Reinking, Ernst; Becker, Marco

Working Paper

Large Language Modelle und unternehmenseigene Daten - Genauere Abfrageergebnisse dank effizientem Datenmanagement und verbesserten technischen Verfahren

IUCF Working Paper, No. 3/2024

Ernst Reinking, Marco Becker[1]

Large Language Modelle und unternehmenseigene Daten

Genauere Abfrageergebnisse dank effizientem Datenmanagement
und verbesserten technischen Verfahren

Zusammenfassung

Retrieval-Augmented Generation (RAG) verbindet auf synergetische Weise das intrinsische Wissen von LLMs mit den riesigen, dynamischen Datenbeständen von Unternehmen. Aufbauend auf dem Grundkonzept einer RAG („Naive RAG“) identifiziert dieses Working Paper kritische Faktoren dieser hochaktuellen Architektur und gibt Hinweise zur Verbesserung. Abschließend werden zukünftige Wege für Forschung und Entwicklung aufgezeigt.

Abstract

Retrieval-Augmented Generation (RAG) synergistically combines the intrinsic knowledge of LLMs with the huge, dynamic databases of companies. Referencing the basic concept of a RAG ("Naive RAG"), this working paper identifies the critical factors of this cutting-edge architecture and gives hints for improvement. Finally, future paths for research and development are outlined.

[1] **Dipl.-Ing. Ernst Reinking** lehrt und forscht als Research Fellow an der NBS Northern Business School – University auf Applied Science in Hamburg zu Themenbereichen Wirtschaftsinformatik, digitale Ökonomie und Künstliche Intelligenz.
Prof. Dr. Marco Becker lehrt und forscht zu den Themen Controlling und Finanzmanagement an der NBS Northern Business School – University auf Applied Science in Hamburg und ist stellvertretender Leiter des Instituts für Unternehmensrechnung, Controlling und Finanzmanagement (IUCF). Darüber hinaus ist er Gründer und Partner der Marco Becker Management Consultants.

Retrieval Augmented Generation

Large Language Modelle (LLMs) haben bewiesen, dass sie ein beeindruckendes Potenzial besitzen, um Arbeitsprozesse in Unternehmensumgebungen grundlegend zu verändern. Indem sie natürliche Sprachabfragen verstehen können, versprechen LLMs den Mitarbeitern im gesamten Unternehmen völlig neue Möglichkeiten: Sie ermöglichen ihnen den Zugriff auf Daten, deren Analyse und die Extraktion von wertvollen Informationen daraus. LLMs wurden mit sehr umfangreichem „allgemeinem Wissen“ trainiert, sie sind also nicht in der Lage Fragen zu dem Domänenwissen eines Unternehmens zu beantworten. Darüber hinaus stehen sie vor Herausforderungen wie Halluzinationen, veraltetem Wissen und nicht transparenten, nicht nachvollziehbaren Argumentationsprozessen.[2]

Retrieval-Augmented Generation (RAG) hat sich als vielversprechende Lösung herausgestellt, indem Wissen aus externen Datenbanken einbezogen wird. Dies erhöht die Genauigkeit und Glaubwürdigkeit der Modelle, insbesondere bei wissensintensiven Aufgaben, und ermöglicht die kontinuierliche Aktualisierung des Wissens und die Integration von Unternehmensspezifischen Informationen.

Die Idee hinter RAG ist recht einfach und funktioniert in drei Schritten:

1. Finde den relevantesten Textabschnitt eines externen Dokuments

2. Rufen ihn ab

3. Füge ihn in die ursprüngliche Eingabeaufforderung des LLM ein, so dass der LLM auf diese Referenztextabschnitte zugreifen und ihn zur Generierung einer Antwort verwenden kann.

Während das LLM umfassende Kompetenzen zum Verständnis natürlicher Sprache, Übersetzung und Argumentation mitbringt, liefert der Retriever wichtige Fakten und Kontext aus dem Unternehmen.

Zusammengefasst bietet RAG eine Möglichkeit, Unternehmens-LLMs mit sich schnell entwickelndem Unternehmenswissen über Nachrichten, interne Systeme, Marktdaten und mehr zu verbinden. Anstatt sich ausschließlich auf das statische Training des LLM zu verlassen, hält der Abrufmechanismus seine Antworten auf den aktuellen Informationsbedarf zugeschnitten. Diese Fähigkeit, aktuelles Kontextwissen zu nutzen, wird RAG in Zukunft wahrscheinlich zu einer unverzichtbaren Fähigkeit für den effektiven Einsatz von LLMs in Unternehmensumgebungen machen.

[2] Ein populäres Beispiel für ein Large Language Model ist ChatGPT.

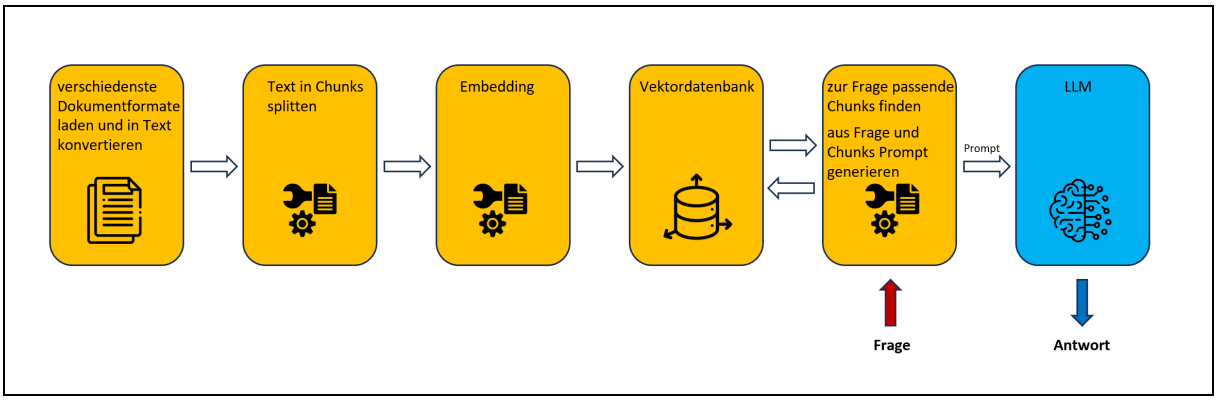


Abbildung 1: Beispielhafter Aufbau eines vorgeschalteten Modells[3]

Die wesentlichen Komponenten des RAG-Modells werden im Folgenden kurz beschrieben:[4]

1. Dokumente laden und konvertieren:
   In einem ersten Schritt müssen die unterschiedlichen Dokumentformate mit ihren unstrukturierten Daten eingelesen werden. Sie reichen von .pdf über Textverarbeitungsformate (.doc, .docx, .odt, .txt, .rtf, .md, ...), Mailformate (.html, .eml, ...), Webtexten bis hin zu Formaten, die mit OCR-Texterkennung verarbeitet werden müssen. Die Textinhalte müssen extrahiert, in reines ASCII-Format umgewandelt und von allen Steuerzeichen befreit werden.

2. Texte splitten:
   Der Text eines eingelesenen Dokuments wird in Teile, sogenannte Chunks, zerlegt, die je nach Einstellung einige hundert bis einige tausend Zeichen umfassen. Die Aufteilung erfolgt so, dass sich die Chunks um eine ebenfalls vorgegebene Anzahl von Zeichen überlappen. Damit soll erreicht werden, dass bei der technisch notwendigen Aufteilung nicht die inhaltliche Kontinuität bzw. der Sinn verloren geht oder verfälscht wird.[5]

3. Embedding:
   Beim Embedding geht es allgemein darum, die Texte der Chunks mit anderen Texten oder Chunks vergleichbar zu machen. Dies geschieht durch die Umwandlung der Texte in jeweils eindeutige numerische Vektoren. Dazu werden sogenannte Embedding-Modelle verwendet, es existiert z.B. ein von OpenAI veröffentlichtes Embedding-Modell, das die Vektoren für einen 1536-dimensionalen Raum generiert.[6]

---

[3] Reinking/Becker (2023a): S. 9 und Reinking/Becker (2023b): S.9.
[4] Leicht modifiziert entnommen aus:
Reinking/Becker (2023a): S. 9f. und Reinking/Becker (2023b): S.9f..
[5] Vgl. Lee (2023).
[6] Vgl. Lee (2023).

