

Sachbericht:

Vorverarbeitungs- und Transkriptionspipeline

Im Rahmen dieser Arbeit wurde eine Pipeline für die Vorverarbeitung und Transkription der Fachtexte erfolgreich entwickelt und implementiert. Die Pipeline erkennt verschiedene Dateitypen und extrahiert deren Inhalte für die weitere Verarbeitung. Dabei werden die Texte bereinigt, strukturiert und in klar definierte Sätze unterteilt, die jeweils mit einer eindeutigen ID und der zugehörigen Seitenzahl versehen sind. Diese standardisierte Aufbereitung gewährleistet eine hohe Datenqualität und ermöglicht eine präzise Weiterverarbeitung in nachgelagerten Modulen.

KI-Funktion für Tokenisierung und Embeddings

Zur semantischen Aufschlüsselung der Eingabetexte wurde eine KI-gestützte Funktion erfolgreich entwickelt und implementiert, die Sätze tokenisiert und kontextbezogene Embeddings für die Tokens generiert. Durch die Anwendung moderner NLP-Modelle werden die Sätze in vektorielle Repräsentationen überführt, die semantische Zusammenhänge (auch zum Gesamtthema des Textes) erfassen und eine effiziente Weiterverarbeitung ermöglichen. Die Embeddings dienen als Grundlage für verschiedene KI-gestützte Analysen, einschließlich der semantischen Suche und Clustering-Methoden, und tragen zur Verbesserung der Textverständnis- und Verarbeitungsfähigkeiten des Systems bei.

Extraktion und Ranking von Schlüsselwörtern aus NLP-Vektoren

Um die relevanten Konzepte des Eingabetexts später als Knoten im Lerngraphen zur Verfügung stellen zu können, wurde eine Methode zur Extraktion und Priorisierung von Schlüsselwörtern aus NLP-Vektoren entwickelt und implementiert. Durch die Analyse vektorisierter Textrepräsentationen werden relevante Begriffe identifiziert und anhand ihrer semantischen Bedeutung sowie ihres Kontexts gewichtet. Das resultierende Ranking ermöglicht eine präzisere Identifikation zentraler Konzepte und verbessert die Effizienz nachgelagerter Verarbeitungsprozesse, wie die thematische Klassifikation oder die Information-Retrieval-Systeme.

Erstellung einer maschinenlesbaren Liste syntaktischer Muster und deren Beziehungen

Für eine noch genauere Extraktion von semantischen Relationen wurde eine syntaktische Mustererkennung entwickelt und implementiert. Dazu wurde eine umfassende, maschinenlesbare Liste syntaktischer Muster erstellt und die Muster mit den jeweiligen semantischen Beziehungen verknüpft, die sie ausdrücken. Die resultierende Struktur ermöglicht eine präzise Identifikation und Interpretation syntaktischer Konstruktionen in Texten und dient als Grundlage für weiterführende KI-gestützte Sprachverarbeitung. Durch diese standardisierte Repräsentation wird die Erkennung sprachlicher Zusammenhänge optimiert und eine effizientere Verarbeitung von Inhalten in nachgelagerten Modulen unterstützt.

Erstellung von Fachontologien und Regelwerken zur Verknüpfung mit Transkriptionsergebnissen

Zur inhaltlichen Verbesserung des Materials für den Lerngraphen wurden von Experten Domänenontologien kuratiert, die die zentralen Konzepte eines Fachgebiets und deren semantische Zusammenhänge repräsentieren. Es wurden Regelwerke entwickelt, die die Inhalte der Ontologien zu den Ergebnissen der Transkriptionspipeline in Beziehung setzen, wodurch eine präzise inhaltliche Zuordnung und Kontextualisierung ermöglicht wird. Diese Arbeit legt die Grundlage für eine verbesserte semantische Analyse und eine effizientere Weiterverarbeitung der extrahierten Informationen.

Herstellung von Beziehungen zwischen extrahierten Konzepten, Syntaktischen Muster und Ontologien

Zur Erstellung des Roh-Graphen wurden Beziehungen zwischen Konzepten aus der Schlüsselwortextraktion, der syntaktischen Mustererkennung und den Ontologien hergestellt. Dabei wurden extrahierte Begriffe systematisch mit den zugrunde liegenden syntaktischen Strukturen und deren semantischen Bedeutungen verknüpft. Durch die Integration in Fachontologien konnten präzise Beziehungen zwischen Konzepten definiert und deren inhaltliche Zusammenhänge besser erfasst werden. Diese Arbeit verbessert die semantische Analyse und legt die Grundlage für eine tiefere, kontextbasierte Verarbeitung und Interpretation von Textinhalten.

Erstellung eines navigierbaren, zoombaren Graphen für Lernpfade

Im Rahmen dieser Arbeit wurde ein interaktiver Graph erfolgreich entwickelt, der die extrahierten Schlüsselwörter und deren Beziehungen visuell darstellt. Die Ergebnisse der semantischen Analyse wurden in eine navigierbare und zoombare Struktur überführt, in der Lernpfade definiert werden können. Nutzer haben die Möglichkeit, Knoten und deren Verbindungen zu überprüfen, zu bearbeiten und anzupassen, um das Wissen gezielt zu strukturieren und zu erweitern. Diese dynamische Visualisierung verbessert die Nachvollziehbarkeit komplexer Zusammenhänge und unterstützt eine intuitive Wissensnavigation und -vermittlung.

Table of Contents

Übersicht über die Transkriptionspipeline:	1
Transcription Pipeline Technical Details	2
Feature Pipeline Overview:	5
Nutzung von Ontologien:	6
Extraktion von Schlüsselwörtern:	7
Erstellung von Relationen aus der Extraktion von Schlüsselwörtern:	9
Syntaktische Extraktion	12
Large-Language-Model Funktionen	15
Large-Language-Model Zusammenfassungen	15
Large-Language-Model Graphen Pruning	16
Wissensgraph	17
Wesentliche Struktur und Zoomfunktion	17
Lernetiketten und personalisierte Lernpfade	17
Zusammenfassungen und Quelleninhalte	19
Lernaktivitäten in der Grafik.....	20

Übersicht über die Transkriptionspipeline:

Die Transkriptionspipeline für das NLP-Modul wurde erfolgreich implementiert und ermöglichte die Verarbeitung von Blocktext in einzelne Sätze mit eindeutigen Bezeichnern. Die Transkriptionspipeline erkannte auch die Inhaltsverzeichnisseite von Büchern, sofern eine solche vorhanden war. Die Kapitelgrenzen dieses Inhaltsverzeichnisses wurden verwendet, um Sätze mit den Kapiteln zu kennzeichnen, aus denen sie extrahiert wurden. Mit dieser Pipeline wird der Text effizient segmentiert, wobei die Satzgrenzen erhalten bleiben und eine genaue Weiterverarbeitung gewährleistet ist. Jedem extrahierten Satz wird ein eindeutiger Bezeichner zugewiesen, der eine präzise Verfolgung und Referenzierung innerhalb des NLP-Workflows ermöglicht.

Robuste Mechanismen zur Textvorverarbeitung wurden integriert, um verschiedene Eingabeformate zu verarbeiten und Konsistenz und Zuverlässigkeit zu gewährleisten. Besonderes Augenmerk wurde auf Randfälle wie Abkürzungen,

Interpunktionsinkonsistenzen und mehrzeilige Textstrukturen gelegt, um eine hohe Segmentierungsgenauigkeit zu gewährleisten. Die Pipeline wurde so konzipiert, dass sie skalierbar und anpassungsfähig ist, so dass sie in bestehende NLP-Komponenten integriert werden kann, ohne dass die Leistungsfähigkeit beeinträchtigt wird.

Die endgültige Implementierung wurde mit verschiedenen Textkorpora getestet, um die Segmentierungsgenauigkeit und die Identifikatorzuweisung zu validieren. Die Ergebnisse bestätigten die Effektivität der Pipeline bei der Vorbereitung strukturierter

Textdaten für die weitere linguistische und analytische Verarbeitung innerhalb des NLP-Moduls.

