Modelle unterstützen Mehrsprachigkeit, Coding, Reasoning und Tool Use. Die größte Variante besitzt ein Kontextfenster von bis zu 128K Tokens [78].

Llama 3.3 ist eine textbasierte instruction-tuned Modellvariante mit 70B Parametern. Das Modell verarbeitet Text als Eingabe und erzeugt Text als Ausgabe. Llama-3.3-70B-Instruct besitzt ein Kontextfenster von 128K Tokens und ist auf mehrsprachige dialogorientierte Anwendungen ausgerichtet²⁶. Llama-3.3-70B-Instruct kann außerdem lokal betrieben werden. Für die lokale Ausführung werden unter anderem quantisierte Modellvarianten und Laufzeitumgebungen wie **llama-cpp-python** eingesetzt. Llama-3.3-70B-Instruct lässt sich lokal mit Quantisierung ausführen und erzeugt dabei JSON-basierte Ausgaben aus radiologischen Freitextberichten [79]. Darüber hinaus lässt sich Llama-3.3-70B-Instruct über Ollama auf einem lokalen Server innerhalb einer geschützten Krankenhausinfrastruktur betreiben [80].

²⁶ <https://huggingface.co/meta-llama/Llama-3.3-70B-Instruct> [Zugriff am 16.05.2026].

Mit Llama 4 wurden neuere multimodale Varianten der Modellfamilie veröffentlicht. Llama 4 Scout und Llama 4 Maverick sind Mixture-of-Experts-Modelle mit nativer multimodaler Verarbeitung. Beide Modelle verfügen über 17B aktive Parameter; Scout nutzt 16 Experten, während Maverick 128 Experten verwendet. Llama 4 Scout ist auf sehr lange Kontextverarbeitung ausgelegt, während Llama 4 Maverick als allgemeines multimodales Modell positioniert wird²⁷.

3.3.2 DeepSeek

DeepSeek ist eine Modellfamilie offen verfügbarer großer Sprach- und multimodaler Modelle, die neben allgemeinen Sprachmodellen auch spezialisierte Varianten für Reasoning, Code, Mathematik und multimodale Verarbeitung umfasst. Die Entwicklungslinie reicht von DeepSeek-LLM über DeepSeek-V2 und DeepSeek-V3 bis zu DeepSeek-R1; ergänzend existieren multimodale Varianten wie DeepSeek-VL und DeepSeek-VL2 [81]. Mit DeepSeek-V3.2 wurde diese Entwicklungslinie um eine neuere Modellgeneration mit Reasoning- und Agent-Fokus erweitert [82].

DeepSeek-V3 ist ein Mixture-of-Experts-Modell mit 671B Gesamtparametern, von denen pro Token 37B Parameter aktiviert werden. Die Architektur ist auf effiziente Inferenz und kostenreduziertes Training ausgelegt und kombiniert dafür Multi-head Latent Attention mit DeepSeekMoE. Das Modell wurde auf 14,8T Tokens vortrainiert, anschließend durch Supervised Fine-Tuning und Reinforcement Learning weiter angepasst und unterstützt eine Kontextlänge von bis zu 128K Tokens [83].

DeepSeek-R1 erweitert die Modellfamilie um eine auf Reasoning ausgerichtete Variante. Die Modellreihe umfasst neben dem vollständigen DeepSeek-R1 auch destillierte Modelle mit 1.5B, 7B, 8B, 14B, 32B und 70B Parametern, die auf Qwen2.5- und Llama-3-Modellen basieren²⁸. Diese destillierten Varianten sind für die Betrachtung lokaler LLMs relevant, da sie im Vergleich zu den vollständigen 671B-Modellen deutlich kleinere Modellgrößen bereitstellen.

Für medizinische Anwendungen findet DeepSeek-R1 Einsatz in medizinischem Wissen, klinischem Reasoning und radiologischer Textverarbeitung. Das Modell ordnet textbasierte klinische Fälle diagnostisch ein, verarbeitet klinische Informationen strukturiert und stellt diagnostische Überlegungen nachvollziehbar dar; Grenzen bestehen jedoch bei offenen Diagnoseaufgaben, bei der Zusammenfassung von Bildgebungsberichten und bei der Reproduzierbarkeit einzelner Ausgaben [84]. Bei der diagnostischen Bearbeitung von MedQA-Fällen zeigt DeepSeek-R1 wiederkehrende

²⁷ <https://ai.meta.com/blog/llama-4-multimodal-intelligence/> [Zugriff am 16.05.2026].

²⁸ <https://huggingface.co/deepseek-ai/DeepSeek-R1> [Zugriff am 17.05.2026].

Schwächen wie Anchoring Bias, Schwierigkeiten bei der Integration widersprüchlicher Befunde und eine begrenzte Berücksichtigung alternativer Diagnosen [85]. In realen Krankenhausanwendungen unterstützt DeepSeek vor allem klinische Entscheidungsfindung, Diagnoseformulierung, klinische Dokumentation und Behandlungsempfehlungen, weist jedoch Defizite bei Transparenz und Validierung auf, da Vorabprüfungen, Bewertungsergebnisse sowie Risiken und Gegenmaßnahmen nicht durchgängig offengelegt werden [86].

Mit DeepSeek-V3.2 wurde eine neuere Modellgeneration veröffentlicht, die Reasoning- und Agent-Fähigkeiten mit höherer Recheneffizienz verbindet. Eine zentrale technische Änderung ist DeepSeek Sparse Attention, wodurch die Verarbeitung langer Kontexte effizienter gestaltet wird. Ergänzend nutzt das Modell ein skalierbares Reinforcement-Learning-Framework und eine Pipeline zur Synthese agentischer Aufgaben. Als stärkere Reasoning-Variante adressiert DeepSeek-V3.2-Speciale besonders rechenintensive Reasoning-Aufgaben. DeepSeek-V3.2 steht zunächst nur über API zur Verfügung; eine spätere Modellveröffentlichung ist angekündigt²⁹ [82].

Für multimodale Verarbeitung existiert mit DeepSeek-VL eine Vision-Language-Modellreihe. DeepSeek-VL ist auf reale Vision-Language-Szenarien ausgelegt und berücksichtigt unter anderem Webseiten-Screenshots, PDFs, OCR, Diagramme und wissenschaftliche Inhalte. Die Modellfamilie umfasst öffentlich verfügbare Varianten mit 1.3B und 7B Parametern [87].

3.3.3 gpt-oss

gpt-oss ist eine von OpenAI veröffentlichte Modellfamilie offengewichteter Reasoning-Modelle. Die Familie umfasst gpt-oss-120b und gpt-oss-20b, die unter der Apache-2.0-Lizenz bereitgestellt werden. Beide Modelle sind textbasierte Mixture-of-Experts-Transformer und unterstützen Reasoning, Tool Use, agentische Workflows sowie strukturierte Ausgaben [88]. gpt-oss-120b besitzt 116,83B Gesamtparameter, von denen 5,13B Parameter pro Token aktiv sind. gpt-oss-20b besitzt 20,91B Gesamtparameter mit 3,61B aktiven Parametern pro Token. Durch die MoE-Architektur wird bei der Inferenz nur ein Teil der Gesamtparameter genutzt. Die Modelle verwenden MXFP4-Quantisierung zur Reduktion des Speicherbedarfs; dadurch kann gpt-oss-120b auf einer einzelnen GPU mit 80 GB Speicher und gpt-oss-20b auf Systemen mit etwa 16 GB Speicher betrieben werden. Beide Modelle unterstützen eine Kontextlänge von 131.072 Tokens [88].

²⁹ <https://api-docs.deepseek.com/news/news251201> [Zugriff am 17.05.2026].

Für medizinische Anwendungen ist gpt-oss vor allem wegen der lokalen Ausführbarkeit relevant. On-Device-LLMs wie gpt-oss-20b und gpt-oss-120b wurden für klinische Entscheidungsunterstützung in Aufgaben wie allgemeiner Diagnostik, ophthalmologischer Diagnose und Management sowie Simulation klinischer Bewertung untersucht. Lokale Ausführung ermöglicht dabei Datenschutz, Reproduzierbarkeit und geringere Abhängigkeit von cloudbasierter Infrastruktur [89]. Auch für sicherheitsbezogene medizinische Dialoganalyse wurden gpt-oss:120b und gpt-oss:20b in einem Vergleich mit weiteren lokalen und cloudbasierten Modellen berücksichtigt. Die betrachtete Aufgabe bestand in der Erkennung medizinischer, ethischer und rechtlicher Risiken in mehrsprachigen medizinischen Dialogen. Lokale LLMs sind für medizinische Anwendungen besonders relevant, da diese Anforderungen an Datenschutz und Datensouveränität stellen [90].

Neben den textbasierten gpt-oss-Modellen existiert mit MEDGPT-OSS eine biomedizinische Erweiterung, die auf dem gpt-oss-Sprachbackbone aufbaut. MEDGPT-OSS ist ein offengewichtiges biomedizinisches Vision-Language-Modell mit 20B Parametern, das den gpt-oss-Sprachkern mit einem visuellen Front-End kombiniert. Dadurch wird gpt-oss selbst nicht als multimodales Modell beschrieben; der Sprachbackbone kann jedoch als Grundlage für biomedizinische multimodale Systeme dienen [88,91].

Für die praktische lokale Ausführung ist außerdem die Effizienz der MoE-Architektur relevant. gpt-oss-20B umfasst 20,9B Gesamtparameter, von denen bei der Inferenz etwa 3,61B aktiv sind. Auf einer einzelnen H100-GPU zeigt das Modell dabei kompakte Latenz-, Durchsatz- und Speicherwerte, was es für den lokalen Betrieb geeignet macht [92].

3.3.4 Qwen

Qwen ist eine von Alibaba entwickelte Modellfamilie offen veröffentlichter großer Sprach- und multimodaler Modelle. Die Modellfamilie umfasst Basismodelle, Chat-Modelle sowie spezialisierte Varianten für Coding, Mathematik und multimodale Verarbeitung. Die ersten Qwen-Modelle umfassen Varianten mit 1.8B, 7B und 14B Parametern. Zusätzlich wurden Qwen-Chat, Code-Qwen, Math-Qwen-Chat und Qwen-VL als erweiterte Modelllinien bereitgestellt [93]. Die Qwen-Chat-Modelle sind auf dialogorientierte Nutzung und menschliche Präferenzen ausgerichtet. Dafür werden Supervised Fine-Tuning und Reinforcement Learning from Human Feedback eingesetzt. Neben allgemeinen Sprachaufgaben unterstützt die Modellfamilie auch Tool Use, Code Interpreter und agentische Anwendungen. Qwen-VL erweitert die Familie um die

Verarbeitung visueller und sprachlicher Instruktionen und unterstützt unter anderem Textverständnis in Bildern, visuelle Verankerung sowie Multi-Image-Dialoge [93].

Für medizinische Anwendungen erzielt Qwen-2.5 in der Chinese National Nursing Licensing Examination anhand von 1200 Multiple-Choice-Fragen aus den Jahren 2019 bis 2023 mit 88,9 % die höchste Gesamtgenauigkeit unter sieben getesteten LLMs; eine XGBoost-basierte Kombination der Modellausgaben steigert diese auf 90,8 %. Die Untersuchung betrachtet damit domänenspezifisches Pflegewissen und klinische Entscheidungsfragen in einem medizinischen Prüfungssetting [94].

Für patientenzentrierte medizinische Sprachverarbeitung existiert mit MedQwen-PE eine auf Qwen3-1.7B basierende Variante. MedQwen-PE passt Qwen3-1.7B mithilfe von Quantized Low-Rank Adaptation (QLoRA) für Zusammenfassung, Fragebeantwortung und Informationsextraktion in multilingualen medizinischen Dialogen an. QLoRA ist eine parameter-effiziente Fine-Tuning-Methode, bei der das vortrainierte Modell nicht vollständig neu trainiert wird, sondern nur zusätzliche Adapterparameter angepasst werden. In MedQwen-PE werden dadurch nur etwa 1,6 % der Modellparameter aktualisiert. Qwen3-1.7B besitzt 1,7B Parameter und ein Kontextfenster von 32.768 Tokens, wodurch längere medizinische Dialoge und Dokumente verarbeitet werden können [95].

Multimodale Qwen-Varianten wurden ebenfalls für medizinische Aufgaben verwendet. Qwen2-VL ist ein Vision-Language-Modell mit mehreren Größenvarianten, darunter 2B, etwa 7/8B und 72B. Das Modell verarbeitet Text-, Bild- und Videoinformationen und nutzt unter anderem dynamische Bildauflösung sowie Multimodal Rotary Position Embedding. Für medizinische Berichtgenerierung passt Qwen2-VL einen dedizierten medizinischen Instruction-Fine-Tuning-Datensatz an und übertrifft dabei andere Vision-Language-Modelle im Vergleich [96].

3.3.5 Mistral

Mistral ist eine von Mistral AI entwickelte Modellfamilie offengewichteter und kommerzieller LLMs. Die Modellfamilie umfasst kleinere dichte Modelle wie Mistral 7B, Mixture-of-Experts-Modelle wie Mixtral 8x7B sowie größere Modelle wie Mistral Small, Mistral Medium und Mistral Large. Mit Mistral 3 wurden zusätzlich kleinere dichte Modelle mit 14B, 8B und 3B Parametern sowie Mistral Large 3 als sparse Mixture-of-Experts-

Modell mit 41B aktiven und 675B Gesamtparametern veröffentlicht. Die Mistral-3-Modelle werden unter Apache-2.0-Lizenz bereitgestellt^{30,31}.

Für die lokale und ressourceneffiziente Verarbeitung medizinischer Freitexte ist Mistral Small 24B relevant. Für die Extraktion onkologischer Attribute aus deutschen Pathologieberichten erzeugen offene, instruction-tuned Modelle in einer sicheren On-Premise-Umgebung strukturierte JSON-Ausgaben. Mistral Small 24B erreicht dabei in Kombination mit Retrieval-Augmented Generation nahezu die Leistung von Llama 3.3 70B, benötigt jedoch deutlich weniger Rechenressourcen [97]. Mistral Large wurde ebenfalls für strukturierte medizinische Extraktion eingesetzt. Mistral-Large-2411 extrahiert ohne Fine-Tuning den Status von Estrogenrezeptor, Progesteronrezeptor und HER2 aus Freitext-Pathologieberichten mithilfe strukturierter JSON-Schemata, Postprocessing und Prompting-Strategien wie Confidence Prompt, Chain-of-Thought und Few-Shot-Beispielen und erzielt dabei auf Dokumentebene F1-Werte von mehr als 95 % [98].

Neben allgemeinen Mistral-Modellen existieren auch domänenspezifische Weiterentwicklungen. CancerLLM ist ein Mistral-style-Modell mit 7B Parametern, das auf klinischen Krebsnotizen und Pathologieberichten trainiert wurde. Das Modell wurde für cancer phenotype extraction und diagnosis generation angepasst und erzielte auf den internen Benchmarks F1-Werte von 91,78 % für die Phänotyp-Extraktion und 86,81 % für die Diagnosegenerierung [99].

Für mehrsprachige medizinische Dokumentverarbeitung wurden Mistral- und Mixtral-Modelle außerdem bei der Übersetzung radiologischer Freitextberichte untersucht. Mistral 7B, Mixtral 8x7B, Mixtral 8x22B und Mistral Large übersetzen CT- und MRT-Berichte in mehrere Sprachen, wobei die Qualität je nach Modell und Sprachpaar variiert und kein Modell durchgehend überlegen ist [100].

Aktuelle Mistral-Modelle erweitern die Familie um längere Kontextfenster und multimodale Fähigkeiten. Mistral Small 3.2 besitzt ein Kontextfenster von 128K Tokens und ist als Aktualisierung der vorherigen Small-Generation beschrieben; Mistral Large 3 wird als multimodales und mehrsprachiges Open-Weight-Modell mit 256K Kontextfenster eingeordnet^{30,32}.

³⁰ <https://mistral.ai/models> [Zugriff am 19.05.2026].

³¹ <https://mistral.ai/news/mistral-3> [Zugriff am 19.05.2026].

³² <https://docs.mistral.ai/models/model-cards/mistral-small-3-2-25-06> [Zugriff am 19.05.2026].

3.3.6 Vision-Language-Modelle für die Bildbeschreibung

Wie der Vergleich der Ausgaben in Abschnitt 4.3 zeigt, überführt keines der lokal einsetzbaren Werkzeuge den eigentlichen Inhalt einer Abbildung in Text oder strukturierte Daten. Die zuvor betrachteten textbasierten Modelle können diese Lücke nicht schließen, da sie ausschließlich Text verarbeiten. Für die Überführung von Bildinhalten in eine textuelle Beschreibung sind daher Vision-Language-Modelle relevant, die visuelle und sprachliche Informationen gemeinsam verarbeiten.

Mehrere der zuvor betrachteten Modellfamilien stellen lokal ausführbare Vision-Language-Varianten bereit, etwa Qwen2-VL sowie DeepSeek-VL (siehe Abschnitte 3.3.2 und 3.3.4). Qwen2-VL verarbeitet Text- und Bildinformationen gemeinsam und wurde in einer Untersuchung zur medizinischen Berichtgenerierung evaluiert, in der es andere Vision-Language-Modelle übertraf [96]. Mit Qwen3-VL setzt dieselbe Reihe diese Entwicklung fort. Die Modellfamilie umfasst dichte Varianten mit 2B, 4B, 8B und 32B Parametern sowie zwei Mixture-of-Experts-Varianten und unterstützt ein Kontextfenster von 256K Tokens³³. Solche Modelle können eine Abbildung als Eingabe aufnehmen und eine textuelle Beschreibung ihres Inhalts erzeugen, die sich anschließend gemeinsam mit den übrigen Dokumentinhalten weiterverarbeiten lässt. Damit ergänzen sie die textbasierte Verarbeitung um einen Zugang zu Bildinhalten, ohne selbst die Aufgabe der Informations- und Kausalitätsextraktion zu übernehmen.

3.3.7 Vergleich der LLMs

Die zuvor betrachteten Modellfamilien unterscheiden sich anhand zentraler Merkmale, darunter Anbieter, Modellgröße, multimodale Verarbeitung und medizinischer Bezug. Tabelle 3 fasst die in den Abschnitten 3.3.1 bis 3.3.5 beschriebenen Eigenschaften zusammen. Berücksichtigt werden ausschließlich die dort dargestellten Modellvarianten und Eigenschaften.

³³ <https://arxiv.org/pdf/2511.21631> [Zugriff am 19.05.2026].

Tabelle 3: Vergleich der betrachteten lokalen LLM-Familien

Modell-familie	Anbieter	Modellgrößen (Beispiele)	Multi-modalität	Medizinischer Bezug
Llama	Meta	8B–405B; Llama 4 (MoE)	ab Llama 4 (Scout, Maverick)	Datenextraktion aus Radiologie-/Lymphombefunden
Deep-Seek	Deep-Seek-AI	1.5B–70B (Distill); V3 671B (MoE)	DeepSeek-VL (1.3B, 7B)	klinisches Reasoning, Diagnostik (MedQA); klinische Entscheidungsunterstützung
gpt-oss	OpenAI	20b, 120b (MoE)	nur als Erweiterung (MEDGPT-OSS)	klinische Entscheidungsunterstützung, Risikoanalyse; MEDGPT-OSS
Qwen	Alibaba	1.8B–14B; Qwen3-1.7B	Qwen-VL, Qwen2-VL (2B, 7/8B, 72B)	Nursing Examination, MedQwen-PE, Berichtgenerierung
Mistral	Mistral AI	3B–14B; Large 3 (MoE)	Mistral Large 3	Onkologie-/Biomarker-Extraktion, CancerLLM

Die Übersicht zeigt, dass alle fünf Modellfamilien offengewichtete und lokal ausführbare Varianten bereitstellen und damit das in Abschnitt [3.3](#) genannte Kriterium der lokalen Nutzbarkeit erfüllen. Unterschiede bestehen vor allem in den verfügbaren Modellgrößen, in der multimodalen Verarbeitung und im medizinischen Bezug. Über die Mixture-of-Experts-Architektur reduzieren Llama 4, DeepSeek-V3, gpt-oss und Mistral Large 3 die Zahl der pro Token aktiven Parameter und senken so den Ressourcenbedarf bei der lokalen Inferenz. Der Kontextumfang reicht von 32.768 Tokens bei Qwen3-1.7B bis zu 256K Tokens bei Mistral Large 3. Strukturierte JSON-Ausgaben stellen Llama, gpt-oss und Mistral bereit und sind für den ETL-Kontext dieser Arbeit besonders relevant. Da die Kausalitätsextraktion eine Teilaufgabe der Informationsextraktion darstellt und für die LLM-basierte Informationsextraktion strukturierte Ausgabeformate und ausreichender Kontextumfang als wichtige Kriterien gelten, sind diese Eigenschaften zugleich Voraussetzung für die kausale Extraktion [8,9]. Die in den Abschnitten [3.3.1](#) bis [3.3.5](#) herangezogenen modellspezifischen Belege beziehen sich überwiegend auf die Informationsextraktion; die grundsätzliche Eignung von LLMs für die kausale Extraktion aus medizinischen Texten ist jedoch belegt [11,77]. Hinsichtlich des medizinischen Bezugs liegen für alle fünf Modellfamilien konkrete Anwendungsfälle aus der medizinischen Informationsextraktion, Berichtverarbeitung und klinischen Entscheidungsunterstützung vor, teilweise ergänzt durch domänenspezifische Varianten

wie MEDGPT-OSS, MedQwen-PE und CancerLLM. Multimodale Verarbeitung stellen Llama, DeepSeek, Qwen und Mistral über dedizierte Vision-Language-Varianten bereit, während gpt-oss textbasiert ausgelegt ist und multimodale Verarbeitung nur über eine biomedizinische Erweiterung anbietet.

Mit dem Überblick über Werkzeuge zur multimodalen PDF-Verarbeitung und lokal ausführbare LLMs für die Informations- und Kausalitätsextraktion sind die fachlichen und technischen Grundlagen für die weitere Bearbeitung gelegt. Aufbauend auf den dargestellten Werkzeugeigenschaften vertieft das folgende Kapitel den Vergleich der KI-gestützten Werkzeuge anhand ihrer Ausgaben, bevor die Konzeption des ETL-Prozesses die geeigneten Werkzeuge und Modelle auswählt. Die betrachteten lokal ausführbaren Modelle erwarten Text beziehungsweise einzelne Bilder als Eingabe und setzen damit eine vorgelagerte Extraktion voraus. Einige cloudbasierte Anbietermodelle nehmen PDF-Dokumente dagegen unmittelbar entgegen. Dies erfordert jedoch die Übergabe des Dokuments an einen externen Dienst und scheidet damit für die lokale Verarbeitung aus (siehe Abschnitt [5.1](#)).

4 Vergleichende Analyse der KI-gestützten Werkzeuge

Aufbauend auf der Werkzeugübersicht in Abschnitt [3.2](#) vergleicht dieses Kapitel die fünf zuvor beschriebenen KI-gestützten Werkzeuge – LlamaParse, LiteParse, Docling, MinerU und Marker – an einer gemeinsamen Beispielseite und betrachtet, wie sich ihre Ausgaben für die weitere Verarbeitung eignen. Während Abschnitt [3.2](#) die Werkzeuge anhand ihrer dokumentierten Eigenschaften gegenüberstellt, untersucht dieses Kapitel ihr tatsächliches Ausgabeverhalten an denselben Dokumentinhalten. Als Grundlage dient dieselbe Seite eines medizinischen PDF-Dokuments, sodass sich die Ausgaben unmittelbar vergleichen lassen. Zunächst betrachtet Abschnitt [4.1](#) die JSON-Ausgaben, anschließend Abschnitt [4.2](#) die Markdown-Ausgaben. Abschnitt [4.3](#) vergleicht die Behandlung von Abbildungen.

4.1 Technischer Vergleich der JSON-Ausgaben

Für den ETL-Kontext dieser Arbeit sind die Ausgabeformate entscheidend. Alle betrachteten Werkzeuge stellen JSON-Ausgaben bereit oder dokumentieren entsprechende JSON-Exportmöglichkeiten. Damit liefern sie grundsätzlich maschinenlesbare Repräsentationen, unterscheiden sich jedoch darin, ob der Schwerpunkt stärker auf positionsbezogenen Textdaten, blockbasierten Dokumentstrukturen oder umfassenderen Dokumentmodellen liegt. Diese Unterschiede werden im Folgenden anhand der erzeugten JSON-Ausgaben der Beispielseite untersucht.

Die JSON-Struktur beschreibt, wie die extrahierten Dokumentinhalte innerhalb der jeweiligen Ausgabe organisiert werden. Dabei unterscheiden sich die betrachteten Werkzeuge vor allem darin, ob sie Inhalte seitenbezogen, elementorientiert, blockorientiert oder als umfassendes Dokumentmodell repräsentieren. Listing [1](#) zeigt die elementorientierte Struktur von LlamaParse, die Dokumentbestandteile als **items** mit Typen wie **header**, **heading**, **text**, **table**, **image** und **link** ablegt.

Listing 1: Ausschnitt der LlamaParse-JSON-Struktur

```

1.  {
2.    "pages": [
3.      {
4.        "page_number": 1,
5.        "items": [
6.          {
7.            "type": "header",
8.            "md": "Clinical Review & Education"
9.          },
10.         {
11.           "type": "heading",
12.           "md": "# Screening for Gestational Diabetes",
13.           "level": 1
14.         },
15.         {
16.           "type": "text",
17.           "md": "***IMPORTANCE** Gestational diabetes ..."
18.         },
19.         {
20.           "type": "table",
21.           "rows": [
22.             ["The USPSTF recommends ...", "B"],
23.             ["The USPSTF concludes ...", "I"]
24.           ]
25.         },
26.         {
27.           "type": "image",
28.           "md": "![arrow icon] (page_1_image_3_v2.jpg)"
29.         },
30.         {
31.           "type": "link",
32.           "md": "<chair@uspstf.net>",
33.           "url": "mailto:chair@uspstf.net"
34.         }
35.       ]
36.     }
37.   ]
38. }

```

Listing 2 zeigt den stärker seitenorientierten Aufbau von LiteParse, der neben dem fortlaufenden Seitentext vor allem **textItems** und **boundingBoxes** bereitstellt. Listing 3 zeigt das umfassendere Dokumentmodell von Docling, das **body**, **texts**, **tables**, **pictures** und **groups** über Referenzen miteinander verbindet. Listing 4 zeigt die blockorientierte Struktur von MinerU mit **para_blocks**, **lines**, **spans** und **discarded_blocks** sowie Blocktypen wie **title**, **text**, **list** und **table**. Listing 5 zeigt die abweichende Ausgabe von Marker, die vor allem aus einer Markdown-Repräsentation sowie ergänzenden **images** und **metadata** besteht.

Listing 2: Ausschnitt der LiteParse-JSON-Struktur

```

1.  {
2.    "pages": [
3.      {
4.        "page": 1,
5.        "width": 612,
6.        "height": 792,
7.        "text": "Clinical Review & Education ...",
8.        "textItems": [
9.          {
10.           "text": "Screening for Gestational Diabetes",
11.           "x": 72.3118,
12.           "y": 74.795,
13.           "width": 218.0348,
14.           "height": 14.9669,
15.           "confidence": 1
16.         }
17.       ],
18.       "boundingBoxes": [
19.         {
20.           "x1": 72.3118,
21.           "y1": 74.795,
22.           "x2": 290.3466,
23.           "y2": 89.7619
24.         }
25.       ]
26.     }
27.   ]
28. }

```

Listing 3: Ausschnitt der Docling-JSON-Struktur

```

1.  {
2.    "schema_name": "DoclingDocument",
3.    "body": {
4.      "children": [
5.        { "$ref": "#/texts/0" },
6.        { "$ref": "#/texts/1" },
7.        { "$ref": "#/tables/0" },
8.        { "$ref": "#/pictures/0" },
9.        { "$ref": "#/groups/0" }
10.     ],
11.    },
12.    "texts": [
13.      {
14.        "self_ref": "#/texts/1",
15.        "label": "section_header",
16.        "text": "JAMA | US Preventive Services Task Force |
17.        RECOMMENDATION STATEMENT"
18.      }
19.    ],
20.    "groups": [
21.      {
22.        "self_ref": "#/groups/0",
23.        "label": "key_value_area"
24.      }
25.    ]
26.  }

```

Listing 4: Ausschnitt der MinerU-JSON-Struktur

```

1. {
2.   "pdf_info": [
3.     {
4.       "para_blocks": [
5.         {
6.           "type": "title",
7.           "bbox": [69, 76, 291, 92],
8.           "lines": [
9.             { "spans": [ { "type": "text",
10.              "content": "Screening for Gestational Diabetes" } ] }
11.           ]
12.         },
13.         { "type": "text", "bbox": [70, 120, 177, 129] },
14.         { "type": "list", "bbox": [309, 233, 539, 451] },
15.         { "type": "table", "bbox": [71, 545, 420, 608] }
16.       ],
17.       "discarded_blocks": [
18.         {
19.           "type": "header",
20.           "content": "Clinical Review & Education"
21.         }
22.       ]
23.     }
24.   ]
25. }

```

Listing 5: Ausschnitt der Marker-JSON-Struktur

```

1. {
2.   "markdown": "### **JAMA | US Preventive Services Task Force |
3.   RECOMMENDATION STATEMENT**\n\n# Screening for Gestational
4.   Diabetes ...",
5.   "images": {},
6.   "metadata": {
7.     "table_of_contents": [],
8.     "page_stats": []
9.   }
10. }

```

Neben der allgemeinen JSON-Struktur ist für den ETL-Kontext relevant, wie die Inhalte der Beispielseite und insbesondere die Tabelle „**Summary of Recommendations**“ abgebildet werden. LlamaParse bildet Überschriften, Abstract-Blöcke, die rechte Informationsspalte, die Tabelle und den Abschnitt „**Importance**“ als eigene Elemente ab. Listing 6 zeigt, wie LlamaParse die Tabelle mit `rows`, `html` und `csv` ausgibt und dabei die Werte „**B**“ und „**I**“ enthält. LiteParse gibt die Seite dagegen als fortlaufenden Text sowie als einzelne Texteinheiten mit Positionsinformationen aus. Listing 7 zeigt, dass LiteParse die Inhalte nicht semantisch gruppiert und die Tabelle nicht als eigenes Tabellenobjekt modelliert, sondern nur als einzelne Texteinheiten ausgibt. Docling trennt Überschriften, Textblöcke, Tabellen, Bilder und Gruppen als Dokumentelemente.