4. Vektordatenbank:
   Eine Vektordatenbank ist ein spezieller Typ von Datenbank, der dazu verwendet wird, Vektordaten zu speichern, zu organisieren und abzufragen. Eine häufige Anwendung von Vektordatenbanken ist die Ähnlichkeitssuche. Man kann eine Abfrage mit einem gegebenen Vektor starten und die Vektoren in der Datenbank finden, die diesem am ähnlichsten sind. Dies wird durch verschiedene Ähnlichkeits- oder Distanzmetriken wie Kosinus-Ähnlichkeit oder euklidische Distanz erreicht.

5. Relevante Chunks finden und Prompt generieren:
   In diesem Schritt wird die Anfrage des Benutzers entgegengenommen und nach vorheriger Aufbereitung eine Ähnlichkeitssuche gegen die in der Vektordatenbank gespeicherten Chunks gestartet. Das Ergebnis, der oder die zur Anfrage passenden Chunks, werden wiederum in Text umgewandelt und als Content zusammen mit der Anfrage wie oben beschrieben zu einem Prompt für das LLM zusammengefasst.[7]

Dokumentenmanagement

In der heutigen Zeit fallen in Unternehmen Unmengen an Daten an, die in unzähligen Dokumenten – teilweise inhaltsähnlich oder sogar redundant gespeichert werden. Die Organisation dieser Datenflut ist i. d. R. manuell nicht zu bewältigen, sodass in der Praxis der Einsatz von Dokumentenmanagement-Systemen unausweichlich ist. Die zentralen Aufgaben eines Dokumentenmanagement-Systems sind in der folgenden Infografik kurz zusammengefasst:

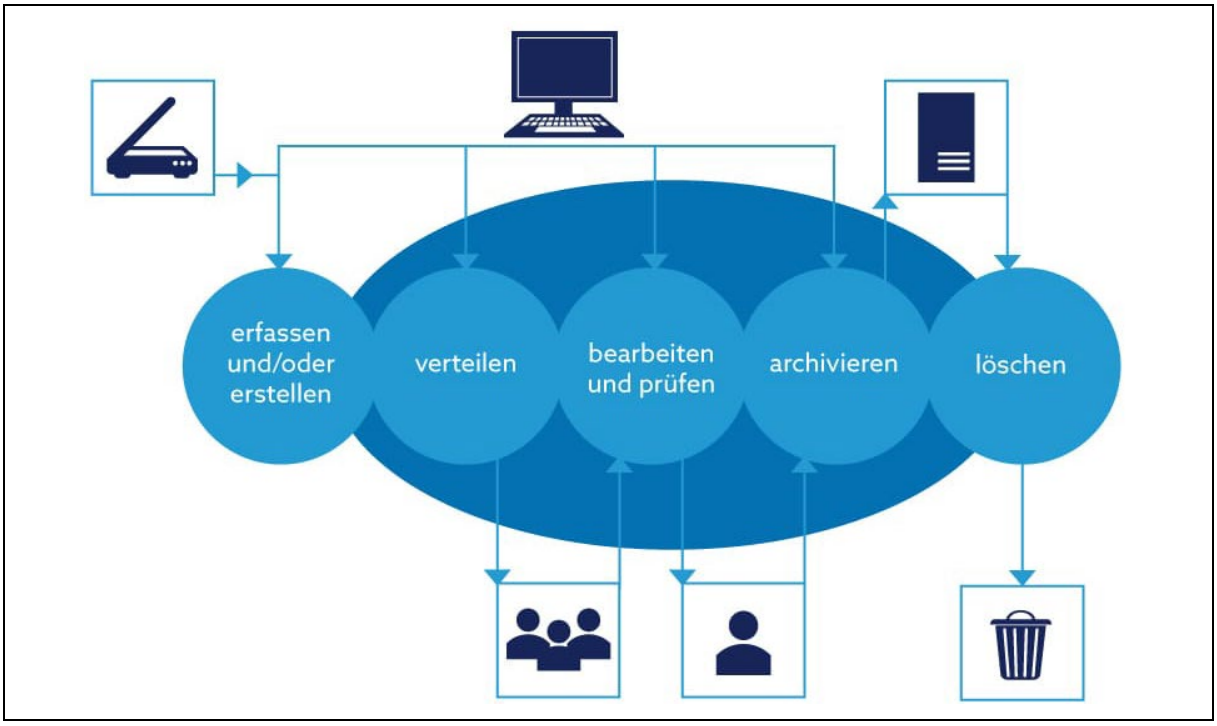


Abbildung 2: Aufgaben des Dokumentenmanagements[8]

[7] Vgl. Lee (2023).
[8] Sevdesk (2024).

Aus der konsequenten Nutzung eines Dokumentenmanagement-Systems im Unternehmen entsteht ein erheblicher Nutzen für das Unternehmen, wie aus der folgenden Grafik ersichtlich ist:

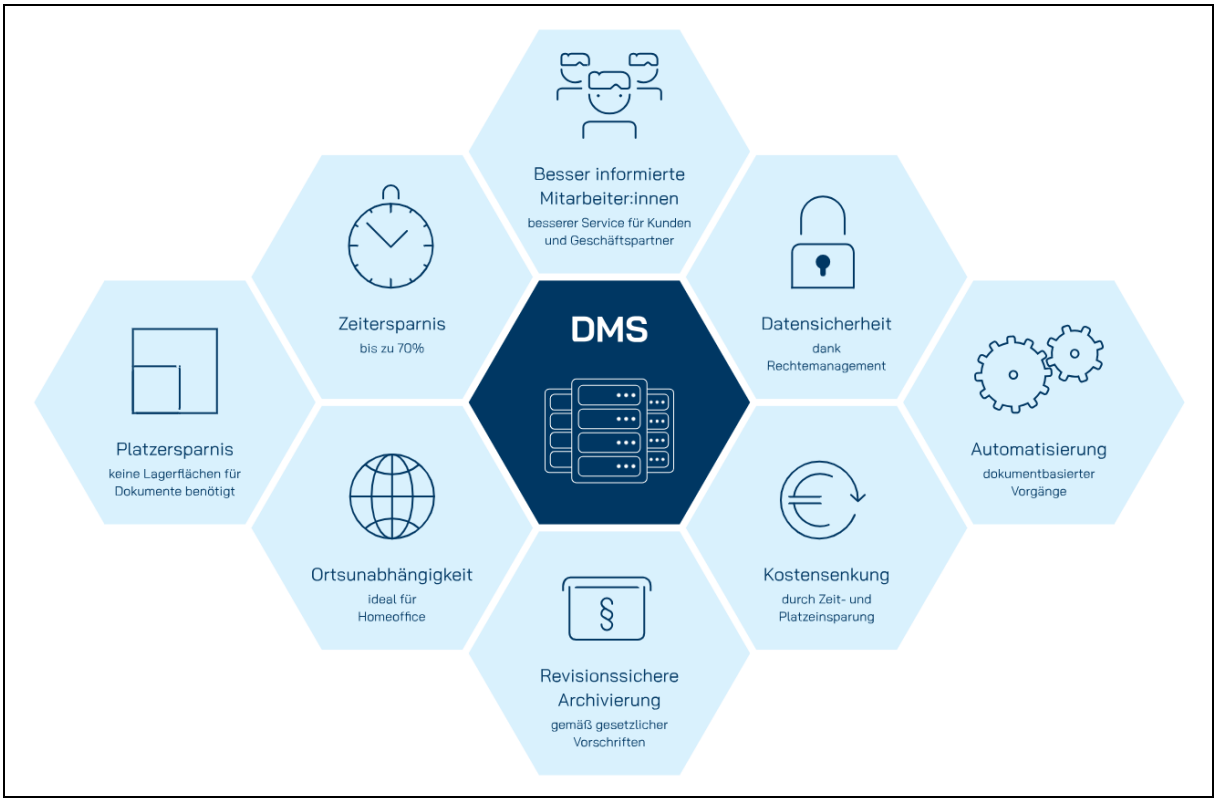


Abbildung 3: Notwendigkeit von Dokumentenmanagement[9]

