Informatik
Hauptcampus

HOCHSCHULE TRIER

Extraktion und Analyse von Symbolnamen in beschreibungslogischen Ontologien

Paria Bolouki

Bachelor-Abschlussarbeit

Betreuer: Prof. Dr. Claudia Schon

Hochschule Trier, 17.04.2025

Inhaltsverzeichnis

1

Einleitung

1.1 Motivation und Problemstellung

Ontologien spielen eine zentrale Rolle bei der Wissensrepräsentation und -verarbeitung, insbesondere in Bereichen wie Biologie, Medizin und Informatik. Sie dienen dazu, Wissen in einer strukturierten Form darzustellen, die für Maschinen interpretierbar ist und in Anwendungen wie der Künstlichen Intelligenz genutzt werden kann [SS09].

Ontologien werden oft von Experten entwickelt, die die Symbolnamen innerhalb der Ontologie gezielt und bedeutungsvoll wählen. Diese Symbolnamen tragen häufig implizit semantische Informationen, die für das Verständnis und die Zielsetzung der Ontologie hilfreich sind. Ein zentrales Problem bei der automatisierten Verarbeitung dieser Ontologien ist jedoch, dass die Bedeutung der Symbolnamen in herkömmlichen Verfahren zur Inferenz kaum beachtet wird.

Claudia Schon beschreibt in [Sch24], dass im Gegensatz zu Menschen, die sich bei der Bewertung logischer Schlussfolgerungen auf relevante Informationen konzentrieren, automatisierte Theorembeweiser oft unnötige Inferenzen einbeziehen, was zu einer ineffizienten Ablenkung bei Beweisaufgaben führt. Um dieses Problem zu lösen, schlägt Schon einen Ansatz vor, der Techniken der natürlichen Sprachverarbeitung (NLP) nutzt, um das Kontextverständnis und die logische Stringenz in automatisierten Beweissystemen zu verbessern.

1.2 Zielsetzung und Forschungsfragen

Ziel dieser Arbeit ist es, zu untersuchen, welche sprachlichen und semantischen Informationen in Symbolnamen und deren Annotationen in beschreibungslogischen Ontologien enthalten sind und wie diese automatisiert erfasst und analysiert werden können. Neben den Symbolnamen stehen dabei insbesondere Annotationen wie `label`, `comment` und `definition` im Fokus, da sie häufig semantisch relevante Zusatzinformationen enthalten.

Teil der Analyse ist die automatische Extraktion von Keywords aus textuellen Annotationen wie Kommentaren und Definitionen. Dazu wird ein vortrainiertes Sprachmodell eingesetzt, das auf die Erkennung von Fachbegriffen spezialisiert

ist. Die extrahierten Keywords werden anschließend mit einem manuell erstellten Goldstandard verglichen und evaluiert.

Im Rahmen dieser Arbeit sollen folgende Forschungsfragen untersucht werden: Welche strukturellen und sprachlichen Muster lassen sich bei Symbolnamen und ihren Annotationen in verschiedenen Ontologien beobachten? Welche Annotationstypen kommen wie häufig in Ontologien vor und wie unterscheiden sie sich hinsichtlich ihrer semantischen Informationsdichte? Wie gut eignet sich ein transformerbasiertes Sprachmodell zur automatischen Keyword-Extraktion aus diesen Annotationen hinsichtlich semantischer Präzision und Relevanz? Und schließlich: Wie schneiden die automatisch extrahierten Keywords im Vergleich zu einem manuell erstellten Goldstandard ab?

1.3 Aufbau der Arbeit

Das zweite Kapitel gibt zunächst einen Überblick über verwandte Arbeiten. Darauf folgt in Kapitel 3 die Darstellung der theoretischen Grundlagen: Neben der Rolle und Struktur von Ontologien werden zentrale Konzepte der Beschreibungslogik eingeführt. Ergänzend werden grundlegende Prinzipien des Natural Language Processing (NLP) erläutert und die Funktionsweise großer Sprachmodelle beschrieben. In Kapitel 4 wird die Methodik der Arbeit beschrieben. Zunächst wird analysiert, welche Symbolnamen und Annotationen in ausgewählten Ontologien vorkommen und wie diese strukturiert sind. Anschließend wird das Verfahren zur automatischen Keywords-Extraktion erläutert, das auf einem transformatorbasierten Sprachmodell beruht. Im fünften Kapitel folgt die Evaluierung der Extraktionsergebnisse anhand eines manuell erstellten Goldstandards. Kapitel 6 bildet den Abschluss mit einer Zusammenfassung der wichtigsten Erkenntnisse und einem Ausblick auf mögliche Weiterentwicklungen.

2

Verwandte Arbeiten

In den letzten Jahren hat sich zunehmend die Erkenntnis durchgesetzt, dass Symbolnamen in Ontologien nicht nur technische Bezeichner, sondern auch Träger semantischer Informationen sind. Diese Erkenntnis hat zu verschiedenen Ansätzen geführt, in denen symbolische Systeme um Verfahren aus der natürlichen Sprachverarbeitung erweitert werden.

Ein Beispiel hierfür ist die Erweiterung klassischer Axiomselektionsmethoden wie SInE um semantische Ähnlichkeitsmaße, z.B. auf der Basis von Word Embeddings. Damit können auch Konzepte mit unterschiedlichen Benennungen, aber ähnlicher Bedeutung berücksichtigt werden. Erste Evaluationen zeigen, dass der Inferenzprozess dadurch zielgerichteter und effizienter wird [FKS19].