Listing 6: LlamaParse, Tabellenabbildung

```

1. {
2.   "type": "table",
3.   "rows": [
4.     [
5.       "The USPSTF recommends screening for gestational diabetes
6.       in asymptomatic pregnant persons at 24 weeks of gestation
7.       or after.",
8.       "B"
9.     ],
10.    [
11.      "The USPSTF concludes that the current evidence is
12.      insufficient to assess the balance of benefits and harms of
13.      screening for gestational diabetes in asymptomatic pregnant
14.      persons before 24 weeks of gestation.",
15.      "I"
16.    ]
17.  ],
18.  "html": "<table>...</table>",
19.  "csv": "\"The USPSTF recommends ...\", \"B\"\\n\"The USPSTF
20.  concludes ...\", \"I\"\""
21. }

```

Listing 7: LiteParse, Texteinheiten statt Tabellenobjekt

```

1. {
2.   "textItems": [
3.     {
4.       "text": "Summary of Recommendations",
5.       "x": 72.3118,
6.       "y": 518.315
7.     },
8.     {
9.       "text": "The USPSTF recommends screening for gestational
10.      diabetes in asymptomatic pregnant persons",
11.       "x": 80.0178,
12.       "y": 551.2747
13.     },
14.     {
15.       "text": "B",
16.       "x": 399.3515,
17.       "y": 555.2815
18.     },
19.     {
20.       "text": "I",
21.       "x": 400.2657,
22.       "y": 580.6062
23.     }
24.   ]
25. }

```

Listing 8 zeigt, wie Docling die Tabelle mit einzelnen Tabellenzellen sowie Zeilen- und Spalteninformationen ausgibt und dabei die Werte „B“ und „I“ enthält. MinerU erfasst Hauptinhalt, rechte Informationsspalte, Tabelle sowie Kopf- und Fußbereiche blockweise. Listing 9 zeigt, wie MinerU die Tabelle als **table**-Block mit HTML-Repräsentation ausgibt und dabei die Werte „B“ und „I“ enthält. Marker gibt die Seite hauptsächlich als Markdown-Repräsentation wieder. Titel, Abstract-Blöcke, rechte Informationsspalte und Abschnitte sind im Text enthalten. Listing 10 zeigt, wie Marker die Tabelle als Markdown-Tabelle ausgibt, den Wert „I“ jedoch nicht sauber in der Tabellenzelle hält, sondern anschließend zusammen mit dem Hinweistext platziert.

Listing 8: Docling, Tabellenzellen

```

1. {
2.   "self_ref": "#/tables/0",
3.   "label": "table",
4.   "data": {
5.     "table_cells": [
6.       {
7.         "text": "The USPSTF recommends screening for gestational
8.           diabetes in asymptomatic pregnant persons at 24 weeks of
9.           gestation or after.",
10.        "start_row_offset_idx": 0,
11.        "end_row_offset_idx": 1,
12.        "start_col_offset_idx": 0,
13.        "end_col_offset_idx": 1
14.      },
15.      {
16.        "text": "B",
17.        "start_row_offset_idx": 0,
18.        "end_row_offset_idx": 1,
19.        "start_col_offset_idx": 1,
20.        "end_col_offset_idx": 2
21.      }
22.    ]
23.  }
24. }

```

Listing 9: MinerU, Tabellenblock mit HTML

```

1. {
2.   "type": "table",
3.   "bbox": [71, 545, 420, 608],
4.   "blocks": [
5.     {
6.       "type": "table_body",
7.       "spans": [
8.         {
9.           "type": "table",
10.          "html": "<table><tr><td>The USPSTF recommends ...
11.            </td><td>B</td> </tr><tr><td>The USPSTF concludes ...
12.            </td><td>I</td></tr></table>"
13.          }
14.        ]
15.      }
16.    ]
17.  }
18. }

```

Listing 10: Marker, Markdown-Tabelle

```

1. {
2.   "markdown": " ## Summary of Recommendations\n\n| The USPSTF
3.     recommends screening for gestational ... | B |\n|---|---|\n|
4.     The USPSTF concludes that the current evidence ... | |\n\nI
5.     See the Figure for a more detailed summary ... "
6. }

```

Beim Positionsbezug und den ETL-Auffälligkeiten zeigen sich vor allem Unterschiede in der Genauigkeit und Weiterverwendbarkeit der räumlichen Informationen. LlamaParse enthält Bounding Boxes, diese liegen jedoch in einer skalierten Seitendarstellung vor und müssen für einen direkten Abgleich mit dem PDF-Seitenmaß normalisiert werden. Zudem erfasst LlamaParse Symbol- und Icon-Elemente teilweise als Bilder, wodurch bei Bedarf eine Filterung erforderlich wird. Listing 11 zeigt, wie LiteParse den Positionsbezug über x- und y-Koordinaten, Breite, Höhe, Fontinformationen und

Confidence-Werte angibt. Für eine strukturierte Tabellen- oder Abschnitsverarbeitung wäre jedoch zusätzliche Nachverarbeitung erforderlich, da die Ausgabe primär aus einzelnen Texteinheiten besteht.

Listing 11: LiteParse, positionsbezogene Texteinheit

```

1. {
2.   "text": "Summary of Recommendations",
3.   "x": 72.3118,
4.   "y": 518.315,
5.   "width": 141.5492,
6.   "height": 10.9757,
7.   "fontName": "g_d0_f3",
8.   "fontSize": 10.9757,
9.   "confidence": 1
10. }
```

Listing 12 zeigt, wie Docling Positionsinformationen über **prov** und Bounding Boxes bereitstellt und den Koordinatenursprung explizit angibt. Auffällig sind einzelne Textzusammenziehungen wie „USPreventive“ oder „CMEQuiz“, die bei einer Weiterverarbeitung korrigiert werden müssten.

Listing 12: Docling, Bounding Box mit Koordinatenursprung

```

1. {
2.   "texts": [
3.     {
4.       "label": "section_header",
5.       "text": "JAMA | USPreventive Services Task Force |
6.       RECOMMENDATION STATEMENT",
7.       "prov": [{ "page_no": 1, "bbox": {
8.         "l": 72.3118, "t": 729.4670, "r": 377.8191, "b":
9.         721.8171,
10.        "coord_origin": "BOTTOMLEFT" } } ]
11.     },
12.     {
13.       "label": "text",
14.       "text": "CMEQuiz",
15.       "prov": [{ "page_no": 1, "bbox": {
16.         "l": 444.4894, "t": 571.5594, "r": 474.9435, "b":
17.         565.1915,
18.         "coord_origin": "BOTTOMLEFT" } } ]
19.     }
20.   ]
21. }
```

Listing 13 zeigt, wie MinerU Bounding Boxes und Seitenmaße angibt, die Beispielseite mit dem PDF-Seitenmaß 612 × 792, und Prozentwerte teilweise als **inline_equation** klassifiziert. Kopf- und Fußzeilen führt MinerU teilweise getrennt in **discarded_blocks**. Auffällig ist außerdem, dass MinerU einzelne Symbol- beziehungsweise Icon-Elemente fehlerhaft erkennen kann. Marker stellt in der untersuchten JSON-Ausgabe keine durchgängigen detaillierten Bounding Boxes für einzelne Text- und Tabellenbestandteile bereit. Die Ausgabe ist daher für eine Markdown-nahe Weiterverarbeitung geeignet, für zellgenaue Tabellenverarbeitung und positionsbezogene Auswertungen jedoch weniger detailliert.

Listing 13: MinerU, Seitenmaß und Inline-Equation

```

1. {
2.   "page_size": [612, 792],
3.   "lines": [
4.     {
5.       "spans": [
6.         {"type": "text", "content": "gestational diabetes in the
7.         US has been estimated at"}, {"type": "inline_equation", "content": "5 . 8 \%"},
8.         {"type": "text", "content": "to"}, {"type": "inline_equation", "content": "9 . 2 \%"}
9.       ]
10.    }
11.  ]
12. }
13. }

```

4.2 Vergleich der Markdown-Ausgaben

Über die JSON-Ausgaben hinaus stellen LlamaParse, Docling, MinerU und Marker auch eine Markdown-Ausgabe bereit, die Dokumentinhalte kompakter und ohne explizite Positions- und Strukturmetadaten abbildet. LiteParse erzeugt dagegen kein echtes Markdown, sondern nur eine Textausgabe, die das räumliche Layout über Leerraum nachbildet; sie wird hier zur Vollständigkeit mit betrachtet. Der Vergleich nutzt dieselbe Beispielseite wie der JSON-Vergleich in Abschnitt [4.1](#) und konzentriert sich auf den Zeichenumfang gegenüber JSON sowie auf die Abbildung der Tabelle „Summary of Recommendations“.

LlamaParse erzeugt eine strukturierte Markdown-Ausgabe mit Überschriftsebenen, Fettauszeichnung der Abschnittsmarker und einer in Markdown eingebetteten HTML-Tabelle. Listing [14](#) zeigt, wie LlamaParse die Tabelle als `<table>` ausgibt und die Werte „B“ und „I“ in getrennten Zellen erhält; Symbol- und Icon-Elemente erscheinen dabei als Textplatzhalter wie `[arrow]` und `[plus]`. Docling gibt eine native Markdown-Pipe-Tabelle aus.

Listing 14: LlamaParse, Markdown mit eingebetteter HTML-Tabelle (Ausschnitt)

```

1. ### Summary of Recommendations
2.
3. <table>
4.   <tbody>
5.     <tr>
6.       <td>The USPSTF recommends screening for ... </td>
7.       <td>B</td>
8.     </tr>
9.     <tr>
10.      <td>The USPSTF concludes that the current ... </td>
11.      <td>I</td>
12.    </tr>
13.  </tbody>
14. </table>
15.
16. [arrow] Editorial page 487
17. [plus] Multimedia

```

Listing 15 zeigt, wie Docling die beiden Empfehlungen als Tabellenzeilen mit den Werten „B“ und „I“ in getrennten Spalten abbildet; die bereits in der JSON-Ausgabe aufgetretenen Textzusammenziehungen wie „USPreventive“ und „CMEQuiz“ bleiben erhalten, und Abbildungen erscheinen lediglich als Platzhalter `<!--image-->`. MinerU bettet die Tabelle ebenfalls als HTML ein. Listing 16 zeigt, wie MinerU die Werte „B“ und „I“ in getrennten `<td>`-Zellen erhält und Prozentwerte sowie hochgestellte Verweise als LaTeX-Ausdrücke wie `\(5.8\%\)` und `\(^{1-3}\)` ausgibt.

Listing 15: Docling, native Markdown-Tabelle

```
1. ## Summary of Recommendations
2.
3. | The USPSTF recommends screening for gestational ... | B |
4. |-----|-----|
5. | The USPSTF concludes that the current evidence is ... | I |
```

Listing 16: MinerU, Markdown mit eingebetteter HTML-Tabelle

```
1. ## Summary of Recommendations
2.
3. <table><tr><td>The USPSTF recommends screening ... </td><td>B</td>
  </tr><tr><td>The USPSTF concludes that the current evidence is
  insufficient ... </td><td>I</td></tr></table>
4.
5. ... estimated at \ (5.8\%) to \ (9.2\%) , based on traditional ...
6. ... develops during pregnancy. \ (^{1-3}) Prevalence ...
```

Marker gibt eine Markdown-Pipe-Tabelle aus, hält den Wert „I“ jedoch nicht in der Tabellenzelle. Listing 17 zeigt, wie die zweite Wertspalte leer bleibt und das „I“ erst nach der Tabelle gemeinsam mit dem Hinweistext erscheint – derselbe Effekt wie in der JSON-Ausgabe (Listing 10); zusätzlich enthält die Ausgabe zahlreiche Tracking-URLs. LiteParse modelliert die Tabelle nicht als Tabelle. Listing 18 zeigt, dass die Werte „B“ und „I“ als rechts positionierter Fließtext erscheinen und die Ausgabe zusätzlich Seitenfuß- und Wasserzeicheninformationen enthält.

Listing 17: Marker, Markdown-Tabelle mit ausgelagertem „I“

```
1. ## Summary of Recommendations
2. | The USPSTF recommends screening ...<br>at 24 weeks of ... | B |
3. |-----|-----|
4. | The USPSTF concludes that the current evidence is ... | |
5.
6. I See the Figure for a more detailed summary of the ...
```

Listing 18: LiteParse, Textausgabe ohne Tabellenstruktur

```
1. Summary of Recommendations
2. The USPSTF recommends screening for ... B
3. at 24 weeks of gestation or after.
4.
5. The USPSTF concludes that the current ... I See the Figure ...
6. and harms of screening for ...
7. weeks of gestation. Summary of the ...
8.
9. See the summary of the recommendations figure.
```

Deutlich unterscheiden sich die Formate im Zeichenumfang. Tabelle 4 stellt die Zeichenanzahl der Markdown- und JSON-Ausgaben je Werkzeug gegenüber. Die Markdown-Ausgaben liegen durchgängig zwischen rund 3.850 und 5.800 Zeichen, während die JSON-Ausgaben von rund 17.000 bis über 938.000 Zeichen reichen. Der Größenunterschied ergibt sich vor allem aus den in Abschnitt 4.1 beschriebenen Positions- und Strukturmetadaten – Bounding Boxes, Koordinaten und Zellindizes –, die die Markdown-Ausgabe nicht enthält.

Tabelle 4: Zeichenumfang der Markdown- und JSON-Ausgaben je Werkzeug

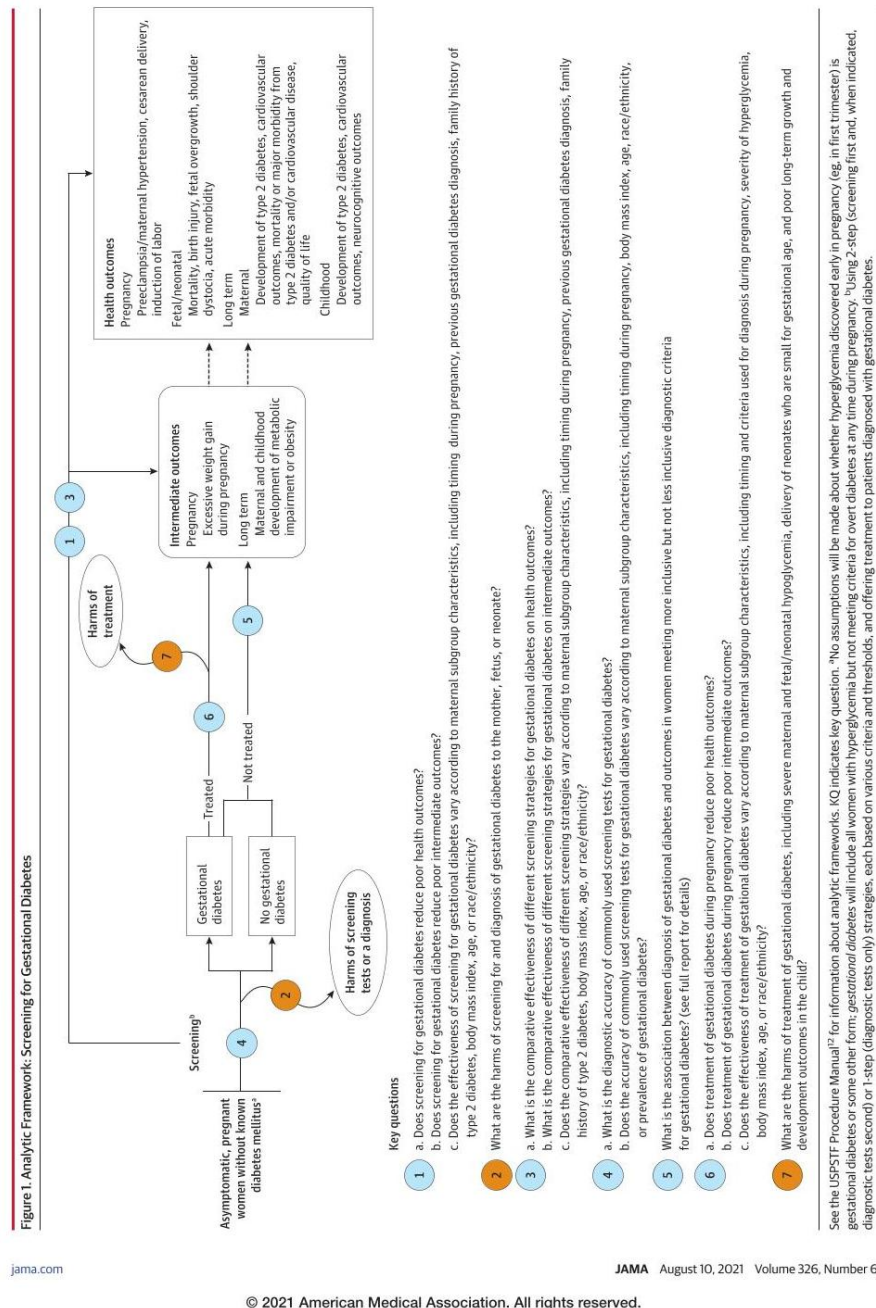
Werkzeug	Markdown (Zeichen)	JSON (Zeichen)	Verhältnis
LlamaParse	≈ 4.250	≈ 17.200	4x
LiteParse	≈ 5.400	≈ 625.300	115x
Docling	≈ 4.300	≈ 346.700	80x
MinerU	≈ 3.850	≈ 938.200	243x
Marker	≈ 5.800	≈ 88.200	15x

Insgesamt bilden LlamaParse, Docling und MinerU die Tabelle auch in Markdown vollständig mit beiden Bewertungswerten ab, während Marker denselben Zellfehler wie in der JSON-Ausgabe zeigt und LiteParse keine tabellarische Struktur bereitstellt. Gegenüber JSON ist die Markdown-Repräsentation deutlich kompakter, verzichtet dafür aber auf Positions- und Zellgrenzeninformationen. Ob sich die kompaktere Markdown- oder die metadatenreiche JSON-Repräsentation besser als Eingabe für die LLM-gestützte Extraktion eignet, betrachtet Kapitel 5.

4.3 Vergleich der Bild- und Abbildungsextraktion

Der letzte Vergleich betrifft die Behandlung von Abbildungen und unterscheidet dabei zwei Fähigkeiten: das Herauslösen einer Abbildung als eigene Bilddatei und die Überführung des Bildinhalts in Text oder Daten. Als Beispiel dient die Abbildung „Figure 1. Analytic Framework: Screening for Gestational Diabetes“, in Abbildung 18 dargestellt, aus dem zugehörigen JAMA-Artikel [75]. Da sich JSON- und Markdown-Ausgabe je Werkzeug deutlich unterscheiden, betrachtet der Vergleich beide Formate; die Listings zeigen jeweils die aussagekräftigere Repräsentation.

Abbildung 18: Beispielabbildung eines PDF-Dokuments



LlamaParse löst die Abbildung nicht als Datei heraus, sondern überführt ihre Struktur in einen Graphen. Listing 19 zeigt, wie LlamaParse das Flussdiagramm als Mermaid-Graph mit Knoten, Kanten und den nummerierten Beziehungen abbildet; JSON- und Markdown-Ausgabe sind dabei identisch. Einzelne Knotenbeschriftungen sind zusammengezogen, etwa „Gestationaldiabetes“, und die Zuordnung ist nicht durchgängig fehlerfrei. Docling registriert die Abbildung als Dokumentelement und kann sie bei entsprechender Konfiguration als Bilddatei herauslösen. In der Standardausgabe reduziert die Markdown-Repräsentation die Abbildung auf den Platzhalter

`<!-- image -->`; wird die Bilderzeugung aktiviert, löst Docling die Abbildung als Bilddatei heraus und referenziert sie im Markdown.

Listing 19: LlamaParse, Abbildung als Mermaid-Graph (JSON und Markdown identisch)

```
1. graph TD
2.   A([Asymptomatic, pregnantwomen without known...]) -- 1 --> B[Screening]
3.   B -- 4 --> C[Gestationaldiabetes]
4.   B -- 4 --> D[No gestationaldiabetes]
5.   C --> E[Treated]
6.   D --> F[Not treated]
7.   ...
8.   E -- 7 --> J((Harms oftreatment))
```

Listing 20 zeigt das `picture`-Element mit `bbox` im JSON sowie den zugehörigen Figurentext, den Docling in Einzelfragmente zerlegt. Den eigentlichen Bildinhalt überführt Docling dabei nicht in strukturierte Daten.

Listing 20: Docling, `picture`-Element im JSON und Markdown mit Platzhalter und Bildreferenz

```
1. // JSON (gekürzt)
2. "pictures": [{ "self_ref": "#/pictures/0", "label": "picture",
3. "prov": [{ "bbox": {"l":72.0,"t":729.7,"r":545.2,"b":65.6} }]}]
4. "texts": [ { "label":"text","text":"Asymptomatic, pregnant"},
5. {"label":"text","text":"women without known"}, ... ]
6. // Markdown
7. <!-- image -->
8. // Markdown (mit aktivierter Bilderzeugung)
9. ![Image](artifacts/image_000000_....png)
```

LiteParse erzeugt kein Bildobjekt und überführt die Abbildung stattdessen per OCR in Text. Listing 21 zeigt, dass die In-Bild-Beschriftungen als ungeordneter, am Layout orientierter Textblock erscheinen; JSON- und Markdown-Ausgabe stimmen hier im Wesentlichen überein. Marker löst die Abbildung als eigene Bilddatei heraus.

Listing 21: LiteParse, Abbildung als ungeordneter OCR-Text (JSON und Markdown gleich)

```
1. Figure 1. Analytic ... Harms of
2.                    treatment 1 3 Pregnancy
3.                    7 Intermediate ... induction of ...
4.                    Pregnancy Fetal/neonatal
5.                    diabetes ... 6 Excessive ... dystocia, ...
6.                    During ... Long term
7. Asymptomatic, ... Gestational ...
```

Listing 22 zeigt, wie Marker das JPEG (573 × 1792) im `images`-Objekt mit Größe und Farbmodus führt und über einen `Figure`-Blockzähler vermerkt, während die Markdown-Ausgabe nur mit `` darauf verweist; den Bildinhalt strukturiert Marker nicht. MinerU löst die Abbildung ebenfalls als Datei heraus und beschreibt sie am strukturiertesten. Listing 23 zeigt, wie MinerU im JSON einen typisierten `image`-Block mit `img_path`, `bbox`

und einer diesmal befüllten **image_caption** ausgibt, während das Feld **content** leer bleibt; die Markdown-Ausgabe zeigt die Bildunterschrift und den Bildverweis.

Listing 22: Marker, herausgelöste Bilddatei im JSON-images-Objekt

```
1. // JSON (gekürzt)
2. "images": {
3.   "_page_0_Figure_2.jpeg": {
4.     "_mode": "RGB",
5.     "_size": [573, 1792]
6.   }
7. }
8. "block_counts": [ ... ["Figure", 1] ]
9.
10. // Markdown
11. 
```

Listing 23: MinerU, typisierter image-Block mit Pfad, bbox und Caption

```
1. {
2.   "type": "image",
3.   "img_path": "images/c0799...cf.jpg",
4.   "image_caption": ["Figure 1. Analytic Framework: Screening for
5.     Gestational Diabetes", ...],
6.   "image_footnote": [],
7.   "content": "",
8.   "bbox": [148, 78, 460, 917]
9. }
```

Über die Werkzeuge hinweg trennen sich die beiden Fähigkeiten deutlich. Das Herauslösen als Datei gelingt Marker, MinerU und Docling (bei entsprechender Konfiguration); LiteParse löst die Abbildung nicht heraus, und LlamaParse ersetzt sie durch eine Graph-Repräsentation. Den Bildinhalt überführt allein LlamaParse in strukturierte Daten, und dies mit Fehlern. Tabelle 5 fasst die Ergebnisse zusammen. Quer zu allen Werkzeugen zeigt sich, dass die JSON-Ausgabe für die Abbildungsbehandlung durchgängig aussagekräftiger ist als die Markdown-Ausgabe, da sie Position, Typ und Metadaten der Abbildung enthält, die die Markdown-Ausgabe auf einen Verweis oder Platzhalter reduziert. Eine zuverlässige Überführung des eigentlichen Bildinhalts in strukturierte Daten leistet keines der lokal einsetzbaren Werkzeuge.

Tabelle 5: Behandlung der Abbildung je Werkzeug

Werkzeug	Als Datei herausgelöst	Position (bbox)	Inhalt überführt
LlamaParse	nein	ja	teilweise (Mermaid-Graph, fehlerhaft)
LiteParse	nein	ja	nein
Docling	ja	ja	nein
MinerU	ja	ja	nein
Marker	ja	ja	nein

Damit liegt für alle fünf Werkzeuge ein Bild ihres tatsächlichen Ausgabeverhaltens vor: der Struktur ihrer JSON-Ausgaben, der Treue ihrer Markdown-Repräsentation und ihrer Behandlung von Abbildungen. Diese Beobachtungen bilden zusammen mit den Anforderungen in Abschnitt [5.1](#) die Grundlage für die Auswahl des Extraktionswerkzeugs in Abschnitt [5.3](#).

5 Konzeption des ETL-Prozesses

Dieses Kapitel entwickelt das Konzept eines lokal ausführbaren ETL-Prozesses für die Verarbeitung multimodaler medizinischer PDF-Dokumente. Aufbauend auf den in Kapitel [3](#) und [4](#) verglichenen Werkzeugen und Modellen überführt das Konzept die zuvor gewonnenen Erkenntnisse in eine konkrete Architektur und beschreibt die einzelnen Verarbeitungsphasen. Zunächst bestimmt Abschnitt [5.1](#) die Systemanforderungen. Anschließend stellt Abschnitt [5.2](#) die Gesamtarchitektur des ETL-Prozesses dar. Abschnitt [5.3](#) begründet die Auswahl geeigneter Werkzeuge und Modelle. Darauf aufbauend beschreiben die Abschnitte [5.4](#) bis [5.6](#) die drei Phasen des Prozesses: Abschnitt [5.4](#) erläutert die Extraktionsphase, in der ein Dokumentenwerkzeug und ein Vision-Language-Modell die multimodalen PDF-Dokumente verarbeiten. Abschnitt [5.5](#) behandelt die Transformationsphase, in der ein lokales LLM Informationen und kausale Zusammenhänge extrahiert. Abschnitt [5.6](#) beschreibt die Ladephase, die die Ergebnisse in ein strukturiertes Datenmodell überführt.

5.1 Systemanforderungen

Dieser Abschnitt bestimmt die Anforderungen, die der ETL-Prozess erfüllen muss, und grenzt zugleich den Umfang dieser Arbeit ab. Die Anforderungen leiten sich aus dem Ziel ab, multimodale medizinische PDF-Dokumente lokal in ein strukturiertes Datenmodell zu überführen, und bilden die Grundlage für die Gesamtarchitektur in Abschnitt [5.2](#) sowie für die Auswahl der Werkzeuge und Modelle in Abschnitt [5.3](#).

In funktionaler Hinsicht liest der Prozess digital erzeugte multimodale medizinische PDF-Dokumente ein und überführt deren Inhalte, wie Texte, Tabellen und Abbildungen, in eine einheitliche, maschinenlesbare Repräsentation. Aus dieser Repräsentation extrahiert ein lokal ausgeführtes LLM anschließend Informationen und kausale Zusammenhänge. Die Extraktionsergebnisse überführt der Prozess abschließend in ein strukturiertes Datenmodell. Diese drei Schritte entsprechen den in den Abschnitten [5.4](#) bis [5.6](#) beschriebenen Phasen der Extraktion, Transformation und des Ladens.