Darüber hinaus wurde ein NLP-Modell selbst in die Transkriptionspipeline aufgenommen. Dieses wurde erfolgreich darauf trainiert, wichtige Strukturelemente in Textdaten zu identifizieren, darunter Referenzen, Vorworte, Verlagsinformationen, Fußnoten, Formeln, Indizes und Glossare. Das Modell zeigte eine starke Leistung bei der Unterscheidung dieser Kategorien und ermöglichte eine effiziente Klassifizierung und Markierung relevanter Abschnitte für die weitere Verarbeitung.

Um ein robustes Training zu gewährleisten, wurde ein vielfältiger Datensatz zusammengestellt, der eine Vielzahl von Dokumenttypen und Formatierungsstilen enthält. Um die Klassifizierungsgenauigkeit zu verbessern, wurden fortschrittliche Techniken zur Verarbeitung natürlicher Sprache, einschließlich Sequenzkennzeichnung und Mustererkennung, eingesetzt. Besonderes Augenmerk wurde auf die Unterscheidung strukturell ähnlicher Abschnitte gelegt, wie z. B. die Unterscheidung zwischen Fußnoten und Referenzen oder Formeln und standardmäßigen numerischen Daten.

Das trainierte Modell wurde in die Verarbeitungspipeline integriert, um bestimmte Kategorien automatisch zu kennzeichnen, so dass ihre Identifizierung die nachfolgenden Analyse- und Transformationsschritte steuern kann. Die Bewertungsmetriken bestätigten eine hohe Präzision und Wiedererkennung für alle Zielkategorien, was eine zuverlässige Erkennung gewährleistet. Die endgültige Implementierung unterstützt eine skalierbare Verarbeitung und kann mit zusätzlichen Trainingsdaten weiter verfeinert werden, um die Anpassungsfähigkeit an neue Dokumentstrukturen zu verbessern.

Transcription Pipeline Technical Details

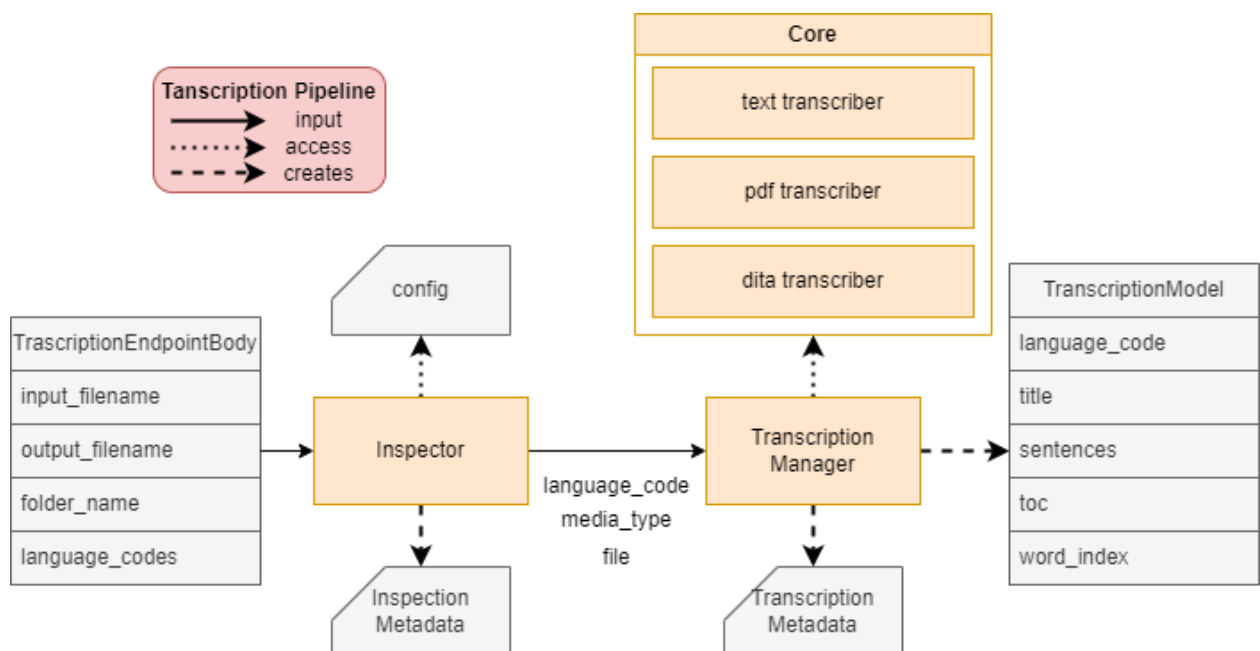


Figure 1: Overview of the Transcription Pipeline

Inspector: prüft und validiert die Datei in folgender Reihenfolge:

- Datei existiert
- Medientyp wird unterstützt
- Dateigröße wird unterstützt
- Sprache wird unterstützt

Transcription Manager: startet die Kerntranskription basierend auf dem Medientyp

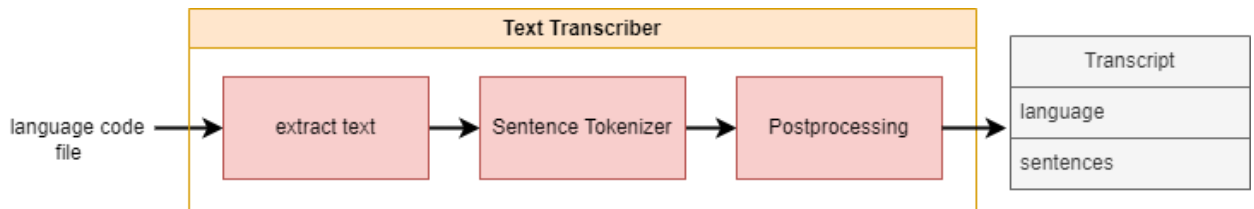


Figure 2: Text Transcriber

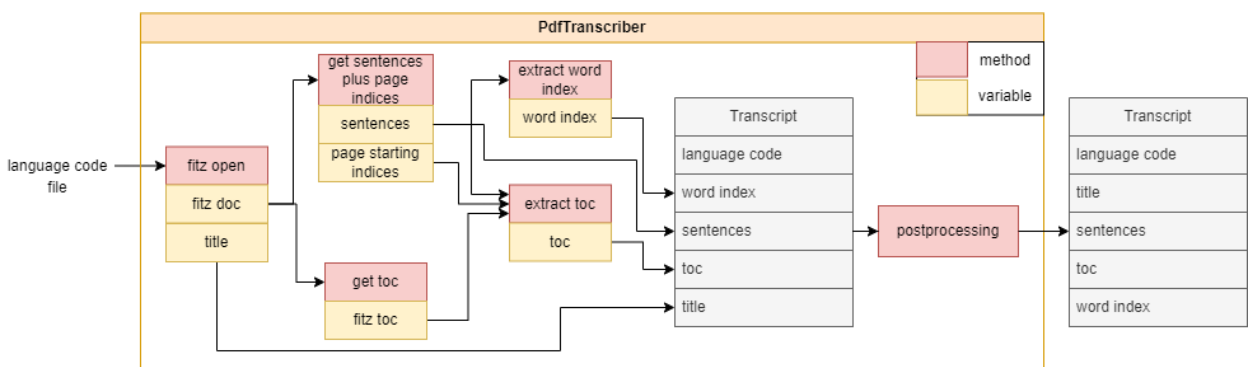


Figure 3: PDF Transcriber

fitz open: PyMuPDF methode, die das fitz doc object erstellt

get_sentences_plus_page_indices: startet die fitz_doc pages macht die folgende Schritte:

1. extrahiert Text
2. Textfehler behandeln
3. Setzte Tokenizer
4. speichert die Kennung des Anfangssatzes der Seite

get_toc: PyMuPDF methode extrahiert toc aus pdf metadata in List Form [level, title, page]

extract_toc: wann ein fitz_toc gibt's, extract_toc iteriert über die fitz_toc Einträge und erstellte eine Liste von Sections (level, title, start_section_id).

extract_word_index:

1. Findet den Index im pdf
2. extrahiert den Index trotz unterschiedlicher Indexstile korrekt

Um den Index zu finden, werden die Sätze nach bestimmten Startmustern („Index“, „Stichwortverzeichnis“ etc.) durchsucht. Wird kein Muster gefunden oder werden zu viele gefunden, schlägt der word_index fehl. Dann werden alle Sätze nach dem letzten gefundenen Startmuster für die Indexerstellung verwendet.