Ein Ansatz orientiert sich an assoziativen Strukturen, wie sie im menschlichen Denken vorkommen. Dabei werden symbolische Begriffe nach semantischer Ähnlichkeit miteinander verknüpft. Ein vektorbasiertes Verfahren nutzt Word Embeddings, um Symbolnamen in Vektoren zu überführen und so inhaltlich verwandte Aussagen zu identifizieren - ein Verfahren, das sich besonders für domänenoffene Wissensbasen eignet [Sch23].

Darauf aufbauend zeigt Claudia Schon in [Sch24], dass Symbolnamen als semantische Hinweise dienen können, um Beweissysteme zielgerichteter zu steuern. In ihrem Ansatz werden Symbolnamen mit Vektorrepräsentationen natürlicher Sprache verknüpft, wodurch ihre semantische Nähe zum jeweiligen Beweisziel quantifiziert werden kann. Auf diese Weise kann die Relevanz einzelner Aussagen kontextsensitiv bewertet und die Auswahl von Inferenzkandidaten deutlich fokussierter gestaltet werden. Dies reduziert unnötige Ableitungen und macht die Beweisführung insgesamt effizienter.

Ein weiterentwickelter Ansatz verfolgt die sogenannte SeVEn-Strategie, bei der große Sprachmodelle wie Sentence-BERT verwendet werden [JS24]. Dabei werden Axiome und Beweisziele zunächst in natürliche Sprache übersetzt und anschließend in Vektoren repräsentiert. Auf Basis dieser kontextabhängigen Repräsentationen können Aussagen mit semantisch verwandtem Inhalt identifiziert werden - auch wenn die Beziehung zwischen ihnen nur implizit vorhanden ist. Dies ermöglicht insbesondere in domänenoffenen Wissensbasen eine gezieltere Auswahl relevanter Wissenseinheiten.

Diese Arbeiten zeigen, dass Symbolnamen mehr als nur technische Bezeichner sind. Sie enthalten sprachliche und konzeptuelle Informationen, die - richtig interpretiert - den Zugang zu semantischem Wissen erheblich verbessern können. Inwieweit sich diese Ansätze systematisch auf große, realweltliche Ontologien übertragen lassen, ist noch offen.

Die Extraktion von Keywords aus Ontologie-Annotationen spielt in dieser Arbeit eine unterstützende Rolle für die semantische Analyse. Aktuelle Studien zeigen, dass große Sprachmodelle (Large Language Models, LLMs) in der Lage sind, solche Annotationen nicht nur syntaktisch, sondern auch semantisch auszuwerten. Chataut [CDG+24] demonstrieren, dass LLMs durch gezieltes Prompt-Design relevante Begriffe aus komplexen, annotierten Texten extrahieren können. Sie weisen jedoch auch auf Herausforderungen wie die Generierung irrelevanter oder halluzinierter Begriffe hin, die einer kritischen Evaluation bedürfen.

Ein verwandter Ansatz wird von [LCL+24] vorgestellt, der mit Generative Pretrained Transformers enhanced with Ontology Narration ein System präsentiert, das GPT-4 nutzt, um formale Ontologiebegriffe in natürlichsprachliche Beschreibungen zu überführen. Diese dienen als Grundlage für eine präzisere Annotation biologischer Daten und erleichtern die Erkennung semantisch passender Konzepte.

Einen breiteren Fokus verfolgt die Studie [GDA24], die das Potenzial großer Sprachmodelle für verschiedene Aufgaben des Ontologie-Lernens untersucht. Dazu gehören die Typisierung von Begriffen (Term Typing), die Ableitung taxonomischer Strukturen sowie die Identifikation nicht-hierarchischer Relationen zwischen Konzepten. Die Ergebnisse zeigen, dass LLMs in der Lage sind, relevante Begriffe aus domänenspezifischen Texten zu erkennen, zu klassifizieren und in ontologische Strukturen zu überführen - ein Ansatz, der insbesondere bei offenen oder unstrukturierten Wissensquellen von Bedeutung ist.

3

Theoretische Grundlagen

3.1 Definition und Rolle von Ontologien

Der Artikel [BHL+14] beleuchtet die Definition von Ontologien aus unterschiedlichen Perspektiven, insbesondere der Philosophie und der Informatik. In der Philosophie bezeichnet Ontologie traditionell die 'Lehre vom Sein', während sie in der Informatik als formale Definition von Begriffen und deren Beziehungen innerhalb einer spezifischen Anwendungsdomäne, beispielsweise im Tourismus, in der Biologie oder im Rechtswesen, verstanden wird. Solche formalen Ontologien dienen als gemeinsame Wissensbasis, die sowohl den Austausch zwischen Computersystemen als auch zwischen Mensch und Maschine erleichtert. Sie sollen semantische Missverständnisse reduzieren und eine standardisierte Kommunikation ermöglichen.