In nichtfunktionaler Hinsicht arbeiten alle Komponenten lokal: Sowohl die Verarbeitungswerkzeuge als auch die Modelle laufen ohne Cloud- oder externe Dienste auf lokal verfügbarer Rechenleistung, sodass die medizinischen Dokumente die lokale Verarbeitungsumgebung nicht verlassen. Der Prozess gliedert sich dabei in die drei Phasen Extraktion, Transformation und Laden, die sich einzeln austauschen lassen,

sodass sich Werkzeuge oder Modelle ersetzen lassen, ohne die Gesamtarchitektur zu verändern.

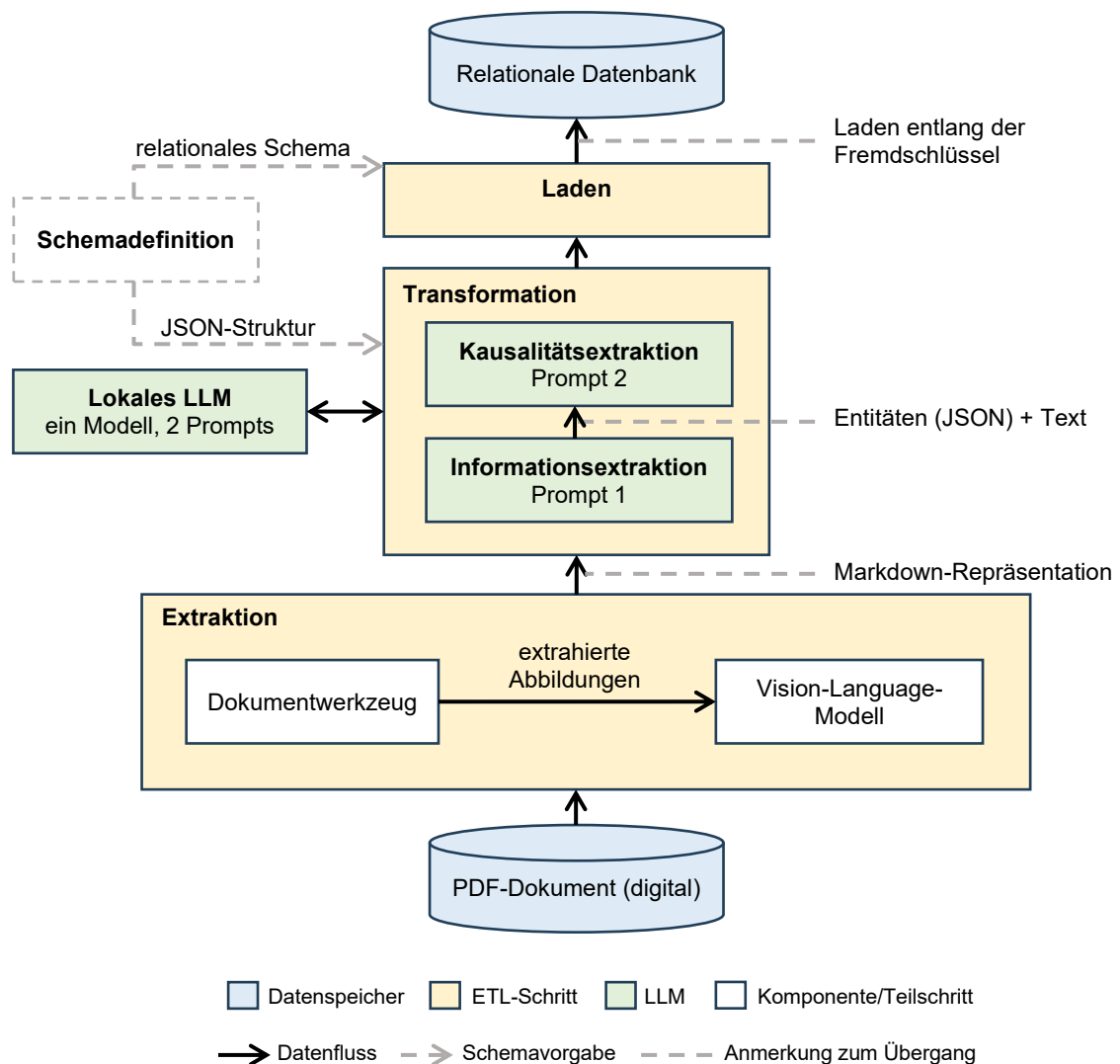
In qualitativer Hinsicht stellt der Prozess Anforderungen an die Güte seiner Ergebnisse. Zunächst soll die Extraktionsphase den fachlich relevanten Dokumentinhalt, insbesondere Text und Tabellen, möglichst vollständig und originalgetreu in die maschinenlesbare Repräsentation überführen, da die Transformationsphase unmittelbar auf dieser Repräsentation aufsetzt. Darüber hinaus soll das LLM die enthaltenen Informationen und kausalen Zusammenhänge korrekt und möglichst vollständig erkennen, da die Güte dieser Extraktion die Qualität des gesamten Prozessergebnisses bestimmt. Schließlich sollen die Ergebnisse nachvollziehbar bleiben, sodass sich die erkannten Informationen, etwa anhand der persistierten Zwischenausgabe der Informationsextraktion (siehe Abschnitt [5.2](#)), mit den Dokumentinhalten abgleichen lassen.

Den Umfang dieser Arbeit bildet der vollständige Weg vom eingelesenen PDF-Dokument bis zum strukturierten Datenmodell. Verarbeitet werden die textuellen, tabellarischen und bildlichen Inhalte der Dokumente. Dieses Datenmodell bildet die definierte Ausgabe und zugleich die Schnittstelle des Prozesses: Es liegt in einer strukturierten, maschinenlesbaren Form vor, die sich durch nachgelagerte Systeme weiterverwenden lässt. Die Verarbeitung beschränkt sich auf digital erzeugte PDF-Dokumente. Für Entwicklung und Erprobung dienen öffentlich verfügbare wissenschaftliche Publikationen als stellvertretende Beispieldaten, da echte medizinische Dokumente mit personenbezogenen Patientendaten für diese Arbeit nicht zur Verfügung stehen. Sie repräsentieren denselben Dokumenttyp, enthalten jedoch keine schützenswerten Daten.

5.2 Gesamtarchitektur des ETL-Prozesses

Der ETL-Prozess setzt die in Kapitel [2](#) beschriebene dreiphasige Struktur aus Extraktion, Transformation und Laden um und spezialisiert sie für die Verarbeitung multimodaler medizinischer PDF-Dokumente mit einem lokalen LLM. Abbildung [19](#) zeigt die Gesamtarchitektur als vertikalen Datenfluss von der PDF-Quelle bis zur relationalen Zieldatenbank. Die Architektur greift dabei den spezialisierten ETL-Prozess für unstrukturierte Daten aus Abschnitt [2.1](#) auf und ersetzt dessen NLP-Vorverarbeitung durch eine LLM-gestützte Transformation.

Abbildung 19: Gesamtarchitektur des lokalen ETL-Prozesses



In der Extraktionsphase überführt ein Dokumentwerkzeug das digital erzeugte PDF-Dokument in eine einheitliche, maschinenlesbare Markdown-Repräsentation. Markdown dient hier als Eingabeformat für die nachfolgenden LLM-Schritte, da es die Dokumentinhalte aus Text und Tabellen kompakt und mit geringem Token-Umfang abbildet und dabei auf Positions- und Strukturmetadaten verzichtet, die ein textbasiertes LLM nicht benötigt (siehe Abschnitt [4.2](#)). Enthält ein Dokument Abbildungen, arbeiten die beiden Komponenten sequenziell zusammen: Das Dokumentwerkzeug extrahiert die Abbildungen als Bilddateien und verweist an deren Position in der Markdown-Repräsentation auf die jeweilige Datei. Anschließend erzeugt ein lokales Vision-Language-Modell zu jeder Bilddatei eine textuelle Beschreibung, die die Bildreferenz an der ursprünglichen Position ersetzt. Auf diese Weise entsteht eine rein textuelle Repräsentation, in der auch die Bildinhalte den nachfolgenden, textbasierten Schritten zur Verfügung stehen. Die Auswahl des konkreten Werkzeugs begründet Abschnitt [5.3](#), die Ausgestaltung der Phase beschreibt Abschnitt [5.4](#).

Die Transformationsphase gliedert sich in zwei aufeinander aufbauende Schritte, die ein einziges lokal ausgeführtes LLM mit zwei unterschiedlichen Prompts bearbeitet. Der erste Schritt extrahiert aus der Markdown-Repräsentation Informationen in Form von Entitäten und gibt sie als JSON aus. Der zweite Schritt extrahiert daraus kausale Zusammenhänge. Diese Reihenfolge ergibt sich aus der Aufgabe selbst, da kausale Beziehungen bereits erkannte Entitäten verbinden und der zweite Schritt deren Ergebnisse benötigt. Der erste Schritt reicht seine Ausgabe daher zusammen mit dem Markdown-Originaltext an den zweiten Schritt weiter, damit dieser die Beziehungen im ursprünglichen Kontext erkennt. Die persistierte JSON-Ausgabe des ersten Schritts bildet zugleich einen prüfbaren Zwischenstand, an dem sich die erkannten Entitäten vor der Kausalitätsextraktion kontrollieren lassen. Die lokale Ausführung des Modells entspricht der in Abschnitt [5.1](#) geforderten lokalen Verarbeitung, sodass die medizinischen Inhalte die Verarbeitungsumgebung nicht verlassen. Die Prompts und die weitere Ausgestaltung beschreibt Abschnitt [5.5](#).

Die Ladephase überführt die Extraktionsergebnisse in das relationale Zieldatenmodell. Sie verarbeitet die JSON-Ausgaben der beiden Transformationsschritte und schreibt sie in mehreren Stufen entlang der Fremdschlüsselstruktur in die Datenbank: zunächst das Dokument, dann die Entitäten und zuletzt die Kausalbeziehungen. Die Trennung von Transformation und Laden hält die Phasen begrifflich rein, da die Transformation die Daten erzeugt und die Ladephase sie persistiert. Das relationale Datenmodell bildet die definierte Ausgabe und Schnittstelle des Prozesses (siehe Abschnitt [5.1](#)); seine Struktur beschreibt Abschnitt [5.6](#).

Über die drei Phasen hinweg nutzt der Prozess für jeden Übergang das Format, das der jeweiligen Aufgabe entspricht: Markdown als kompakte Eingabe in die LLM-Schritte, JSON als strukturierte und persistierbare Ausgabe dieser Schritte und das relationale Schema als abfragbares Ziel. Transformations- und Ladephase greifen dabei auf eine gemeinsame, zentrale Schemadefinition zurück, aus der sich sowohl die im Prompt vorgegebene JSON-Struktur als auch das relationale Datenbankschema ableiten, sodass beide stets übereinstimmen (siehe Abbildung [19](#)). Der Prozess verarbeitet die Dokumente sequenziell, sodass jedes PDF-Dokument die Phasen vollständig durchläuft, bevor das nächste folgt. Die klare Trennung der Phasen erlaubt es zudem, einzelne Werkzeuge oder Modelle auszutauschen, wie in Abschnitt [5.1](#) gefordert.

5.3 Auswahl geeigneter Werkzeuge und Modelle

Dieser Abschnitt wählt die beiden zentralen Komponenten des ETL-Prozesses aus: das Werkzeug für die Extraktionsphase und das Sprachmodell für die Transformationsphase. Beide Entscheidungen leiten sich aus den Anforderungen in Abschnitt 5.1 sowie aus den Vergleichen in Kapitel 3 und 4 ab. Die Ladephase erfordert demgegenüber keine vergleichende Auswahl aus mehreren Kandidaten, da sie die Ergebnisse lediglich in ein relationales Datenbanksystem schreibt. Aus Konsistenz mit der Anforderung der lokalen Ausführbarkeit genügt hierfür ein lokal betreibbares relationales System; die Gestaltung des Datenmodells beschreibt Abschnitt 5.6.

5.3.1 Werkzeug für die Extraktionsphase

Für die Extraktionsphase ist die lokale Ausführbarkeit das ausschlaggebende Kriterium. Da der Prozess für die Verarbeitung medizinischer Dokumente mit personenbezogenen Patientendaten ausgelegt ist, muss er ohne Cloud- oder externe Dienste arbeiten, damit diese Daten die lokale Verarbeitungsumgebung nicht verlassen. Damit scheidet LlamaParse (siehe Abschnitt 3.2.3.1) als cloudgebundenes Werkzeug aus. Die verbleibenden lokal ausführbaren Werkzeuge bewertet die Auswahl anhand der originalgetreuen Wiedergabe von Tabellen, der Qualität der Markdown-Ausgabe als Eingabeformat für die Transformationsphase, der Strukturqualität des zugrunde liegenden Dokumentmodells und der Artefaktarmut des erzeugten Texts.

LiteParse (siehe Abschnitt 3.2.3.2) erfüllt diese Kriterien am wenigsten, da es Inhalte nicht semantisch gruppiert, Tabellen nicht als eigene Tabellenobjekte modelliert und kein echtes Markdown erzeugt. Marker (siehe Abschnitt 3.2.3.5) liefert eine brauchbare Markdown-Ausgabe, hält jedoch einzelne Tabellenwerte nicht zuverlässig in der zugehörigen Zelle, sodass ein Bewertungswert aus der Tabelle herausfällt; bei tabellarisch aufgebauten medizinischen Befunden ist dies ein relevantes Risiko. MinerU (siehe Abschnitt 3.2.3.4) bildet Tabellen vollständig ab und liefert ein strukturiertes Dokumentmodell, gibt jedoch Fließtextzeichen wie Prozentwerte und hochgestellte Verweise als LaTeX-Ausdrücke aus, sodass die Ausgabe vor der Weiterverarbeitung eine zusätzliche Normalisierung erfordert. Docling (siehe Abschnitt 3.2.3.3) bildet Tabellen sowohl in der JSON- als auch in der Markdown-Ausgabe originalgetreu ab und behält die Bewertungswerte in getrennten Zellen, erzeugt saubere native Markdown-Tabellen mit geringem Token-Umfang (siehe Tabelle 4 in Abschnitt 4.2) und verfügt über das umfassendste Dokumentmodell der betrachteten Werkzeuge.

Der Prozess setzt daher in der Extraktionsphase Docling ein. Maßgeblich sind die für medizinische Befunde entscheidende Tabellentreue und die kompakte, saubere Markdown-Ausgabe, die unmittelbar als Eingabe für die Transformationsphase dient. MinerU bildet aufgrund seiner vergleichbaren Tabellen- und Strukturqualität die nächstgelegene Alternative.

Neben dem Dokumentwerkzeug für Text und Tabellen sieht die Extraktionsphase eine zweite Komponente für die Verarbeitung von Abbildungen vor. Da das gewählte Transformationsmodell rein textbasiert arbeitet (siehe Abschnitt [5.3.2](#)), überführt ein lokales Vision-Language-Modell den Inhalt einer Abbildung in eine textuelle Beschreibung, die anschließend gemeinsam mit den übrigen Inhalten in die Markdown-Repräsentation eingeht. Als lokal ausführbare Vision-Language-Modelle kommen die in Abschnitt [3.3](#) vorgestellten multimodalen Varianten in Betracht, etwa Qwen2-VL, Qwen3-VL oder DeepSeek-VL. Entsprechend der modularen Struktur des Prozesses (siehe Abschnitt [5.1](#)) bleibt diese Komponente austauschbar, sodass sich das eingesetzte Vision-Language-Modell ohne Anpassung der übrigen Phasen ersetzen lässt.

5.3.2 Modell für die Transformationsphase

Für die Transformationsphase ist die Ausführbarkeit unter begrenzten lokalen Ressourcen das ausschlaggebende Kriterium, da das Modell ohne Cloud-Anbindung auf lokaler Hardware arbeiten soll. Damit scheiden die ressourcenintensiven Varianten der betrachteten Modellfamilien aus, etwa Llama-3.3-70B, DeepSeek-V3 oder Mistral Large 3, deren Parameterzahl den lokalen Betrieb auf begrenzter Hardware verhindert. Die verbleibenden, lokal ausführbaren Modelle bewertet die Auswahl anhand der Eignung für strukturierte JSON-Ausgaben, eines ausreichenden Kontextumfangs für längere medizinische Dokumente und eines effizienten Ressourcenbedarfs bei der Inferenz.

Unter diesen Kriterien eignet sich gpt-oss-20b besonders. Das Modell stellt strukturierte Ausgaben bereit, die der ETL-Prozess für die JSON-basierte Weitergabe zwischen den Transformationsschritten und für die Ladephase benötigt, und unterstützt ein Kontextfenster von 131.072 Tokens, das auch längere Dokumentrepräsentationen aufnimmt. Durch die Mixture-of-Experts-Architektur aktiviert das Modell pro Token nur etwa 3,61B seiner 20,9B Parameter, wodurch sich der Ressourcenbedarf bei der Inferenz deutlich reduziert und der lokale Betrieb auf begrenzter Hardware möglich wird. Als rein textbasiertes Modell entspricht gpt-oss-20b zudem der textbasierten Auslegung der Transformationsphase, die auf der textuellen und tabellarischen Markdown-Repräsentation aufsetzt.

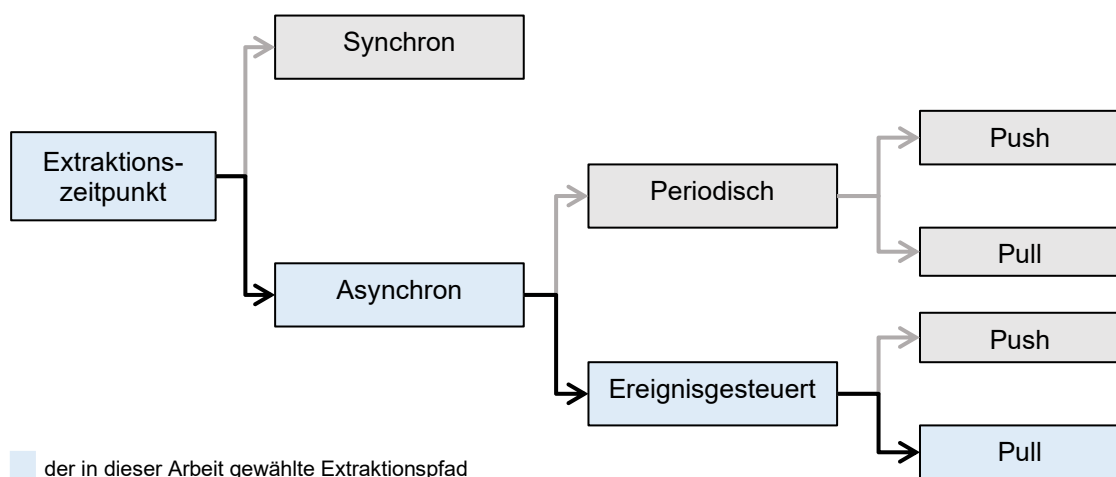
Der Prozess setzt daher in der Transformationsphase gpt-oss-20b ein. Innerhalb der lokal ausführbaren Modelle kommen grundsätzlich auch kompaktere Varianten anderer Familien in Betracht, etwa Qwen3-1.7B oder kleinere Mistral-3- und DeepSeek-R1-Distill-Modelle; diese senken den Ressourcenbedarf weiter, verfügen jedoch über kleinere Kontextfenster oder sind nicht durchgängig auf strukturierte Ausgaben ausgelegt.

5.4 Extraktionsphase: Verarbeitung multimodaler PDF-Dokumente

Die Extraktionsphase bildet den ersten Schritt des ETL-Prozesses und überführt ein digital erzeugtes PDF-Dokument in eine einheitliche, maschinenlesbare Markdown-Repräsentation. Sie nutzt dafür Docling sowie ein lokales Vision-Language-Modell für die Verarbeitung von Abbildungen und stellt die Grundlage für die nachfolgende Transformationsphase bereit. Der Abschnitt beschreibt zunächst, wie ein Dokument in den Prozess gelangt, anschließend die Konvertierung selbst und schließlich die Ausgabe der Phase.

Der Dateneingang lässt sich über den Extraktionszeitpunkt einordnen. Dieser unterscheidet zwischen synchroner Extraktion, bei der eine kooperative Quelle jede Änderung unmittelbar weitergibt, und asynchroner Extraktion, die entweder periodisch oder ereignisgesteuert erfolgt. Wer die Übertragung auslöst, unterscheidet sich nach dem Push- und dem Pull-Prinzip: Beim Push-Prinzip benachrichtigt die Quelle das Zielsystem aktiv über Änderungen, während beim Pull-Prinzip das Zielsystem die Quelle von sich aus abfragt [20]. Abbildung 20 ordnet den Dateneingang in diese Systematik ein und hebt den in dieser Arbeit gewählten Pfad hervor.

Abbildung 20: Einordnung des Dateneingangs nach dem Extraktionszeitpunkt (in Anlehnung an [20])

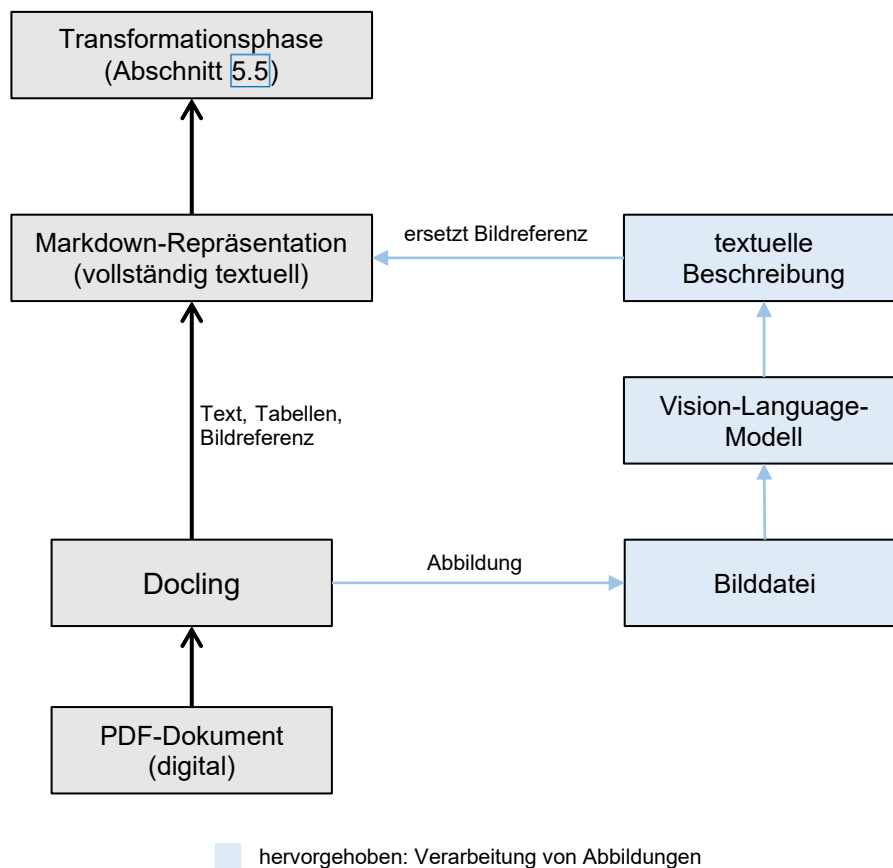


Der Provider der PDF-Dokumente propagiert keine Änderungen und tritt damit nicht als kooperative Quelle auf, sodass die Extraktion asynchron erfolgt. Da kein festes Intervall vorliegt, sondern ein manueller Start die Verarbeitung auslöst, ist sie ereignisgesteuert. Bei diesem Ereignis liest der Prozess das Dokument aktiv ein; die Quelle sendet es nicht von sich aus. Der Dateneingang folgt damit einer asynchronen, ereignisgesteuerten Extraktion nach dem Pull-Prinzip. Die Unterscheidung der Datenart in Snapshot und Log betrifft demgegenüber die Erkennung von Änderungen bei wiederholter Extraktion einer veränderlichen Quelle und entfällt hier, da der Prozess jedes PDF-Dokument als in sich abgeschlossenes Einzeldokument behandelt und nicht als veränderlichen Datenbestand fortschreibt.

Die modulare Struktur des Prozesses lässt eine spätere Erweiterung um das Push-Prinzip zu. Eine kooperative Quelle, die neue Dokumente periodisch oder nach bestimmten Ereignissen von sich aus liefert, ließe sich der Extraktionsphase vorschalten, ohne die nachfolgenden Phasen zu verändern.

Die Konvertierung überführt das eingeleseene Dokument mit Docling in die Markdown-Repräsentation. Da der Prozess ausschließlich digital erzeugte PDF-Dokumente verarbeitet, liegt eine eingebettete Textebene vor, aus der Docling den Inhalt direkt gewinnt, ohne dass eine vorgelagerte Texterkennung notwendig ist. Docling überführt dabei Fließtext, Überschriften und Tabellen in eine einheitliche Markdown-Struktur und bildet die Tabellen originalgetreu ab (siehe Abschnitt [4.2](#)). Enthält das Dokument Abbildungen, extrahiert Docling diese zusätzlich als einzelne Bilddateien und hinterlegt an ihrer jeweiligen Position eine Referenz auf die zugehörige Datei in der Markdown-Repräsentation. Für jede dieser Referenzen erzeugt anschließend ein lokales Vision-Language-Modell aus der Bilddatei eine textuelle Beschreibung, die die Referenz an der ursprünglichen Position ersetzt. Dadurch liegen die Bildinhalte als Text an der passenden Stelle im Dokument vor und stehen den nachfolgenden, textbasierten Schritten zur Verfügung. Die so erzeugte Markdown-Repräsentation bildet die Ausgabe der Extraktionsphase und dient als definierte Eingabe für die Transformationsphase (siehe Abschnitt [5.5](#)). [Abbildung 21](#) veranschaulicht den Ablauf der Extraktionsphase und hebt die Verarbeitung von Abbildungen hervor.

Abbildung 21: Ablauf der Extraktionsphase mit Text-, Tabellen- und Bildverarbeitung



5.5 Transformationsphase: LLM-basierte Informations- und Kausalitätsextraktion

Die Transformationsphase überführt die Markdown-Repräsentation der Extraktionsphase in strukturierte Informationen und kausale Zusammenhänge. Sie gliedert sich in zwei aufeinander aufbauende Schritte, die ein einziges lokal ausgeführtes Modell (siehe Abschnitt 5.3.2) mit zwei unterschiedlichen Prompts bearbeitet: Der erste Schritt extrahiert die Entitäten, der zweite die kausalen Beziehungen zwischen ihnen. Diese Trennung folgt der in Abschnitt 2.3 beschriebenen Unterscheidung zwischen Informationsextraktion und Kausalitätsextraktion als spezieller Form der Relationsextraktion. Die Aufteilung in zwei Schritte gegenüber einer gemeinsamen Extraktion in einem einzigen Prompt hat mehrere Gründe: Jeder Prompt bearbeitet eine klar umrissene Teilaufgabe, wodurch sich seine Formulierung gezielter auf diese Aufgabe ausrichten lässt. Zudem setzt die Kausalitätsextraktion auf bereits erkannten Entitäten auf und kann deren Bezeichner unmittelbar referenzieren, statt Entitäten und Beziehungen gleichzeitig bestimmen zu müssen. Schließlich entsteht zwischen den

Schritten ein prüfbarer Zwischenstand, der bei einer kombinierten Extraktion nicht vorläge.

Der erste Schritt führt die Informationsextraktion durch. Aus der Markdown-Repräsentation erkennt das Modell die relevanten medizinischen Entitäten und gibt sie als JSON aus. Jede Entität erhält einen eindeutigen Bezeichner, den erkannten Textausschnitt und einen Typ, etwa Krankheit, Symptom oder Medikament. Die Typvergabe erfolgt geführt offen: Der Prompt gibt gängige Typen als Orientierung vor, erlaubt dem Modell jedoch, bei Bedarf einen passenden eigenen Typ zu vergeben. Dadurch bleibt die Extraktion auch bei Entitäten anwendbar, die sich keinem der vorgegebenen Typen zuordnen lassen. Listing 24 zeigt den entsprechenden Beispiel-Prompt.

Listing 24: Beispielhafter Prompt für die Informationsextraktion

```
1. You are an information extraction system for medical documents.
2. Extract all relevant medical entities from the text below.
3.
4. For each entity, provide:
5. - id: a unique identifier (e1, e2, ...)
6. - text: the exact mention as it appears in the text
7. - type: the entity category. Use one of the following where it
   fits: Disease, Symptom, Medication, Treatment, RiskFactor,
   Finding. If none fits, assign a suitable type of your own.
8.
9. Return only valid JSON in this format:
10. {
11.   "entities": [
12.     { "id": "e1", "text": "...", "type": "..." }
13.   ]
14. }
15.
16. Text:
17. <<MARKDOWN>>
```

Der zweite Schritt führt die Kausalitätsextraktion durch. Er erhält die im ersten Schritt erkannten Entitäten zusammen mit dem Markdown-Originaltext und erkennt gerichtete Ursache-Wirkungs-Beziehungen zwischen diesen Entitäten. Die Übergabe des Originaltexts ist notwendig, da kausale Beziehungen im ursprünglichen Kontext stehen und sich nicht allein aus der Entitätsliste erschließen lassen. Jede Beziehung verweist über die Bezeichner auf eine Ursache- und eine Wirkungsentität und hält den kausalen Marker fest, also den sprachlichen Ausdruck, der die Beziehung anzeigt; bei impliziter Kausalität ohne eindeutigen Marker bleibt dieses Feld leer. Diese Unterscheidung greift die in Abschnitt 2.3.2 beschriebene explizite und implizite Kausalität auf. Nicht jede Entität ist Teil einer kausalen Beziehung; Entitäten ohne kausalen Bezug bleiben erhalten, ohne in einer Relation aufzutreten. Listing 25 zeigt den entsprechenden Beispiel-Prompt.