Redundanzen

Eine Redundanz liegt vor, wenn dieselben Daten mehrfach in einem Dokumentenmanagement-System gespeichert werden. Identische Daten sind somit in mehreren Dokumenten vorhanden oder Dokumente sind mehrfach vorhanden, im schlimmsten Fall ohne, dass es je bemerkt wird. Eine zentrale Aufgabe von Dokumentenmanagement-Systemen ist u.a. die Vermeidung von Redundanzen. Neben einem vermeidbaren zusätzlichen Speicherbedarf für das Archivieren der redundanten Daten führen Redundanzen in der Praxis häufig zu Inkonsistenzen in den Daten. Dieser Effekt ist um so ausgeprägter, je höher die Änderungshäufigkeit der zu Grunde liegenden Daten ist. Es gilt somit Redundanzen so weit wie möglich zu vermeiden.

Metadaten

Metadaten sind strukturierte Informationen. Sie beschreiben, erklären, lokalisieren, oder helfen dabei, es sonst wie einfacher zu machen, eine Informationsquelle abzurufen, zu verwenden, oder zu verwalten. Metadaten werden oft Daten zu bestimmten Daten oder Informationen zu bestimmten Informationen genannt. Metadaten können sich dabei auf alle Bestandteile eines Dokuments beziehen.[10]

[9] Eigene Darstellung in Anlehnung an: Optimal Systems GmbH (2024).
[10] Vgl. Europäische Union (2012): S. 5.

Zur Steuerung innerhalb des Dokumentenlebenszyklus werden Meta-Daten eingesetzt. Metadaten beschreiben und ggf. ergänzen die inhaltlichen Daten.

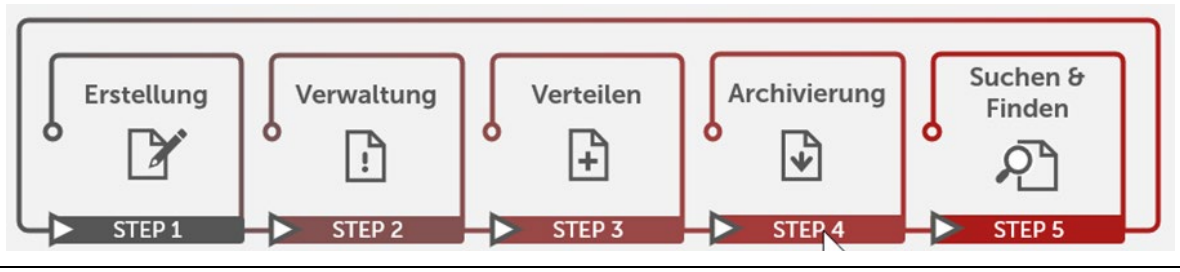
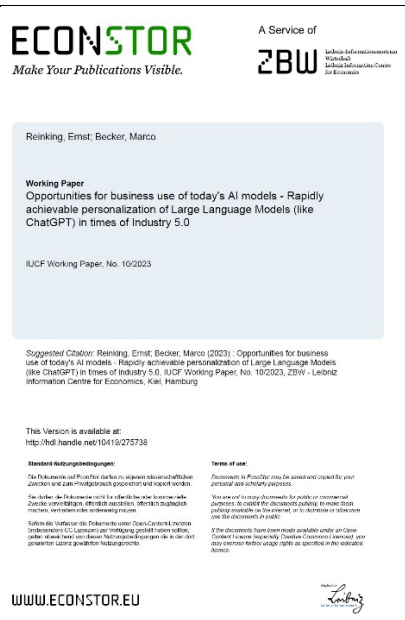


Abbildung 4: Beispielhafter Dokumentenlebenszyklus[11]

Metadaten sind auch ein zentraler Bestandteil im Kontext von RAG.[12]

Textextraktion

Für RAG ist die Extraktion von Informationen aus Dokumenten ein unumgängliches Szenario. Die Sicherstellung der Effizienz bei der Extraktion von Inhalten aus der Quelle ist entscheidend für die Verbesserung der Qualität des Endprodukts. Dieser Prozess darf nicht unterschätzt werden. Bei der Implementierung von RAG kann eine schlechte Informationsextraktion während des Parsing-Prozesses zu einem eingeschränkten Verständnis und einer eingeschränkten Nutzung der in den PDF-Dateien enthaltenen Informationen führen.


ECONSTOR
Make Your Publications Visible.

A Service of
ZBW

Reinking, Ernst; Becker, Marco

Working Paper
Opportunities for business use of today's AI models - Rapidly achievable personalization of Large Language Models (like ChatGPT) in times of Industry 5.0

IUCF Working Paper, No. 10/2023

Suggested Citation: Reinking, Ernst; Becker, Marco (2023) : Opportunities for business use of today's AI models - Rapidly achievable personalization of Large Language Models (like ChatGPT) in times of Industry 5.0, IUCF Working Paper, No. 10/2023, ZBW - Leibniz Information Centre for Economics, Kiel, Hamburg

This Version is available at:
http://hdl.handle.net/10419/275738

Standard-Nutzungsbedingungen:

Terms of use:

WWW.ECONSTOR.EU

PDF-Datei

Reinking, Ernst; Becker, Marco Working Paper Opportunities for business use of today's AI models - Rapidly achievable personalization of Large Language Models (like ChatGPT) in times of Industry 5.0 IUCF Working Paper, No. 10/2023 Suggested Citation: Reinking, Ernst; Becker, Marco (2023) : Opportunities for business use of today's AI models - Rapidly achievable personalization of Large Language Models (like ChatGPT) in times of Industry 5.0, IUCF Working Paper, No. 10/2023, ZBW - Leibniz Information Centre for Economics, Kiel, Hamburg This Version is available at: http://hdl.handle.net/10419/275738 Standard-Nutzungsbedingungen: Die Dokumente auf EconStor dürfen zu eigenen wissenschaftlichen Zwecken und zum Privatgebrauch gespeichert und kopiert werden. Sie dürfen die Dokumente nicht für öffentliche oder kommerzielle Zwecke vervielfältigen, öffentlich ausstellen, öffentlich zugänglich machen, vertreiben oder anderweitig nutzen. Sofern die Verfasser die Dokumente unter Open-Content-Lizenzen (insbesondere CC-Lizenzen) zur Verfügung gestellt haben sollten, gelten abweichend von diesen Nutzungsbedingungen die in der dort genannten Lizenz gewährten Nutzungsrechte. Terms of use: Documents in EconStor may be saved and copied for your personal and scholarly purposes. You are not to copy documents for public or commercial purposes, to exhibit the documents publicly, to make them publicly available on the internet, or to distribute or otherwise use the documents in public. If the documents have been made available under an Open Content Licence (especially Creative Commons Licences), you may exercise further usage rights as specified in the indicated licence.