Ontologien spielen eine zentrale Rolle im Semantic Web, das darauf abzielt, die Bedeutung von Informationen zu erfassen, anstatt wie herkömmliche Suchmaschinen lediglich Schlüsselwörter abzugleichen. Busse et al. illustrieren dies am Beispiel einer Hotelsuche: Während eine klassische Suchmaschine Begriffe wie „kinderfreundlich“ und „Strand“ nur syntaktisch vergleicht und dadurch irrelevante Ergebnisse liefern kann, nutzt das Semantic Web Ontologien, um Begriffe semantisch zu strukturieren. So werden etwa Synonyme wie „kinderfreundlich“ und „familienfreundlich“ als gleichbedeutend erkannt oder geografische Beziehungen wie „Norddeutschland“ und „Nord- und Ostsee“ sinnvoll verknüpft. Dadurch können Anfragen so interpretiert werden, dass sie dem menschlichen Verständnis möglichst nahekommen: Begriffe werden nicht nur analysiert, sondern auch im Kontext erfasst, sofern dieser auch explizit in der Ontologie repräsentiert ist, und logisch miteinander in Beziehung gesetzt. Ontologien dienen dabei als Wissensbasis, die präzisere und kontextbezogene Antworten ermöglicht [BHL+14].

3.2 Beschreibungslogische Ontologien

Beschreibungslogiken (Description Logics, DLs) sind formale Wissensrepräsentationssprachen, die zur präzisen Modellierung von Konzepten, Relationen und Individuen entwickelt wurden. Sie ermöglichen nicht nur die strukturierte Darstellung von Wissen, sondern auch die Ableitung neuer Informationen durch logische

Schlussfolgerungen. DLs bilden die theoretische Grundlage für Ontologiesprachen wie die Web Ontology Language (OWL), die im Semantic Web eine zentrale Rolle spielt. Ein wesentliches Merkmal von DLs ist ihre Fähigkeit zur logischen Schlussfolgerung, die es erlaubt, aus vorhandenen Fakten neues Wissen abzuleiten. Aufgrund ihrer formalen Eigenschaften sind DLs heute ein zentrales Werkzeug der Ontologie-Modellierung und spielen eine wichtige Rolle in wissensbasierten Systemen [BCM+03, KSH14].

$\mathcal{ALC}$

$\mathcal{ALC}$ ist eine der einfachsten und gleichzeitig am weitesten verbreiteten Beschreibungslogiken. Der Name $\mathcal{ALC}$ steht für *Attributive Concept Language with Complements* und beschreibt eine Basissprache, die zur formalen Beschreibung von Wissen entwickelt wurde. Sie ermöglicht die Modellierung von Konzepten, Rollen und Individuen in einer Domäne und wird zur Wissensrepräsentation verwendet. Aufgrund ihrer Klarheit und Entscheidbarkeit bildet die $\mathcal{ALC}$ die Grundlage für viele Erweiterungen und spielt eine zentrale Rolle in Ontologiesprachen wie OWL. Die wesentlichen Bausteine zur Definition einer logischen Sprache sind die Syntax und die Semantik [BCM+03].

Syntax von $\mathcal{ALC}$-Konzepten

In $\mathcal{ALC}$ bestehen Konzepte aus atomaren Konzepten, Rollen und komplexen Konstruktoren, die zur Definition neuer Konzepte verwendet werden können [BCM+03]. Die folgende Grammatik definiert formal die möglichen Konzepte in $\mathcal{ALC}$ [BCM+03]:

In der Darstellung stehen C und D für beliebige Konzeptbeschreibungen und R für eine Rolle. Atomare Konzepte werden durch A bezeichnet.

- A – Atomare Konzepte
- $\top$ – Universelles Konzept
- $\bot$ – Leeres Konzept
- $\neg C$ – Komplement
- $C \sqcap D$ – Schnittmenge
- $C \sqcup D$ – Vereinigung
- $\exists R.C$ – Existenzquantifizierung
- $\forall R.C$ – Allquantifizierung

Angenommen, wir betrachten eine Domäne, in der es die atomaren Konzepte Mensch und Student gibt, sowie die Rolle hatKind, die eine Beziehung zwischen Individuen ausdrückt. Beispielsweise beschreibt das Konzept Mensch $\sqcap \neg$Student alle Individuen, die als Menschen, aber nicht als Studenten klassifiziert sind. Die Existenzquantifizierung kann verwendet werden, um auszudrücken, dass es Individuen gibt, die mindestens ein Kind haben:

$$\exists \text{hatKind}.\top$$

beschreibt alle Menschen, für die eine Beziehung hatKind existiert, unabhängig davon, welche Eigenschaften dieses Kind hat. Eine spezifischere Aussage wäre:

$$\exists \text{hatKind}.\text{Student}$$

Diese Aussage beschreibt alle Personen, die mindestens ein Kind haben, das ein Student ist.

Semantik von $\mathcal{ALC}$-Konzepten

Die Semantik der $\mathcal{ALC}$ wird durch Interpretationen definiert. Eine Interpretation ist ein Paar $I = (\Delta^I, \cdot^I)$, wobei Δ^I die Domäne - die Menge aller möglichen Objekte - und $\cdot^I$ eine Interpretationsfunktion ist, die Konzepte auf Teilmengen von Δ^I und Rollen auf binäre Relationen abbildet [BCM+03]. Die Interpretationen der Konzeptkonstruktoren sind [BCM+03]:

- $\top^I = \Delta^I$ (Jedes Individuum gehört zu $\top$.)
- $\bot^I = \emptyset$ (Kein Individuum gehört zu $\bot$.)
- $(\neg C)^I = \Delta^I \setminus C^I$ (Die Menge aller Objekte, die nicht Teil von C sind)
- $(C \sqcap D)^I = C^I \cap D^I$ (Die Menge der Individuen, die zu beiden Konzepten gehören)
- $(C \sqcup D)^I = C^I \cup D^I$ (Die Menge der Individuen, die zu mindestens einem der beiden Konzepte gehören)
- $(\exists R.C)^I = \{x \in \Delta^I \mid \exists y \in C^I, (x, y) \in R^I\}$ (Die Menge aller Objekte, die mindestens eine Beziehung R zu einem Individuum aus C haben)
- $(\forall R.C)^I = \{x \in \Delta^I \mid \forall y \in \Delta^I, (x, y) \in R^I \Rightarrow y \in C^I\}$ (Die Menge aller Objekte, die nur Beziehungen R zu Elementen aus C haben)