Listing 25: Beispielhafter Prompt für die Kausalitätsextraktion

```

1. You are a causal relation extraction system for medical
   documents.
2. You receive a list of previously extracted entities and the
   original text.
3. Identify directed cause-effect relations between these entities
   as stated or implied in the text.
4.
5. For each causal relation, provide:
6. - id: a unique identifier (r1, r2, ...)
7. - cause: the id of the cause entity
8. - effect: the id of the effect entity
9. - marker: the textual cue expressing causality (e.g. "causes",
   "leads to"), or null if the relation is implicit
10.
11. Use only entity ids from the provided list. Not every entity must
   take part in a relation. Return only valid JSON in this format:
12. {
13.   "relations": [
14.     { "id": "r1", "cause": "e1", "effect": "e2", "marker": "..."}
15.   ]
16. }
17.
18. Entities:
19. <<ENTITIES_JSON>>
20.
21. Text:
22. <<MARKDOWN>>

```

Die Prompts, in Listing [24](#) und Listing [25](#) dargestellt, zeigen die konzeptionelle Gestaltung der beiden Extraktionsschritte beispielhaft. Ihre konkrete Formulierung ist nicht festgelegt, sondern wird in Kapitel [6](#) iterativ erprobt und im Hinblick auf die Extraktionsqualität optimiert. Die in den Prompts geforderte JSON-Struktur ist dabei nicht fest im Prompt hinterlegt, sondern leitet sich aus einer zentralen Schemadefinition ab. Dieselbe Definition beschreibt das Zieldatenmodell und dient sowohl der Erzeugung des relationalen Datenbankschemas als auch der im Prompt vorgegebenen Ausgabestruktur. Dadurch stimmen die vom Modell erzeugte JSON-Ausgabe und das Zielschema der Ladephase stets überein, ohne dass beide getrennt gepflegt werden müssen (siehe Abschnitt [5.6](#)).

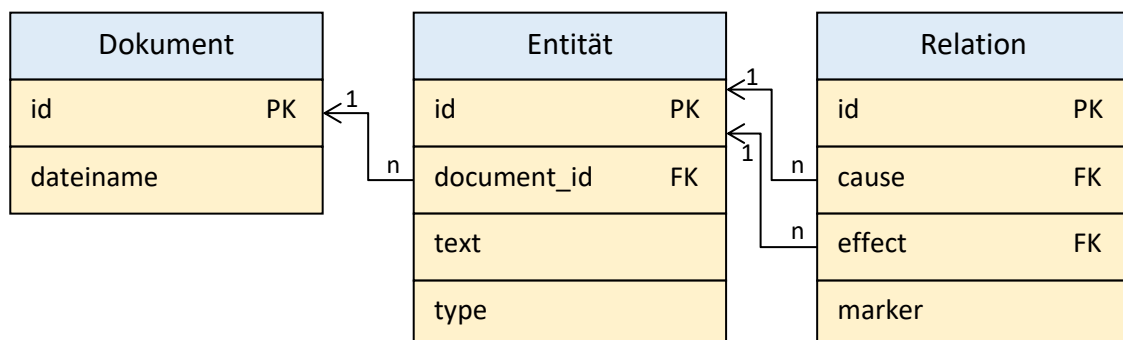
Die persistierte JSON-Ausgabe des ersten Schritts bildet zugleich einen prüfbaren Zwischenstand (siehe Abschnitt [5.2](#)), an dem sich die erkannten Entitäten vor der Kausalitätsextraktion kontrollieren lassen. Die so erzeugten Entitäten und kausalen Beziehungen bilden die Ausgabe der Transformationsphase und dienen als Eingabe für die Ladephase, die sie in das relationale Datenmodell überführt (siehe Abschnitt [5.6](#)).

5.6 Ladephase: Überführung in ein strukturiertes Datenmodell

Die Ladephase bildet den letzten Schritt des ETL-Prozesses und überführt die Ausgaben der Transformationsphase in das relationale Zieldatenmodell. Sie verarbeitet die beiden JSON-Ausgaben der Transformationsschritte, die Entitäten und die kausalen Beziehungen, und persistiert sie dauerhaft in der Datenbank.

Das relationale Datenmodell besteht aus drei Tabellen, deren Aufbau Abbildung 22 zeigt. Das dargestellte Schema ist eine konkrete Ausgestaltung des Zieldatenmodells: Über die zentrale Schemadefinition (siehe Abschnitt 5.2) lässt sich die Struktur der Tabellen und Felder an andere Anwendungsfälle anpassen. Das zugrunde liegende Entity-Relation-Modell aus Entitäten und den kausalen Beziehungen zwischen ihnen bleibt dabei bestehen, da es den fachlichen Kern der Informations- und Kausalitätsextraktion abbildet. Die Tabelle **Dokument** hält jedes verarbeitete PDF-Dokument mit einem eindeutigen Bezeichner und dem Dateinamen als Quelle fest. Die Tabelle **Entität** speichert die im ersten Transformationsschritt erkannten Entitäten mit Bezeichner, Textausschnitt und Typ und verweist über einen Fremdschlüssel auf das zugehörige Dokument. Die Tabelle **Relation** speichert die im zweiten Schritt erkannten kausalen Beziehungen; sie verweist über zwei Fremdschlüssel auf die Ursache- und die Wirkungsentität und hält den kausalen Marker fest, der bei impliziter Kausalität leer bleibt. Die Tabelle **Dokument** erfüllt dabei zwei Aufgaben. Sie ordnet jede Entität ihrem Herkunftsdocument zu und erlaubt es so, die Inhalte mehrerer Dokumente in derselben Datenbank zu halten und gezielt je Dokument abzufragen. Über die Zuordnung der Entitäten zum Dokument sind auch die kausalen Beziehungen mittelbar ihrem Ursprungsdocument zugeordnet.

Abbildung 22: Beispielhaftes relationales Zieldatenmodell der Ladephase



Die Abbildung der JSON-Ausgaben auf das relationale Schema erfolgt direkt (siehe Abschnitt 5.2). Jedes Objekt im JSON-Array wird zu einer Tabellenzeile, und jedes Feld eines Objekts zu einem Spaltenwert der entsprechenden Tabelle. Die Entitätsobjekte des ersten Transformationsschritts füllen mit ihren Feldern **id**, **text** und **type** die Tabelle **Entität**; die Relationsobjekte des zweiten Schritts füllen mit **id**, **cause**, **effect** und **marker** die Tabelle **Relation**. Die Felder **cause** und **effect**, die im JSON auf Entitätsbezeichner verweisen, werden dabei zu den Fremdschlüsseln auf die Tabelle **Entität**. Der Dateiname des verarbeiteten Dokuments bildet zusammen mit einem erzeugten Bezeichner den Eintrag in der Tabelle **Dokument**, auf den die Entitäten über ihren Dokument-Fremdschlüssel verweisen. Da JSON-Struktur und Tabellenschema aus derselben Schemadefinition hervorgehen, sind Transformation und Laden über ein gemeinsames Datenmodell gekoppelt und müssen nicht getrennt synchron gehalten werden.

Der Ladevorgang erfolgt in drei aufeinanderfolgenden Stufen, die der Fremdschlüsselstruktur des Datenmodells folgen. Zunächst schreibt der Prozess das Dokument in die Datenbank, anschließend die zugehörigen Entitäten und zuletzt die kausalen Beziehungen. Diese Reihenfolge ist notwendig, da jede Stufe über Fremdschlüssel auf die vorige verweist: Die Entitäten verweisen auf das Dokument, die Beziehungen auf die Ursache- und Wirkungsentität. Die referenzierten Einträge müssen daher jeweils bereits in der Datenbank vorliegen, damit die Fremdschlüsselbedingungen erfüllt sind. Der Prozess lädt die Daten dabei transaktional mit aktiver Prüfung der Fremdschlüsselbedingungen und nicht als Masseladevorgang (Bulk-Load) mit ausgesetzter Integritätsprüfung. Ausschlaggebend ist weniger die Ladegeschwindigkeit, da pro Dokument nur eine überschaubare Zahl von Entitäten und Beziehungen anfällt und die Dokumente einzeln verarbeitet werden (siehe Abschnitt 5.4), als vielmehr die referentielle Integrität: Da die zu ladenden Daten aus den Ausgaben eines Sprachmodells stammen, stellt die aktive Prüfung sicher, dass gespeicherte Beziehungen stets auf vorhandene Entitäten verweisen.

Das relationale Datenmodell bildet die definierte Ausgabe und Schnittstelle des Prozesses (siehe Abschnitt 5.1). Es liegt in einer strukturierten, abfragbaren Form vor, in der sich die extrahierten Entitäten und ihre kausalen Beziehungen je Dokument auswerten und durch nachgelagerte Systeme weiterverwenden lassen.

Damit ist der ETL-Prozess von der Extraktion über die Transformation bis zum Laden vollständig konzipiert. Das Kapitel hat die Systemanforderungen bestimmt, die Gesamtarchitektur festgelegt, die eingesetzten Werkzeuge und Modelle begründet und die drei Phasen im Einzelnen ausgestaltet. Mehrere Bestandteile der Konzeption sind dabei bewusst als beispielhafte Ausgestaltung angelegt und noch nicht abschließend festgelegt, etwa die konkrete Formulierung der Prompts und die konkrete Ausgestaltung des Zieldatenmodells. Das folgende Kapitel überführt die Konzeption in eine prototypische Implementierung und erprobt dabei diese offenen Bestandteile. Anschließend wird der Prozess exemplarisch auf Dokumente mit unterschiedlichen Bestandteilen wie Fließtext, Tabellen und Abbildungen angewendet, um sein Verhalten anhand konkreter Beispiele zu veranschaulichen.

6 Prototypische Implementierung

Dieses Kapitel beschreibt die prototypische Implementierung des in Kapitel [5](#) konzipierten ETL-Prozesses. Aufbauend auf der zuvor entwickelten Architektur setzt der Prototyp die drei Phasen Extraktion, Transformation und Laden als lokal ausführbare Verarbeitungskette um und überführt ein digital erzeugtes, multimodales medizinisches PDF-Dokument in eine relationale Datenbank. Der Schwerpunkt liegt dabei auf den für das Verständnis wesentlichen Codeausschnitten und den Erkenntnissen aus der Erprobung. Zunächst beschreibt Abschnitt [6.1](#) die technischen Rahmenbedingungen aus Hardware, Ausführungsumgebung sowie eingesetzten Werkzeugen und Modellen. Anschließend erläutern die Abschnitte [6.2](#), [6.3](#) und [6.4](#) die Umsetzung der Extraktions-, der Transformations- und der Ladephase. Abschließend beschreibt Abschnitt [6.5](#), wie ein Steuerskript die drei Phasen zu einem durchgängigen Ablauf zusammenführt und wie sich der Prototyp verhält, wenn eine Phase scheitert.

6.1 Technische Rahmenbedingungen

Dieser Abschnitt beschreibt die technischen Rahmenbedingungen des Prototyps, bestehend aus der eingesetzten Hardware, der Ausführungsumgebung sowie den verwendeten Werkzeugen und Modellen. Die Umsetzung folgt der Anforderung einer vollständig lokalen Verarbeitung, sodass die Dokumentinhalte die lokale Umgebung zu keinem Zeitpunkt verlassen (siehe Abschnitt [5.1](#)).

6.1.1 Hardware und Ausführungsumgebung

Der Prototyp wird auf einem Notebook mit einem Intel-Core-i5-8250U-Prozessor und 16 Gigabyte Arbeitsspeicher unter Windows 11 ausgeführt. Das System verfügt über keine dedizierte Grafikeinheit, sondern lediglich über eine integrierte Grafiklösung ohne eigenen Grafikspeicher. Die Sprach- und Vision-Language-Modelle werden daher vollständig auf dem Prozessor ausgeführt. Der Datenschutz erfordert eine lokale Verarbeitung; die eingesetzte Hardware zeigt, dass diese auch unter begrenzten Ressourcen möglich ist (siehe Abschnitt [2.4](#)). Sie bestimmt zugleich die praktische Verarbeitungsgeschwindigkeit: Die rein prozessorbasierte Ausführung führt insbesondere bei großen Eingaben zu erheblichen Verarbeitungszeiten.

Als Programmiersprache dient Python in der Version 3.14. Die lokale Ausführung der Modelle übernimmt die Laufzeitumgebung Ollama in der Version 0.31.2; die Ansteuerung aus dem Python-Prozess erfolgt über den zugehörigen Python-Client ollama in der Version 0.6.2. Die Unterscheidung ist relevant, da die Laufzeitumgebung und der Python-Client eigenständige Komponenten mit getrennten Versionsständen sind. Die Dokumentkonvertierung nutzt das Werkzeug Docling, die Vorbereitung der Bilddateien das Bildverarbeitungspaket Pillow. Die über den Paketmanager pip installierten Abhängigkeiten sind mit ihren Versionsständen in der Datei requirements.txt festgehalten, die dem Projekt beiliegt (siehe Listing 26).

Listing 26: Verwendete Python-Pakete mit Versionsständen

```
1.# python-pakete des Prototyps
2.
3.docling 2.118.0
4.docling-core 2.86.0
5.ollama (Client) 0.6.2
6.pillow 12.3.0
```

6.1.2 Verwendete Modelle

Der Prototyp setzt zwei Modelle ein, die nacheinander in unterschiedlichen Phasen zum Einsatz kommen. In der Extraktionsphase erzeugt das Vision-Language-Modell Qwen3-VL-8B (siehe Abschnitt 3.3.6) die textuellen Beschreibungen der Abbildungen (siehe Abschnitt 6.2). In der Transformationsphase extrahiert das Sprachmodell gpt-oss-20b die Entitäten und kausalen Beziehungen (siehe Abschnitt 6.3). Beide Modelle werden über Ollama geladen und ausgeführt.

Da beide Modelle in getrennten Phasen arbeiten, müssen sie nicht gleichzeitig im Arbeitsspeicher vorgehalten werden. Der Prozess gibt das Vision-Language-Modell nach der Extraktionsphase wieder frei, sodass der Arbeitsspeicher für das nachfolgende Sprachmodell zur Verfügung steht. Die Freigabe erfolgt über den Parameter **keep_alive** der Modellaufrufe, der das jeweilige Modell unmittelbar nach dem Aufruf entlädt (siehe Listing 28 in Abschnitt 6.2 und Listing 35 in Abschnitt 6.3). Diese sequenzielle Ausführung ermöglicht den Betrieb beider Modelle innerhalb des verfügbaren Arbeitsspeichers bei der Verarbeitung eines einzelnen Dokuments und entspricht der linearen, phasenweisen Verarbeitung des ETL-Prozesses (siehe Abschnitt 5.4). Sie ist damit auf den sequenziellen Einzelbetrieb ausgelegt; ein gleichzeitiger Betrieb mehrerer Verarbeitungsvorgänge, wie er in einer serverseitigen Umgebung entstünde, ist nicht Gegenstand des Prototyps (siehe Abschnitt 8.4).

6.1.3 Eingabedokumente

Der Prototyp setzt digital erzeugte PDF-Dokumente mit eingebetteter Textebene voraus. Gescannte Dokumente ohne Textebene liegen außerhalb des betrachteten Anwendungsfalls. Die technische Umsetzung dieser Voraussetzung in der Dokumentkonvertierung beschreibt Abschnitt [6.2](#).

6.2 Extraktionsphase

Die Extraktionsphase überführt ein digital erzeugtes PDF-Dokument in eine rein textuelle Markdown-Repräsentation und bildet damit die definierte Eingabe für die nachfolgende Transformationsphase (siehe Abschnitt [6.3](#)). Zwei Komponenten wirken dabei zusammen: das Dokumentwerkzeug Docling gewinnt Fließtext, Überschriften und Tabellen und löst enthaltene Abbildungen als Bilddateien heraus, und ein lokales Vision-Language-Modell erzeugt aus diesen Bilddateien textuelle Beschreibungen (siehe Abschnitt [5.4](#)).

6.2.1 Dokumentkonvertierung mit Docling

Docling konvertiert das eingelesene PDF-Dokument in eine strukturierte Dokumentrepräsentation, aus der sich das Markdown ableitet. Da der Prozess ausschließlich digital erzeugte PDF-Dokumente ohne aktivierte optische Zeichenerkennung verarbeitet (siehe Abschnitt [6.1](#)), gewinnt Docling den Inhalt unmittelbar aus der Textebene und überführt ihn in eine einheitliche Markdown-Struktur. Die Erprobung bestätigt, dass die Text- und Tabelleninhalte auf diesem Weg auch ohne optische Zeichenerkennung vollständig gewonnen werden.

Für die Behandlung von Abbildungen ist die Konfiguration der Konvertierung entscheidend. Docling reduziert Abbildungen in der Standardausgabe auf einen Platzhalter und löst sie erst bei aktivierter Bilderzeugung als eigenständige Bilddateien heraus (siehe Abschnitt [4.3](#)). Die Konvertierung aktiviert daher die Bilderzeugung und exportiert das Markdown mit Bildreferenzen anstelle von Platzhaltern, sodass Docling an der jeweiligen Position der Abbildung eine Referenz auf die zugehörige Bilddatei einträgt. Zusätzlich bestimmt der Skalierungsfaktor `images_scale` die Auflösung der von Docling erzeugten Bilder: Ein höherer Wert erhöht die Auflösung, während der Wert 1,0 der Standardqualität entspricht. Der gewählte Wert 2,0 liefert damit eine gegenüber der Standardqualität erhöhte Auflösung der herausgelösten Bilder, um eine für die

anschließende Bildbeschreibung ausreichende Bildqualität zu erhalten³⁴. Ein Vergleichslauf mit einem niedrigeren Wert belegt, dass diese Erhöhung notwendig ist (siehe Abschnitt 6.2.4). Listing 27 zeigt die zugehörige Konfiguration und den Export.

Listing 27: Docling-Konfiguration und Markdown-Export mit Bildreferenzen

```
1. pipeline_options = PdfPipelineOptions()
2. pipeline_options.images_scale = 2.0
3. pipeline_options.generate_picture_images = True
4. pipeline_options.do_ocr = False
5.
6. doc.save_as_markdown(md_path, image_mode=ImageRefMode.REFERENCED)
```

6.2.2 Bildbeschreibung mit dem Vision-Language-Modell

Die herausgelösten Bilddateien überführt anschließend das Vision-Language-Modell in textuelle Beschreibungen. Das in Abschnitt 6.1 eingeführte Vision-Language-Modell Qwen3-VL-8B verarbeitet Text- und Bildinformationen gemeinsam. Das Modell erhält je Abbildung die Bilddatei zusammen mit einer Anweisung, den Bildinhalt sachlich und knapp zu beschreiben und dabei erkennbare Beschriftungen, Achsen, Pfeile und Beziehungen zu berücksichtigen, ohne nicht sichtbare Informationen zu ergänzen. Listing 28 zeigt den Modellaufruf; die Angabe zur Speicherfreigabe gibt das Modell nach dem Aufruf wieder frei (siehe Abschnitt 6.1).

Listing 28: Aufruf des Vision-Language-Modells zur Bildbeschreibung (Ausschnitt)

```
1. prompt = (
2.     "You are describing a figure from a medical document. "
3.     "Describe the content of this image factually and concisely in "
4.     "plain text. "
5.     "Include any labels, axes, arrows, and relationships shown. "
6.     "Do not add information that is not visible in the image."
7. )
8. antwort = ollama.chat(
9.     model="qwen3-vl:8b",
10.    messages=[{"role": "user", "content": prompt, "images": [
11.        str(bildpfad)]}],
12.    options={"temperature": 0},
13.    keep_alive=0
14. )
14. beschreibung = antwort["message"]["content"].strip()
```

Der Prozess durchsucht das von Docling erzeugte Markdown nach Bildreferenzen und ersetzt jede Referenz durch die zugehörige Beschreibung an derselben Position (siehe Listing 29). Dadurch stehen die Bildinhalte als Text an der ursprünglichen Stelle im Dokument, und die Markdown-Repräsentation enthält keine verbleibenden Bildverweise mehr.

³⁴ https://docling-project.github.io/docling/reference/pipeline_options/#docling.datamodel.pipeline_options.PaginatedPipelineOptions.images_scale, https://docling-project.github.io/docling/generated/examples/export_figures/ [Zugriff am 14.07.2026].

Listing 29: Ersetzen der Bildreferenzen durch die textuellen Beschreibungen

```
1. bild_pattern = re.compile(r"![[\^]]*\]\(\([^\)]+\)\)")  
2. markdown_final = bild_pattern.sub(ersetze_referenz, markdown)
```

6.2.3 Robustheit gegenüber realen Dokumenteigenschaften

Die Verarbeitung realer Dokumente erfordert zusätzliche Maßnahmen, die sich in der Erprobung als notwendig erwiesen haben. Docling löst nicht nur inhaltstragende Abbildungen heraus, sondern auch kleinformatige grafische Elemente wie Symbole. Solche Elemente von wenigen Pixeln Kantenlänge tragen keinen beschreibbaren Inhalt und werden daher vor dem Aufruf des Vision-Language-Modells anhand ihrer geringen Abmessungen aussortiert.

Großformatige Abbildungen mit hoher Detaildichte führten in der Entwicklung bei der Verarbeitung auf der eingesetzten Hardware vereinzelt zu einem Abbruch der Modellausführung. Der genaue Punkt, ab dem die Ausführung abbricht, wurde nicht bestimmt, da er von der jeweiligen Abbildung und der verfügbaren Hardware abhängt. Der Prozess begrenzt daher vorsorglich die Bildgröße und verkleinert zu große Abbildungen vor der Beschreibung proportional, sodass die längste Kante einen festen Grenzwert von 1600 Pixeln nicht überschreitet. Da die Skalierung das Seitenverhältnis beibehält und das Bild nicht beschneidet, bleibt der vollständige Bildinhalt erhalten und lediglich die Auflösung verringert sich. Bei den betrachteten Dokumenten greift diese Begrenzung nicht, da die herausgelösten Abbildungen mit höchstens 1328 Pixel Kantenlänge darunter bleiben; sie wirkt als Vorsorge für umfangreichere Vorlagen.

Ergänzend fängt der Prozess Fehler bei der Beschreibung einzelner Bilder ab, sodass ein fehlgeschlagener Einzelaufruf den Ablauf nicht unterbricht, sondern die betroffene Bildstelle mit einem Hinweis versieht und die Verarbeitung fortsetzt. Denselben Hinweis setzt der Prozess, wenn das Modell eine leere Beschreibung zurückgibt: Dieser Fall löst keine Ausnahme aus und bliebe andernfalls unbemerkt. Eine Wiederholung des Aufrufs mit verringerter Auflösung nimmt der Prozess dagegen bewusst nicht vor. Das Modell beschreibt eine Abbildung bei geringerer Auflösung nicht ungenauer, sondern erfindet sie (siehe Abschnitt [6.2.4](#)); ein Wiederholungsversuch mit einem kleineren Bild ersetzt damit einen erkennbaren Fehlschlag durch eine frei erfundene Beschreibung.

6.2.4 Ergebnis der Extraktionsphase

Das Ergebnis der Extraktionsphase ist eine rein textuelle Markdown-Repräsentation, die Fließtext, Überschriften und Tabellen des Dokuments sowie die textuellen Beschreibungen der enthaltenen Abbildungen an ihrer jeweiligen Position enthält. Docling zeichnet die Überschriften als Markdown-Überschriften aus, überführt die

Tabellen in die Markdown-Tabellenschreibweise und gibt den Fließtext absatzweise in der Lesereihenfolge des Dokuments wieder. Listing 30 zeigt einen Ausschnitt vor und nach der Ersetzung der Bildreferenz auf die erste Abbildung des Dokuments (siehe Abbildung 18): Die Überschrift erscheint als Markdown-Überschrift (Zeile 2), und an die Stelle der Bildreferenz (Zeile 5) tritt die Beschreibung der Abbildung (Zeile 12).

Listing 30: Ausschnitt der Markdown-Repräsentation vor und nach der Ersetzung der Bildreferenz

```

1. # Vor der Ersetzung
2. ## Data Synthesis and Analysis
3. For intervention effects using relative risks (RRs), meta-
4. analyses used random-effects models in Review Manager version 5.1
5. (The Cochrane Collaboration). ...
6. Fordiagnosticaccuracy, analyseswerestratifiedbytimingofthe index
7. test in pregnancy and comparison, including different test
8. thresholds. ...
9. ![Image](phase1_out\markdown_mit_referenzen_artifacts\image_00000
10. 3_3616df20...png)
11.
12. # Nach der Ersetzung
13. ## Data Synthesis and Analysis
14. For intervention effects using relative risks (RRs), meta-
15. analyses used random-effects models in Review Manager version 5.1
16. (The Cochrane Collaboration). ...
17. Fordiagnosticaccuracy, analyseswerestratifiedbytimingofthe index
18. test in pregnancy and comparison, including different test
19. thresholds. ...
20. [Bildbeschreibung: Figure 1: Analytic Framework for Gestational
21. Diabetes is a flowchart illustrating screening considerations. ...]

```

Die Erprobung an einem Dokument mit zwei inhaltstragenden Abbildungen und mehreren Symbolen bestätigt den Ablauf: die Symbole werden übersprungen, beide Abbildungen als Bilddateien herausgelöst und beschrieben, und die Beschreibungen ersetzen die Referenzen positionsgenau. Die erzeugten Beschreibungen geben die grobe Struktur der Diagramme zuverlässig wieder, können jedoch einzelne Detailelemente wie Nummerierungen fehlerhaft wiedergeben.

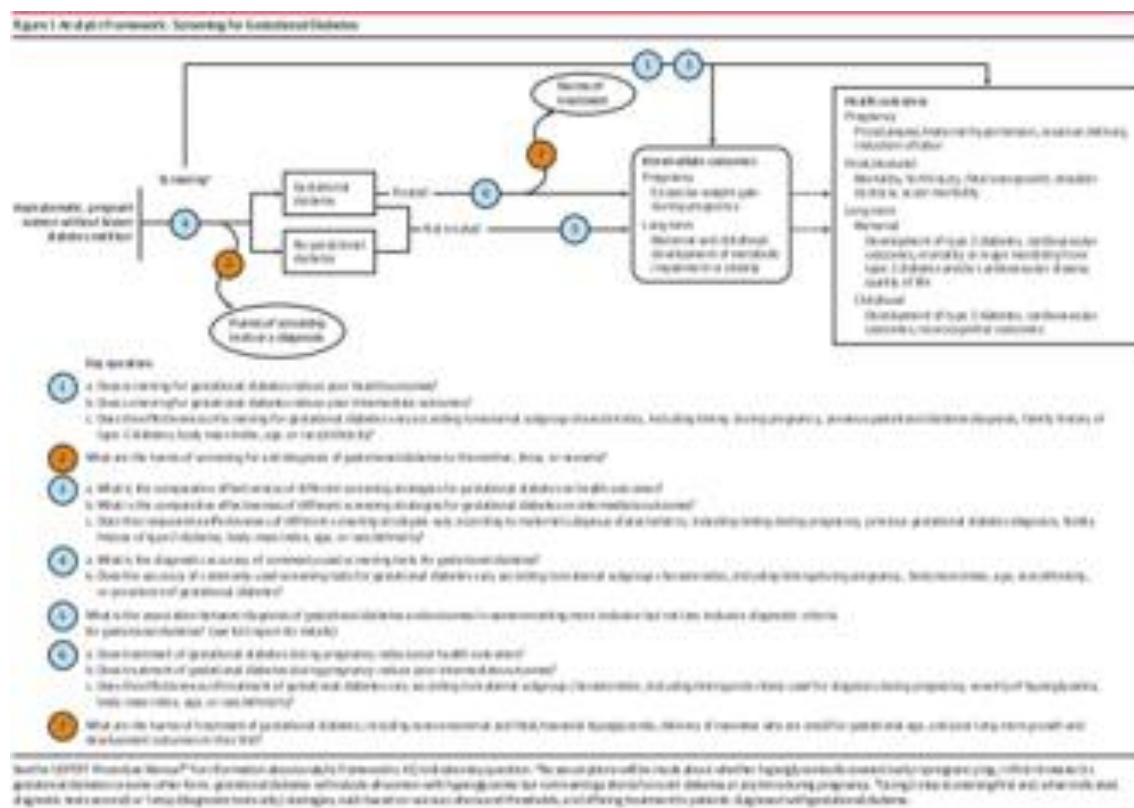
Die Auflösung der herausgelösten Abbildungen entscheidet darüber, ob die Beschreibung den Bildinhalt überhaupt trifft. Ein Vergleichslauf mit dem Skalierungsfaktor 0,5 anstelle von 2,0 verkleinert dieselbe Abbildung (siehe Abbildung 18) von 946 auf 237 Pixel Breite (siehe Abbildung 23). Das Modell beschreibt sie daraufhin nicht ungenauer, sondern erfindet ihren Inhalt: Aus dem Analyserahmen zum Screening auf Gestationsdiabetes (Listing 31, Zeile 2) wird ein Modell eines Notfallsystems für die gemeindenahе Versorgung (Listing 31, Zeile 7). Die grobe Bildstruktur gibt das Modell dabei zutreffend wieder – Kästen, Ovale, blaue und orange Knoten sowie eine Legende sind tatsächlich vorhanden –, sämtliche Beschriftungen erfindet es jedoch, vom Titel bis zur Bedeutung der Legende. Für die zweite Abbildung liefert das Modell bei dieser Auflösung eine leere Antwort. Beide Läufe verwenden

dieselbe Eingabe und dieselbe Modellkonfiguration, sodass der Unterschied allein auf die Auflösung zurückgeht. Das Verhalten entspricht der in Abschnitt 2.4 beschriebenen Halluzination: Das Modell erzeugt eine plausibel wirkende, aber faktisch falsche Ausgabe und zeigt die fehlende Lesbarkeit nicht an.