Zunächst wird der Text bereinigt und formatiert und dann durch Zeilenumbruch getrennt. Alle in Frage kommenden Indexwörter werden dann durch Regex gefiltert (Wort und Zahl durch Leerzeichen getrennt). Auf diese Weise wird eine Liste von Indexwörtern erstellt. Anschließend werden die Wörter auf ihre alphabetische Reihenfolge hin überprüft; sind sie nicht in Ordnung, wird kein word_index zurückgegeben.

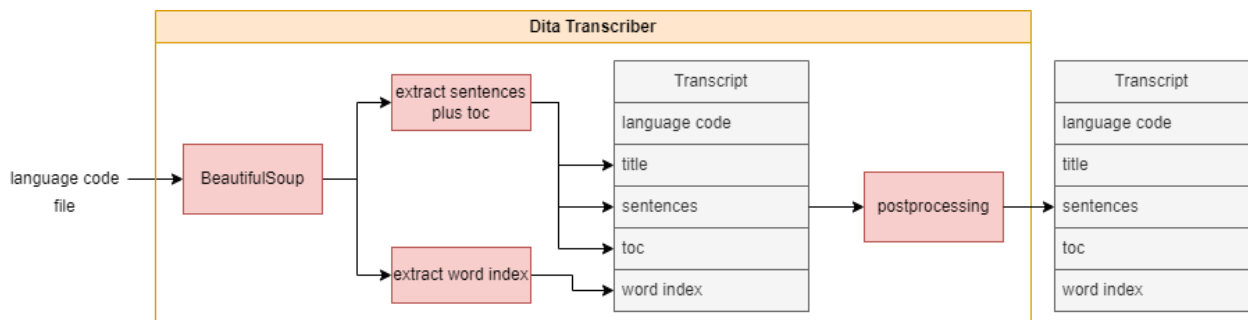


Figure 4: Dita Transcriber

BeautifulSoup: Bibliothek zum Extrahieren von Daten aus html- und xml-Dateien. Erzeugt eine Baumstruktur, die in der gesamten Pipeline verwendet wird

extract_sentences_plus_toc: iteriert durch den Baum und sucht nach allen Themen mit dem Tag „topic“, „concept“ oder „glossentry“. Alle Textkörper werden sentinisiert und an die Satzliste angehängt. Wenn das Tag „topic“ lautet, wird der Titel an die toc-Liste angehängt.

extract_word_index: alle Titel der Themen mit dem Tag „concept“ oder „glossentry“ werden an die word_index-Liste angehängt. Die Seiten werden hier nur aufgezählt und haben keine wirkliche Bedeutung, da Dita-Dateien nicht seitenbasiert sind.

Sentence Tokenizer: Der nltk.sent_tokenizer verwendet, da die Tokenisierung langer Eingaben viel Zeit in Anspruch nimmt. Für ein normales pdf-Lehrbuch benötigte der spacy sentenizer mehrere Sekunden, während der nltk sentenizer dies in weniger als einer Sekunde erledigen kann.

Postprocessing: Folgende Schritte werden im Postprocessing durchgeführt (alle hier verwendeten Schritte werden hoffentlich durch eine neue Architektur und die Integration der Blockklassifizierung ersetzt):

1. den Anfang löschen: alles vor dem ersten Satz kann als Titelblatt, Vorwort und schriftliches Inhaltsverzeichnis eingestuft und daher gelöscht werden
2. Ende löschen: wenn bestimmte Schlüsselwörter in den toc-Titeln gefunden werden („Glossar“, „Danksagungen“, „Quellen“ usw.), kann alles danach als

nicht informativer Text für die Feature-Pipeline eingestuft werden (oder wurde bereits anderweitig extrahiert) und wird gelöscht

3. Iteration über die Sätze mit folgenden Schritten:
 - a. Bereinigen des Satzanfangs, der dem Regex entspricht
 - b. Löschen des ganzen Satzes, der mit dem Regex übereinstimmt
 - c. Zusammenfügen von Sätzen, die vom Sentenizer falsch aufgeteilt wurden
 - d. Entfernt Zeilenumbrüche
 - e.

Digital and Physical Page Number: Da digitale PDFs verwendet werden, gibt es zwei Arten von Seitenzahlen:

digitale Seitenzahl: Seitenzahl, die angegeben wird, wenn die PDF-Datei in einem Editor geöffnet wird. Die Titelseite ist die Seitennummer 1

physische Seitenzahl: Seitenzahl, die auf der Seite angegeben ist (bei physischen Kopien der PDF-Datei). Die Titelseite muss nicht unbedingt die Seitenzahl 1 sein.

Manche PDFs beginnen mit der Seitennummerierung nach Einleitung und TOC. So kann die Seite mit der physischen Seitenzahl 1 zum Beispiel die digitale Seitenzahl 7 haben.

Feature Pipeline Overview:

Die Feature Pipeline enthält die Logik und die Schritte des Hauptprozesses. Sie nimmt als Eingabe die Ergebnisse der Transkriptionspipeline und gibt als Ausgabe eine Liste von Knoten und Kanten mit verschiedenen Eigenschaften aus. Die Feature Pipeline hat 3 Hauptfunktionen, die Begriffe extrahieren und Beziehungen zwischen ihnen herstellen.

Dabei handelt es sich um i) Ontology Leveraging, ii) Syntactic Patterns und iii) Vector Embeddings.

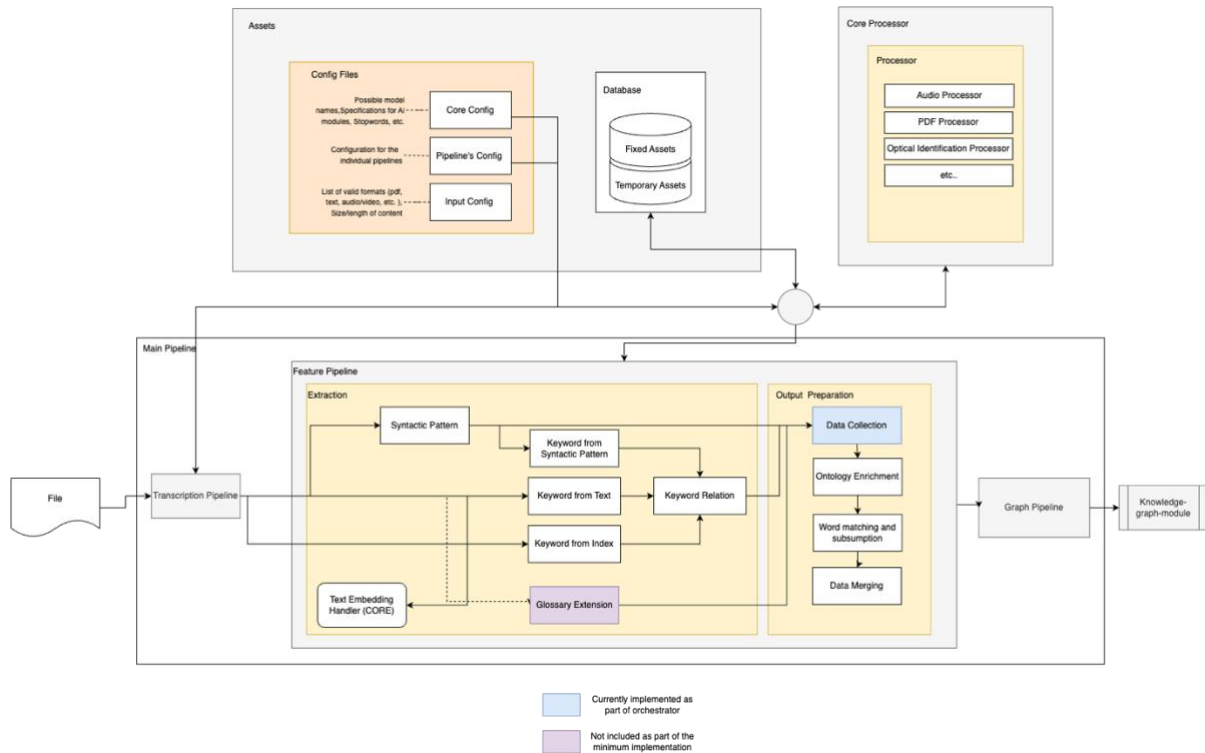


Figure 5: Feature Pipeline Overview

Nutzung von Ontologien:

Dieses Arbeitspaket lieferte erfolgreich maschinenlesbare Ontologien, die auf bestimmte Bereiche zugeschnitten sind und eine strukturierte Darstellung der wichtigsten Konzepte, Begriffe und Beziehungen ermöglichen. Diese Ontologien wurden selbst in einem standardisierten Graphen-Framework entwickelt, wobei ein Tool zur Erstellung von Ontologien verwendet wurde, das entwickelt wurde. Dadurch wurde die Interoperabilität und die einfache Integration in den Rest der Feature Pipeline gewährleistet. Das bereichsspezifische Wissen wurde anhand von internen Modellierungsleitfäden und -regeln modelliert, die im Rahmen des Arbeitspakets erstellt wurden. Diese Ontologien erfassen

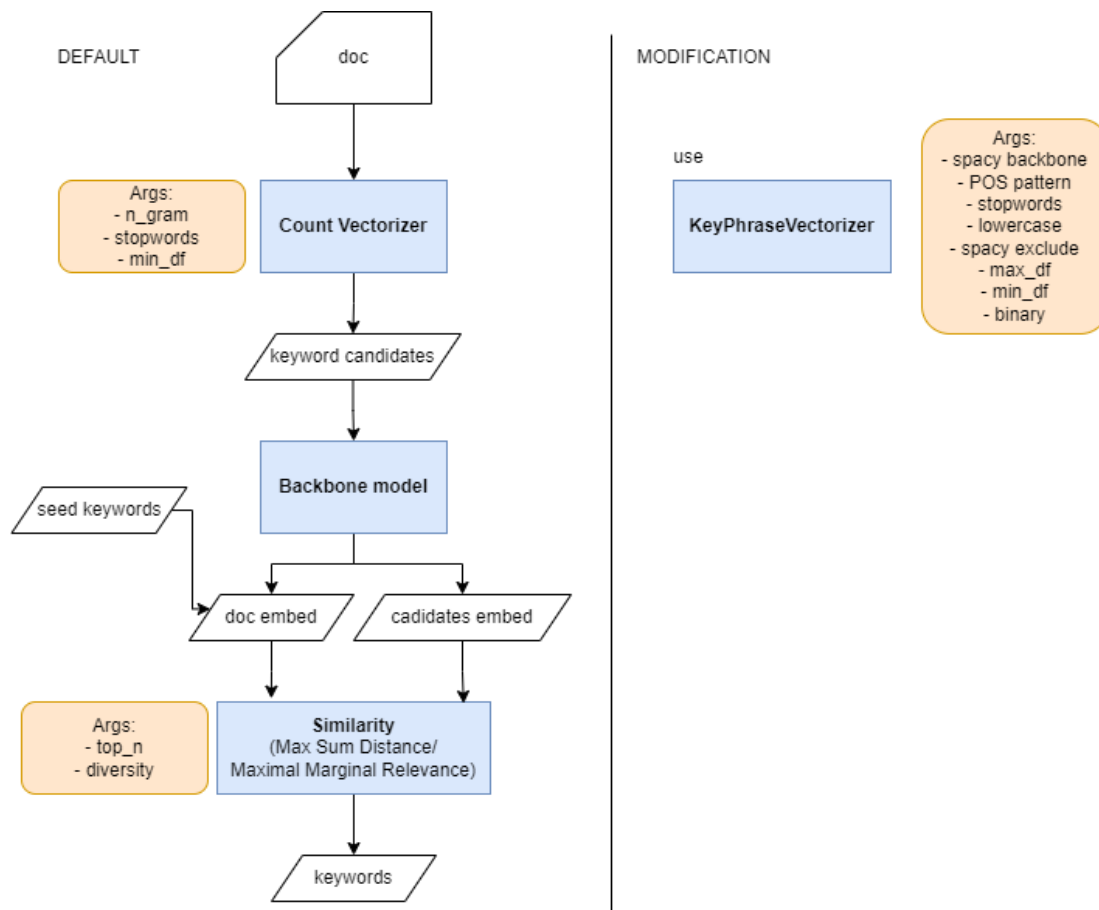


Figure 7: Keybert Overview

Wie oben beschrieben, besteht der Prozess aus 3 Hauptschritten:

1. find keyword candidates with a Count Vectorizer
2. calculate candidate embeddings and document embedding (if chapters are used as doc chapter embeddings are calculated instead of 1 single document embedding)
3. order keywords based on their similarity to the document embedding

Explanations of some terms are below:

Count Vectorizer:

Searches the document for all occurring `n_gram` phrases
 filters out `n_gram` phrases including given stopwords
 returns all `n_gram` phrases with `min_df` occurrences

KeyPhraseVectorizer:

searches the document for all occurring phrases which match POS pattern (based on spacy backbone)
 same workflow as Count Vectorizer otherwise

Backbone model:

word embedding model (usually a bert model)

Similarity:

Different methods are possible to calculate similarity (currently MMR is used)
keywords are ranked based on similarity and top_n keywords become output
Seed keywords:

list of keywords which are already representative of the document
optional input which can be used if useful

can be inputted as flat list (all keywords influence all docs) or as nested list (list of seed keywords per doc)

Erstellung von Relationen aus der Extraktion von Schlüsselwörtern:

Wörter, die mit der Vektor-Embedding-Methode der Schlüsselwort-Extraktion extrahiert wurden, können dann diese Einbettungen verwenden, um Ähnlichkeiten zu ermitteln.

Schlüsselwort-Ähnlichkeiten:

Nachdem die Schlüsselwörter in Wörteinbettungen umgewandelt wurden, gibt es eine numerische Darstellung jedes Wortes in einem 384-dimensionalen Raum. Die Koordinaten eines Schlüsselworts in diesem hochdimensionalen Raum stellen abstrakte Sprachmerkmale dar, die in ihrer Gesamtheit die semantische Bedeutung des Wortes repräsentieren.

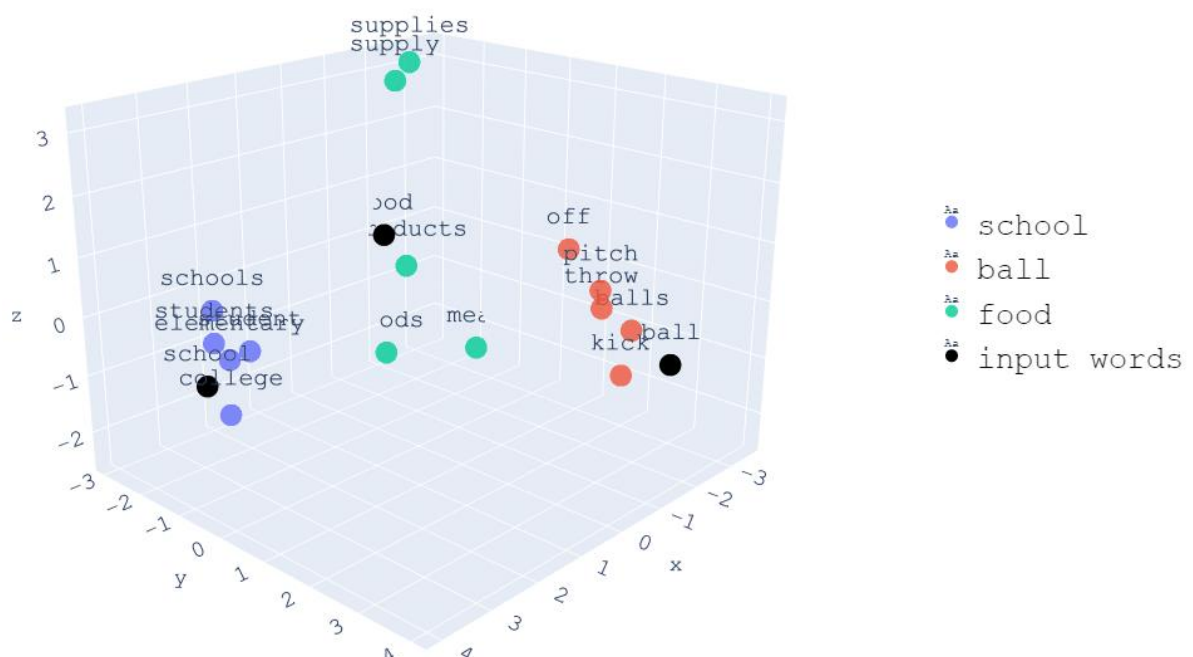


Figure 8: Example of Semantic Similarity in Vector Embeddings

Die numerischen Darstellungen der Schlüsselwörter werden verwendet, um die Ähnlichkeit zwischen ihnen zu quantifizieren, indem der Abstand zwischen den Paaren von Schlüsselwörtern berechnet wird. Dies geschieht durch die Berechnung der Cosinus-Ähnlichkeit, die den Winkel zwischen Vektoren betrachtet. Nach diesem ersten Schritt der Kantenbildung ergibt sich eine Ähnlichkeitsmatrix, in der jedes

Komplexität des Buches variiert. Die Ausgabe dieses Schrittes ist ein Datenrahmen mit Paaren von Quell- und Zielknoten. Die Zielknoten sind hier die Zentralknoten. Jeder Knoten eines Clusters erhält eine Kante, die ihn mit dem Zentralknoten des Clusters verbindet.