Das folgende Beispiel zeigt, wie Bedeutung durch Syntax und Semantik ausgedrückt wird. Betrachten wir dazu eine Domäne mit den Individuen Gol, Raana und Paria:

$$\Delta^{\mathcal{I}} = \{\text{Gol}, \text{Raana}, \text{Paria}\}$$

und folgende Interpretationen:

$$\begin{aligned} \text{Student}^{\mathcal{I}} &= \{\text{Gol}, \text{Paria}\} \\ \text{Arbeitnehmer}^{\mathcal{I}} &= \{\text{Paria}, \text{Raana}\} \\ \text{hatKind}^{\mathcal{I}} &= \{(\text{Gol}, \text{Raana})\} \end{aligned}$$

Dann gilt:

$$\begin{aligned} (\text{Student} \sqcap \text{Arbeitnehmer})^{\mathcal{I}} &= \{\text{Paria}\} \\ (\exists \text{hatKind}.\top)^{\mathcal{I}} &= \{\text{Gol}\} \\ (\exists \text{hatKind}.\text{Arbeitnehmer})^{\mathcal{I}} &= \{\text{Gol}\} \end{aligned}$$

Nach der Einführung der grundlegenden Bausteine der $\mathcal{ALC}$-Beschreibungslogik stellt sich die Frage, wie ein solches formales System zur Modellierung von Wissen in einer konkreten Domäne eingesetzt werden kann. Wie in [BCM+03] beschrieben, besteht eine Wissensbasis in der Beschreibungslogik typischerweise aus zwei zentralen Komponenten: der TBox (Terminological Box) und der ABox (Assertional Box). Beide Elemente sind eng miteinander verknüpft und bilden die Grundlage für die Repräsentation und Verarbeitung von Wissen in vielen wissensbasierten Systemen.

TBox

Die TBox (Terminological Box) ist ein zentraler Bestandteil einer Wissensbasis in der Beschreibungslogik. Sie enthält intensionales Wissen, d.h. allgemeine Definitionen von Konzepten und deren Beziehungen. Die Syntax der TBox beschreibt die zulässige Struktur von Konzeptdefinitionen, d. h. wie Konzepte mit Hilfe von logischen Konstruktoren gebildet werden können. Die Semantik legt fest, welche Bedeutung diesen Ausdrücken im Interpretationsmodell zukommt - z. B. wann ein Individuum zu einem bestimmten Konzept gehört [BCM+03].

Die grundlegenden Aufgaben der TBox umfassen [BCM+03]:

1. **Definition von Konzepten:**
   Konzepte werden in der TBox definiert, oft durch logische Ausdrücke, die andere Konzepte oder Rollen kombinieren. Zum Beispiel kann das Konzept „Mutter“ definiert werden als:

   $$\text{Mutter} \equiv \text{Person} \sqcap \exists \text{hatKind}.\top \sqcap \text{Weiblich}$$

   Dies beschreibt eine Mutter als eine Person, die mindestens ein Kind hat und weiblich ist.

2. **Subsumtion und Hierarchien:**
   Die Konzepte in einer TBox können in einer hierarchischen Struktur angeordnet werden. Diese Struktur basiert auf Subsumtion, d.h. auf Beziehungen zwischen Ober- und Unterklassen. Beispielsweise kann das Konzept Student als Unterklasse des Konzepts Person definiert werden:

   $$\text{Student} \sqsubseteq \text{Person}$$

   Semantisch bedeutet dies, dass jede Instanz von Student auch eine Instanz von Person ist.

3. **Klassifikation:**
   Neue Konzepte können in die Hierarchie eingefügt werden, indem ihre Beziehung zu bestehenden Konzepten überprüft wird. Dies geschieht durch die Subsumtionsprüfung, die sicherstellt, dass die neuen Konzepte korrekt in die bestehende Struktur integriert werden.

ABox

Die ABox (Assertional Box) enthält das extensionale Wissen einer Wissensbasis in der Beschreibungslogik. Sie beschreibt konkrete Aussagen über Individuen, d.h. welche Objekte existieren und wie sie zueinander in Beziehung stehen. Die Semantik legt dabei fest, welche Individuen zu welchen Konzepten gehören und welche Rollenbeziehungen zwischen ihnen gelten. Im Gegensatz zur TBox, die allgemeine Konzepte und ihre Beziehungen beschreibt, konzentriert sich die ABox auf konkrete Aussagen über Individuen. Diese werden in Form von *concept assertion* und *role assertion* festgehalten.

Eine Concept Assertion legt fest, dass ein bestimmtes Individuum zu einem bestimmten Konzept gehört [BCM+03]. Zum Beispiel haben die Aussagen:

$$\text{Person}(\text{Gol})$$
$$\text{Weiblich}(\text{Gol})$$

die Bedeutung, dass Gol eine weibliche Person ist. Falls es in der TBox bereits die Definition

$$\text{Mutter} \equiv \text{Person} \sqcap \exists \text{hatKind}.\top \sqcap \text{Weiblich}$$

gibt, kann aus dieser Assertion abgeleitet werden, dass Gol eine Mutter ist, wenn eine passende *role assertion* wie hatKind(Gol, Raana) existiert.