Listing 31: Beschreibungen derselben Abbildung bei unterschiedlicher Auflösung (gekürzt)

1. **# Skalierungsfaktor 2,0 - Abbildung mit 946x1328 Pixel**
2. Figure 1: Analytic Framework for Gestational Diabetes is a flowchart illustrating screening considerations.
3. - Central flow: Starts with "Asymptomatic, pregnant women without known diabetes mellitus." A step labeled "Screening" (step 4) branches into two paths:
4. - "Gestational diabetes" -> splits into "Treated" (step 6) and "Not treated" (step 5).
- 5.
6. **# Skalierungsfaktor 0,5 - dieselbe Abbildung mit 237x332 Pixel**
7. The image is a flowchart titled "Figure 1. A Model of a Rapid Response System for Community-Based Care." It includes rectangular boxes: one labeled "Rapid Response Team" with bullet points, another labeled "Inpatient Admission Process," and two lower rectangles ("Discharge Planning" and "No Inpatient Admission"). Circular nodes (blue and orange) represent process steps ... A legend at the bottom identifies blue circles as steps a-e and the orange circle as step f.

Abbildung 23: Analyserahmen bei Skalierung 0,5 (237 Pixel Breite)



Bei der Textextraktion zeigt sich zugleich eine Eigenschaft des Ausgangsdokuments. Der Textinhalt liegt vollständig vor, in einem Teil des Fließtextes fehlen jedoch die Zwischenräume zwischen den Wörtern, sodass mehrere Wörter zu einer Zeichenkette zusammenlaufen (siehe Listing [30](#), Zeile [4](#)). Diese Zwischenräume fehlen bereits in der Textebene des PDF-Dokuments: Eine Textsuche im Originaldokument findet die betroffenen Passagen ausschließlich ohne Leerzeichen. Die im Layout sichtbaren Wortabstände sind dort folglich nicht als Zeichen hinterlegt, und die Konvertierung gibt die Textebene unverändert wieder. Die Überschriften, die Tabellen und die Absatzstruktur bleiben davon unberührt. Dieses Verhalten tritt dokumentabhängig auf und zeigt sich bei den in Kapitel [7](#) verarbeiteten Dokumenten nicht. Diese Repräsentation bildet die Eingabe für die Transformationsphase (siehe Abschnitt [6.3](#)).

6.3 Transformationsphase

Die Transformationsphase überführt die textuelle Markdown-Repräsentation aus der Extraktionsphase in eine strukturierte Repräsentation aus Entitäten und kausalen Beziehungen. Ein lokal ausgeführtes Sprachmodell verarbeitet dazu die Markdown-Eingabe in zwei aufeinander aufbauenden Schritten: der erste Schritt extrahiert die Entitäten, der zweite die kausalen Beziehungen zwischen diesen Entitäten (siehe Abschnitt [5.5](#)). Als Modell dient das in Abschnitt [6.1](#) eingeführte Sprachmodell gpt-oss-20b.

6.3.1 Zentrale Schemadefinition und Erzwingung gültiger Ausgaben

Für beide Schritte gibt eine zentrale Schemadefinition die Struktur der erwarteten Ausgabe vor. Aus dieser Definition leiten sich sowohl das Ausgabeformat der Modellaufrufe als auch das relationale Zielschema der Ladephase ab, sodass Prompt und Datenbank auf einer gemeinsamen Grundlage beruhen (siehe Abschnitt [5.6](#)).

Listing [32](#) zeigt die Definition der Entitäts- und Relationsstruktur. Jedes Feld trägt zwei Typangaben: Die erste benennt den Typ im JSON-Schema des Modellaufrufs und folgt dessen Schreibweise (**string**, **integer**), die zweite den Spaltentyp der Datenbank in der Schreibweise von SQLite (**TEXT**, **INTEGER**). Hinzu kommen ein etwaiger Fremdschlüsselbezug sowie die Angaben, ob das Modell das Feld selbst erzeugt und ob die Datenbank es verpflichtend führt.

Listing 32: Zentrale Schemadefinition der Zieltabellen des Prototyps

```

1. @dataclass
2. class Feld:
3.     # Feldname (in JSON und Datenbank identisch)
4.     name: str
5.     # Typ im JSON-Schema: "string" | "integer"
6.     json_typ: str
7.     # Typ in SQLite: "TEXT" | "INTEGER"
8.     sql_typ: str
9.     # Zieltabelle bei Fremdschlüssel
10.    referenziert: str = None
11.    # vom Modell erzeugtes Feld
12.    im_prompt: bool = True
13.    # NOT-NULL-Bedingung
14.    pflicht: bool = False
15.
16. ENTITAET = Tabelle("entitaet", [
17.    Feld("document_id", "integer", "INTEGER", referenziert=
18.        "dokument", im_prompt=False, pflicht=True),
19.    Feld("text", "string", "TEXT", pflicht=True),
20.    Feld("type", "string", "TEXT")
21. ])
22.
23. RELATION = Tabelle("relation", [
24.    Feld("cause", "integer", "INTEGER", referenziert="entitaet",
25.        pflicht=True),
26.    Feld("effect", "integer", "INTEGER", referenziert="entitaet",
27.        pflicht=True),
28.    Feld("marker", "string", "TEXT")
29. ])

```

Aus dieser Definition erzeugen zwei Ableitungen das JSON-Schema für die Modellaufufe und die Anweisung zur Erstellung der Datenbanktabellen (siehe Listing 33). Beide gehen aus derselben Quelle hervor, sodass eine Änderung an der Struktur unmittelbar auf das Ausgabeformat und das Datenbankschema wirkt; die daraus hervorgehenden Zieltabellen beschreibt Abschnitt 6.4.1.

Listing 33: Ableitung von JSON-Schema und Datenbankschema aus der Definition

```

1. # JSON-Schema für den Modellaufruf
2. format_schema = json_schema(ENTITAET, "entities")
3.
4. # CREATE TABLE für die Datenbank
5. tabellen_sql = create_table(ENTITAET)

```

Listing 34 zeigt das für den ersten Schritt erzeugte JSON-Schema. Es führt neben dem Textausschnitt und dem Typ einen Bezeichner, den das Modell selbst vergibt. Das Feld für den Dokumentbezug entfällt hier, da die Definition es von der Modellausgabe ausnimmt; die Ladephase ergänzt es (siehe Abschnitt 6.4.1).

Listing 34: Aus der Definition erzeugtes JSON-Schema für den ersten Schritt

```
1. {
2.   "type": "object",
3.   "properties": {
4.     "entities": {
5.       "type": "array",
6.       "items": {
7.         "type": "object",
8.         "properties": {
9.           "id": {"type": "integer"},
10.          "text": {"type": "string"},
11.          "type": {"type": "string"}
12.        },
13.        "required": ["id", "text", "type"]
14.      }
15.    }
16.  },
17.  "required": ["entities"]
18. }
```

Der jeweilige Modellaufruf übergibt das zugehörige Schema als verbindliches Ausgabeformat, wodurch das Modell seine Antwort als strukturiertes JSON nach diesem Schema erzeugt. Die Erprobung zeigt, wie die Vorgaben wirken. Erhält das Modell weder die Schemavorgabe noch eine Anweisung im Systemprompt, gibt es die Entitäten als nummerierte Textliste anstelle eines strukturierten JSON aus. Die Anweisung im Systemprompt, ausschließlich gültiges JSON ohne erklärenden Text auszugeben, führt für sich genommen zu einer JSON-Ausgabe, die jedoch das umschließende Objekt der Definition auslässt. Die Schemavorgabe erzeugt dagegen auch ohne diese Anweisung die festgelegte Struktur. Der Prototyp übergibt beide Vorgaben gemeinsam und macht die erwartete Ausgabe damit sowohl formal als auch in der Anweisung verbindlich.

Der dem Modell eigene Schließvorlauf muss dabei zugelassen bleiben (**think=True**). Setzt der Modellaufruf den Parameter **think** auf **false**, bleibt die Antwort gemeinsam mit der Schemavorgabe leer: Das Modell erzeugt Token, von denen keines in der Ausgabe erscheint. Ohne Schemavorgabe wirkt sich derselbe Parameter dagegen nicht aus, denn das Modell stellt seiner Antwort auch dann einen Schließvorlauf voran. Erst mit zugelassenem Schließvorlauf erzeugt das Modell gemeinsam mit der Schemavorgabe gültiges JSON. Listing [35](#) zeigt den Modellaufruf mit beiden Vorgaben.

Zusätzlich verarbeitet der Prozess die Modellausgabe nachgiebig, da gpt-oss-20b das umschließende Objekt der Schemadefinition gelegentlich weglässt und die Ausgabe unmittelbar als Liste liefert; die Auswertung akzeptiert daher sowohl die vollständige Objektstruktur als auch die unmittelbare Liste (siehe Listing [36](#)). Lässt sich die Ausgabe keiner der beiden Formen zuordnen, fängt der Prozess den Fehler bewusst nicht ab: Die Phase endet mit einem Fehlercode, woraufhin das Steuerskript den Ablauf abbricht und die folgende Phase nicht startet (siehe Abschnitt [6.5](#)). Da die Transformationsphase ihre

Ausgabe erst nach erfolgreicher Auswertung schreibt, gelangen unvollständige Ergebnisse auf diesem Weg nicht in die Ladephase.

Listing 35: Modellaufruf mit Systemprompt und abgeleitetem Schema (Ausschnitt)

```
1. system_prompt = {
2.     "role": "system",
3.     "content": "You are a precise information extraction system. "
4.               "You always respond with valid JSON only, no
                    explanations, no markdown, no numbered lists."
5. }
6.
7. antwort = ollama.chat(
8.     model="gpt-oss:20b",
9.     messages=[system_prompt, {"role": "user", "content":
10.                               prompt+markdown}],
11.     format=json_schema(ENTITAET, "entities"),
12.     options={"temperature": 0},
13.     keep_alive=0,
14.     think=True
15. )
```

Listing 36: Nachgiebige Auswertung der Modellausgabe

```
1. daten = json.loads(antwort["message"]["content"])
2. if isinstance(daten, dict):
3.     entities = daten.get("entities", [])
4. elif isinstance(daten, list):
5.     entities = daten
6. else:
7.     print(f"[Fehler] Unerwartetes JSON-Format: {type(daten)}")
8.     raise SystemExit(1)
```

6.3.2 Erster Schritt: Extraktion der Entitäten

Der erste Schritt übergibt dem Modell die Markdown-Repräsentation zusammen mit der Anweisung, klinisch oder wissenschaftlich relevante Einheiten wie Krankheiten, Symptome, Medikamente, Behandlungen, Verfahren, diagnostische Tests, Patientengruppen oder Ergebnisgrößen zu extrahieren. Für jede Entität erzeugt das Modell einen Bezeichner, den Textausschnitt weitgehend im Wortlaut des Dokuments und einen Typ. Der Typ ist geführt offen: die Anweisung nennt mögliche Typen, überlässt dem Modell jedoch die konkrete Vergabe, sodass auch nicht vorab benannte Typen entstehen können.

Die Erprobung an einem medizinischen Beispieldokument liefert eine konsistente Menge typisierter Entitäten. Das Modell erfasst dabei dieselbe Institution teilweise mehrfach als eigenständige Entitäten, etwa einmal als Organisation und einmal als benannte Leitlinie und ordnet die Typen weitgehend nachvollziehbar zu. Eine Verbindung zwischen beiden Formen stellt der Prozess nicht her: Die Schemadefinition sieht für eine Entität allein den Textausschnitt und den Typ vor, sodass sich eine Gleichsetzung zweier Entitäten nicht ausdrücken lässt (siehe Abschnitt [6.3.1](#)). Dieselbe Sache kann dadurch als zwei getrennte Entitäten in der Datenbank stehen. Einzelne Zuordnungen sind fachlich

diskutierbar, etwa die Einordnung eines Ereignisses als Krankheit; dies entspricht der erwartbaren Unschärfe einer geführten Typisierung.

6.3.3 Zweiter Schritt: Extraktion der kausalen Beziehungen

Der zweite Schritt übergibt dem Modell die Markdown-Repräsentation zusammen mit der im ersten Schritt erzeugten Entitätsliste und der Anweisung, Ursache-Wirkungs-Beziehungen zwischen diesen Entitäten zu erkennen. Für jede Beziehung erzeugt das Modell einen Bezeichner, einen Verweis auf die verursachende und einen auf die betroffene Entität sowie einen Marker. Die Verweise nutzen die Bezeichner aus dem ersten Schritt, wodurch die Beziehungen unmittelbar an die zuvor extrahierten Entitäten anknüpfen und deren wiederholte Erfassung entfällt (siehe Abschnitt [5.5](#)). Dazu bereitet der Prozess die Entitäten als Referenzliste mit ihren Bezeichnern auf und stellt sie dem Modell voran (siehe Listing [37](#)). Der Marker enthält das kausale Signalwort aus dem Text und bleibt leer, wenn die Kausalität implizit und ohne erkennbares Signalwort vorliegt (siehe Abschnitt [2.3.2](#)).

Listing 37: Aufbereitung der Entitäten als Referenzliste für den zweiten Schritt

```
1.entity_ref = "\n".join(f"{e['id']}: {e['text']} ({e['type']})" for  
  e in entities)
```

Die Erprobung bestätigt den Ablauf: das Modell löst die Verweise korrekt auf die zuvor extrahierten Entitäten auf und erkennt inhaltlich zutreffende, gerichtete Ursache-Wirkungs-Beziehungen, die den Kernaussagen des Dokuments entsprechen. Das Marker-Feld füllt das Modell überwiegend mit einer kurzen kausalen Phrase; eine durchgängige Reduktion auf ein einzelnes Signalwort gelingt nicht in allen Fällen.

6.3.4 Verarbeitung im gemeinsamen Kontext

Beide Schritte verarbeiten die Markdown-Repräsentation vollständig in einem gemeinsamen Kontext, ohne das Dokument in kleinere Abschnitte zu zerlegen. Diese Verarbeitung erhält satzübergreifende kausale Beziehungen, deren Ursache und Wirkung sich über mehrere Sätze oder Abschnitte verteilen (siehe Abschnitt [2.3.2](#)); eine abschnittsweise Verarbeitung würde solche Beziehungen an den Abschnittsgrenzen trennen. Mit der Dokumentgröße steigt jedoch der Verarbeitungsaufwand deutlich an, sodass die vollständige Verarbeitung großer Dokumente auf ressourcenbeschränkter Hardware zeitintensiv ist (siehe Abschnitt [6.1](#)).

6.3.5 Ergebnis der Transformationsphase

Das Ergebnis der Transformationsphase sind zwei strukturierte Ausgaben: eine Menge typisierter Entitäten und eine Menge kausaler Beziehungen, die über die Entitätsbezeichner miteinander verknüpft sind. Listing 38 zeigt einen Ausschnitt beider Ausgaben. Die Felder *cause* und *effect* einer Beziehung tragen die Bezeichner der zuvor extrahierten Entitäten, sodass sich jede Beziehung auf die Entitätsmenge zurückführen lässt; die Entität mit dem Bezeichner 1 tritt dabei in einer Beziehung als Ursache und in einer anderen als Wirkung auf. Beide folgen der zentralen Schemadefinition und bilden die Eingabe für die Ladephase, die sie in das relationale Zielschema überführt (siehe Abschnitt 6.4).

Listing 38: Ausschnitt der beiden Ausgaben der Transformationsphase

```
1. // Entitäten
2. [
3.   { "id": 1, "text": "Gestational diabetes", "type": "disease" },
4.   { "id": 4, "text": "1-step screening", "type": "procedure" },
5.   { "id": 16, "text": "primary cesarean deliveries", "type":
6.     "outcome" },
7.   { "id": 31, "text": "neonatal hypoglycemia", "type": "outcome" }
8. ]
9. // Kausale Beziehungen
10. [
11.   { "id": 1, "cause": 1, "effect": 16, "marker": "associated
12.     with" },
13.   { "id": 8, "cause": 4, "effect": 1, "marker": "associated with
14.     increased likelihood of" },
15.   { "id": 9, "cause": 4, "effect": 31, "marker": "associated with
16.     increased risk for" }
17. ]
```

6.4 Ladephase

Die Ladephase überführt die strukturierten Ausgaben der Transformationsphase in eine relationale Datenbank. Als Datenbanksystem dient SQLite, das über das in Python enthaltene Standardmodul angesprochen wird und damit ohne zusätzliche Softwarekomponente lokal betrieben werden kann (siehe Abschnitt 6.1). Das Zielschema besteht aus den drei Tabellen **Dokument**, **Entität** und **Relation**, die über Fremdschlüssel miteinander verbunden sind (siehe Abschnitt 5.6).

6.4.1 Ableitung des Zielschemas aus der zentralen Schemadefinition

Das relationale Zielschema und das Ausgabeformat der Transformationsphase gehen aus derselben zentralen Schemadefinition hervor (siehe Abschnitt 5.6 sowie Listing 32 in Abschnitt 6.3.1). Die Felder der extrahierten Entitäten und Beziehungen entsprechen

daher unmittelbar den Spalten der Zieltabellen: Der Textausschnitt und der Typ einer Entität bilden die Spalten der Entitätstabelle, und die verursachende Entität, die betroffene Entität sowie der Marker einer Beziehung bilden die Spalten der Relationstabelle. Die Dokumenttabelle ergänzt das Modell um den Dateinamen, den Pfad und den Ladezeitpunkt des verarbeiteten Dokuments und macht dadurch die Herkunft der gespeicherten Daten nachvollziehbar. Eine Prüfung, ob ein Dokument bereits verarbeitet wurde, nimmt der Prototyp dabei nicht vor: Die Dokumenttabelle lässt denselben Dateinamen mehrfach zu, sodass ein wiederholter Durchlauf einen zweiten Eintrag mit eigenen Entitäten und Beziehungen erzeugt. Listing 39 zeigt die aus der Definition erzeugten Anweisungen zur Erstellung der drei Zieltabellen.

Listing 39: Aus der Definition erzeugte Anweisungen zur Erstellung der Zieltabellen

```
1. CREATE TABLE IF NOT EXISTS dokument (
2.   id INTEGER PRIMARY KEY AUTOINCREMENT,
3.   dateiname TEXT NOT NULL,
4.   pfad TEXT,
5.   geladen_am TEXT NOT NULL
6. );
7.
8. CREATE TABLE IF NOT EXISTS entitaet (
9.   id INTEGER PRIMARY KEY AUTOINCREMENT,
10.  document_id INTEGER NOT NULL,
11.  text TEXT NOT NULL,
12.  type TEXT,
13.  FOREIGN KEY (document_id) REFERENCES dokument(id)
14. );
15.
16. CREATE TABLE IF NOT EXISTS relation (
17.  id INTEGER PRIMARY KEY AUTOINCREMENT,
18.  cause INTEGER NOT NULL,
19.  effect INTEGER NOT NULL,
20.  marker TEXT,
21.  FOREIGN KEY (cause) REFERENCES entitaet(id),
22.  FOREIGN KEY (effect) REFERENCES entitaet(id)
23. );
```

6.4.2 Dreistufiges Laden entlang der Fremdschlüssel

Der Ladevorgang erfolgt in drei aufeinanderfolgenden Stufen, die der Fremdschlüsselstruktur des Datenmodells folgen. Zunächst schreibt der Prozess das Dokument in die Datenbank, anschließend die zugehörigen Entitäten und zuletzt die kausalen Beziehungen. Jede Entität verweist über einen Fremdschlüssel auf das zugehörige Dokument, jede Beziehung auf die verursachende und die betroffene Entität; die referenzierten Einträge liegen daher zum Zeitpunkt des Einfügens bereits vor.

Der Prozess lädt die Daten transaktional mit aktiver Prüfung der Fremdschlüsselbedingungen, die bei SQLite eigens zu aktivieren ist. Alle Einfügungen eines Dokuments bilden eine gemeinsame Transaktion, die erst nach der letzten Stufe abgeschlossen wird. Tritt während des Ladens ein Fehler auf, macht der Prozess die

gesamte Transaktion rückgängig, sodass kein unvollständig geladenes Dokument in der Datenbank verbleibt (siehe Listing 40).

Listing 40: Transaktionales Laden mit aktiver Fremdschlüsselprüfung

```
1. verbindung.execute("PRAGMA foreign_keys = ON;")
2. try:
3.     # Stufe 1 Dokument, Stufe 2 Entitaeten, Stufe 3 Relationen ...
4.     verbindung.commit()
5. except sqlite3.Error:
6.     verbindung.rollback()
7.     raise
```

6.4.3 Vergabe eindeutiger Bezeichner

Die Bezeichner der extrahierten Entitäten sind innerhalb eines Dokuments eindeutig, beginnen jedoch je Dokument erneut und wären daher über mehrere Dokumente hinweg nicht eindeutig. Die Datenbank vergibt deshalb beim Einfügen eigene, global eindeutige Bezeichner. Der Prozess führt dabei eine Zuordnung zwischen dem ursprünglichen Bezeichner aus der Transformationsphase und dem von der Datenbank vergebenen Bezeichner. Beim Einfügen der Beziehungen löst er die Verweise auf die verursachende und die betroffene Entität über diese Zuordnung auf die vergebenen Bezeichner auf (siehe Listing 41). Dadurch bleiben die Beziehungen korrekt, und die Datenbank kann die Ergebnisse mehrerer Dokumente aufnehmen, ohne dass sich die Bezeichner überschneiden.

Listing 41: Übersetzung der Bezeichner und Auflösung der Verweise

```
1. id_map = {}
2. for e in entities:
3.     cursor.execute("INSERT INTO entitaet (document_id, text, type)
4.         VALUES (?, ?, ?);",
5.             (document_id, e["text"], e.get("type")))
6.     # lokale id -> vergebene DB-id
7.     id_map[e["id"]] = cursor.lastrowid
8. for r in relations:
9.     cause_db, effect_db = id_map.get(r["cause"]), id_map.get(
10.         r["effect"])
11.     if cause_db is None or effect_db is None:
12.         # Verweis auf nicht vorhandene Entitaet -> verwerfen
13.         continue
14.     cursor.execute("INSERT INTO relation (cause, effect, marker)
15.         VALUES (?, ?, ?);",
16.             (cause_db, effect_db, r.get("marker")))
```

Verweist eine Beziehung auf einen Bezeichner, der nicht unter den geladenen Entitäten vorliegt, verwirft der Prozess diese Beziehung, anstatt sie zu speichern. Dadurch verweisen gespeicherte Beziehungen stets auf vorhandene Entitäten (siehe Abschnitt 5.6).

6.4.4 Ergebnis der Ladephase

Das Ergebnis der Ladephase ist eine relationale Datenbank, in der die extrahierten Entitäten und kausalen Beziehungen strukturiert vorliegen und über Fremdschlüssel mit dem Herkunftsdocument sowie untereinander verbunden sind. Die Erprobung bestätigt den Ablauf: die drei Tabellen werden entlang der Fremdschlüssel befüllt, die Entitäten erhalten global eindeutige Bezeichner, und die Beziehungen verweisen über die aufgelösten Bezeichner auf die zugehörigen Entitäten. Die gespeicherten Daten stehen damit für strukturierte Abfragen zur Verfügung, etwa für die gemeinsame Abfrage einer kausalen Beziehung mit ihren verursachenden und betroffenen Entitäten.

6.5 Gesamtablauf des Prototyps

Die vorangegangenen Abschnitte beschreiben die drei Phasen einzeln. Dieser Abschnitt beschreibt, wie ein Steuerskript sie zu einem durchgängigen Ablauf zusammenführt und wie sich der Prototyp verhält, wenn eine Phase scheitert.

Ein Steuerskript startet die Phasen nacheinander und reicht das zu verarbeitende Dokument über eine Umgebungsvariable an jede von ihnen durch (siehe Listing 42). Jede Phase läuft dabei als eigenständiger Prozess, sodass der Arbeitsspeicher einer Phase vollständig frei ist, bevor die folgende startet; dies ergänzt die Freigabe der Modelle unmittelbar nach dem jeweiligen Aufruf (siehe Abschnitt 6.1.2). Der Prototyp verarbeitet jedes Dokument als abgeschlossene Einheit und durchläuft den Ablauf je Dokument einmal.

Listing 42: Steuerskript: Aufruf der Phasen und Abbruch bei einem Fehler

```

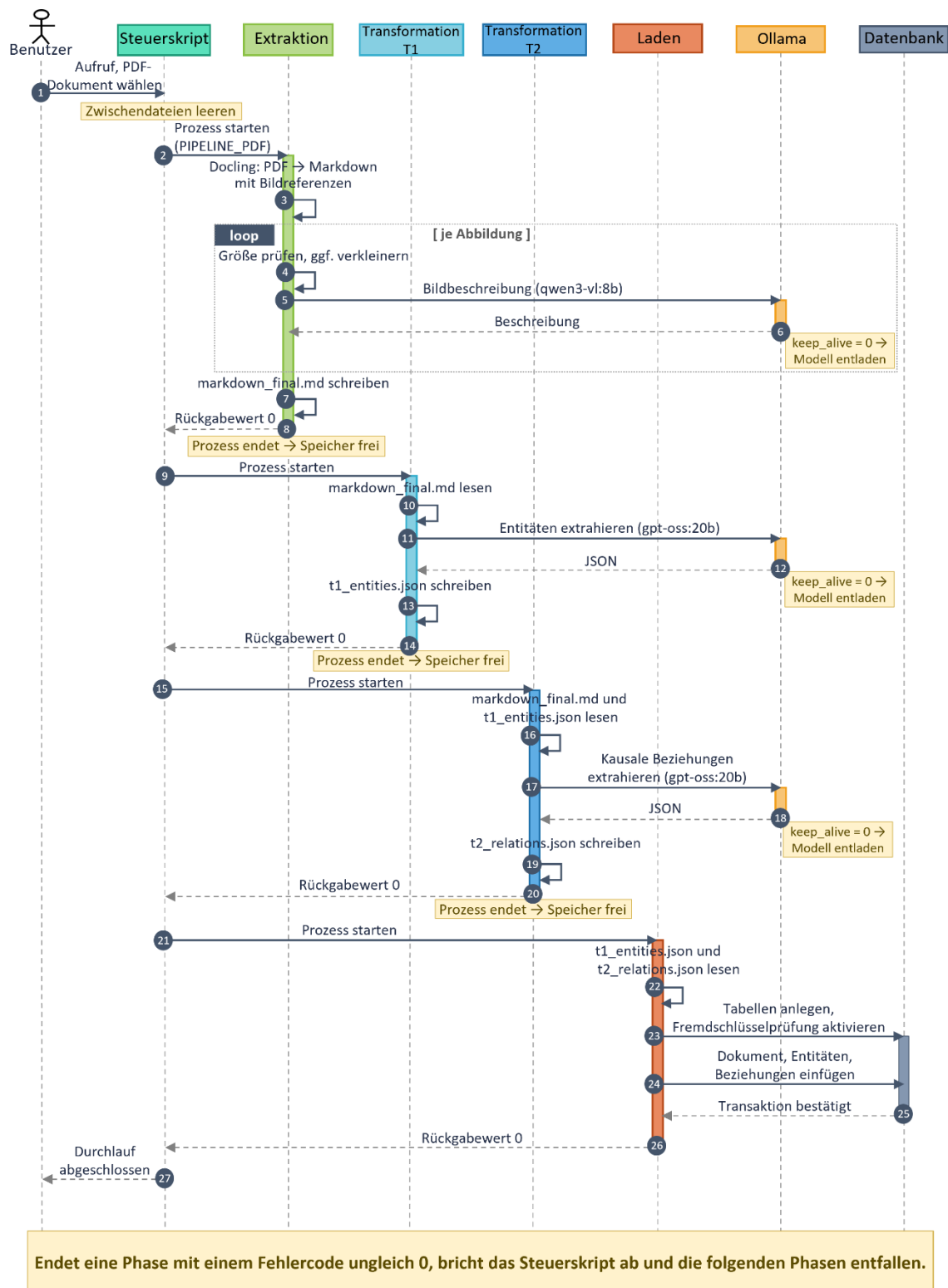
1. PHASEN = [
2.     ("extraktion.py", "Phase 1 - Extraktion"),
3.     ("transformation_t1.py", "Phase 2 - Transformation T1"),
4.     ("transformation_t2.py", "Phase 2 - Transformation T2"),
5.     ("laden.py", "Phase 3 - Laden")
6. ]
7.
8. umgebung = os.environ.copy()
9. # Dokument an alle Phasen durchreichen
10. umgebung["PIPELINE_PDF"] = str(pdf.resolve())
11. for skript, beschriftung in PHASEN:
12.     ergebnis = subprocess.run([sys.executable, skript],
13.                               env=umgebung)
14.     if ergebnis.returncode != 0:
15.         print(f"[Abbruch] {beschriftung} mit Fehlercode {ergebnis.
16.             returncode} beendet.")
17.         raise SystemExit(ergebnis.returncode)

```

Abbildung 24 stellt den zeitlichen Ablauf dar. Die Aktivierungsbalken der vier Teilschritte überlappen sich nicht, und jedes Modell ist nur während der zugehörigen Aufrufe geladen: Das Vision-Language-Modell gibt der Prototyp am Ende der Extraktionsphase frei, bevor die Transformationsphase das Sprachmodell lädt.