ASCII-Text

Abbildung 5: Beispiel für Textextraktion[13]

[11] Eigene Darstellung in Anlehnung an: IPI GmbH (2024).
[12] Vgl. Mathur (2024).
[13] Eigene Darstellung.

In diesem Beispiel wurden nur die im PDF eingebetteten Textelemente extrahiert. Die als Bild eingebetteten Texte (z.B. das Logo von ECONSTOR) müssten zusätzlich per OCR-Erkennung ausgelesen und als ASCI-Text bereitgestellt werden. Dies wurde im Beispiel **nicht** durchgeführt.

Im Laufe der Jahre wurden zudem verschiedene Variante von PDF-Dokumenten für unterschiedliche Anwendungszwecke entwickelt. Alle basieren auf der ISO 3200, haben aber bestimmte – für den jeweiligen Anwendungszweck – angepasste Eigenschaften. Die folgende Grafik gibt einen Überblick über die verschiedenen Varianten:[14]

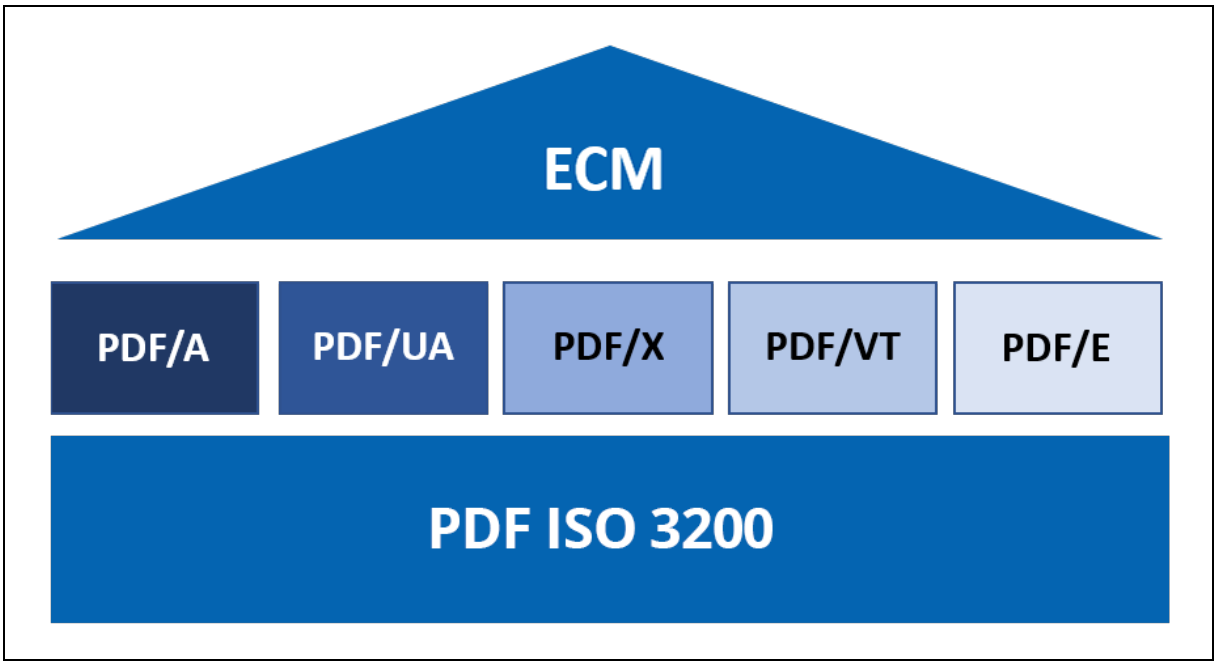


Abbildung 6: Varianten von PDF-Dokumenten[15]

Darüber hinaus gibt es jedes dieser PDF-Formate jeweils in einer textbasierten und einer bildbasierten Variant. Ob eine PDF-Datei in einer textbasierten oder bildbasierten Version vorliegt, hängt in erster Linie von der Methode ab, mit der die PDF-Dateien erzeugt wurde. Die wesentlichen Unterschiede zwischen textbasierten und bildbasierten PDF-Dateien sind in der folgenden Tabelle zusammengefasst:[16]

[14] Vgl. Foxit Software Inc. (2024).
[15] Eigene Darstellung in Anlehnung an: Foxit Software Inc. (2024).
[16] Vgl. ABBYY Europe GmbH (2024).

Textbasierte PDFs	Bildbasierte PDFs
Digital erzeugte PDFs	Gescannte PDFs
Texte und Bilder können im PDF-Dokument einzelnd angewählt werden.	Texte und Bilder können im PDF-Dokument nicht ausgewählt werden. (Text kann nur mit einer Texterkennungssoftware (OCR) ausgelesen werden)
Speicherbedarf des PDF-Dokuments ist eher gering	Speicherbedarf des PDF-Dokuments ist grundsätzlich groß

Abbildung 7: Textbasierte vs. bildbasierte PDFs[17]

Innerhalb einer PDF-Datei werden die verschiedenen Elemente (wie Text, Tabellen, Bilder etc.) auf unterschiedlichen Ebenen gespeichert. Diese werden als Layer bezeichnet und enthalten neben den jeweiligen Datenelements (Text, Tabellen, Bilder etc.) insbesondere Metadaten, die die Anordnung des jeweiligen Datenelements auf der Seite beschreiben:[18]

- Encoded Characters: Eine Folge von Bytes, welche die eigentlichen Zeichen repräsentieren
- Font Data: Eine Gruppe von Glyphen, die für die graphische Visualisierung einzelner Zeichen zuständig sind
- Einer Tabelle, die die kodierten Zeichen mit den Glyphen verbindet
- Stream Objects: Beinhalten Daten für die Darstellung von größeren Inhalten

Die vollständige Seite entsteht somit durch das Übereinanderlegen der einzelnen Layer mit den Datenelementen, wie die folgende Abbildung veranschaulichen soll:[19]

[17] Eigene Darstellung in Anlehnung an Miro Engineering (2024).
[18] Vgl. ABBYY Europe GmbH (2024) und Lehenmeier/Lohmüller (2012): S. 1.
[19] Vgl. ABBYY Europe GmbH (2024).

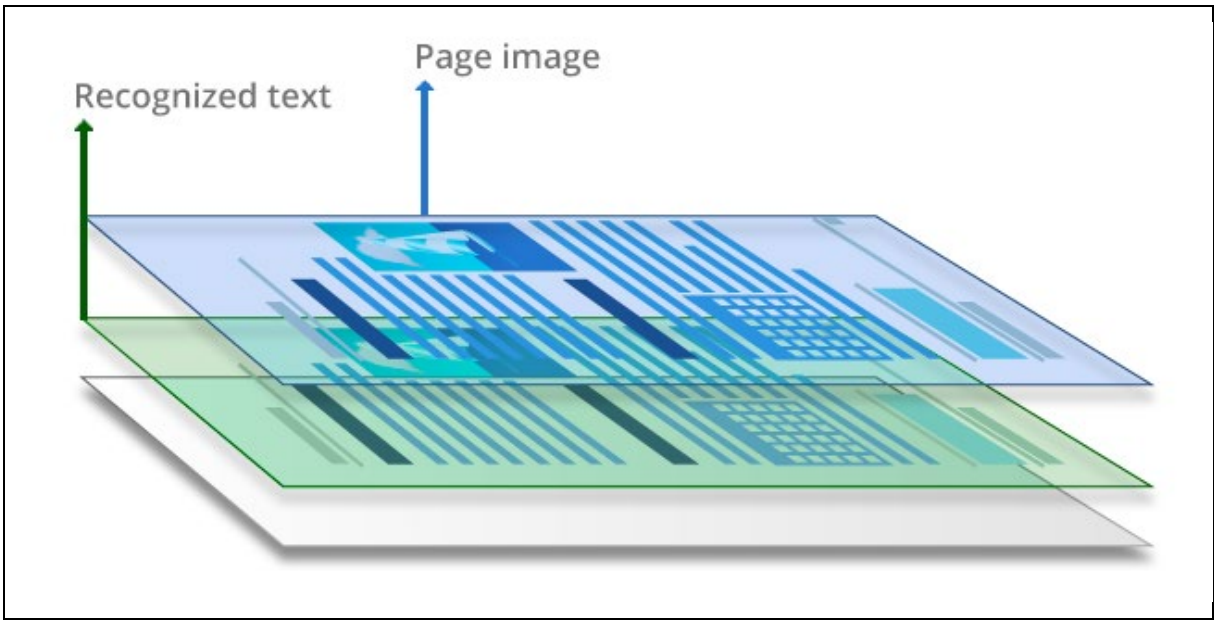


Abbildung 8: kombinierte Layer-Architektur in PDFs[20]