Eine Role Assertion beschreibt Beziehungen zwischen Individuen [BCM+03]. Zum Beispiel:

$$\text{hatKind}(\text{Gol}, \text{Raana})$$

bedeutet, dass Gol ein Kind namens Raana hat.

Die zentrale Schlussfolgerungsaufgabe in der ABox ist das *Instance Checking*, also die Überprüfung, ob ein bestimmtes Individuum zu einem Konzept gehört. Darüber hinaus gibt es weitere wichtige Inferenzdienste, die häufig auf Instance Checking basieren [BCM+03]:

Knowledge base consistency: Prüfung, ob die ABox für jedes Konzept mindestens ein gültiges Individuum enthält.

Gegeben:

- $\text{Mutter}(\text{Jonas})$
- $\text{Vater}(\text{Jonas})$
- $\text{Mutter} \sqcap \text{Vater} \sqsubseteq \bot$

Problem: Jonas kann nicht gleichzeitig Mutter und Vater sein, da diese Konzepte disjunkt definiert sind.

Realization: Bestimmung des spezifischsten Konzepts: Ermittlung des spezifischsten Konzepts, zu dem ein Individuum gehört.

Gegeben:

- Mensch(Paria)
- Student(Paria)
- Student $\sqsubseteq$ Mensch

Das spezifischste Konzept, zu dem Paria gehört, ist Student, weil es eine Unterklasse von Mensch ist.

Retrieval: Suche nach allen Individuen, die zu einem bestimmten Konzept gehören.

Gegeben:

- Student(Gol)
- Student(Paria)
- Arbeitnehmer(Jonas)

Alle Individuen, die Mitglieder des Konzepts Student sind: Paria, Gol.

Erweiterungen von $\mathcal{ALC}$

Die Beschreibungssprache $\mathcal{ALC}$ bildet die Grundlage für viele erweiterte Beschreibungslogiken, die entwickelt wurden, um die Ausdrucksstärke für verschiedene Anwendungen zu erhöhen. Obwohl $\mathcal{ALC}$ bereits über grundlegende Konzepte und Rollenoperatoren verfügt, sind diese für viele Anwendungen nicht ausreichend [BCM+03].
Erweiterungen von $\mathcal{ALC}$ lassen sich in zwei Kategorien einteilen [BCM+03]:

- Klassische Erweiterungen, deren Semantik innerhalb des bestehenden modelltheoretischen Rahmens von $\mathcal{ALC}$ definiert werden kann.
- Nicht-klassische Erweiterungen, die eine Erweiterung des modelltheoretischen Ansatzes erfordern.

Im Folgenden werden einige der klassischen Erweiterungen näher erläutert.

1. **Inverse Rollen**: Inverse Rollen ermöglichen Relationen in beide Richtungen zu modellieren. Die erweiterte Syntax von $\mathcal{ALC}$ mit inversen Rollen erlaubt es, zu jeder Rolle R die inverse Rolle R^- zu definieren, die ebenfalls eine gültige Rolle ist [BCM+03]. Die Semantik inverser Rollen R^- wird mathematisch wie folgt definiert [BCM+03]:

$$(R^-)^I = \{(o', o) \in \Delta^I \times \Delta^I \mid (o, o') \in R^I\}$$

Dies bedeutet, dass ein Paar (o', o) genau dann zur Interpretation von R^- gehört, wenn das umgekehrte Paar (o', o) zur Interpretation von R gehört [BCM+03].

Beispiel:

Während in $\mathcal{ALC}$ eine Relation wie hatKind nur in einer Richtung existiert, erlaubt eine inverse Rolle hatKind$^-$ die Umkehrung dieser Beziehung [BCM+03]:

$$\text{hatKind}^- = \text{istKindVon}$$

In $\mathcal{ALC}$ kann nicht ausgedrückt werden, dass zwei Rollen wie hatKind und istKindVon logisch miteinander verknüpft sind. Auch wenn beide getrennt definiert werden, bleibt der Zusammenhang zwischen ihnen unbeachtet. Erst die Erweiterung durch inverse Rollen in $\mathcal{ALCI}$ ermöglicht es, solche Beziehungen explizit als gegenseitig umkehrbar zu modellieren. Dadurch können zusätzliche Schlussfolgerungen gezogen werden, die in $\mathcal{ALC}$ nicht möglich wären. Die Ausdruckskraft der Logik wird dadurch erheblich erweitert. Inverse Rollen sind insbesondere für Datenbanken, die Wissensrepräsentation und die Ontologieentwicklung von großer Bedeutung und werden daher in modernen Ontologiesprachen wie OWL unterstützt [BCM+03].

2. **Rollenhierarchien**: In vielen Anwendungen stehen Rollen in einer hierarchischen Beziehung zueinander, so dass jede Instanz einer spezifischeren Rolle automatisch auch die allgemeinere Rolle erfüllt. Solche Strukturen werden durch Unterrollen modelliert, die eine Spezialisierung von Oberrollen darstellen. Die Beziehung zwischen einer Unterrolle und einer Oberrolle wird durch die Notation $R_1 \sqsubseteq R_2$ ausgedrückt, was bedeutet, dass jede Instanz von R_1 gleichzeitig auch eine Instanz von R_2 ist [BCM+03].
   Die Semantik dieser Hierarchie besagt, dass eine Unterrolle immer eine Teilmenge der Oberrolle ist. Formal wird dies durch

$$R_1^I \subseteq R_2^I$$

   ausgedrückt. Das bedeutet, dass für alle Individuen a, b, wenn $(a, b) \in R_1^I$ erfüllt ist, dann auch $(a, b) \in R_2^I$ erfüllt sein muss [BCM+03].