Die Fehlerbehandlung verteilt sich auf zwei Ebenen. Innerhalb einer Phase fängt der Prototyp diejenigen Fehler ab, die den weiteren Ablauf nicht gefährden: Eine fehlgeschlagene Bildbeschreibung unterbricht die Extraktion nicht (siehe Abschnitt 6.2.3), eine nicht verwertbare Modellausgabe führt zu einem kontrollierten Abbruch der Transformation durch das Steuerskript (siehe Abschnitt 6.3.1), und ein Fehler beim Laden macht die gesamte Transaktion rückgängig, sodass kein unvollständig geladenes Dokument in der Datenbank verbleibt (siehe Abschnitt 6.4.2). Auf der Ebene des Gesamtablaufs prüft das Steuerskript den Rückgabewert jeder Phase und bricht bei einem Fehler ab, statt die folgende Phase mit unvollständigen Zwischenergebnissen zu starten (siehe Abbildung 24).

Abbildung 24: Sequenzdiagramm des Gesamtablaufs



Die Erprobung bestätigt den Gesamtablauf: Ein einzelner Aufruf des Steuerskripts durchläuft die vier Teilschritte ohne Eingriff und erzeugt aus dem PDF-Dokument eine befüllte relationale Datenbank. Endet eine Phase mit einem Fehler, startet die folgende nicht. Die inhaltliche Betrachtung der Ergebnisse für unterschiedliche Dokumentbestandteile erfolgt in Kapitel 7.

7 Beispielhafte Anwendung

Dieses Kapitel wendet den in Kapitel [6](#) umgesetzten Prototyp beispielhaft auf medizinische PDF-Dokumente an und betrachtet, wie der Prozess die unterschiedlichen Bestandteile dieser Dokumente verarbeitet. Zunächst beschreibt Abschnitt [7.1](#) das Vorgehen bei der Auswahl und Verarbeitung der Dokumente. Anschließend betrachten die Abschnitte [7.2](#), [7.3](#) und [7.4](#) die Verarbeitung der textuellen, der tabellarischen und der bildlichen Bestandteile. Abschließend fasst Abschnitt [7.5](#) die Beobachtungen zu einem Gesamtbild der Stärken und Grenzen zusammen und leitet zur Schlussbetrachtung über.

7.1 Vorgehen bei der Anwendung

Die Anwendung dient der beispielhaften Betrachtung des Prozessverhaltens und nimmt keine quantitative Bewertung vor. Sie verzichtet daher auf Kennzahlen, auf einen Vergleich mit einer Referenzannotation und auf statistische Aussagen und betrachtet stattdessen an konkreten Beispielen, wie der Prozess die einzelnen Dokumentbestandteile verarbeitet (siehe Abschnitt [1.1](#)).

Ein einzelnes Dokument bildet nur eine bestimmte Ausprägung jedes Bestandteils ab: eine bestimmte Tabellenstruktur, eine bestimmte Art von Abbildung und eine bestimmte sprachliche Form kausaler Aussagen. Um das Verhalten des Prozesses gegenüber der Bandbreite realer Dokumente nachvollziehbar zu machen, betrachtet die Anwendung drei Dokumente mit unterschiedlich strukturierten Bestandteilen. Diese Vielfalt macht sowohl die Tragfähigkeit als auch die Grenzen des Prozesses sichtbar, ohne eine quantitative Bewertung vorzunehmen.

Als Beispieldokumente dienen drei frei zugängliche medizinische Publikationen, die den in Abschnitt [5.1](#) beschriebenen Testdaten entsprechen. Das erste Dokument ist eine randomisierte klinische Studie zur glykämischen Reaktion auf zwei Lokalanästhetika bei Zahnextraktionen³⁵. Das zweite Dokument ist eine Zusammenfassung von Empfehlungen zur Eisenmangelanämie bei Schwangeren und Kleinkindern³⁶. Das dritte Dokument ist eine systematische Übersichtsarbeit zur Bewertung von Leitlinien für das Screening auf Gestationsdiabetes³⁷. Die drei Dokumente unterscheiden sich gezielt in

³⁵ <https://www.researchgate.net/publication/346418357> [Zugriff am 11.07.2026].

³⁶ <https://doi.org/10.3945/ajcn.117.155788> [Zugriff am 11.07.2026].

³⁷ <https://bmjopen.bmj.com/content/9/5/e023014> [Zugriff am 11.07.2026].

ihren Bestandteilen: Das erste enthält eine Werttabelle mit verschachtelter Struktur, ein Flussdiagramm und eine überwiegend verneinende Aussage zur Kausalität, das zweite eine rechteckige Empfehlungstabelle und ein Fachdiagramm, das dritte eine umfangreiche Bewertungstabelle sowie ein Flussdiagramm.

Der Prozess verarbeitet jedes Dokument als abgeschlossene Einheit von der Extraktion bis zur relationalen Datenbank. Beim dritten Dokument³⁷ dienen nur die ersten vier Seiten des Originaldokuments als Eingabe, da das vollständige Dokument für die rein prozessorbasierte Verarbeitung zu umfangreich ist (siehe Abschnitt 6.1); die vier Seiten enthalten mit dem einleitenden Text, einer Bewertungstabelle und einem Flussdiagramm alle drei Bestandteile. Die folgenden Abschnitte greifen die Ergebnisse der drei Durchläufe nach Bestandteilen geordnet auf.

7.2 Verarbeitung textueller Bestandteile

Die textuellen Bestandteile bilden den Kern der Transformationsphase, da aus ihnen sowohl die Entitäten als auch die kausalen Beziehungen hervorgehen (siehe Abschnitt 6.3). Die drei Dokumente zeigen dabei ein deutlich unterschiedliches Verhalten, das eng an die sprachliche Form der jeweiligen Aussagen gebunden ist.

7.2.1 Extraktion der Entitäten

In allen drei Dokumenten gewinnt der Prozess eine zusammenhängende Menge typisierter Entitäten. Die geführte offene Typisierung richtet sich dabei nach dem jeweiligen Dokument. Das zweite Dokument³⁶ kommt mit den im Prompt genannten Typen aus und ordnet sie nachvollziehbar zu, etwa die Einordnung von Übelkeit und Verstopfung als Symptom. Das dritte Dokument³⁷ bildet dagegen überwiegend eigene Typen wie Leitlinie, Kriterium oder Studie, für die der Prompt keinen passenden Eintrag vorgibt.

Zugleich zeigt die Typisierung ihre Grenzen. Im ersten Dokument³⁵ entstehen wenig trennscharfe Typen wie Gerät oder Dosierung, und ein Herstellername mit vollständiger Anschrift wird als eine einzelne Entität erfasst (siehe Abbildung 25 und Listing 43). Im dritten Dokument, das zahlreiche Leitlinien einzeln aufzählt, erzeugt der Prozess für jede genannte Leitlinie eine eigene Entität, sodass ein erheblicher Teil der Entitätsmenge aus einzelnen Leitlinien besteht. Bei aufzählungslastigen Dokumenten erhöht die Extraktion damit die Zahl feingranularer Entitäten, was die Typvielfalt steigert, aber die Trennschärfe verringert.

Abbildung 25: Belegstellen der Entitäten aus Listing 43 im ersten Dokument

Materials and Methods: A total of 60 simple extractions (n=60), performed on teeth of 30 patients with type II diabetes mellitus, were included in the study. The sample was divided according to the used anesthetic solutions into two groups: Group A (lidocaine with epinephrine group; n=30) and Group B (lidocaine with norepinephrine group; n=30). A vasoconstrictor concentration of 1:100,000 was applied in both groups. Consistently with each extraction, blood glucose levels measurements, using portable glucometer, were done three times: immediately before local anesthesia administration, 5 minutes after local anesthesia, and 5 minutes after tooth extraction.

Conclusion: Administration of 4 ml of 2% lidocaine with epinephrine 1:100,000 or 2% lidocaine with norepinephrine 1:100,000 in the context of simple dental extraction did not strikingly increase nor decrease the blood glucose levels in patients with type II diabetes. According to the results of this study, both local anesthetics may be used safely in type II diabetics, with no preference for one over the other.

Glycemic levels by capillarity were evaluated for each patient three times on each appointment: before local anesthetic administration, 5 minutes after administration of local anesthesia, 5 minutes after dental extraction. Blood glucose levels were measured by using a glucometer (GlucoLab, Infopia Co. Ltd., Anyang, Kyunggi, Korea). Local anesthesia and dental extraction were always done by one fixed operator. The same volume of anesthetic solution (4 ml) was applied using fixed technique in both study groups. After each simple dental extraction, the difficulty of extraction was

Listing 43: Ausschnitt der Entitäten des ersten Dokuments mit wenig trennscharfen Typen

```
1. {"id": 11, "text": "GlucoLab, Infopia Co. Ltd., Anyang, Kyunggi, Korea", "type": "device" },
2. {"id": 12, "text": "4 ml of 2% lidocaine with epinephrine 1:100,000", "type": "dosage" },
3. {"id": 14, "text": "vasoconstrictor concentration of 1:100,000", "type": "dosage" }
```

7.2.2 Extraktion der kausalen Beziehungen

Die Kausalitätsextraktion hängt unmittelbar davon ab, ob der Text kausale Aussagen enthält, und spiegelt so die inhaltliche Ausrichtung des jeweiligen Dokuments wider. Das dritte Dokument³⁷ benennt zahlreiche Risikobeziehungen und verknüpft entsprechend den Gestationsdiabetes mit verschiedenen Folgeerkrankungen. Die Marker geben dabei die im Text verwendeten Formulierungen wieder und variieren zwischen erhöht, tritt eher auf und entwickelt eher, sodass der Prozess unterschiedliche sprachliche Formen derselben Grundaussage aufgreift.

Das zweite Dokument³⁶ zeigt darüber hinaus die in Abschnitt 2.3.2 beschriebene Unterscheidung zwischen expliziter und impliziter Kausalität. Neben Beziehungen mit ausdrücklichem Marker, etwa der Erhöhung hämatologischer Kennwerte durch die Eisengabe, erfasst der Prozess auch Beziehungen ohne erkennbares Signalwort und kennzeichnet diese mit einem leeren Marker, den die Ausgabe als implizit ausweist. So entstehen aus der Aufzählung der Nebenwirkungen der Eisengabe mehrere implizite Beziehungen, für die der Text kein kausales Signalwort enthält (siehe Listing 44). Die erste Beziehung im Ausschnitt verweist mit dem Marker **reduction** auf ein Signalwort,

die zweite trägt einen leeren Marker und erscheint dadurch als implizit, da der Text an dieser Stelle keine ausdrückliche kausale Formulierung enthält.

Listing 44: Explizite und implizite kausale Beziehung im zweiten Dokument

1.	2	iron supplementation	--[reduction]-->	low birth weight
2.	3	iron supplementation	--[(implicit)]-->	nausea

7.2.3 Grenzen der Kausalitätsextraktion

Das erste Dokument³⁵ macht eine grundlegende Grenze der Kausalitätsextraktion sichtbar. Als Studie mit überwiegend verneinendem Ergebnis stellt es fest, dass die untersuchten Anästhetika den Blutzucker nicht wesentlich beeinflussen, und enthält daher kaum bejahende kausale Aussagen. Der Prozess erfasst dennoch eine Beziehung, nach der Adrenalin einen Blutzuckeranstieg verursacht (siehe Listing 45). Diese Beziehung stammt aus einem Satz, der eine frühere Annahme wiedergibt, welche die Studie im weiteren Verlauf gerade widerlegt. Die Kausalitätsextraktion arbeitet damit auf der Ebene einzelner Sätze und erfasst nicht den argumentativen Zusammenhang, in dem eine Aussage steht: Sie findet, was der Text an einer Stelle formuliert, nicht, was die Publikation als Ganzes schlussfolgert.

Listing 45: Aus dem ersten Dokument extrahierte kausale Beziehung

1.	1	epinephrine	--[cause]-->	blood glucose levels
----	---	-------------	--------------	----------------------

Eine verwandte Verkürzung zeigt sich im dritten Dokument³⁷. Dort bezieht sich eine Aussage auf die Nachkommen von Müttern mit Gestationsdiabetes, während der Prozess die Beziehung unmittelbar zwischen Gestationsdiabetes und der Folgeerkrankung herstellt und das Zwischenglied auslässt. In beiden Fällen erfasst der Prozess die kausale Kernaussage korrekt auf sprachlicher Ebene, ohne einschränkende oder vermittelnde Bestandteile des Kontexts abzubilden.

7.3 Verarbeitung tabellarischer Bestandteile

Die tabellarischen Bestandteile durchlaufen zunächst die Extraktionsphase, in der das Dokumentwerkzeug die Tabellen in eine Markdown-Struktur überführt, bevor die Transformationsphase ihren Inhalt weiterverarbeitet (siehe Abschnitt 6.2). Die drei Dokumente zeigen, dass die Struktur der Tabelle die Qualität dieser Überführung bestimmt.

Die rechteckigen Tabellen der beiden letzten Dokumente überführt das Dokumentwerkzeug so, dass die inhaltliche Zuordnung erhalten bleibt. Die Empfehlungstabelle des zweiten Dokuments³⁶ mit den Empfehlungsgraden behält ihre zeilenweise Struktur vollständig, sodass jeder Grad seiner Definition eindeutig zugeordnet bleibt. Die umfangreiche Bewertungstabelle des dritten Dokuments³⁷ mit den Bewertungskriterien fasst das Dokumentwerkzeug dagegen je Bereich zusammen: Die im Original zeilenweise aufgeführten Einzelkriterien einer Domäne erscheinen gebündelt in einer Zelle, während die Zuordnung zwischen Domäne und zugehörigen Kriterien korrekt bleibt (siehe Listing 46). Die Zuordnung geht damit nicht verloren, die ursprüngliche Zeilengliederung der Einzelkriterien bleibt jedoch nicht erhalten.

Listing 46: Ausschnitt der Bewertungstabelle des dritten Dokuments (Zellinhalte gekürzt)

1.	AGREE II domain	AGREE II item
2.	-----	-----
3.	Scope and purpose	The overall objective of the ...
4.	Stakeholder involvement	The guideline development group ...
5.	Rigour of development	Systematic methods were used to ...
6.	Clarity of presentation	The recommendations are specific ...

Die Werttabelle des ersten Dokuments³⁵ zeigt dagegen die Grenze der Tabellenüberführung. Diese Tabelle stellt die Blutzuckerwerte über mehrere Messzeitpunkte und Untergruppen dar und verbindet dazu mehrere Kopfzeilen mit zusammengefassten Zellen. Das Dokumentwerkzeug erfasst zwar die enthaltenen Werte vollständig, gibt ihre Anordnung jedoch nicht durchgängig korrekt wieder, sodass einzelne Werte den zugehörigen Zeilen und Spalten nicht mehr eindeutig zuzuordnen sind (siehe Listing 47). Die zusammengefasste Kopfzeile erscheint in mehrere Zellen zerteilt, und die erste Spalte enthält leere Zellen, während die Zahlenwerte selbst vollständig vorliegen. Der Inhalt bleibt damit erhalten, die strukturelle Zuordnung geht bei stark verschachtelten Tabellen jedoch teilweise verloren.

Der Vergleich mit der Bewertungstabelle des dritten Dokuments³⁷ zeigt, dass nicht jede verschachtelte Tabelle zu einer fehlerhaften Anordnung führt. Beide Tabellen sind im Original mehrspaltig verschachtelt, das Ergebnis unterscheidet sich jedoch: Bei der Bewertungstabelle bleibt die Zuordnung trotz Zusammenfassung erhalten, während bei der Werttabelle die Zuordnung einzelner Zahlenwerte verloren geht. Ausschlaggebend ist damit weniger die Verschachtelung an sich als die Art der zusammengefassten Zellen: Eine Zusammenfassung ganzer Textblöcke bleibt eindeutig, während sich überlagernde Kopf- und Wertzellen die Anordnung stören.

Listing 47: Ausschnitt der Werttabelle des ersten Dokuments (Zellinhalte gekürzt)

1.	Study ...	Blood ...	Blood ...	Level ...	Blood ...		
2.			Baseline	Post- ...	Change ...	P-Value**	P-Value***
3.	Group A ...		164.37 ...	166.43 ...	2.06 ...	0.355	0.448
4.	Group B ...		163.03 ...	161.93 ...	-1.10 ...	0.612	0.448

Für die nachfolgende Transformationsphase wirkt sich diese Grenze unterschiedlich aus. Da die Entitäts- und Kausalitätsextraktion auf den textuellen Inhalt der Tabelle zugreift und nicht auf ihre geometrische Anordnung, gelangen die in den Tabellen enthaltenen Begriffe grundsätzlich in die Verarbeitung. Eine verlässliche Auswertung der in der Anordnung kodierten Beziehungen, etwa der Zuordnung eines Messwerts zu einer bestimmten Gruppe und einem bestimmten Zeitpunkt, ist bei verschachtelten Tabellen jedoch nicht gewährleistet.

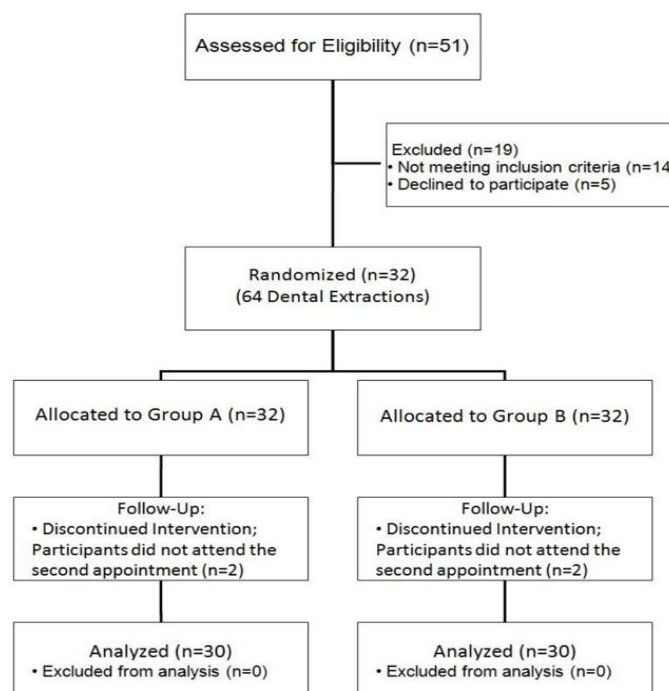
7.4 Verarbeitung bildlicher Bestandteile

Die bildlichen Bestandteile durchlaufen den in Abschnitt 6.2 beschriebenen Pfad, in dem das Dokumentwerkzeug die Abbildungen als Bilddateien herauslöst und das Vision-Language-Modell sie in textuelle Beschreibungen überführt. Die drei Dokumente enthalten je ein inhaltstragendes Diagramm und zeigen, dass die Struktur der Abbildung die Zuverlässigkeit der Beschreibung bestimmt. Zusätzlich zeigt sich eine Grenze der vorgelagerten Filterung kleinformatiger Elemente.

7.4.1 Klar strukturierte Flussdiagramme

Die beiden Flussdiagramme des ersten³⁵ und des dritten³⁷ Dokuments folgen einem klaren, überwiegend senkrechten Ablauf mit kurzen Verbindungen zwischen benachbarten Kästen. Abbildung 26 zeigt das Flussdiagramm des ersten Dokuments zur Teilnehmerverfolgung, dessen Beschreibung Listing 48 wiedergibt.

Abbildung 26: Vom Dokumentwerkzeug herausgelöstes Flussdiagramm des ersten Dokuments

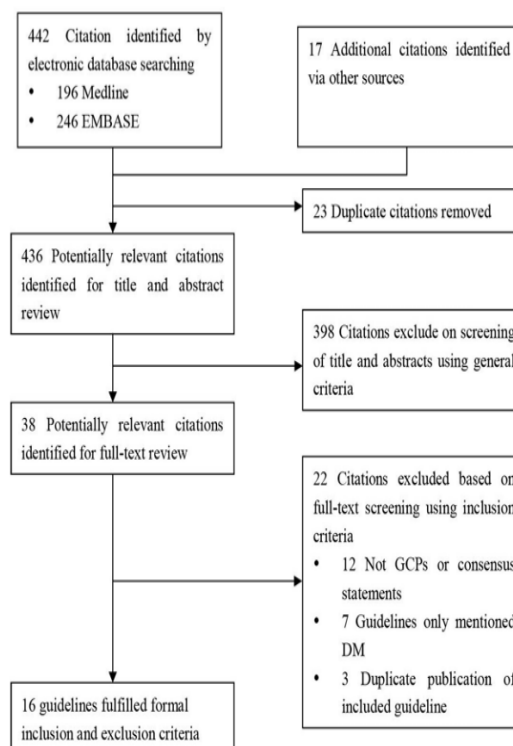


Listing 48: Ausschnitt der Bildbeschreibung des Flussdiagramms (Abbildung 26)

1. - "Assessed for Eligibility (n=51)"
2. - leads to "Excluded (n=19)": "Not meeting inclusion criteria (n=14)", "Declined to participate (n=5)".
3. - leads to "Randomized (n=32) (64 Dental Extractions)", splitting into:
4. - "Allocated to Group A (n=32)" -> Follow-up (n=2) -> "Analyzed (n=30)".
5. - "Allocated to Group B (n=32)" -> Follow-up (n=2) -> "Analyzed (n=30)".

Das Modell erfasst die Kästen, die Verzweigung in zwei Gruppen und sämtliche Fallzahlen der einzelnen Schritte vollständig und korrekt. Ebenso gibt das Modell beim Flussdiagramm des dritten Dokuments zur Literatursauswahl alle Zahlenangaben der Auswahlsschritte zutreffend wieder (siehe Abbildung 27 und Listing 49). Zugleich ergänzt das Modell eine nicht im Bild enthaltene Zwischensumme, indem es zwei Ausgangswerte addiert und die Summe im Text nennt (siehe Listing 49, Zeile 3). Diese Ergänzung ist im vorliegenden Fall zutreffend, verweist aber darauf, dass das Modell über die reine Beschreibung hinausgeht und Inhalte ableitet, die im Bild nicht unmittelbar dargestellt sind.

Abbildung 27: Vom Dokumentwerkzeug herausgelöstes Flussdiagramm des dritten Dokuments



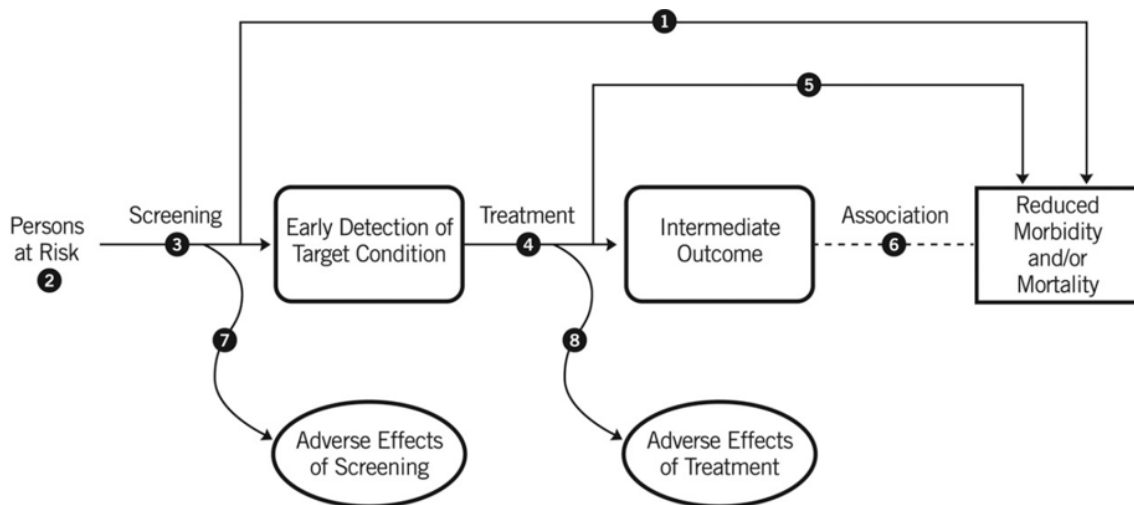
Listing 49: Ausschnitt der Bildbeschreibung des Flussdiagramms (Abbildung 27)

1. - Top-left box: "442 citation identified by electronic database searching" (with subpoints: 196 Medline, 246 EMBASE).
2. - Top-right box: "17 Additional citations identified via other sources".
3. - An arrow connects both boxes to a central box labeled "23 Duplicate citations removed", reducing the total from 459 (442 + 17) to 436.

7.4.2 Komplexes Diagramm mit abzweigenden Beziehungen

Das Fachdiagramm des zweiten Dokuments³⁶ stellt den Zusammenhang zwischen einer Screeningmaßnahme und ihren Folgen über nummerierte Beziehungen dar und enthält mehrere lange, rechtwinklig über die Breite des Bildes geführte Pfeile, die nicht an einem Kasten, sondern auf einer anderen Verbindung beginnen (siehe Abbildung 28). Hier zeigt die Bildbeschreibung ihre Grenze. Das Modell erfasst zwar die Kästen, ihre Beschriftungen und die Art der Verbindungen zutreffend, darunter die Unterscheidung zwischen durchgezogenen Pfeilen und einer gestrichelten Assoziationslinie. Bei zwei Pfeilen weicht die Beschreibung vom Bild ab (siehe Listing 50). Der lange obere Pfeil verläuft im Bild vom Screeningpfad zur verringerten Morbidität und Mortalität, während die Beschreibung ihn in umgekehrter Richtung als Rückführung von dort zur frühen Erkennung darstellt (Zeile 3); ein weiterer Pfeil beginnt im Bild am Behandlungspfad, wird in der Beschreibung jedoch der Zwischengröße zugeordnet (Zeile 1) und verläuft damit auf derselben Strecke wie die Assoziationslinie (Zeile 2).

Abbildung 28: Vom Dokumentwerkzeug herausgelöstes Fachdiagramm des zweiten Dokuments



Listing 50: Ausschnitt der Bildbeschreibung des Fachdiagramms (Abbildung 28)

1. A solid arrow (item 5) extends from "Intermediate Outcome" to "Reduced Morbidity and/or Mortality".
2. A dashed line labeled "Association" (item 6) also connects "Intermediate Outcome" to "Reduced Morbidity and/or Mortality".
3. An arrow (item 1) loops back from "Reduced Morbidity and/or Mortality" to "Early Detection of Target Condition"

Die Genauigkeit der Bildbeschreibung hängt damit von der Struktur der Abbildung ab: Klar gegliederte Flussdiagramme mit kurzen Verbindungen werden zuverlässig erfasst, während bei einem Diagramm mit langen, an anderen Verbindungen ansetzenden Pfeilen einzelne Zuordnungen fehlerhaft ausfallen. Diese Beobachtung schränkt die

textuelle Erfassung des Bildinhalts nicht grundsätzlich ein, verweist aber darauf, dass die Verlässlichkeit der Beschreibung mit der Komplexität des Diagramms abnimmt.