Beachten Sie, dass aufgrund der Art und Weise, wie bei der Affinitätsausbreitung nach Ähnlichkeiten gesucht wird, das Endergebnis vom Zufallszustand und der Reihenfolge, in der die Daten eingegeben werden, abhängig ist. Um reproduzierbare Ergebnisse zu erzielen, sollten die Eingabedaten für die Clustering-Methode zunächst sortiert und dann der Zufallsstatus festgelegt werden.

Cluster-Reinigung

Nachdem die Cluster erstellt wurden, werden sie bereinigt, um die Qualität des Graphen zu verbessern. Dies geschieht in drei Schritten:

1. Selbst-Verbindungen löschen

Verbindungen, bei denen der Quell- und der Zielknoten derselbe Knoten sind, werden gelöscht.

2. Verwerfen von kleinen Clustern

Cluster müssen aus mindestens zwei Knoten (einschließlich des Zentralknotens) bestehen, um ein echtes Cluster zu sein. Cluster, die nur aus einem Knoten bestehen, werden verworfen.

3. Verwerfen von schwach verbundenen Clustern

Dies ist der „Müllbeseitigungs“-Schritt, der im obigen Abschnitt über das Clustering erwähnt wurde. Schlüsselwörter ohne starke Verbindungen zum Rest des Textes neigen dazu, in einem „Müll-Cluster“ gruppiert zu werden. Dieser Cluster zeichnet sich dadurch aus, dass die Schlüsselwörter scheinbar nichts miteinander zu tun haben und dass die Verbindungen innerhalb des Clusters schwach sind. Cluster mit einer mittleren Ähnlichkeit unter einem festgelegten Schwellenwert werden verworfen.

Generierung weiterer Kanten innerhalb von Clustern

Schlüsselwörter sollten nicht nur mit ihrem Cluster-Mittelknoten verbunden sein, sondern auch Beziehungen zu anderen Schlüsselwörtern aufweisen. Im Schritt „Intracluster-Beziehungen“ wird der $n_{\text{intracluster}}$ nächste Nachbar zu den Verbindungen innerhalb jedes Clusters hinzugefügt. Das bedeutet, dass die $n_{\text{intracluster}}$ und ihre höchsten Ähnlichkeiten zwischen den Knoten im Cluster (ohne Zentralknoten) genommen und als Verbindungen in den Graphen aufgenommen werden. Die Anzahl der Intracluster-Verbindungen skaliert mit der Clustergröße und ist definiert als: $n_{\text{intracluster_relations}} = \text{number of nodes in cluster} // 2$

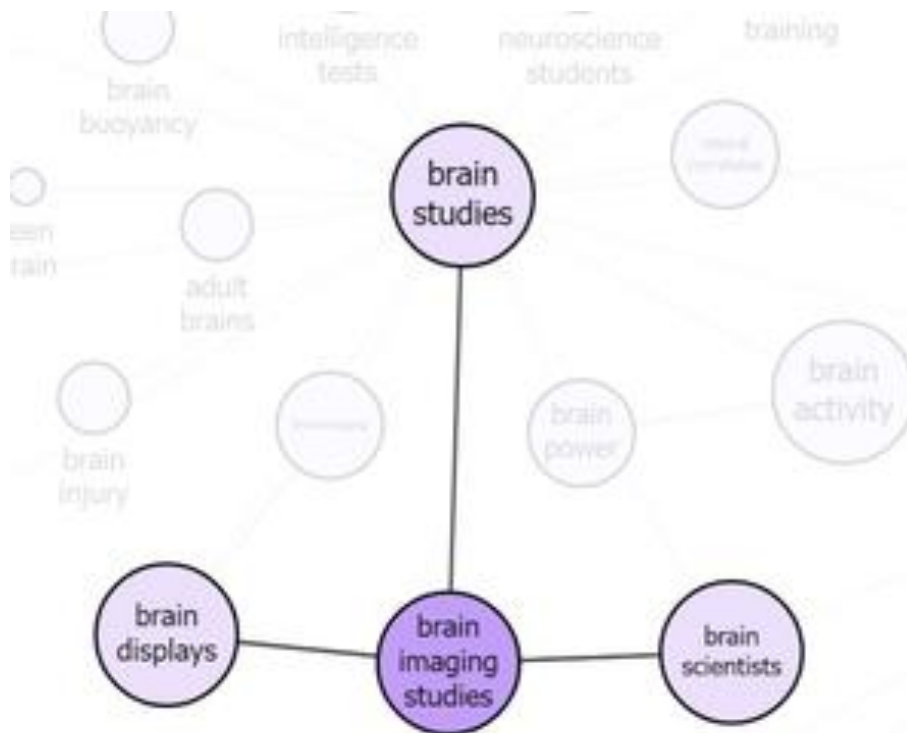


Figure 10: Example of a connected cluster

Generierung weiterer Kanten zwischen Clustern

Die Clustermethode erzeugt isolierte Cluster von Stichwörtern, die noch nicht zu einem zusammenhängenden Graphen verbunden sind. Starke Verbindungen zwischen Stichwörtern, die sich nicht im selben Cluster befinden, sollten einbezogen werden. Daher werden starke Verbindungen zwischen Clustern hinzugefügt, indem die $n_{\text{intercluster}}$ ähnlichsten Knoten, die sich nicht im selben Cluster befinden, einbezogen werden. Die Anzahl der Intercluster-Verbindungen skaliert mit der Anzahl der Cluster und ist definiert als:

$$n_{\text{intercluster_relations}} = \text{number of clusters} // 3 * 2.$$

Diese relative Anzahl von Verbindungen wurde pragmatisch bestimmt, indem man die sinnvollen Ergebnisse in den resultierenden Diagrammen betrachtete.

Syntaktische Extraktion

Die syntaktische Extraktionskomponente extrahiert ähnlich wie die oben beschriebenen Einbettungsmethoden, die das KI-Modul anwendet, Schlüsselwörter und Beziehungen zwischen ihnen, um sie als Knoten bzw. Kanten im resultierenden Wissensgraphen zu verwenden. Im Gegensatz zu den Einbettungsmethoden, die auf der Ähnlichkeit zwischen Schlüsselwörtern beruhen und daher ungerichtete Kanten liefern, kann die syntaktische Extraktion auch gerichtete Kanten liefern (d.h. Taxonomie / Typ 1 oder Lernordnung / Typ 2).

Bei der syntaktischen Extraktion werden semantische Beziehungen zwischen Begriffen, die in einem Text kommuniziert werden, anhand von syntaktischen Mustern ermittelt. Die Organisation basiert auf der klischeehaften Unterscheidung zwischen Syntax und Semantik: Eine semantische Relation, wie z. B. Kausalität, wird durch eine

entsprechende syntaktische Relation erfasst, d. h. eine Gruppe von grammatischen Mustern, die Kausalität in einer bestimmten Sprache widerspiegeln sollten.

Die Komponente Syntaktische Extraktion folgt einem NLP-Ansatz, der Part-of-Speech-Tagging und Dependency Parsing auf der Grundlage der spaCy-Bibliothek verwendet.

Linguistic Background:

In einem natürlichsprachlichen Text ist die Semantik einer Beziehung wie

has_subtype(x, y)

entspricht einer bestimmten Menge von syntaktischen Realisierungen in einer bestimmten natürlichen Sprache. In einer Dependenzgrammatik dargestellt, umfasst diese Menge für die englische Sprache z. B.