Beispiel:

Ein Beispiel für eine Rollenhierarchie ist die Beziehung zwischen den Rollen hatSohn und hatKind:

$$\text{hatSohn} \sqsubseteq \text{hatKind}$$

Diese Hierarchie bedeutet, dass jede Person, die einen Sohn hat, automatisch auch ein Kind hat.

Rollenhierarchien ermöglichen es, Rollen als eigenständige Entitäten zu behandeln und in einer Taxonomie zu organisieren. Damit können nicht nur Konzepte, sondern auch deren Relationen systematisch strukturiert werden, was die Modellierung komplexer Wissensstrukturen erleichtert und Inferenzmechanismen verbessert [BCM+03].

3. **Transitive Rollen**: In vielen Anwendungen beziehen sich Relationen nicht nur auf direkte Verbindungen, sondern sollen auch über mehrere Zwischenschritte hinweg gültig bleiben. Dies wird durch transitive Rollen ermöglicht, die sicherstellen, dass eine Beziehung auch für indirekte Verbindungen gültig ist [BCM+03].

   Die Syntax einer transitiven Rolle besteht darin, eine bestehende Rolle R explizit als transitiv zu deklarieren, was durch die Notation $Trans(R)$ ausgedrückt wird. Semantisch bedeutet dies, dass für alle a, b, c gilt: Wenn $(a, b) \in R^I$ und $(b, c) \in R^I$ gilt, dann muss auch $(a, c) \in R^I$ gelte [BCM+03].

   Beispiel:

   Angenommen, *Gol* ist die Vorgesetzte von *Raana*, und *Raana* ist die Vorgesetzte von *Paria*, dann folgt aus der Transitivität automatisch, dass *Gol* auch die Vorgesetzte von *Paria* ist. Dies kann formal als transitive Rolle definiert werden:

   $$\text{VorgesetzterVon}^+ = \text{VorgesetzterVon}$$

   Wenn also gilt:

   $$(\text{Gol}, \text{Raana}) \in \text{VorgesetzterVon}^I \quad \text{und} \quad (\text{Raana}, \text{Paria}) \in \text{VorgesetzterVon}^I$$

   dann folgt daraus:

   $$(\text{Gol}, \text{Paria}) \in \text{VorgesetzterVon}^I$$

   Dies zeigt, dass die Beziehung *VorgesetzterVon* nicht nur für direkte Vorgesetzte gilt, sondern auch für indirekte Beziehungen gilt.

4. **Nominals**: Nominals sind eine Erweiterung von Beschreibungslogiken, die es ermöglichen, bestimmte Individuen explizit als Teil eines Konzeptes zu definieren. Während in klassischen Beschreibungslogiken Konzepte in der Regel als Mengen von Individuen interpretiert werden, erlauben Nominals eine direkte Referenzierung einzelner Individuen innerhalb der Konzeptdefinition. Dies wird durch den *one-of*-Konstruktor repräsentiert, der die Form

   $$\{a_1, \ldots, a_n\}$$

   hat und ein Konzept bezeichnet, das genau aus den genannten Individuen besteht [BCM+03].
   Die Syntax eines Nominals verwendet die {}-Notation, um einzelne Individuen explizit zu definieren. Dadurch können Konzepte als Mengen definiert werden, die genau diese Individuen enthalten [BCM+03].
   Die Semantik eines Nominals wird wie folgt interpretiert:

$$\{a_1, \ldots, a_n\}^I = \{a_1^I, \ldots, a_n^I\}$$

Dies bedeutet, dass das Konzept genau die aufgelisteten Individuen enthält und keine weiteren Elemente umfasst [BCM+03].

Zusätzlich gibt es das *fills*-Konstrukt, das angibt, dass ein Individuum eine bestimmte Rolle in Bezug auf eine andere Entität erfüllt. Dies wird durch

$$\exists R.\{a\}$$

dargestellt und beschreibt alle Individuen, die mit dem Individuum a über die Rolle R verbunden sind [BCM+03].

Beispiel:

Ein Beispiel für Nominals wäre die Modellierung eines Teams in einem Unternehmen. Angenommen, es gibt ein Team, das aus genau drei Personen besteht:

$$\text{TeamMitglieder} \equiv \{\text{Gol}, \text{Raana}, \text{Paria}\}$$

Das bedeutet, dass das Konzept *TeamMitglieder* ausschließlich die Individuen Gol, Raana und Paria umfasst und keine weiteren Personen dazugehören.

Nominals bieten eine flexible Möglichkeit, sich direkt auf einzelne Entitäten zu beziehen. Sie ermöglichen eine präzisere Modellierung und verbessern die Ausdruckskraft der Beschreibungslogik, indem sie explizite Bezüge zu spezifischen Individuen herstellen [BCM+03].

3.3 Grundlagen der Natural Language Processing

Natural Language Processing (NLP) ist ein interdisziplinäres Forschungsgebiet, das Methoden der Informatik, der Künstlichen Intelligenz (KI) und der Linguistik kombiniert, um natürliche Sprache zu verarbeiten und zu verstehen [JM09].