Soll nun der Text aus einem PDF-Dokument extrahiert werden, um es beispielsweise in einem LLM weiterzuverarbeiten, so müssen in einem ersten Schritt zunächst die Layer wieder voneinander getrennt werden, um sie einzeln zu untersuchen. In jedem Layer kann sich Text befinden, wobei dieser im Idealfall direkt ausgelesen werden kann; im komplizierteren Fall aber erst aus Tabellen oder Grafiken extrahiert werden muss:[21]

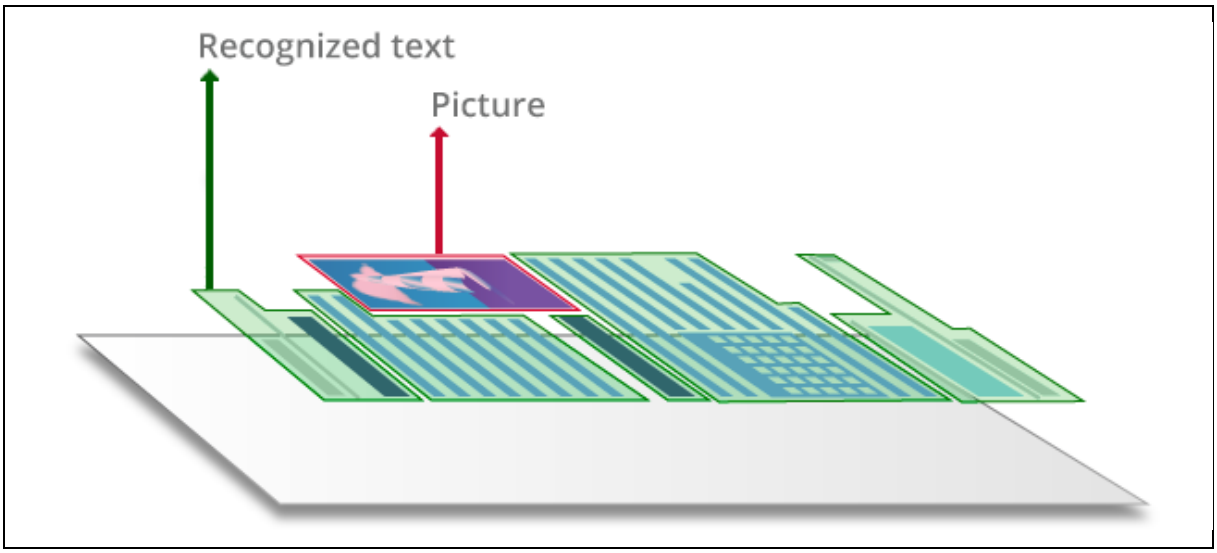


Abbildung 9: Separierte Layer-Architektur in PDFs[22]

„Zwar ist es möglich, Text aus PDF-Dateien zu extrahieren, allerdings wird man dabei vor einige Probleme gestellt:

[20] ABBYY Europe GmbH (2024).
[21] Vgl. Adobe Software Systems Ireland Limited (2024).
[22] ABBYY Europe GmbH (2024).

- Gruppierung von Texten verläuft nicht nach inhaltlichen Aspekten
- Konvertierung von Schriften in das gewünschte Ausgabeformat
- Parsen von Tabellen, Kopf- und Fußzeilen sowie Grafiken und Bildern

[…]

Eine Lösung, um das Parsen von PDF-Dokumente zu optimieren wäre, das Dokument zuerst in kleinere logische Elemente zu segmentieren. Je nachdem wie die einzelnen Elemente untereinander positioniert sind, werden sie zu logischen Gruppen verknüpft. Das heißt, der Inhalt wird nicht mehr nach dessen graphischen Anordnung im PDF-Dokument, sondern nach dessen semantischen Zusammenhängen geparst.“ [23]

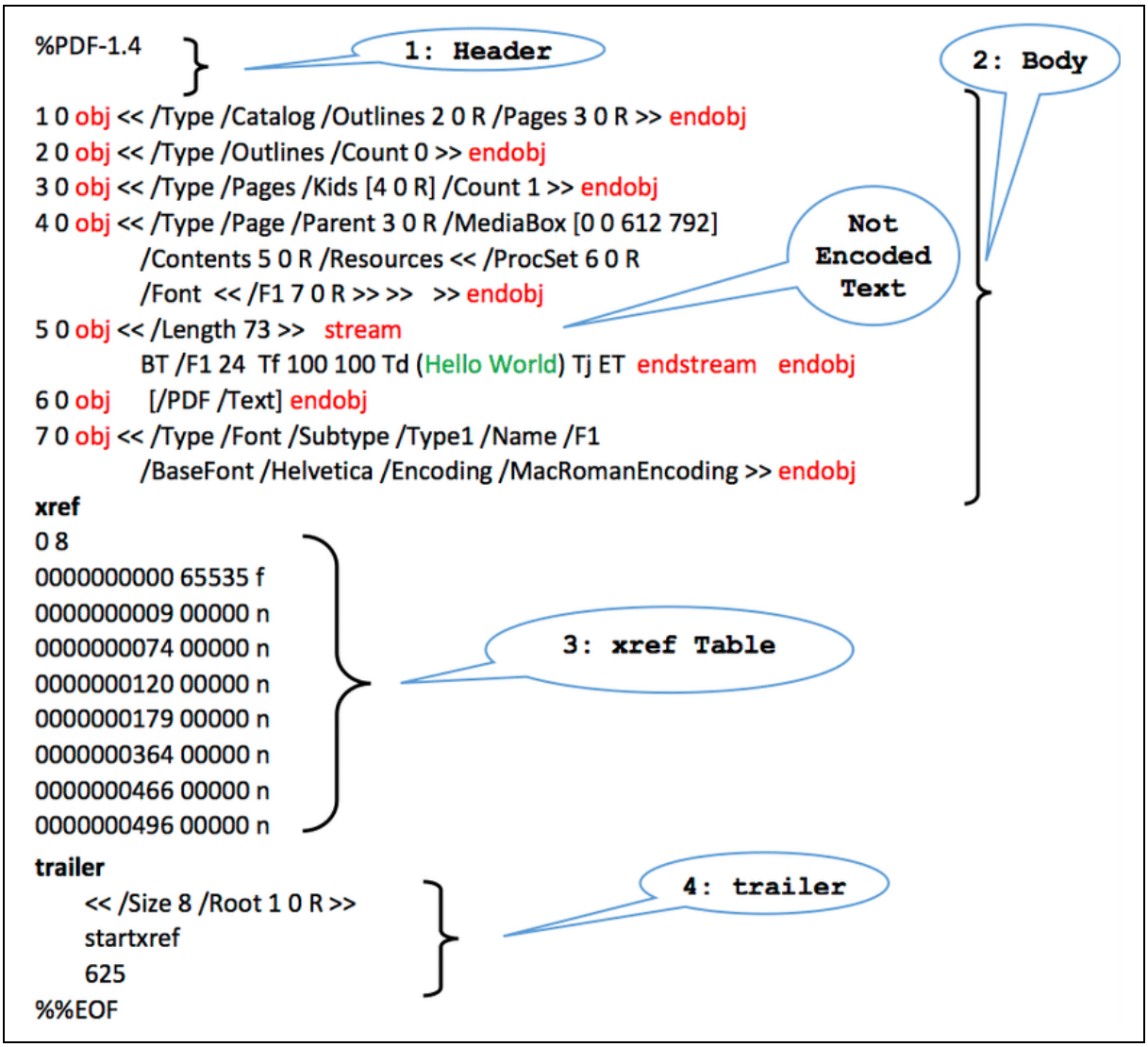


Abbildung 10: Technischer Aufbau einer PDF-Datei[24]