$(y)_N \leftarrow (is)_V \rightarrow (a\ type)_N \rightarrow (of)_{ADP} \rightarrow (x)_N$

$(y)_N \leftarrow (is)_V \rightarrow (a\ category)_N \rightarrow (within)_{ADP} \rightarrow (x)_N$

$(x)_N \leftarrow (includes)_V \rightarrow (the\ class)_N \rightarrow (of)_{ADP} \rightarrow (y)_N$

$(y)_N (is)_{AUX} \leftarrow (defined)_V \rightarrow [(as)_{ADP} \rightarrow (an\ x)_N] [(which)_{ADP} \leftarrow (...)_V \rightarrow \dots\dots\dots]$

etc.

Jede sprachspezifische Menge von syntaktischen Mustern ist als unbegrenzt zu betrachten, da es für jede semantische Relation eine unendliche Anzahl von möglichen syntaktischen Realisierungen gibt. Das Ziel der syntaktischen Extraktionskomponente ist es, die häufigsten und zuverlässigsten Muster im Text zu erfassen.

Übersicht Architektur und Workflow:

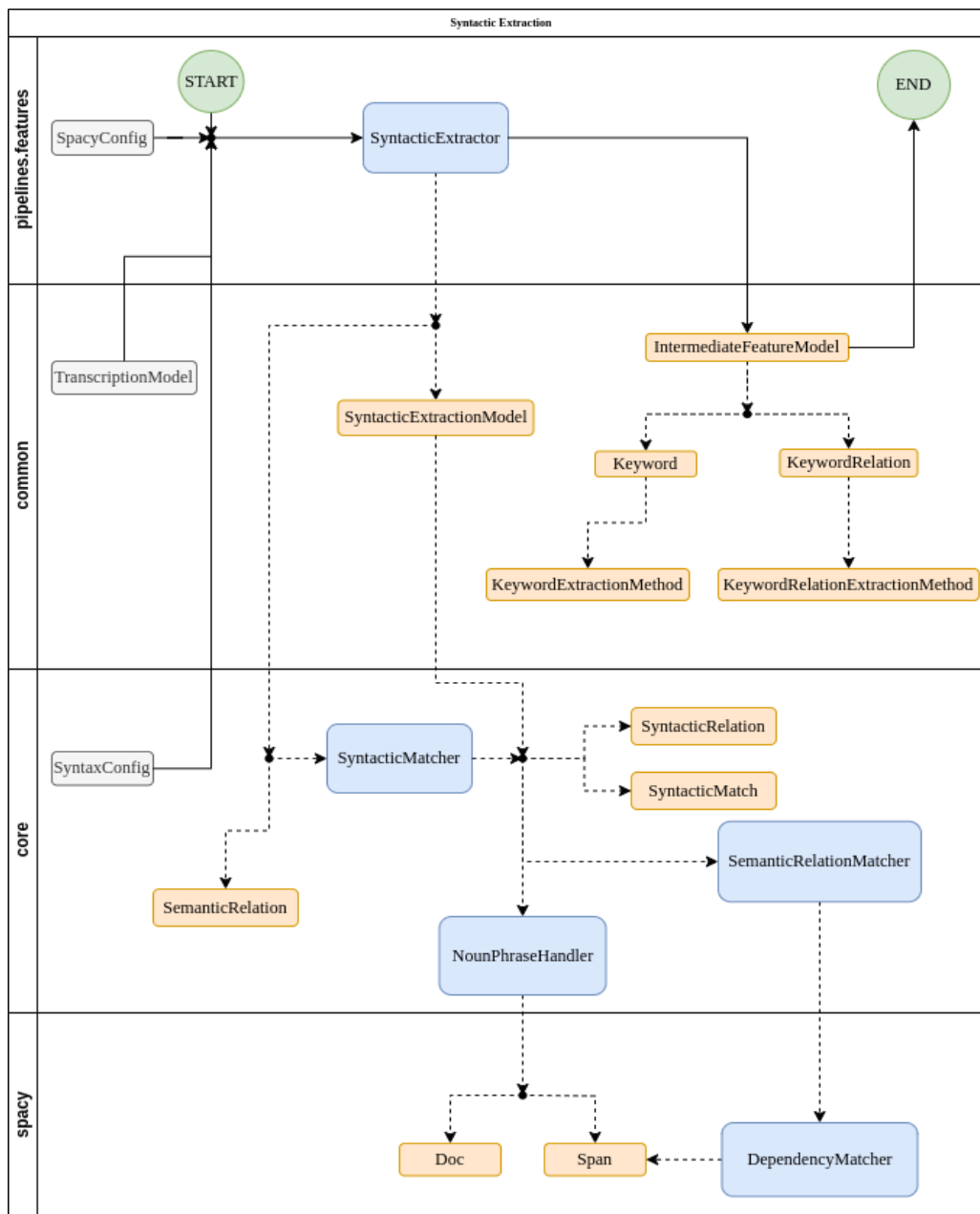


Figure 11: Syntactic Pattern Architecture and Workflow

Die Klasse `syntactic extractor` bietet die notwendige Schnittstelle zur Feature-Pipeline. Sie nimmt ein Standard-`TranscriptionModel` und gibt ein `IntermediateFeatureModel` zurück, dessen Schlüsselwörter und Schlüsselwortrelationen auf der Grundlage der gegebenen Konfiguration syntaktisch extrahiert werden.

Der eigentliche Musterabgleich wird an den syntaktischen `Matcher` delegiert, eine Kernfunktion des `feature_processors`, bei dem `spacy` zum Einsatz kommt. Der syntaktische Abgleicher liefert eine Liste syntaktischer Übereinstimmungen auf der Grundlage des konfigurierten Sprachmodells und der syntaktischen Beziehungen. Der syntaktische `Matcher` umfasst außerdem zwei Hilfsklassen. Der Substantivphrasen-Handler ist für die Zusammenführung von Substantivphrasen einer

bestimmten Form vor dem Abgleich verantwortlich, während der semantische Beziehungsabgleicher ein Wrapper des Abhängigkeitsabgleichers von spacy ist, der benötigt wird, um zusätzliche Informationen zu übermitteln, die dem letzteren fehlen, vor allem die Richtungsabhängigkeit des Abgleichs.

Die Syntax für syntaktische Muster wird in der Konfigurationsdatei `syntax_config.yml` des syntaktischen Extraktionsmoduls verwendet, anstatt direkt die (für unsere Zwecke) unnötig aufgeblähte Syntax von spaCy zu verwenden. Die wichtigsten Bausteine sind *Token expressions* which are represented as a tuple (`dependency_label`, `raw_token`, `pos_tag`), e.g. (`prep`, `to`, `ADP`), that may further be designated as a source (`prep`, `to`, `ADP`, `S`) or as a target (`prep`, `to`, `ADP`, `T`); and

Pattern expressions which are tree structures recursively built from token expressions: `token_expr_root [pattern_expr_1, ..., pattern_expr_m]`

Zum Beispiel in dem Satz: "Atoms form molecules through chemical bonds" in einem Ausgangstext, der Musterausdruck

`(*, form, VERB)[(nsubj, *, NOUN, T), (dobj, *, NOUN, S)]`

extrahiert die semantische Beziehung `consists_of(molecules, atoms)`.

Large-Language-Model Funktionen

Aufgrund des technologischen Durchbruchs von Large-Language-Models (LLMs) während der Projektentwicklungszeit wurde die Entscheidung getroffen, diese Technologie zu nutzen, um die Graphenerstellungspipeline voranzutreiben. Es war von größter Bedeutung, generative KI nicht ohne Kontrolle und Transparenz einzusetzen. Aus diesem Grund wurden LLMs auf zwei Arten eingesetzt. Die erste bestand darin, Zusammenfassungen zu erstellen, um mehr Inhalt hinter den Knoten einzuschließen, aber durch die Nutzung der Satz-IDs und Kapiteltitle aus der Transkriptionspipeline konnte die Eingabe für diese Zusammenfassungen kontrolliert werden und die Quellen konnten transparent eingesehen werden. Die zweite Methode verwendet LLMs, um Graphen zu bereinigen, indem Knoten und Kanten von geringerer Qualität entfernt werden.