Ein frühes Beispiel für NLP ist ELIZA, ein Chatbot, der in den 1960er Jahren entwickelt wurde. ELIZA konnte einfache Muster in den Benutzereingaben erkennen und darauf reagieren, ohne die tatsächliche Bedeutung der Texte zu verstehen.

Moderne NLP-Systeme verwenden dagegen leistungsfähige Sprachmodelle, die mit maschinellen Lernverfahren trainiert wurden, um Sprache auf einer tieferen Ebene zu analysieren [JM09].

Bevor ein Text in natürlicher Sprache verarbeitet werden kann, muss er normalisiert werden, was als Textnormalisierung bezeichnet wird. Die Textnormalisierung umfasst in der Regel drei Hauptaufgaben: Tokenisierung, Normalisierung von Wortformaten und Satzsegmentierung [JM09].

1. **Tokenisierung:** Die Tokenisierung ist ein wesentlicher erster Schritt im NLP. Sie bezeichnet den Prozess der Segmentierung von fortlaufendem Text in kleinere Einheiten, sogenannte Tokens. Diese Tokens können ganze Wörter, Wortbestandteile oder einzelne Zeichen sein. Die Tokenisierung ist von zentraler Bedeutung, da sie die Grundlage für die nachfolgenden Verarbeitungsschritte bildet. In der Praxis lassen sich im Wesentlichen zwei Arten der Tokenisierung unterscheiden: die regelbasierte (top-down) Tokenisierung und die subwordbasierte (bottom-up) Tokenisierung [JM09].

   - **Regelbasierte (top-down) Tokenisierung:** Die regelbasierte Tokenisierung folgt vordefinierten Regeln zur Segmentierung von Text. Dabei werden sprachspezifische Konventionen berücksichtigt, um eine möglichst genaue Zerlegung des Textes zu gewährleisten. Beispielsweise werden Satzzeichen wie Kommas und Punkte als eigene Token behandelt, da sie für die syntaktische Analyse relevant sind. Gleichzeitig ist es notwendig, Zahlen und Sonderzeichen in bestimmten Kontexten zusammenzuhalten, z.B. bei Währungsangaben, Datumsformaten oder Internetadressen [JM09].
     Ein weiteres Beispiel für die regelbasierte Tokenisierung ist die Behandlung von Clitic-Konstruktionen[1], d. h. verkürzten Wortformen mit Apostrophen. Hierbei werden Ausdrücke wie *„couldn't“* in die zwei Tokens *„could“* und *„n't“* aufgeteilt. Für bestimmte Anwendungen kann es auch erforderlich sein, mehrteilige Ausdrücke (*multiword expressions*) wie *„ice cream“* oder *„high school“* als eine Einheit zu belassen, um ihre semantische Bedeutung korrekt zu erfassen [JM09].
     Auch bei der Tokenisierung spielen sprachspezifische Herausforderungen eine wichtige Rolle. Während im Deutschen und Englischen Leerzeichen eine natürliche Wortgrenze markieren, gibt es in Sprachen wie Chinesisch, Japanisch oder Thai keine explizite Worttrennung. Stattdessen besteht die Herausforderung darin, Wortgrenzen korrekt zu identifizieren, da die chinesischen Schriftzeichen (Hanzi) nicht einzelnen Buchstaben, sondern Morphemen[2] entsprechen [JM09].

   - **Subword-basierte (bottom-up) Tokenisierung:** Neben der regelbasierten Tokenisierung gibt es die Möglichkeit, Wörter in kleinere Subwords oder Zeichenfolgen zu zerlegen. Diese bottom-up Ansätze sind besonders relevant, wenn es darum geht, unbekannte oder seltene Wörter effizient zu verarbeiten. Während regelbasierte Verfahren davon ausgehen, dass alle Wörter im Vokabular vorhanden sind, erlauben subword-basierte Verfahren eine flexiblere Modellierung, indem Wortbestandteile wie Präfixe, Suffixe oder andere bedeutungstragende Segmente identifiziert werden [JM09].
     Ein bekannter Ansatz zur bottom-up Tokenisierung ist das Byte-Pair Encoding. Es geht von einem Minimalvokabular aus, das aus einzelnen Buchsta-

[1] Clitic-Konstruktionen sind Wortteile, die nicht allein, sondern nur in Verbindung mit einem anderen Wort verwendet werden können
[2] Ein Morphem ist die kleinste bedeutungstragende Einheit einer Sprache.

ben oder Zeichen besteht, und kombiniert diese iterativ zu häufig auftretenden Sequenzen. Dadurch entstehen Subword-Einheiten, die es ermöglichen, auch unbekannte Wörter durch Kombination bekannter Segmente darzustellen [JM09].