7.4.3 Grenze der vorgelagerten Filterung

Eine weitere Grenze zeigt sich nicht beim Modell, sondern bei der vorgelagerten Filterung kleinformatiger Grafikelemente (siehe Abschnitt 6.2). Im dritten Dokument³⁷ übersteht ein fachlich nicht einschlägiges Bedienelement in Form einer Schaltfläche die Größenfilterung und gelangt an das Vision-Language-Modell (siehe Abbildung 29). Das Modell beschreibt die Schaltfläche und verneint dabei die im Prompt erfragten Bestandteile (siehe Listing 51). Ein daneben liegendes Verlagslogo wird dagegen korrekt als kleinformatiges Element aussortiert. Die Filterung sortiert ein Element nur aus, wenn beide Kanten unter dem Schwellenwert liegen. Bei der Schaltfläche liegt allein die Höhe darunter, sodass ihre Breite sie im Ablauf hält. Über den Verbleib entscheidet damit die längere Kante – ein Maß, das mit der fachlichen Einschlägigkeit nicht zusammenhängt und sich auch durch einen anderen Schwellenwert nicht daran ausrichten lässt. So entstehen vereinzelt Beschreibungen ohne fachlichen Bezug.

Abbildung 29: Als Abbildung behandeltes Bedienelement des dritten Dokuments



Listing 51: Ausschnitt der Bildbeschreibung des Bedienelements (Abbildung 29)

1. The image displays a circular logo on the left with three curved color segments surrounding a central bookmark-shaped icon. To the right, the text "Check for updates" appears. No additional labels, axes, arrows, or relationships are visible.

7.5 Zusammenfassende Betrachtung

Die beispielhafte Anwendung auf drei Dokumente mit unterschiedlich strukturierten Bestandteilen zeigt ein durchgängig funktionsfähiges Zusammenspiel der drei Phasen: In allen drei Fällen gelangen die Inhalte des PDF-Dokuments als typisierte Entitäten und kausale Beziehungen in die relationale Datenbank. Zugleich macht die Vielfalt der Dokumente sichtbar, wovon die Qualität der einzelnen Verarbeitungsschritte abhängt. Tabelle 6 fasst die Beobachtungen je Bestandteil zusammen.

Tabelle 6: Beobachtete Stärken und Grenzen je Dokumentbestandteil

Bestandteil	Beobachtete Stärke	Beobachtete Grenze
Text – Entitäten	zusammenhängende, typisierte Entitätsmengen; dokumentspezifische Typen	wenig trennscharfe Typen; viele feingranulare Entitäten bei Aufzählungen
Text – Kausalität	explizite und implizite Beziehungen erkannt; Marker aus dem Text	Extraktion auf Satzebene ohne argumentativen Kontext
Tabellen	Zuordnung bei rechteckigen und blockverschachtelten Tabellen erhalten	Wertanordnung bei überlagernden Kopf-/Wertzellen verloren
Abbildungen	klar gegliederte Flussdiagramme fehlerfrei beschrieben	komplexe Diagramme: einzelne Pfeile falsch zugeordnet
Filterung	fachlich nicht einschlägige Kleingrafiken überwiegend aussortiert	vereinzelt gelangen solche Bedienelemente in die Beschreibung

Die Verarbeitung der textuellen Bestandteile trägt am weitesten. Der Prozess gewinnt in allen Dokumenten zusammenhängende Entitätsmengen und erkennt kausale Beziehungen einschließlich der Unterscheidung zwischen expliziter und impliziter Kausalität. Die Ergebnisse spiegeln dabei die sprachliche Form des Dokuments wider: Ein kausalitätsreiches Dokument liefert zahlreiche Beziehungen, ein Dokument mit überwiegend verneinendem Ergebnis nur wenige. Die Grenze liegt darin, dass die Extraktion auf der Ebene einzelner Sätze arbeitet und den argumentativen Zusammenhang nicht erfasst, sodass eine im Kontext widerlegte oder eingeschränkte Aussage als kausale Beziehung erscheinen kann. Die geführte offene Typisierung erweist sich als flexibel, erzeugt bei aufzählungslastigen Dokumenten jedoch viele feingranulare Entitäten und wenig trennscharfe Typen.

Die Verarbeitung der tabellarischen und bildlichen Bestandteile trägt bei klar strukturierten Vorlagen zuverlässig. Bei rechteckigen und blockweise verschachtelten Tabellen bleibt die Zuordnung zwischen Kategorie und Inhalt erhalten, und klar gegliederte Flussdiagramme werden fehlerfrei beschrieben. Die Grenzen zeigen sich an der Struktur der Vorlage und an der vorgelagerten Verarbeitung: Tabellen mit überlagernden Kopf- und Wertzellen behalten ihren Inhalt, verlieren aber die eindeutige Anordnung der Werte, ein Diagramm mit langen, an anderen Verbindungen ansetzenden Pfeilen führt zu Fehlern bei der Zuordnung einzelner Pfeile, und die Größenfilterung trennt fachlich nicht einschlägige Grafikelemente nicht in allen Fällen zuverlässig ab.

Insgesamt zeigt die Anwendung, dass der Prozess die betrachteten medizinischen Dokumente über alle drei Bestandteile hinweg verarbeitet und dabei eine strukturierte, abfragbare Repräsentation erzeugt. Die beobachteten Grenzen betreffen weniger die grundsätzliche Funktionsfähigkeit als die Zuverlässigkeit bei bestimmten Strukturen und Kontexten. Die folgende Schlussbetrachtung ordnet diese Beobachtungen in Bezug auf die Forschungsfragen ein und leitet daraus Ansätze für weiterführende Arbeiten ab (siehe Kapitel 8).

8 Fazit und Ausblick

Dieses Kapitel schließt die Arbeit ab. Zunächst fasst Abschnitt 8.1 die zentralen Ergebnisse zusammen. Anschließend beantwortet Abschnitt 8.2 die drei Forschungsfragen. Danach ordnet Abschnitt 8.3 die Grenzen des Prototyps ein, bevor Abschnitt 8.4 einen Ausblick auf weiterführende Arbeiten gibt.

8.1 Zusammenfassung

Die Arbeit hat einen lokal ausführbaren ETL-Prozess konzipiert und prototypisch umgesetzt, der multimodale medizinische PDF-Dokumente verarbeitet und dabei sowohl strukturierte Informationen als auch kausale Beziehungen extrahiert und in ein relationales Datenmodell überführt. Der wissenschaftliche Beitrag liegt in der Verbindung von multimodaler Dokumentverarbeitung und LLM-gestützter Kausalitätsextraktion innerhalb einer durchgängigen ETL-Architektur.

Ausgehend von den theoretischen Grundlagen (siehe Kapitel 2) und einem Vergleich verfügbarer Werkzeuge und lokaler Modelle (siehe Kapitel 3 und 4) entstand ein dreiphasiges Konzept (siehe Kapitel 5), das die Extraktion, die Transformation und das Laden voneinander trennt. Die Extraktionsphase überführt ein PDF-Dokument mit einem Dokumentwerkzeug und einem Vision-Language-Modell in eine einheitliche textuelle Repräsentation, in der Text, Tabellen und Abbildungen gemeinsam vorliegen. Die Transformationsphase gewinnt daraus mit einem lokalen Sprachmodell in zwei Schritten die Entitäten und die kausalen Beziehungen. Die Ladephase überführt diese Ergebnisse transaktional in eine relationale Datenbank. Die prototypische Umsetzung (siehe Kapitel 6) setzt dieses Konzept vollständig lokal um und stützt sich auf eine zentrale Schemadefinition, aus der sowohl die Vorgaben für das Sprachmodell als auch das Datenbankschema hervorgehen.

Die beispielhafte Anwendung auf drei Dokumente mit unterschiedlich strukturierten Bestandteilen (siehe Kapitel 7) hat gezeigt, dass der Prozess in allen Fällen die Inhalte des PDF-Dokuments in die relationale Datenbank überführt und dass die Qualität der einzelnen Verarbeitungsschritte von der Struktur des jeweiligen Bestandteils abhängt.

8.2 Beantwortung der Forschungsfragen

Die drei Forschungsfragen aus Abschnitt 1.1 lassen sich auf Grundlage der vorangegangenen Kapitel wie folgt beantworten:

Forschungsfrage 1: Welche Werkzeuge eignen sich für die Verarbeitung multimodaler medizinischer PDF-Dokumente?

Für die Verarbeitung multimodaler medizinischer PDF-Dokumente eignet sich die Kombination eines Dokumentwerkzeugs mit einem Vision-Language-Modell (siehe Kapitel 3 und 4). Das Dokumentwerkzeug überführt Text und Tabellen in eine strukturierte Markdown-Repräsentation und löst enthaltene Abbildungen als Bilddateien heraus, während das Vision-Language-Modell diese Abbildungen in textuelle Beschreibungen überführt (siehe Abschnitt 6.2). Auf diese Weise gelangen alle drei Bestandteile eines Dokuments in eine gemeinsame textuelle Repräsentation. Die Anwendung hat bestätigt, dass sich digital erzeugte medizinische Dokumente auf diesem Weg tragfähig verarbeiten lassen, wobei die Struktur der jeweiligen Bestandteile die Qualität der Überführung beeinflusst (siehe Abschnitte 7.3 und 7.4).

Forschungsfrage 2: Welche lokal ausführbaren LLMs eignen sich für die Informations- und Kausalitätsextraktion aus medizinischen Dokumentinhalten?

Für die Informations- und Kausalitätsextraktion eignet sich ein lokal ausführbares Sprachmodell, das die Aufgabe in zwei aufeinanderfolgenden Schritten bearbeitet: die Extraktion typisierter Entitäten und die anschließende Extraktion kausaler Beziehungen zwischen diesen Entitäten (siehe Kapitel 3 und Abschnitt 6.3). Die lokale Ausführung ohne externe Dienste entspricht dabei den Anforderungen an den Schutz medizinischer Daten. Die Anwendung hat gezeigt, dass das Modell zusammenhängende Entitätsmengen gewinnt und kausale Beziehungen erkennt, einschließlich der Unterscheidung zwischen expliziter und impliziter Kausalität (siehe Abschnitt 7.2). Die geführte offene Typisierung erlaubt dem Modell, dokumentspezifische Typen zu bilden. Die Ergebnisse spiegeln die sprachliche Form des jeweiligen Dokuments wider, sodass ein kausalitätsreiches Dokument zahlreiche Beziehungen liefert und ein Dokument mit überwiegend verneinendem Ergebnis nur wenige.

Forschungsfrage 3: Wie lassen sich die ausgewählten Werkzeuge und Modelle in einem lokal ausführbaren ETL-Prozess zusammenführen, und wie verhält sich dieser bei der Anwendung auf medizinische PDF-Dokumente mit unterschiedlichen Bestandteilen?

Die Werkzeuge und Modelle lassen sich in einem dreiphasigen ETL-Prozess zusammenführen, der die Extraktion, die Transformation und das Laden sequenziell durchläuft und jedes Dokument als abgeschlossene Einheit verarbeitet (siehe Kapitel [5](#)). Die zentrale Schemadefinition verbindet die Transformations- und die Ladephase, indem sie sowohl die Vorgaben für das Sprachmodell als auch das Datenbankschema aus einer gemeinsamen Quelle ableitet (siehe Abschnitt [6.3](#)). Bei der Anwendung auf Dokumente mit unterschiedlichen Bestandteilen verhält sich der Prozess durchgängig funktionsfähig und erzeugt in allen betrachteten Fällen eine strukturierte, abfragbare Repräsentation. Sein Verhalten hängt dabei von der Struktur der Bestandteile ab: Die textuelle Verarbeitung trägt am weitesten, während die Verarbeitung tabellarischer und bildlicher Bestandteile bei klar strukturierten Vorlagen zuverlässig ist und bei stark verschachtelten oder komplexen Vorlagen an Grenzen stößt (siehe Abschnitt [7.5](#)). Das Besondere der medizinischen Anwendung liegt dabei weniger im Ablauf des Prozesses als in seinen Randbedingungen. Da medizinische Dokumente personenbezogene Patientendaten enthalten können, ist die vollständig lokale Ausführung keine Optimierung, sondern eine Voraussetzung, die cloudgebundene Werkzeuge von vornherein ausschließt (siehe Abschnitte [5.1](#) und [5.3](#)). Zugleich verschiebt der fachliche Kontext das Gewicht der beobachteten Grenzen: Eine Werttabelle, deren Zuordnung zwischen Messwert, Gruppe und Zeitpunkt verloren geht, oder eine kausale Aussage, die eine im Text anschließend widerlegte Annahme wiedergibt, berühren unmittelbar den fachlichen Aussagegehalt des Dokuments (siehe Abschnitte [7.2.3](#) und [7.3](#)). Der Prozess erzeugt daher eine strukturierte Repräsentation, die als Ausgangspunkt einer fachlichen Prüfung dient und nicht als bereits gesicherte Aussage.

8.3 Grenzen

Der Prototyp und seine Anwendung unterliegen mehreren Grenzen, die sich aus der Konzeption, der eingesetzten Technik und der Art der betrachteten Dokumente ergeben.

Eine erste Grenze betrifft die Kausalitätsextraktion. Der Prozess erkennt kausale Aussagen auf der Ebene einzelner Sätze, erfasst jedoch nicht den argumentativen Zusammenhang, in dem eine Aussage steht. Eine im Kontext widerlegte oder eingeschränkte Aussage kann dadurch als kausale Beziehung erscheinen (siehe Abschnitt [7.2](#)). Diese Beobachtung bestätigt die bereits in der Einleitung benannte Schwierigkeit, dass Kausalität in medizinischen Texten häufig implizit oder satzübergreifend auftritt (siehe Kapitel [1](#)).

Eine zweite Grenze betrifft die extrahierten Entitäten. Der Prozess führt gleichbedeutende Bezeichnungen nicht zusammen. Die Schemadefinition sieht für eine Entität allein den Textausschnitt und den Typ vor, sodass sich eine Gleichsetzung zweier Entitäten nicht ausdrücken lässt (siehe Abschnitt [6.3.1](#)). Erscheint dieselbe Sache im Dokument einmal ausgeschrieben und einmal abgekürzt, kann sie dadurch als zwei getrennte Entitäten in der Datenbank stehen, zwischen denen keine Verbindung besteht (siehe Abschnitt [6.3.2](#)).

Eine dritte Grenze betrifft die Struktur der Bestandteile. Stark verschachtelte Tabellen behalten zwar ihren Inhalt, verlieren aber die eindeutige Anordnung der Werte, sodass die in der Anordnung kodierten Beziehungen nicht verlässlich auswertbar sind (siehe Abschnitt [7.3](#)). Bei Abbildungen ordnet das Vision-Language-Modell die Beziehungen in klar gegliederten Flussdiagrammen zuverlässig zu, während es bei einem komplexen Diagramm mit langen, an anderen Verbindungen ansetzenden Pfeilen einzelne Pfeile falsch zuordnet (siehe Abschnitt [7.4](#)). Hinzu kommt, dass die vorgelagerte Filterung kleinformatiger Grafikelemente fachlich nicht einschlägige Bedienelemente nicht in allen Fällen aussortiert (siehe Abschnitt [7.4](#)).

Eine vierte Grenze betrifft die Verarbeitung selbst. Der Verzicht auf eine Aufteilung großer Dokumente erhält zwar satzübergreifende kausale Zusammenhänge, führt aber bei umfangreichen Dokumenten auf der eingesetzten Hardware zu einer zeitintensiven Verarbeitung, sodass beim dritten Dokument³⁷ nur die ersten vier Seiten als Eingabe dienen (siehe Abschnitte [6.1](#) und [7.1](#)). Zudem setzt die Extraktion digital erzeugte Dokumente mit vorhandener Textebene voraus, da die optische Zeichenerkennung deaktiviert ist (siehe Abschnitt [6.1](#)).

Eine fünfte Grenze betrifft die Aussagekraft der Anwendung. Die Betrachtung dient der beispielhaften Untersuchung des Prozessverhaltens und nimmt keine quantitative Bewertung vor (siehe Abschnitt [7.1](#)). Die Ergebnisse beruhen auf drei Dokumenten und erlauben daher keine verallgemeinernden Aussagen über die Genauigkeit des Prozesses, sondern zeigen sein Verhalten an konkreten Beispielen.

8.4 Ausblick

Aus den benannten Grenzen ergeben sich mehrere Ansätze für weiterführende Arbeiten. Ein erster Ansatz betrifft die Kausalitätsextraktion über die Satzebene hinaus. Ein Verfahren, das den argumentativen Zusammenhang einer Aussage berücksichtigt, könnte zwischen einer behaupteten und einer im Kontext widerlegten kausalen

Beziehung unterscheiden und so die Verlässlichkeit der extrahierten Beziehungen erhöhen.

Ein zweiter Ansatz betrifft die Zusammenführung gleichbedeutender Entitäten. Ein Verfahren zur Entitätsauflösung, das eine ausgeschriebene Bezeichnung und ihre Abkürzung derselben Sache zuordnet, könnte die extrahierten Entitäten konsolidieren und die daran anknüpfenden Beziehungen an einem gemeinsamen Eintrag bündeln. Dies erforderte eine Erweiterung des Datenmodells um eine entsprechende Zuordnung.

Ein dritter Ansatz betrifft die Verarbeitung tabellarischer und bildlicher Bestandteile. Eine gezielte Behandlung verschachtelter Tabellen sowie eine verbesserte Beschreibung komplexer Diagramme könnten die in Kapitel 7 beobachteten Grenzen verringern. Für Tabellen käme dabei ein eigener Verarbeitungspfad in Betracht, der sie nicht als Fließtext an die Transformationsphase weiterreicht, sondern ihre Struktur gesondert auswertet: Aus den Kopfzeilen ließe sich ein Zielschema ableiten und die Zeilen ließen sich unmittelbar als Datensätze in die Datenbank laden. Ein solcher Pfad würde die in der Anordnung kodierten Beziehungen erhalten, die bei der textuellen Weitergabe verloren gehen, und setzte voraus, dass das Datenmodell neben Entitäten und kausalen Beziehungen auch tabellarische Datensätze aufnimmt. Ebenso könnte eine Filterung, die nicht allein die Abmessungen heranzieht, sondern den Dokumentzusammenhang eines Bildes berücksichtigt – etwa das Vorhandensein einer Bildunterschrift –, Grafikelemente ohne fachlichen Bezug zuverlässiger aussortieren.

Ein vierter Ansatz betrifft die Skalierbarkeit. Eine Untersuchung von Strategien zur Verarbeitung umfangreicher Dokumente, die satzübergreifende Zusammenhänge erhalten und zugleich den Ressourcenbedarf begrenzen, könnte die Anwendbarkeit auf vollständige Dokumente verbessern. Ergänzend könnte die Aufnahme einer optischen Zeichenerkennung die Verarbeitung auf gescannte Dokumente erweitern. Über den lokalen Einzelbetrieb hinaus stellt zudem ein serverseitiger Betrieb einen eigenen Untersuchungsgegenstand dar: Werden mehrere Dokumente aus verschiedenen Prozessen gleichzeitig verarbeitet, ist die sequenzielle Freigabe der Modelle nicht mehr ausreichend, da mehrere Vorgänge zeitgleich auf die Modelle zugreifen. Ein solcher Betrieb erfordert eine geeignete Ressourcenverwaltung, etwa das gemeinsame Vorhalten der Modelle für mehrere Anfragen oder eine Einreihung der Vorgänge in eine Warteschlange, und wäre auf einer leistungsfähigeren Serverumgebung umzusetzen.

Ein fünfter Ansatz betrifft den wiederholten Betrieb. Da der Prototyp jedes Dokument als abgeschlossene Einheit verarbeitet und keine Prüfung auf bereits geladene Dokumente vornimmt, führt ein erneuter Durchlauf zu doppelten Einträgen. Eine Duplikaterkennung, etwa über eine Eindeutigkeitsbedingung auf der Dokumentquelle oder über eine

Prüfsumme des Dateiinhalts, könnte solche Wiederholungen erkennen und den Bestand entweder unverändert lassen oder gezielt aktualisieren. Damit käme zugleich die in Abschnitt [2.1](#) beschriebene Unterscheidung zwischen vollständigem Neuladen und inkrementeller Aktualisierung in den Blick, die der Prototyp bislang nicht ausgestaltet.

Ein sechster Ansatz betrifft die Bewertung. Eine systematische Evaluation des Prozesses anhand einer Referenzannotation und geeigneter Kennzahlen könnte die in dieser Arbeit exemplarisch betrachteten Beobachtungen quantitativ absichern und die Genauigkeit der Informations- und Kausalitätsextraktion messbar machen.

Literaturverzeichnis

- [1] I. Li, J. Pan, J. Goldwasser, N. Verma, W. P. Wong, M. Y. Nuzumlali, B. Rosand, Y. Li, M. Zhang, D. Chang, R. A. Taylor, H. M. Krumholz und D. Radev, „Neural Natural Language Processing for Unstructured Data in Electronic Health Records: A Review“ *Computer Science Review*, Bd. 46, p. 100511, 2022.
- [2] M. Zubke, M. Katzensteiner und O. J. Bott, „Integration of Unstructured Data into a Clinical Data Warehouse for Kidney Transplant Screening - Challenges & Solutions“ *Digital Personalized Health and Medicine: Proceedings of MIE 2020*, Bd. 270, pp. 272-276, 2020.
- [3] G. M. Silverman, H. S. Sahoo, N. E. Ingraham, M. Lupei, M. A. Puskarich, M. Usher, J. Dries, R. L. Finzel, E. Murray, J. Sartori, G. Simon, R. Zhang, G. B. Melton, C. J. Tignanelli und S. Pakhomov, „NLP Methods for Extraction of Symptoms from Unstructured Data for Use in Prognostic COVID-19 Analytic Models“ *Journal of Artificial Intelligence Research*, Bd. 72, pp. 429-474, 2021.
- [4] T. Hiipala, *The Structure of Multimodal Documents: An Empirical Approach*, New York: Routledge, 2015.
- [5] J. Rausch, O. Martinez, F. Bissig, C. Zhang und S. Feuerriegel, „DocParser: Hierarchical Document Structure Parsing from Renderings“ *Proceedings of the AAAI Conference on Artificial Intelligence*, Bd. 35, Nr. 5, pp. 4328-4338, 2021.
- [6] D. Chen, S. A. Alnassar, K. E. Avison, R. S. Huang und S. Raman, „Large Language Model Applications for Health Information Extraction in Oncology: Scoping Review“ *JMIR Cancer*, Bd. 11, p. e65984, 2025.
- [7] M. Garg, S. Raza, S. Rayana, X. Liu und S. Sohn, „The Rise of Small Language Models in Healthcare: A Comprehensive Survey [Article 100999]“ *Computer Science Review*, Bd. 62, 2026.
- [8] A. Yu, J. Weile und M. Courtot, „Benchmarking LLM-based Information Extraction Tools for Medical Documents“ 2026, doi: 10.64898/2026.01.19.26344287.
- [9] A. Akkasi und M.-F. Moens, „Causal Relationship Extraction from Biomedical Text using Deep Neural Models: A Comprehensive Survey“ *Journal of Biomedical Informatics*, Bd. 119, p. 103820, 2021.
- [10] I. Kunle-John, J. Paul, O. Agbaakin, P. Aina, I. Odezuligbo und S. Anuyah, „PubMedCausal: A Span-Level Annotated Corpus for Causal Relation Extraction in Biomedical Text“ arXiv, arXiv:2605.28363, 2026.
- [11] S. Gopalakrishnan, L. Garbayo und W. Zadrozny, „Causality Extraction from Medical Text using Large Language Models (LLMs)“ *Information*, Bd. 16, Nr. 1, p. 13, 2025.
- [12] P. Vassiliadis und A. Simitsis, „Extraction, Transformation, and Loading“ in *Encyclopedia of Database Systems*, Springer, 2009, pp. 1095-1101.
- [13] S. H. A. El-Sappagh, A. M. A. Hendawi und A. H. El Bastawissy, „A Proposed Model for Data Warehouse ETL Processes“ *Journal of King Saud University - Computer and Information Sciences*, Bd. 23, Nr. 2, pp. 91-104, 2011.
- [14] U. Dayal, M. Castellanos, A. Simitsis und K. Wilkinson, „Data Integration Flows for Business Intelligence“ in *Proceedings of the 12th International Conference on Extending Database Technology: Advances in Database Technology*, 2009.
- [15] A. R. Post, M. Ai, A. Kalsanka Pai, M. Overcash und D. S. Stephens, „Architecting the Data Loading Process for an i2b2 Research Data Warehouse: Full Reload versus Incremental Updating“ *AMIA Annual Symposium Proceedings*, Bd. 2017, pp. 1411-1420, 2017.
- [16] M. Macura, „Integration of Data from Heterogeneous Sources using ETL Technology“ *Computer Science*, Bd. 15, Nr. 2, pp. 109-132, 2014.
- [17] A. Doan, J. F. Naughton, R. Ramakrishnan, A. Baid, X. Chai, F. Chen, T. Chen, E. Chu, P. DeRose, B. Gao, C. Gokhale, J. Huang, W. Shen und B.-Q. Vuong, „Information Extraction Challenges in Managing Unstructured Data“ *SIGMOD Record*, Bd. 37, Nr. 4, pp. 14-20, 2009.

- [18] J. Song, Y. Bao und J. Shi, „A Triggering and Scheduling Approach for ETL in a Real-time Data Warehouse“ *2010 10th IEEE International Conference on Computer and Information Technology*, pp. 91-98, 2010.
- [19] K. Adnan und R. Akbar, „Limitations of Information Extraction Methods and Techniques for Heterogeneous Unstructured Big Data“ *International Journal of Engineering Business Management*, Bd. 11, pp. 1-23, 2019.
- [20] A. Bauer und H. Günzel, *Data-Warehouse-Systeme*, Heidelberg: dpunkt, 2013.
- [21] N. Ladas, F. Borchert, S. Franz, A. Rehberg, N. Strauch, K. K. Sommer, M. Marschollek und M. Gietzelt, „Programming Techniques for Improving Rule Readability for Rule-Based Information Extraction Natural Language Processing Pipelines of Unstructured and Semi-Structured Medical Texts“ *Health Informatics Journal*, Bd. 29, Nr. 2, pp. 1-14, 2023.
- [22] H. Bongertmann, B. Nast, L. Griesch, H. Rotzoll und K. Sandkuhl, „Large Language Models for Structuring and Integration of Heterogeneous Data“ in *Empowering the Interdisciplinary Role of ISD in Addressing Contemporary Issues in Digital Transformation: How Data Science and Generative AI Contributes to ISD (ISD2025 Proceedings)*, Belgrade, Serbia, 2025.
- [23] M. Suzuki und K. Yamaguchi, „Recognition of E-Born PDF Including Mathematical Formulas“ in *Computers Helping People with Special Needs*, Cham, Springer International Publishing, 2016, pp. 35-42.
- [24] H. Cheng, P. Zhang, S. Wu, J. Zhang, Q. Zhu, Z. Xie, J. Li, K. Ding und L. Jin, „M6Doc: A Large-Scale Multi-Format, Multi-Type, Multi-Layout, Multi-Language, Multi-Annotation Category Dataset for Modern Document Layout Analysis“ *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15138-15147, 2023.
- [25] Z. Wang, Y. Xu, L. Cui, J. Shang und F. Wei, „LayoutReader: Pre-training of Text and Layout for Reading Order Detection“ in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, 2021, pp. 4735-4744.
- [26] N. Livathinos, C. Auer, M. Lysak, A. Nassar, M. Dolfi, P. Vagenas, C. Berrospi Ramis, M. Omenetti, K. Dinkla, Y. Kim, S. Gupta, R. Teixeira de Lima, V. Weber, L. Morin und I. Meijer, „Docling: An Efficient Open-Source Toolkit for AI-driven Document Conversion“ *arXiv*, arXiv:2501.17887, 2025.
- [27] S.-F. Sung, Y.-H. Hu und C.-Y. Chen, „Disambiguating Clinical Abbreviations by One-to-All Classification: Algorithm Development and Validation Study“ *JMIR Medical Informatics*, Bd. 12, p. e56955, 2024.
- [28] H. Goodrum, K. Roberts und E. V. Bernstam, „Automatic Classification of Scanned Electronic Health Record Documents“ *International Journal of Medical Informatics*, Bd. 144, p. 104302, 2020.
- [29] F. Adamo, F. Attivissimo, A. Di Nisio und M. Spadavecchia, „An Automatic Document Processing System for Medical Data Extraction“ *Measurement*, Bd. 61, pp. 88-99, 2015.
- [30] A. Neustein, S. S. Imambi, M. Rodrigues, A. Teixeira und L. Ferreira, „Application of Text Mining to Biomedical Knowledge Extraction: Analyzing Clinical Narratives and Medical Literature“ in *Text Mining of Web-Based Medical Content*, Berlin, München, Boston, De Gruyter, 2014, pp. 3-32.
- [31] H. Xu, P. D. Stetson und C. Friedman, „A Study of Abbreviations in Clinical Notes“ *AMIA Annual Symposium Proceedings*, pp. 821-825, 2007.
- [32] A. McCallum, „Information Extraction: Distilling Structured Data from Unstructured Text“ *ACM Queue*, Bd. 3, pp. 48-57, 2005.
- [33] J. Jiang, „Information Extraction from Text“ in *Mining Text Data*, Boston, MA, Springer US, 2012, pp. 11-41.
- [34] M. H. A. Abdullah, N. Aziz, S. J. Abdulkadir, H. S. A. Alhussian und N. Talpur, „Systematic Literature Review of Information Extraction from Textual Data: Recent Methods, Applications, Trends, and Challenges“ *IEEE Access*, Bd. 11, pp. 10535-10562, 2023.
- [35] Y. Yang, Z. Wu, Y. Yang, S. Lian, F. Guo und Z. Wang, „A Survey of Information Extraction Based on Deep Learning“ *Applied Sciences*, Bd. 12, Nr. 19, p. 9691, 2022.
- [36] J. L. Martinez-Rodriguez, A. Hogan und I. Lopez-Arevalo, „Information Extraction Meets the Semantic Web: A Survey“ *Semantic Web*, Bd. 11, Nr. 2, pp. 255-335, 2020.