Large-Language-Model Zusammenfassungen

Das Arbeitspaket implementierte erfolgreich ein Zusammenfassungssystem, das ein großes Sprachmodell (LLM) verwendet, um Zusammenfassungen aus extrahierten Sätzen und Kapiteltitle zu erstellen. Durch die Nutzung kontrollierter Eingaben gewährleistet dieser Ansatz einen strukturierten und transparenten Zusammenfassungsprozess, bei dem die Nutzer den Inhalt der Zusammenfassung bis zu ihren ursprünglichen Quellen zurückverfolgen können.

Das System wurde so konzipiert, dass es Schlüsselsätze und Kapiteltitle als Input extrahiert und so einen kuratierten und eingeschränkten Datensatz für das LLM bereitstellt, um kohärente und kontextbezogene Zusammenfassungen zu erstellen. Diese Methode minimierte Halluzinationen und bewahrte die Treue zum Originaltext, was die Zuverlässigkeit erhöhte. Es wurde ein Gleichgewicht zwischen Prägnanz und

Informativität erreicht, das sicherstellte, dass die Zusammenfassungen wesentliche Details enthielten und dennoch lesbar blieben. Die Bewertung umfasste qualitative und quantitative Beurteilungen, die bestätigten, dass die Zusammenfassungen den extrahierten Inhalt genau wiedergaben und gleichzeitig den logischen Fluss beibehielten. Die Implementierung unterstützt künftige Verfeinerungen, einschließlich einer domänenspezifischen Abstimmung und einer benutzergesteuerten Eingabegewichtung, wodurch die Anpassungsfähigkeit und Vertrauenswürdigkeit bei der automatischen Zusammenfassung weiter verbessert wird.

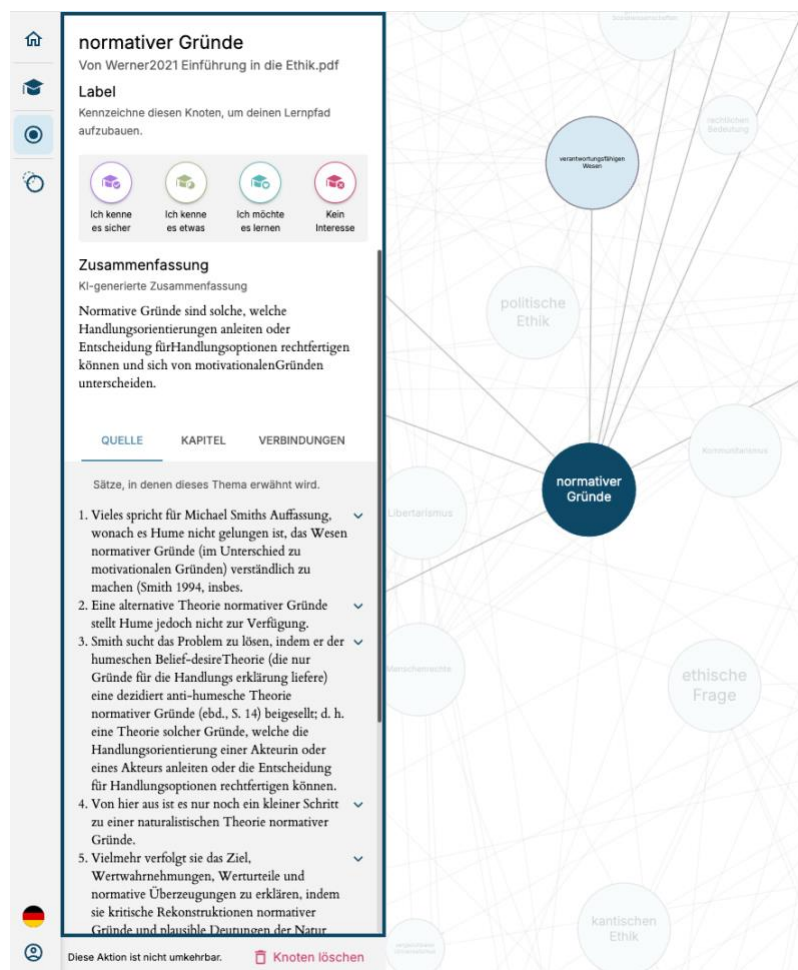


Figure 12: Example of a Generated Summary

Large-Language-Model Graphen Pruning

Mit Hilfe eines LLM wurde ein System zur Bereinigung von Wissensgraphen implementiert, um komplexe Graphen zu verfeinern, indem die weniger relevanten Knoten und Kanten identifiziert und entfernt werden. Dieser Ansatz gewährleistete die Erhaltung wesentlicher Strukturen und optimierte den Graphen hinsichtlich Effizienz und Interpretierbarkeit.

Das LLM wurde eingesetzt, um die Relevanz von Knoten und Kanten auf der Grundlage von kontextueller Bedeutung, Konnektivität und domänenspezifischen Kriterien systematisch zu bewerten. Die Pruning-Kontrollen wurden sorgfältig kalibriert, um eine übermäßige oder unzureichende Entfernung zu verhindern und die Integrität des Graphen zu erhalten, während die Komplexität reduziert wurde. Das System passte die

Schwellenwerte dynamisch an, um Schwankungen in der Graphendichte und Informationsbedeutung zu berücksichtigen.

Es wurde eine Validierungslogik erstellt, um die Auswirkungen des Pruning zu bewerten und sicherzustellen, dass Schlüsselbeziehungen und wichtiges Wissen erhalten bleiben. Vor allem aber wurde eine harte logische Regel eingeführt, um zu verhindern, dass das LLM während des Bereinigungsprozesses weitere Informationen hinzufügt. Die Funktion war nur dazu gedacht, Inhalte zu bereinigen (oder zu entfernen), nicht aber, sie hinzuzufügen. Aufgrund der unvorhersehbaren Natur von LLMs neigen sie jedoch dazu, dennoch Inhalte hinzuzufügen. Um dies zu verhindern und sicherzustellen, dass Inhalte nur beschnitten und nicht erweitert werden, wurde eine harte logische Regel eingeführt, die sicherstellt, dass alle zusätzlichen Schlüsselwörter oder Beziehungen, die hinzugefügt wurden, anschließend von einem Standardkontron außerhalb des LLM gelöscht werden. Die endgültige Implementierung zeigte eine verbesserte Handhabbarkeit des Graphen, die nachgelagerte analytische Prozesse unterstützt und gleichzeitig den Nutzen der ursprünglichen Wissensstruktur bewahrt. Der Ansatz ermöglicht eine weitere Anpassung und Verfeinerung auf der Grundlage der sich entwickelnden Anforderungen des Bereichs.

Wissensgraph

Wesentliche Struktur und Zoomfunktion

Es wurde ein interaktiver Wissensgraph erstellt, der eine nahtlose Navigation und Erkundung der verknüpften Daten ermöglicht. Das System ermöglicht die Visualisierung von Knoten und Kanten, die die von den oben beschriebenen KI-Pipelines geschaffenen Beziehungen effektiv darstellen. Die Benutzer können intuitiv mit dem Graphen interagieren und durch die Struktur mit sanften Übergängen zwischen verschiedenen Bereichen von Interesse navigieren.

Ein wesentliches Merkmal dieser Implementierung ist die Zoomfunktion, die es den Benutzern ermöglicht, den Graphen in vier verschiedenen Zoomstufen zu betrachten. Auf der höchsten Stufe bietet das System einen breiten Überblick, der wichtige Knotencluster und allgemeine Konnektivitätsmuster anzeigt. Beim Heranzoomen werden weitere Knoten mit unterschiedlichen Zoomstufen sichtbar, die eine immer feinere Granularität aufweisen. Auf der detailliertesten Zoomstufe können die Nutzer Knoten niedrigerer Ebenen untersuchen, die von der KI-Pipeline aufgrund ihrer höheren Granularität als solche gekennzeichnet werden.

Das Benutzererlebnis wird durch dynamische Rendering-Techniken verbessert, die fließende Übergänge zwischen den Zoomstufen gewährleisten und gleichzeitig die Klarheit und Lesbarkeit erhalten. Knoten und Kanten sind für die Sichtbarkeit auf jeder Ebene optimiert und passen sich dynamisch an, um Unordnung zu vermeiden und ein intuitives Layout zu erhalten. Leistungsoptimierungen wurden implementiert, um eine reaktionsschnelle Interaktion auch bei größeren Diagrammen zu gewährleisten.