[23] Lehenmeier/Lohmüller (2012): S. 1.
[24] Al-Sharif et al. (2018): S. 699.

Die folgende Abbildung zeigt das Ergebnis der Text-Extraktion:

Standard-Nutzungsbedingungen:

Die Dokumente auf EconStor dürfen zu eigenen wissenschaftlichen Zwecken und zum Privatgebrauch gespeichert und kopiert werden.

Sie dürfen die Dokumente nicht für öffentliche oder kommerzielle Zwecke vervielfältigen, öffentlich ausstellen, öffentlich zugänglich machen, vertreiben oder anderweitig nutzen.

Sofern die Verfasser die Dokumente unter Open-Content-Lizenzen (insbesondere CC-Lizenzen) zur Verfügung gestellt haben sollten, gelten abweichend von diesen Nutzungsbedingungen die in der dort genannten Lizenz gewährten Nutzungsrechte.

Terms of use:

Documents in EconStor may be saved and copied for your personal and scholarly purposes.

You are not to copy documents for public or commercial purposes, to exhibit the documents publicly, to make them publicly available on the internet, or to distribute or otherwise use the documents in public.

If the documents have been made available under an Open Content Licence (especially Creative Commons Licences), you may exercise further usage rights as specified in the indicated licence.

Abbildung 11: Teil des Beispieldokument mit Kennzeichnung der Objekteaufteilung[25]

PDF-Dateien können somit eher als eine Sammlung von Druckanweisungen, denn als ein Datenformat bezeichnet werden. Der Inhalt wird in Objekte zerlegt, die Positionsangaben und andere Darstellungsinformationen enthalten. Diese Objekte können sogar in der Datei anders angeordnet sein, als sie im Ausdruck gelesen werden sollen.

Für die Extraktion von Text ist das PDF-Format eigentlich ungeeignet, aber aufgrund seiner extremen Verbreitung spielt es in diesem Umfeld eine dominierende Rolle und es muss dennoch eine Lösung gefunden werden. Es existieren verschiedene Tools und Programmbibliotheken, jede mit ihren eigenen Vor- und Nachteilen bis hin zur Nutzung künstlicher Intelligenz.

Unter den verfügbaren Ressourcen haben sich bei den Arbeiten der Verfasser die Bibliothek PyMuPDF[26] für allgemeine Textextraktionsaufgaben und pytesseract[27] für die Schrifterkennung in Bildern als relativ vielseitig und praktisch erwiesen. Dennoch ist keine Lösung bekannt, die eine fehlerfreie Extraktion aus allen PDF-Varianten garantieren kann.

Für eine optimale Textextraktion sind PDF-Dokumente, die eine minimale Vielfalt an Objekttypen wie Bilder und Tabellen aufweisen und komplexe Layouts wie mehrspaltige Texte und Fußnoten vermeiden, deutlich besser geeignet. Bei Dokumenten, die einer OCR-basierten Schrifterkennung unterzogen werden müssen, ist zudem auf eine hohe Bildqualität zu achten. Hier kann die OpenCV-Bibliothek zur Verbesserung der Bildqualität beitragen und somit die Effizienz der Textextraktion signifikant steigern.

Chunks

Im Zusammenhang mit der RAG -Anwendungen ist Chunking der Prozess, bei dem große Textabschnitte in kleinere Segmente zerlegt werden. Es ist aus zwei Gründen wichtig, die richtige Wahl hinsichtlich der Chunking-Strategie zu treffen:

[25] Eigene Darstellung.
[26] PyMuPDF (2024).
[27] pytesseract (2024).

- Erstens wird dadurch bestimmt, ob der Kontext tatsächlich für die Eingabeaufforderung relevant, hinreichend prägnant und gleichzeitig umfänglich ist.

- Zweitens wird bestimmt, ob der abgerufene Text in den Kontext eingebettet werden kann, bevor er an ein LLM gesendet wird. Der Grund dafür ist, dass die Anzahl der Token dadurch begrenzt ist, dass nur eine bestimmte Menge pro Anfrage gesendet werden kann.

Die Chunk-Size[28], also die Größe der Datenblöcke oder Textsegmente, in die Informationen unterteilt werden, hat einen signifikanten Einfluss auf die Qualität der Antworten von RAG-Systemen, er manifestiert sich in verschiedenen Aspekten der Antwortqualität.

Wenn ein vollständiger Absatz oder ein Dokument eingebettet wird, berücksichtigt der Einbettungsprozess sowohl den Gesamtkontext als auch die Beziehungen zwischen den Sätzen und Phrasen im Text. Dies kann zu einer umfassenderen Vektordarstellung führen, die die umfassendere Bedeutung und Themen des Textes erfasst. Größere Chunkgrößen können andererseits zu Rauschen führen und die Bedeutung einzelner Sätze oder Phrasen verwässern, was das Finden präziser Übereinstimmungen bei der Abfrage erschwert.

Eine kleinere Chunkgröße, beispielsweise ein einzelner Satz oder eine Phrase, konzentriert sich auf Einzelheiten und eignet sich möglicherweise besser für den Abgleich mit Einbettungen auf Satzebene. Allerdings können sie auch dazu führen, dass wichtiger Kontext außerhalb des betrachteten Chunks liegt, was wiederum die Relevanz und Genauigkeit der Antworten beeinträchtigen kann.

Die heute gebräuchlichste Methode besteht darin, eine feste Anzahl von Token in einem Chunk zu definieren und optional festzulegen, ob es zwischen diesen Token eine Überlappung geben soll. Im Allgemeinen wird eine gewisse Überlappung zwischen den Blöcken gewünscht, um sicherzustellen, dass der semantische Kontext zwischen den Chunks möglichst nicht verloren geht.

Es gibt keine allgemeingültige Lösung für das Chunking. Was für einen Anwendungsfall optimal ist, kann für einen anderen nicht funktionieren. Man sollte sich die Frage nach der erwarteten Länge und Komplexität der Benutzeranfragen stellen. Werden sie kurz und spezifisch oder lang und komplex sein? Dies wirkt sich auch auf die Art und Weise aus, wie die Inhalte aufgeteilt werden. Hier empfiehlt es sich, mit Testabfragen zu experimentieren und im jeweiligen Anwendungsfall eine Fein-abstimmung vorzunehmen. Üblich sind Chunk-Größen zwischen 500 und 2000 Token.

Vektordatenbank

In einem Retrieval-Augmented Generation (RAG) System erfüllt eine Vektordatenbank eine zentrale Funktion, indem sie einerseits die im Embedding in hochdimensionale Vektoren umgewandelten Chunks speichert und andererseits mit hoch performanten Algorithmen das schnelle Auffinden ähnlicher oder relevanter Vektoren sicherstellt.