- [37] J. Yang, S. C. Han und J. Poon, „A survey on extraction of causal relations from natural language text“ *Knowledge and Information Systems*, Bd. 64, Nr. 5, pp. 1161-1186, 2022.
- [38] V. Khetan, M. I. Rizvi, J. Huber, P. Bartusiak, B. Sacaleanu und A. Fano, „MIMICause: Representation and Automatic Extraction of Causal Relation Types from Clinical Notes“ *Findings of the Association for Computational Linguistics: ACL 2022*, pp. 764-773, 2022.
- [39] C. Mihăilă, T. Ohta, S. Pyysalo und S. Ananiadou, „BioCause: Annotating and Analysing Causality in the Biomedical Domain“ *BMC Bioinformatics*, Bd. 14, Nr. 1, p. 2, 2013.
- [40] M. A. Kabir, A. Almulhim, X. Luo und M. Al Hasan, „Informative Causality Extraction from Medical Literature via Dependency-Tree-Based Patterns“ *Journal of Healthcare Informatics Research*, Bd. 6, Nr. 3, pp. 295-316, 2022.
- [41] Y. Miura, E. Aramaki, T. Ohkuma, M. Tonoike, D. Sugihara, H. Masuichi und K. Ohe, „Adverse-Effect Relations Extraction from Massive Clinical Records“ in *Proceedings of the Second Workshop on NLP Challenges in the Information Explosion Era (NLPiX 2010)*, Beijing, China, Coling 2010 Organizing Committee, 2010, pp. 75-83.
- [42] S. Minaee, T. Mikolov, N. Nikzad, M. Chenaghlu, R. Socher, X. Amatriain und J. Gao, „Large Language Models: A Survey“ *arXiv*, arXiv:2402.06196, 2025.
- [43] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser und I. Polosukhin, „Attention is All You Need“ *Advances in Neural Information Processing Systems*, Bd. 30, 2017.
- [44] P. Sahoo, A. K. Singh, S. Saha, V. Jain, S. Mondal und A. Chadha, „A Systematic Survey of Prompt Engineering in Large Language Models: Techniques and Applications“ 2025.
- [45] L. Qin, Q. Chen, X. Feng, Y. Wu, Y. Zhang, Y. Li, M. Li, W. Che und P. S. Yu, „Large Language Models Meet NLP: A Survey“ *Frontiers of Computer Science*, Bd. 20, Nr. 11, p. 2011361, 2026.
- [46] S. Nerella, S. Bandyopadhyay, J. Zhang, M. Contreras, S. Siegel, A. Bumin, B. Silva, J. Sena, B. Shickel, A. Bihorac, K. Khezeli und P. Rashidi, „Transformers and Large Language Models in Healthcare: A Review“ *Artificial Intelligence in Medicine*, Bd. 154, p. 102900, 2024.
- [47] A. J. Thirunavukarasu, D. S. J. Ting, K. Elangovan, L. Gutierrez, T. F. Tan und D. S. W. Ting, „Large Language Models in Medicine“ *Nature Medicine*, Bd. 29, Nr. 8, pp. 1930-1940, 2023.
- [48] S. Bedi, Y. Liu, L. Orr-Ewing, D. Dash, S. Koyejo, A. Callahan, J. A. Fries, M. Wornow, A. Swaminathan, L. S. Lehmann, H. J. Hong, M. Kashyap, A. R. Chaurasia, N. R. Shah, K. Singh und T. Tazbaz, „Testing and Evaluation of Health Care Applications of Large Language Models: A Systematic Review“ *JAMA*, Bd. 333, Nr. 4, pp. 319-328, 2025.
- [49] L. Huang, W. Yu, W. Ma, W. Zhong, Z. Feng, H. Wang, Q. Chen, W. Peng, X. Feng, B. Qin und T. Liu, „A Survey on Hallucination in Large Language Models: Principles, Taxonomy, Challenges, and Open Questions“ *ACM Transactions on Information Systems*, Bd. 43, Nr. 2, p. 42, 2025.
- [50] L. Chen, R. He, P. Lu, Y. Jin, L. Zhou, N. Li, P. Wu und B. Hu, „Operationalizing Large Language Models for Clinical Research Data Extraction: Methods, Quality Control, and Governance“ *Journal of Medical Systems*, Bd. 50, Nr. 1, p. 25, 2026.
- [51] I. Villanueva Miranda, Z. Gu, D. Yang, K. Nezafati, J. Huang, P. Ruan, X. Zhan, G. Xiao und Y. Xie, „ClinBench: A Standardized Multi-Domain Framework for Evaluating Large Language Models in Clinical Information Extraction“ *Advances in Neural Information Processing Systems*, Bd. 38, 2025.
- [52] S. Tenckhoff, M. Koddenbrock und E. Rodner, „LLMStructBench: Benchmarking Large Language Model Structured Data Extraction“ *arXiv*, arXiv:2602.14743, 2026.
- [53] J. B. Balasubramanian, D. Adams, I. Roxanis, A. B. de Gonzalez, P. Coulson, J. S. Almeida und M. García-Closas, „Leveraging Large Language Models for Structured Information Extraction from Pathology Reports“ *Journal of Pathology Informatics*, Bd. 19, p. 100521, 2025.
- [54] N. S. Adhikari und S. Agarwal, „A Comparative Study of PDF Parsing Tools Across Diverse Document Categories“ *arXiv*, arXiv:2410.09871, 2025.

- [55] Z. Guo und H. Jin, „A Rule-based Framework of Metadata Extraction from Scientific Papers“ in *2011 10th International Symposium on Distributed Computing and Applications to Business, Engineering and Science*, 2011.
- [56] D. D. A. Bui, G. Del Fiol und S. Jonnalagadda, „PDF Text Classification to Leverage Information Extraction from Publication Reports“ *Journal of Biomedical Informatics*, p. 141–148, 2016.
- [57] M. S. U. Miah, J. Sulaiman, T. B. Sarwar, A. Naseer, F. Ashraf, K. Z. Zamli und R. Jose, „Sentence Boundary Extraction from Scientific Literature of Electric Double Layer Capacitor Domain: Tools and Techniques“ *Applied Sciences*, Bd. 12, Nr. 1352, 2022.
- [58] A. Ospan, K. Auyesbay, T. Sarsembayeva und A. Mussa, „Application of Rule-Based Method for Automatic Extraction of Tags from Column-Style PDF-Documents“ *Journal of Problems in Computer Science and Information Technologies*, Bd. 3, Nr. 3, p. 52–67, 2025.
- [59] US Preventive Services Task Force, „Screening for Gestational Diabetes: US Preventive Services Task Force Recommendation Statement“ *JAMA*, Bd. 326, Nr. 6, pp. 531-538, 2021.
- [60] M. Zhu und J. M. Cole, „PDFDataExtractor: A Tool for Reading Scientific Text and Interpreting Metadata from the Typeset Literature in the Portable Document Format“ *Journal of Chemical Information and Modeling*, Bd. 62, pp. 1633-1643, 2022.
- [61] W. Yang, F. Cao und X. Zhao, „Extraction of PDF Table Data Based on the Pdfplumber Method“ in *Proceedings of the 2024 4th International Joint Conference on Robotics and Artificial Intelligence (JCRAI 2024)*, 2024.
- [62] M. A. Suryani, S. Hahne, C. Beth, K. Wallmann und M. Renz, „DAF: Data Acquisition Framework to Support Information Extraction from Scientific Publications“ in *Proceedings of the 15th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2023) - Volume 1: KDIR*, 2023.
- [63] C. Cui, T. Sun, M. Lin, T. Gao, Y. Zhang, J. Liu, X. Wang, Z. Zhang, C. Zhou, H. Liu, Y. Zhang, W. Lv, K. Huang, Y. Zhang, J. Zhang, J. Zhang, Y. Liu, D. Yu und Y. Ma, „PaddleOCR 3.0 Technical Report“ arXiv, arXiv:2507.05595, 2025.
- [64] Z. Shen, R. Zhang, M. Dell, B. C. G. Lee, J. Carlson und W. Li, „LayoutParser: A Unified Toolkit for Deep Learning Based Document Image Analysis“ arXiv, arXiv:2103.15348, 2021.
- [65] Z. Shen, K. Lo, L. L. Wang, B. Kuehl, D. S. Weld und D. Downey, „VILA: Improving Structured Content Extraction from Scientific PDFs Using Visual Layout Groups“ *Transactions of the Association for Computational Linguistics*, Bd. 10, pp. 376-392, 2022.
- [66] Y. Dubey, V. Wath, Y. Vyawahare, V. Deogade, G. Loya, A. Pratape und D. Chhatani, „An Integrated OCR-Based Assistive System for Visually Impaired Individuals with Enhanced Accessibility“ *Journal of Machine Engineering*, Bd. 25, Nr. 3, pp. 97-113, 2025.
- [67] C. Li, R. Guo, J. Zhou, M. An, Y. Du, L. Zhu, Y. Liu, X. Hu und D. Yu, „PP-StructureV2: A Stronger Document Analysis System“ arXiv, arXiv:2210.05391, 2022.
- [68] J. T. Moreira-Filho, D. Ranganath, R. S. Tieghi, R. Patton, V. Sutherland, C. Schmitt, A. A. Rooney, J. Fostel, V. R. Walker, T. Saddler, D. Reif, K. Mansouri und N. Kleinstreuer, „Automating Data Extraction From Scientific Literature and General PDF Files Using Large Language Models and KNIME: An Application in Toxicology“ *WIREs Computational Molecular Science*, Bd. 15, Nr. 5, p. e70047, 2025.
- [69] V. Rezanezhad, K. Baierer, M. Gerber, K. Labusch und C. Neudecker, „Document Layout Analysis with Deep Learning and Heuristics“ in *Proceedings of the 7th International Workshop on Historical Document Imaging and Processing*, 2023.
- [70] B. Wang, C. Xu, X. Zhao, L. Ouyang, F. Wu, Z. Zhao, R. Xu, K. Liu, Y. Qu, F. Shang, B. Zhang, L. Wei, Z. Sui, W. Li, B. Shi, Y. Qiao, D. Lin und C. He, „MinerU: An Open-Source Solution for Precise Document Content Extraction“ arXiv, arXiv:2409.18839, 2024.
- [71] B. Petracchi, High Performance Computing for Human Sensible Applications [Ph.D. dissertation], Università degli Studi di Pavia, 2025.
- [72] B. Zhang, S. G. Acosta, P. Carlson, S. Bron, P.-L. Doucet, D. B. Ospina und S. Suo, „ParseBench: A Document Parsing Benchmark for AI Agents“ arXiv, arXiv:2604.08538, 2026.

- [73] C. Auer, M. Lysak, A. Nassar, M. Dolfi, N. Livathinos, P. Vagenas, C. Berrospi Ramis, M. Omenetti, F. Lindlbauer, K. Dinkla, L. Mishra, Y. Kim, S. Gupta, R. Teixeira de Lima und V. Weber, „Docling Technical Report“ arXiv, arXiv:2408.09869, 2024.
- [74] C. Duan, „Accelerating End-to-End PDF to Markdown Conversion Through Assisted Generation“ in *Natural Language Processing and Information Systems*, Springer Nature Switzerland, 2025, pp. 34-48.
- [75] J. Pillay, L. Donovan, S. Guitard, B. Zakher, M. Gates, A. Gates, B. Vandermeer, C. Bougatsos, R. Chou und L. Hartling, „Screening for Gestational Diabetes: Updated Evidence Report and Systematic Review for the US Preventive Services Task Force“ *JAMA*, Bd. 326, Nr. 6, pp. 539-562, 2021.
- [76] Q. Zhang, B. Wang, V. S.-J. Huang, J. Zhang, Z. Wang, H. Liang, C. He und W. Zhang, „Document Parsing Unveiled: Techniques, Challenges, and Prospects for Structured Information Extraction“ arXiv, arXiv:2410.21169, 2026.
- [77] X. Shi, W. Zhao, T. Chen, C. Yang und J. Du, „A Large Language Model Approach to Extracting Causal Evidence across Study Designs for Evidence Triangulation“ medRxiv, 2024.
- [78] A. Grattafiori, A. Dubey, A. Jauhri, A. Pandey, A. Kadian und A. Al-Dahle, „The Llama 3 Herd of Models“ arXiv, arXiv:2407.21783, 2024.
- [79] J. Wihl, E. Rosenkranz, S. Schramm, C. Berberich, M. Griessmair, P. Woźnicki, F. Pinto, S. Ziegelmayer, L. C. Adams, K. K. Bressem, J. S. Kirschke und C. Zimmer, „Data Extraction from Free-Text Stroke CT Reports using GPT-4o and Llama-3.3-70B: The Impact of Annotation Guidelines“ *European Radiology Experimental*, Bd. 9, Nr. 1, p. 61, 2025.
- [80] P. Prucker, K. K. Bressem, S. H. Kim, D. Weller, A. Kader, F. J. Dorfner und S. Ziegelmayer, „Privacy-Preserving Generation of Structured Lymphoma Progression Reports from Cross-sectional Imaging: A Comparative Analysis of Llama 3.3 and Llama 4“ *Journal of Imaging Informatics in Medicine*, Bd. 39, pp. 1868-1878, 2025.
- [81] Z. Deng, W. Ma, Q.-L. Han, W. Zhou, X. Zhu, S. Wen und Y. Xiang, „Exploring DeepSeek: A Survey on Advances, Applications, Challenges and Future Directions“ *IEEE/CAA Journal of Automatica Sinica*, Bd. 12, Nr. 5, pp. 872-893, 2025.
- [82] DeepSeek-AI, „DeepSeek-V3.2: Pushing the Frontier of Open Large Language Models“ arXiv, arXiv:2512.02556, 2025.
- [83] DeepSeek-AI, „DeepSeek-V3 Technical Report“ arXiv, arXiv:2412.19437, 2025.
- [84] M. Tordjman, Z. Liu, M. Yuce, S. Hong, C. Harder, P. Dansoko, C. Silcox, P. A. Testa, J. Cheng, T. Savage, P. Kim, F. Stanford, K. Wang, M. Cruz und P. Hager, „Comparative Benchmarking of the DeepSeek Large Language Model on Medical Tasks and Clinical Reasoning“ *Nature Medicine*, Bd. 31, pp. 2550-2555, 2025.
- [85] B. Moëll, F. Sand Aronsson und S. Akbar, „Medical Reasoning in LLMs: An In-Depth Analysis of DeepSeek R1“ *Frontiers in Artificial Intelligence*, Bd. 8, 2025.
- [86] W. Jiang, D. Wang, Y. Zeng, J. Huang, C. Xu und C. Liu, „Promoting Responsible DeepSeek Deployment in Health Care: Scoping Review Comparing Grey and White Literature“ *Journal of Medical Internet Research*, Bd. 27, p. e80770, 2025.
- [87] H. Lu, W. Liu, B. Zhang, B. Wang, K. Dong, B. Liu, J. Sun, T. Ren, Z. Li, H. Yang, Y. Sun, C. Deng, H. Xu, Z. Xie und C. Ruan, „DeepSeek-VL: Towards Real-World Vision-Language Understanding“ arXiv, arXiv:2403.05525, 2024.
- [88] OpenAI, „gpt-oss-120b & gpt-oss-20b Model Card“ arXiv, arXiv:2508.10925, 2025.
- [89] A. Munim, J. Ma, O. Ibrahim, A. Abdalla, S. Yin, L. Chen und B. Wang, „Benchmarking and Adapting On-Device LLMs for Clinical Decision Support“ arXiv, arXiv:2601.03266, 2026.
- [90] H. Tanioka, „Towards Safe and Trustworthy Healthcare AI: Risk Assessment of Medical Dialogue Using LLMs“ *Human-Centric Intelligent Systems*, 2025.
- [91] K. Zhang, Z. Yuan, C. Peng, S. Zhao, M. Lyu, Z. Chen, Y. Ye, W. Liu, Y. Zhang, K. E. Smith, L. He, L. Sun und Y. Wu, „MedGPT-oss: Training a General-Purpose Vision-Language Model for Biomedicine“ arXiv, arXiv:2603.00842, 2026.
- [92] D. Kumar, D. Yadav und Y. Patel, „GPT-OSS-20B: A Comprehensive Deployment-Centric Analysis of OpenAI's Open-Weight Mixture of Experts Model“ arXiv, arXiv:2508.16700, 2025.

- [93] J. Bai, S. Bai, Y. Chu, Z. Cui, K. Dang, X. Deng, Y. Fan, W. Ge, Y. Han, F. Huang, B. Hui, L. Ji, M. Li und J. Lin, „Qwen Technical Report“ arXiv, arXiv:2309.16609, 2023.
- [94] S. Zhu, W. Hu, Z. Yang, J. Yan und F. Zhang, „Qwen-2.5 Outperforms Other Large Language Models in the Chinese National Nursing Licensing Examination: Retrospective Cross-Sectional Comparative Study“ *JMIR Medical Informatics*, Bd. 13, p. e63731, 2025.
- [95] V. B. Ulli und A. Mondal, „MedQwen-PE: Medical Qwen for Parameter-Efficient Multilingual Patient-Centric Summarization, Question Answering and Information Extraction“ in *NLP-AI4Health*, 2025.
- [96] Y. Yu, C. Shi, J. Tang und S. Zheng, „Qwen-VL2 Model with NEFTune Technique For Medical Report Generation“ in *2025 4th International Symposium on Computer Applications and Information Technology (ISCAIT)*, 2025.
- [97] S. Bartels und J. Carus, „From Text to Data: Open-Source Large Language Models in Extracting Cancer Related Medical Attributes from German Pathology Reports“ *International Journal of Medical Informatics*, Bd. 203, p. 106022, 2025.
- [98] É. Joseph, P. Vallee, T. Perennec, N. Wagneur, J.-S. Frenel, M. Campone, F. Bocquet und F. Le Borgne, „Development and Assessment of a Pipeline for Extracting Structured Data From Free-Text Medical Reports Using a Large Language Model“ *JCO Clinical Cancer Informatics*, Bd. 10, p. e2500133, 2026.
- [99] M. Li, J. Huang, J. Yeung, A. Blaes, S. Johnson, H. Liu, H. Xu und R. Zhang, „CancerLLM: A Large Language Model in Cancer Domain“ *npj Digital Medicine*, Bd. 9, Nr. 1, p. 266, 2026.
- [100] A. Meddeb, S. Lüken, F. Busch, L. Adams, L. Ugga, E. Koltsakis, A. Tzortzakakis, S. Jelassi, I. Dkhil, M. E. Klontzas und M. Triantafyllou, „Large Language Model Ability to Translate CT and MRI Free-Text Radiology Reports Into Multiple Languages“ *Radiology*, Bd. 313, Nr. 3, p. e241736, 2024.

Abbildungsverzeichnis

Abbildung 1: Grundlegende Architektur eines ETL-Prozesses (in Anlehnung an [20]).....	6
Abbildung 2: Spezialisierter ETL-Prozess für unstrukturierte Daten (in Anlehnung an [20]).....	7
Abbildung 3: Beispiel: Verarbeitung eines PDF-Dokuments zu strukturierten Daten (eigene Darstellung)	10
Abbildung 4: Schematischer Aufbau eines medizinischen Laborberichts (eigene Darstellung)	11
Abbildung 5: Beispielseite eines PDF-Dokuments	22
Abbildung 6: Mit PyMuPDF extrahierte Textblöcke auf der Beispielseite aus Abbildung 5.....	23
Abbildung 7: Mit pdfminer.six erkannte textuelle Layoutbereiche auf der Beispielseite aus Abbildung 5... ..	24
Abbildung 8: Mit pdfplumber extrahierte Texteinheiten auf der Beispielseite aus Abbildung 5	26
Abbildung 9: Mit docTR erkannte Textbereiche auf der Beispielseite aus Abbildung 5.....	28
Abbildung 10: Mit PaddleOCR erkannte Textbereiche auf der Beispielseite aus Abbildung 5.....	30
Abbildung 11: Mit Surya erkannte Textbereiche auf der Beispielseite aus Abbildung 5.....	31
Abbildung 12: Mit LayoutParser erkannte Layoutbereiche auf der Beispielseite aus Abbildung 5.....	33
Abbildung 13: Mit LlamaParse erkannte Dokumentbereiche auf der Beispielseite aus Abbildung 5	35
Abbildung 14: Mit LiteParse extrahierte Textbereiche auf der Beispielseite aus Abbildung 5.....	37
Abbildung 15: Mit Docling extrahierte Text- und Strukturbereiche auf der Beispielseite aus Abbildung 5 .	39
Abbildung 16: Mit MinerU extrahierte Dokumentbereiche auf der Beispielseite aus Abbildung 5.....	42
Abbildung 17: Mit Marker extrahierte Dokumentbereiche auf der Beispielseite aus Abbildung 5	43
Abbildung 18: Beispielabbildung eines PDF-Dokuments	67
Abbildung 19: Gesamtarchitektur des lokalen ETL-Prozesses.....	73
Abbildung 20: Einordnung des Dateneingangs nach dem Extraktionszeitpunkt (in Anlehnung an [20])	77
Abbildung 21: Ablauf der Extraktionsphase mit Text-, Tabellen- und Bildverarbeitung	79
Abbildung 22: Beispielhaftes relationales Zieldatenmodell der Ladephase.....	82
Abbildung 23: Analyserahmen bei Skalierung 0,5 (237 Pixel Breite)	91
Abbildung 24: Sequenzdiagramm des Gesamtablaufs.....	102
Abbildung 25: Belegstellen der Entitäten aus Listing 43 im ersten Dokument	105
Abbildung 26: Vom Dokumentwerkzeug herausgelöstes Flussdiagramm des ersten Dokuments	108
Abbildung 27: Vom Dokumentwerkzeug herausgelöstes Flussdiagramm des dritten Dokuments	109
Abbildung 28: Vom Dokumentwerkzeug herausgelöstes Fachdiagramm des zweiten Dokuments	110
Abbildung 29: Als Abbildung behandeltes Bedienelement des dritten Dokuments	111

Tabellenverzeichnis

Tabelle 1: Allgemeiner Vergleich der KI-gestützten Werkzeuge	45
Tabelle 2: Unterstützte Dokumentelemente der KI-gestützten Werkzeuge	46
Tabelle 3: Vergleich der betrachteten lokalen LLM-Familien.....	54
Tabelle 4: Zeichenumfang der Markdown- und JSON-Ausgaben je Werkzeug	66
Tabelle 5: Behandlung der Abbildung je Werkzeug.....	69
Tabelle 6: Beobachtete Stärken und Grenzen je Dokumentbestandteil	112

Listing-Verzeichnis

Listing 1: Ausschnitt der LlamaParse-JSON-Struktur.....	58
Listing 2: Ausschnitt der LiteParse-JSON-Struktur.....	59
Listing 3: Ausschnitt der Docling-JSON-Struktur.....	59
Listing 4: Ausschnitt der MinerU-JSON-Struktur	60
Listing 5: Ausschnitt der Marker-JSON-Struktur.....	60
Listing 6: LlamaParse, Tabellenabbildung.....	61
Listing 7: LiteParse, Texteinheiten statt Tabellenobjekt	61
Listing 8: Docling, Tabellenzellen	62
Listing 9: MinerU, Tabellenblock mit HTML	62
Listing 10: Marker, Markdown-Tabelle	62
Listing 11: LiteParse, positionsbezogene Texteinheit.....	63
Listing 12: Docling, Bounding Box mit Koordinatenursprung.....	63
Listing 13: MinerU, Seitenmaß und Inline-Equation	64
Listing 14: LlamaParse, Markdown mit eingebetteter HTML-Tabelle (Ausschnitt)	64
Listing 15: Docling, native Markdown-Tabelle	65
Listing 16: MinerU, Markdown mit eingebetteter HTML-Tabelle.....	65
Listing 17: Marker, Markdown-Tabelle mit ausgelagertem „I“.....	65
Listing 18: LiteParse, Textausgabe ohne Tabellenstruktur.....	65
Listing 19: LlamaParse, Abbildung als Mermaid-Graph (JSON und Markdown identisch)	68
Listing 20: Docling, picture-Element im JSON und Markdown mit Platzhalter und Bildreferenz.....	68
Listing 21: LiteParse, Abbildung als ungeordneter OCR-Text (JSON und Markdown gleich)	68
Listing 22: Marker, herausgelöste Bilddatei im JSON-images-Objekt.....	69
Listing 23: MinerU, typisierter image-Block mit Pfad, bbox und Caption	69
Listing 24: Beispielhafter Prompt für die Informationsextraktion.....	80
Listing 25: Beispielhafter Prompt für die Kausalitätsextraktion.....	81
Listing 26: Verwendete Python-Pakete mit Versionsständen	86
Listing 27: Docling-Konfiguration und Markdown-Export mit Bildreferenzen	88
Listing 28: Aufruf des Vision-Language-Modells zur Bildbeschreibung (Ausschnitt)	88
Listing 29: Ersetzen der Bildreferenzen durch die textuellen Beschreibungen	89
Listing 30: Ausschnitt der Markdown-Repräsentation vor und nach der Ersetzung der Bildreferenz	90
Listing 31: Beschreibungen derselben Abbildung bei unterschiedlicher Auflösung (gekürzt).....	91

Listing 32: Zentrale Schemadefinition der Zieltabellen des Prototyps	93
Listing 33: Ableitung von JSON-Schema und Datenbankschema aus der Definition	93
Listing 34: Aus der Definition erzeugtes JSON-Schema für den ersten Schritt	94
Listing 35: Modellaufruf mit Systemprompt und abgeleitetem Schema (Ausschnitt)	95
Listing 36: Nachgiebige Auswertung der Modellausgabe.....	95
Listing 37: Aufbereitung der Entitäten als Referenzliste für den zweiten Schritt.....	96
Listing 38: Ausschnitt der beiden Ausgaben der Transformationsphase	97
Listing 39: Aus der Definition erzeugte Anweisungen zur Erstellung der Zieltabellen	98
Listing 40: Transaktionales Laden mit aktiver Fremdschlüsselprüfung	99
Listing 41: Übersetzung der Bezeichner und Auflösung der Verweise	99
Listing 42: Steuerskript: Aufruf der Phasen und Abbruch bei einem Fehler	100
Listing 43: Ausschnitt der Entitäten des ersten Dokuments mit wenig trennscharfen Typen.....	105
Listing 44: Explizite und implizite kausale Beziehung im zweiten Dokument	106
Listing 45: Aus dem ersten Dokument extrahierte kausale Beziehung.....	106
Listing 46: Ausschnitt der Bewertungstabelle des dritten Dokuments (Zellinhalte gekürzt).....	107
Listing 47: Ausschnitt der Werttabelle des ersten Dokuments (Zellinhalte gekürzt)	107
Listing 48: Ausschnitt der Bildbeschreibung des Flussdiagramms (Abbildung 26)	109
Listing 49: Ausschnitt der Bildbeschreibung des Flussdiagramms (Abbildung 27)	109
Listing 50: Ausschnitt der Bildbeschreibung des Fachdiagramms (Abbildung 28).....	110
Listing 51: Ausschnitt der Bildbeschreibung des Bedienelements (Abbildung 29).....	111

Selbständigkeitserklärung

Hiermit versichere ich, die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie die Zitate deutlich kenntlich gemacht zu haben. Ich erkläre weiterhin, dass die vorliegende Arbeit in gleicher oder ähnlicher Form noch nicht im Rahmen eines anderen Prüfungsverfahrens eingereicht wurde.

Zur sprachlichen und grammatikalischen Überarbeitung der Arbeit wurde das KI-gestützte Sprachmodell ChatGPT (OpenAI, Version GPT-5.5) eingesetzt. Die Nutzung beschränkte sich ausschließlich auf die Verbesserung von Sprache, Grammatik, Rechtschreibung und Formulierungen. Sämtliche fachlichen Inhalte, wissenschaftlichen Argumentationen, Analysen, Schlussfolgerungen sowie die Auswahl, Auswertung und Interpretation der verwendeten Quellen wurden eigenständig erarbeitet. Es wurden keine inhaltlichen Beiträge, Ideen, wissenschaftlichen Bewertungen oder Ergebnisse der KI übernommen.

Rostock, den 03.08.2026

Carla Kalaf

Carla Kalaf