Lernetiketten und personalisierte Lernpfade

Es wurde eine Funktion geschaffen, die eine Zuordnung des Lernstatus im Wissensgraphen ermöglicht. Dies ermöglicht es den Nutzern, Knoten auf der Grundlage ihrer Lerninteressen zu kategorisieren. Jeder Knoten kann mit einem von vier Status gekennzeichnet werden, der anzeigt, ob sie bereits etwas wissen oder es lernen wollen.

Diese Funktion ermöglicht es den Benutzern, ihre Erkundung des Graphen auf ihre persönlichen Lernziele und Wissensfortschritte abzustimmen. Durch die Zuweisung von Statuswerten schaffen die Benutzer ein dynamisches, personalisiertes Overlay auf dem Wissensgraphen, das sich an ihre Bedürfnisse anpasst. Knoten, die mit „Möchte lernen“ gekennzeichnet sind, weisen auf Themen hin, die den Nutzer interessieren und mit denen er sich befassen möchte, während Knoten mit der Kennzeichnung „Weiß ich schon“ dabei helfen, den Lernfortschritt zu verfolgen. Knoten, die mit „Nicht interessiert“ gekennzeichnet sind, werden visuell ausgeblendet, um sicherzustellen, dass das Diagramm auf relevantes Material fokussiert bleibt. Das System bietet eine intuitive Schnittstelle für die Aktualisierung dieser Status, so dass die Benutzer ihre Auswahl ändern können, wenn sich ihr Wissen weiterentwickelt.

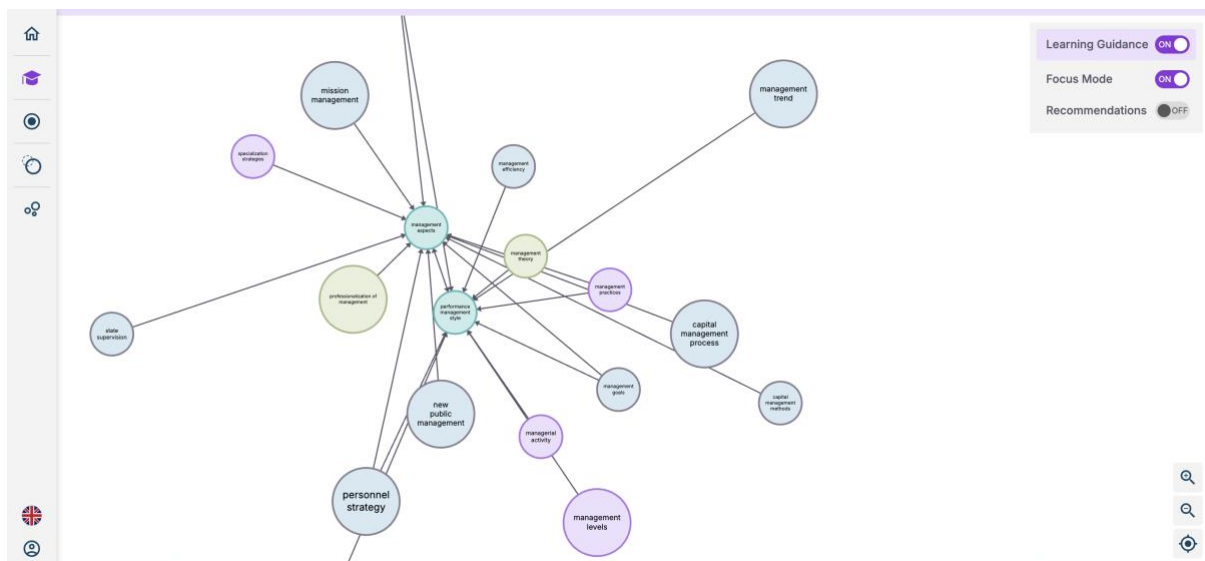


Figure 13: Personalised Pathways

Auf dieser Grundlage generiert der Wissensgraph bei Bedarf personalisierte Lernpfade, indem er die Struktur des Graphen und die Beziehungen zwischen den Knoten analysiert. Das System ermittelt optimale Lernsequenzen auf der Grundlage von Abhängigkeiten zwischen Konzepten. Wenn ein Benutzer ein bestimmtes Thema lernen möchte, kann der Graph vorausgesetzte Knoten vorschlagen, die zuerst verstanden werden sollten, und ihn durch eine logische Progression führen. Umgekehrt kann das System, wenn ein Benutzer bestimmte Themen bereits kennt, den Lernpfad so anpassen, dass redundantes Material übersprungen und der Fokus auf neue Konzepte gelenkt wird. Der Benutzer kann den Pfad auf verschiedenen Ebenen betrachten und Knoten außerhalb der Pfadstruktur anzeigen oder ausblenden. Die Anpassungsfähigkeit dieses Ansatzes gewährleistet, dass jeder Lernende ein individuelles Erlebnis erhält. Der Wissensgraph berechnet die Pfade dynamisch neu, wenn sich der Status ändert, und ermöglicht so eine flexible Erkundung und selbstbestimmtes Lernen. Durch die Nutzung der Verbindungen innerhalb des Graphen stellt das System nicht nur Informationen dar, sondern fördert auch einen strukturierten und effizienten Weg, sich Wissen anzueignen, das auf individuelle Interessen und Vorwissen zugeschnitten ist.

Durch die Nutzung von KI-generierten Zusammenfassungen bei gleichzeitigem Zugriff auf das Quellmaterial schafft diese Funktion ein Gleichgewicht zwischen Effizienz und Transparenz. Sie stellt sicher, dass die Nutzer Wissen schnell aufnehmen können, ohne auf die Möglichkeit zu verzichten, Details zu überprüfen und zu erforschen, was letztlich den Nutzen und die Zuverlässigkeit des Wissensgraphen als Lernwerkzeug erhöht.

Lernaktivitäten in der Grafik

Eine Funktion zur Bewertung von Verbindungen ermöglicht es den Benutzern, die Struktur des Wissensgraphen aktiv zu verfeinern und zu verbessern. Diese Funktion ermöglicht es den Nutzern, bestehende Verbindungen zwischen Knoten zu bewerten, neue Beziehungen hinzuzufügen, wo sie es für angebracht halten, und Verbindungen zu entfernen, die möglicherweise falsch oder irrelevant sind. Dadurch, dass die Benutzer die Kontrolle über die Konnektivität des Graphen haben, entwickelt sich das System dynamisch weiter und passt sich an neue Erkenntnisse und Perspektiven an.

Über die einfachen Verbindungen hinaus können die Benutzer auch Beziehungen durch Hinzufügen weiterer Details anreichern. Jede Kante zwischen Knoten kann zusätzliche Kontextinformationen enthalten, z. B. Erklärungen, wie Konzepte zusammenhängen, die Stärke oder Art der Verbindung oder Zitate, die die Verbindung unterstützen. Diese zusätzliche Detailebene verwandelt den Graphen von einer statischen Visualisierung in ein reichhaltiges, interaktives Wissensnetzwerk, in dem Beziehungen nicht nur sichtbar sind, sondern auch sinnvoll beschrieben werden.

Die Benutzeroberfläche für diese Funktion ist auf intuitive Interaktion ausgelegt, so dass die Benutzer die Verbindungen mit minimalem Aufwand nahtlos ändern können. Bei der Bewertung einer Verbindung können sie die vorhandenen Details überprüfen und entweder ihre Relevanz bestätigen oder Anpassungen vornehmen. Das Hinzufügen einer neuen Verbindung ist so einfach wie die Auswahl zweier Knoten und die Definition der Art ihrer Beziehung, während das Löschen einer Verbindung sicherstellt, dass veraltete oder falsche Assoziationen nicht bestehen bleiben.

Diese Funktion ermöglicht es den Nutzern, die Struktur des Graphen zu gestalten, und fördert so eine genauere und individuellere Darstellung des Wissens. Im Laufe der Zeit wird der Graph zu einem kollaborativen, sich entwickelnden Wissensraum, der sowohl etablierte Beziehungen als auch neue Erkenntnisse widerspiegelt, was ihn zu einem leistungsstarken Werkzeug für das Lernen und Entdecken macht.