[28] Finardi et al. (2024).

```
[[0.78964165, 0.79994387, 0.99869725, 0.87417403, ...........
 [0.69841495, 0.96025049, 0.61727824, 0.23508716, ...........
 [0.87848775, 0.20182608, 0.40274576, 0.05230779, ...........
 [0.65808922, 0.3960142 , 0.88229928  , 0.81538429, ...........
 [0.67959394, 0.10604208, 0.42918272, 0.510735525, ...........
                              ⋮
```

Abbildung 12: beispielhafte Darstellung eines Vektors[29]

Durch die Fähigkeit, relevante Informationen effizient zu identifizieren und abzurufen, trägt die Vektordatenbank wesentlich dazu bei, die Relevanz und Qualität der generierten Inhalte des gesamten RAG-Systems zu erhöhen.

Vektordatenbanken sind in der Regel hoch skalierbar und können mit zunehmender Datenmenge effizient erweitert werden. Dies ist besonders wichtig für RAG-Systeme, die mit großen und ständig wachsenden Informationsmengen arbeiten.

Grundsätzlich stehen mit Vektor-Datenbanken, Vektor-Suchbibliotheken und Vektor-Suchplugins verschiedene Ansätze für die Anwendung in einem RAG zur Verfügung. Jeder dieser Typen hat seine eigenen Vor- und Nachteile, abhängig von den spezifischen Anforderungen der Anwendung, wie Skalierbarkeit, Geschwindigkeit der Suchanfragen und Integration in Systeme.

- Vektor-Datenbanken sind spezialisierte Datenbanken, die für die Speicherung, Abfrage und Verwaltung von Vektordaten optimiert sind. Sie ermöglichen die schnelle Suche in großen Mengen von Vektordaten, wie sie in Anwendungen des maschinellen Lernens und der künstlichen Intelligenz verwendet werden.

- Vektor-Suchbibliotheken wurden besonders vor dem Aufkommen von Vektordatenbanken verwendet. Mit ihrer Hilfe können schnell leistungsstarke Prototypen eines Vektorsuchsystems erstellen. FAISS[30] als Beispiel ist Open Source und wurde von Meta für eine effiziente Ähnlichkeitssuche und dichtes Vektor-Clustering entwickelt.

- Vektor-Suchplugins erweitern die Funktionalität von bestehenden Datenbanksystemen oder Suchmaschinen, indem sie die Möglichkeit hinzufügen, innerhalb dieser Systeme Vektorsuchen durchzuführen. Diese Plugins sind als Erweiterungen bestehender Architekturen gedacht wodurch sie eingeschränkt sind und nicht optimiert werden konnten. Beispiele sind Elasticsearch und PostgreSQL, auch die Lösungen der namhaften Cloudanbieter sind in der Regel nicht als hochspezialisierte Vektordatenbanken konzipiert, sondern vielmehr als durch Plugins erweiterte Systeme zu betrachten.

[29] Eigene Darstellung.
[30] FAISS (2024).

Während alle Arten von Systemen ihre Anwendungen haben, sollte eine spezialisierte Vektordatenbank die erste Wahl für datenintensive Unternehmen sein, die mit Hunderten von Millionen von Vektoren arbeiten und Antworten in Echtzeit benötigen.

Häufig verwendete Vektordatenbaken sind Pinecone, Milvus, Chroma, Weaviate, Deep Lake und qdrant, wobei diese Aufzählung keinen Anspruch auf Vollständigkeit erhebt und keine Wertung darstellt. Darunter befinden sich sowohl proprietäre als auch Open-Source-Produkte. Einige werden, als gehostete Lösung angeboten, andere als selbst zu betreibende Anwendung

So haben die Autoren eigene Entwicklungen FAISS für schnell zu erstellende Prototypen verwendet und für RAG-Anwendungen mit höheren Anforderungen an Performance und Ergebnisqualität die Vektordatenbanken Chroma und Deep Lake eingesetzt. Auch hier kann nur wärmstens empfohlen werden, für den jeweiligen Anwendungsfall eigene Evaluierungen durchzuführen, um die am besten geeignete Vektordatenbank zu finden.

Verbesserung des Retrieving

Während bisher die „Fallstricke“ und wesentlichen Verbesserungspotentiale des RAG-Grundkonzeptes (Naive RAG) diskutiert wurden, sollen im Folgenden stellvertretend einige konzeptionelle Verbesserungsansätzenaufgegriffen und vorgestellt werden, an denen derzeit vielerorts geforscht wird. [31]

- Nutzung von Metadaten
  Hier geht es Metadaten, die bereits in den Ursprungsdokumenten enthalten sind oder im Rahmen des Dokumentenmanagements ergänzt werden. Sie werden von dort in die Chunks übernommen, so dass sie beim Retrieval zusätzlich herangezogen werden können.

- Dynamisches Chunking
  Bisher wurde das fixe Chunking mit festen Längen und Überlappungen diskutiert. Trotz der vorgesehenen Überlappungen kann diese Strategie bei heterogenen Datensätzen zu Problemen führen, wenn relevante Informationen über mehrere Chunks verteilt sind. Beim dynamischen Chunking variiert die Größe der Chunks, um eine möglichst optimale Informationsmenge zu erhalten.
  In einer einfachen Lösung legt man das Ende eines Chunks grundsätzlich auf ein Satzende, um so keine Information zu zerschneiden.
  Mit deutlich höherem Aufwand versucht das semantische Chunking, die Daten nach ihrer Bedeutung oder ihrem semantischen Zusammenhang zu unterteilen.

[31] Anantha et al. (2023) und Gao et al. (2023).

- Parent Document Retriever
  Ziel eines Parent Document Retrievers ist es, die Relevanz und Kontextualität der Chunks gleichzeitig zu verbessern. Zunächst werden Teile des Originaldokuments als Parent-Chunks mit größerer Chunk-Länge erzeugt. Aus diesen Parent-Chunks werden Child-Chunks mit deutlich kleinerer Chunk-Länge erzeugt. Im Retrievalprozess wird dann zunächst auf Basis der Child-Chunks nach passenden Chunks gesucht. Dadurch wird ein genaues und sehr relevantes Retrieval gewährleistet. Anschließend wird jedoch der zu-gehörige Parent-Chunk verwendet, um diesen dann im Prompt für das LLM zu verwenden. Dadurch wird ein größerer Kontext gewährleistet, von dem wiederum umfassendere und differenziertere Antworten des RAG erwartet werden können.

- Rerank[32]
  Beim Retrievalprozess liefert die Vektordatenbank bei der Suche in der Regel mehrere Treffer. Diese haben in der Regel eine unterschiedliche Relevanz und sind nicht nach Relevanz geordnet. Das Reranking ist ein Prozess, der in Verbindung mit dem LLM und der gestellten Suchanfrage die Suchergebnisse in einer zusätzlichen Vorabanfrage bewertet. Nur der am höchsten bewertete Chunk wird dann in der eigentlichen RAG-Abfrage verwendet.

Zusammenfassung und Ausblick

Die Zukunft von RAG in der Unternehmenswelt ist sehr vielversprechend, da die Technologie das Potenzial hat, viele Aspekte der Geschäftstätigkeit zu verändern. Insbesondere ermöglicht sie erstmals die gezielte Nutzung und Auswertung der riesigen Menge an unstrukturierten Daten, über die jedes Unternehmen verfügt.

Erste, meist sehr positive Ergebnisse sind bereits mit einfachen Prototypen schnell sichtbar. Die technische Basis stammt in der Regel aus einer sehr großen Open Source Community. Technische Optimierungen und Verbesserungen[33], wie sie in diesem Working Paper beschrieben werden, werden weltweit mit sehr hoher Geschwindigkeit vorangetrieben und können sehr schnell umgesetzt werden.

Für eine erfolgreiche Integration von RAG in Unternehmen ist jedoch eine sorgfältige Planung und Vorbereitung sowie insbesondere die Einführung eines konsequenten Datenmanagements unerlässlich. Dazu gehören auch Überlegungen und Entscheidungen zu Dokumentenformaten und deren vollständige maschinelle Lesbarkeit.

[32] Qin et al. (2023).
[33] Chen et al. (2023).

Literaturverzeichnis

ABBYY Europe GmbH (2024): Arten von PDFs, Online-Quelle: https://pdf.abbyy.com/de/learning-center/pdf-types/, letzter Zugriff am: 11.03.2024.

Adobe Systems Software Ireland Limited (2024): PDF-Ebenen, Online-Quelle: https://helpx.adobe.com/de/acrobat/using/pdf-layers.html, letzter Abruf am: 11.03.2024.

Anantha, Raviteja; Bethi, Tharun; Vodianik, Danil; Chappidi; Danil (2023): Context Tuning for Retrieval Augmented Generation, Online-Quelle: https://arxiv.org/abs/2312.05708; letzter Zugriff am: 11.03.2024.

Al-Sharif,Ziad A.; Al-Khalee, Attaa Y.; Al-Saleh, Mohammed I.; Al-Ayyoub, Mahmoud (2018): Carving and clustering files in ram für memory forensics, in: Far East Journal of Electronics and Communications, Vol. 18, No. 5, S. 695-722, Online-Quelle: https://www.researchgate.net/publication/326102942_CARVING_AND_CLUSTERING_FILES_IN_RAM_FOR_MEMORY_FORENSICS/link/5bc1b225a6fdcc2c91fb5c53/download?_tp=eyJjb250ZXh0Ijp7ImZpcnN0UGFnZSI6InB1YmxpY2F0aW9uIiwicGFnZSI6InB1YmxpY2F0aW9uIn19, letzter Zugriff am: 11.03.2024.

Chen, Jiawei; Lin, Hongyu; Han, Xianpei; Sun, Le (2023): Benchmarking Large Language Models in Retrieval-Augmented Generation, Online-Quelle: https://arxiv.org/abs/2309.01431, letzter Zugriff am: 11.03.2024.

Europäische Union (2012): Einführung in das Metadatenmanagement, Online-Quelle: https://data.europa.eu/sites/default/files/d2.1.2_training_module_1.4_introduction_to_metadata_management_de_edp.pdf, letzter Zugriff am: 11.03.2024.

FAISS (2024): Faiss Documentation, Online-Quelle: https://faiss.ai/, letzter Zugriff am: 11.03.2024.

Finardi, Paulo; Avila, Leonardo; Castaldoni, Rodrigo; Gengo, Pedro; Larcher, Celio; Piau, Marcos; Costa, Pablo; Caridá, Vinivius (2024): The Chronicles of RAG: The Retriever, The Chunk an The Generator, Online-Quelle: https://arxiv.org/pdf/2401.07883.pdf, letzter Abruf am: 11.03.2024.

Foxit Software Inc. (2024): PDF-Pyramide, Online-Quelle: https://www.foxit.com/blog/wp-content/uploads/2020/08/PDFPyramid2-1.png, letzter Abruf am: 11.03.2024.

Gao, Yunfan; Xiong, Yun; Gao, Xinyu; Jia, Kangxiang; Pan, Jinliu; Bi, Yuxi; Dai, Yi; Sun, Jiawei; Guo, Qianyu; Wang, Meng; Wang, Haofen (2023): Retrieval-Augmented Generation for Large Language Models: A Survey, Online-Quelle: https://arxiv.org/abs/2312.10997v3, letzter Zugriff am: 11.03.2024.

IPI GmbH (2024): Dokumentenlebenszyklus, Online-Quelle: https://www.ipi-gmbh.com/wp-content/uploads/2024/02/Dokumentenmanagement_Dokumenten-Lebenszyklus.png, letzter Zugriff am: 11.03.2024.

Lee, Ernesto (2023): Chunking Strategies for LLM Applications, Online-Quelle: https://drlee.io/chunking-strategies-for-llm-applications-7a37d56e2b15, letzter Zugriff am: 11.03.2024.

Lehenmeier, Constantin; Hohmüller. Valentin (2012): Herausforderungen beim Parsen von PDF-Dateien, Online-Quelle: https://wiki.mi.ur.de/_media/lehre/seminar_plagiate_ss_12/handout_pdf.pdf, letzter Zugriff am: 11.03.2024.

Mathur, Akash (2024): Advanced RAG: Optimizing Retrieval with Additional Context & MetaData using LlamaIndex, - Using Open Source LLM Zephyr-7b-alpha and Instruct Embeddings hkunlp/instructor-large, Online-Quelle: https://akash-mathur.medium.com/advanced-rag-optimizing-retrieval-with-additional-context-metadata-using-llamaindex-aeaa32d7aa2f, letzter Abruf am: 11.03.2024.

Meuschke, Norman; Jagdale, Apurva; Spinde, Timo; Mitrović, Jelena; Gipp, Bela (2023): A Benchmark of PDF Information Extraction Tools using a Multi-Task and Multi-Domain Evaluation Framework for Academic Documents, in Information for a Better World: Normality, Virtuality, Physicality, Inclusivity, LNCS, vol. 13972, Cham: Springer Nature Switzerland, 2023, S. 383–405.

Miro Engineering (2024): Text-Based PDFs vs. Inage-Based PDFs, Online-Quelle: https://miro.medium.com/v2/resize:fit:1400/1*BckE1ZKEuaBSFdQ8NvHP1w.png, letzter Abruf am: 11.03.2024.

Optimal Systems GmbH (2024): Was bringt ein DMS?, Online-Quelle: https://www.optimal-systems.de/wp-content/uploads/2022/10/Was-bringt-ein-DMS-4.svg, letzter Abruf am: 11.03.2024.

PyMuPDF (2024): Welcome to PyMuPDF PyMuPDF 1.23.26 Documentation, Online-Quelle: https://pymupdf.readthedocs.io/en/latest/ letzter Zugriff am: 11.03.2024.

pytesseract (2024): Projekt-Beschreibung Online-Quelle: https://pypi.org/project/pytesseract/, letzter Zugriff am: 11.03.2024.

Qin, Zhen; Jagerman, Rolf; Hui, Kai; Zhuang, Honglei; Wu, Junru; Shen, Jiaming; Liu, Tianqi; Liu, Jialu; Metzler, Donald; Wang, Xuanhui; Bendersky, Michael (2023): Large Language Models are Effective Text Rankers with Pairwise Ranking Prompting, Online-Quelle: https://arxiv.org/abs/2306.17563v1, letzter Zugriff am: 11.03.2023.

Reinking, Ernst; Becker, Marco (2023a): Opportunities for business use of today's AI models - Rapidly achievable personalization of Large Language Models (like ChatGPT) in times of Industry 5.0, IUCF Working Paper, No. 10/2023, ZBW – Leibniz Information Centre for Economics, Kiel, Hamburg.

Reinking Ernst; Becker, Marco (2023b): Einsatzmöglichkeiten von KI in Unternehmen - Zeitnah erreichbare Personalisierung von Large Language Models (wie ChatGPT) in Zeiten der Industrie 5.0, IUCF Working Paper, No. 9/2023, ZBW – Leibniz Information Centre for Economics, Kiel, Hamburg.

Sevdesk (2024): Dokumentenmanagementsystem, Online-Quelle: https://cdn.sevdesk.de/uploads/Blog-Infografik-Dokumentensystem-1.jpg?auto=format,compress, letzter Zugriff am: 11.03.2024.