2. **Normalisierung von Wortformaten:** Die Normalisierung von Wortformaten ist ein wichtiger Schritt im NLP, um unterschiedliche Schreibweisen eines Wortes zu vereinheitlichen. Eine einfache Methode ist das *Case Folding*, bei dem alle Buchstaben in Kleinbuchstaben umgewandelt werden. Dies verbessert die Generalisierung in vielen Anwendungen, kann aber in der Sentimentanalyse und im Information Retrieval nachteilig sein, da die Groß- und Kleinschreibung bedeutungsrelevant sein kann, z. B. bei *US* für das Land und *us* als Pronomen [JM09].
   Ein weiterer wichtiger Aspekt der Normalisierung ist die *Lemmatisierung*, bei der verschiedene Wortformen auf ihre Grundform zurückgeführt werden. Beispielsweise werden die Wörter *ging*, *gegangen* und *geht* auf das Lemma *gehen* reduziert. Dies ist besonders für Sprachen mit komplexer Morphologie relevant, in denen Wörter je nach grammatischem Kontext unterschiedliche Formen annehmen. Eine einfachere, aber weniger genaue Alternative zur *Lemmatisierung* ist das *Stemming*, bei dem Wortendungen regelbasiert entfernt werden. Methoden wie der Porter-Stemmer kürzen Wörter auf ihren Wortstamm zurück, können aber Fehler verursachen, da sie keine vollständige morphologische Analyse durchführen [JM09].
   Die Wahl der Normalisierungsmethode hängt von der jeweiligen NLP-Anwendung ab. Während *Case Folding* für viele Anwendungen nützlich sind, ermöglicht die *Lemmatisierung* eine genauere Behandlung der Wortformen. *Stemming* ist eine schnelle Alternative, kann aber zu Ungenauigkeiten führen.

3. **Satzsegmentierung:** Die Satzsegmentierung unterteilt einen Text in einzelne Sätze, wobei Satzzeichen wie Punkte, Fragezeichen und Ausrufezeichen als Indikatoren dienen. Während Frage- und Ausrufezeichen meist eindeutige Satztrenner sind, kann der Punkt „.“ auch in Abkürzungen wie *„Mr.“* vorkommen, was die Segmentierung erschwert. Regelbasierte Verfahren oder maschinelles Lernen helfen, solche Mehrdeutigkeiten aufzulösen. Beispielsweise verwendet das *Stanford CoreNLP Toolkit* Regeln, um zu erkennen, ob ein Punkt eine Satzgrenze oder Teil einer Abkürzung ist [JM09].

Verteilungshypothese

Neben der strukturellen Verarbeitung von Texten spielt die semantische Analyse eine wesentliche Rolle im NLP. Eine zentrale Frage ist dabei, wie die Bedeutung von Wörtern modelliert werden kann. Eine Grundannahme ist, dass die Bedeutung eines Wortes eng mit seinem sprachlichen Kontext verknüpft ist. Dies führt zur

Verteilungshypothese, die besagt, dass Wörter in ähnlichen Kontexten tendenziell ähnliche Bedeutungen haben [JM09]. Dieses Konzept wurde erstmals von Harris ([Har81]) formuliert und später von Firth ([Fir57]) mit dem berühmten Zitat *„You shall know a word by the company it keeps.“* weiterentwickelt.

Die Verteilungshypothese bildet die Grundlage vieler moderner semantischer Modelle, insbesondere für *Vektorraum-Modelle (Vector Space Models, VSMs)*, die Wörter durch numerische Repräsentationen in hochdimensionalen Räumen abbilden [TP10].

In der Praxis wird die Verteilungshypothese in NLP-Anwendungen genutzt, indem die Kookkurrenz[3] von Wörtern in großen Korpora[4] analysiert wird. Ein bekanntes Beispiel ist die Erstellung von *Wortvektoren*, bei denen semantische Ähnlichkeit als räumliche Nähe im Vektorraum interpretiert wird. Die Verteilungshypothese hat zahlreiche praktische Anwendungen, unter anderem in der Synonymerkennung, der maschinellen Übersetzung und bei Suchmaschinen [TP10].

3.4 Große Sprachmodelle

Große Sprachmodelle (Large Language Models, LLMs) sind vortrainierte Sprachmodelle, die Sprach- und Weltwissen aus großen Textmengen lernen. Ein wesentliches Merkmal großer Sprachmodelle ist ihre Fähigkeit zur konditionalen Generierung, d. h. sie erzeugen kontinuierlich neue Wörter oder Sätze aus einem vorgegebenen Text (*Prompt*) [JM09].

Durch die Transformer-Architektur sind diese Modelle in der Lage, umfangreiche Kontextinformationen zu verarbeiten und konsistente, kohärente Texte zu erzeugen. Diese Eigenschaft ermöglicht den Einsatz von LLMs in einer Vielzahl von NLP-Aufgaben wie Textvervollständigung, Frage-Antwort-Systemen und automatischer Textzusammenfassung [JM09].

Indem das Modell aus früheren Eingaben lernt, kann es fundierte Vorhersagen treffen und Texte sinnvoll fortsetzen. Um die Auswahl der generierten Wörter zu steuern, werden verschiedene *Sampling-Techniken* verwendet [JM09].

Sampling

Sampling ist ein Verfahren im Decoding-Prozess großer Sprachmodelle, bei dem das nächste Wort basierend auf Wahrscheinlichkeiten ausgewählt wird. Anstatt deterministisch immer das wahrscheinlichste Wort zu wählen, ermöglicht Sampling eine variablere Textgenerierung, indem es alternative Wörter mit einer bestimmten Wahrscheinlichkeit berücksichtigt. Dies trägt dazu bei, dass die generierten Texte natürlicher und weniger vorhersehbar klingen [JM09].

[3] Kookkurrenz bezeichnet das gemeinsame Auftreten von Wörtern in einem bestimmten Kontext, z. B. in einem Satz oder einem Dokument.

[4] Ein Korpus (Plural: Korpora) ist eine Sammlung von Texten zur linguistischen Analyse und NLP-Verarbeitung.

Es gibt verschiedene Sampling-Techniken, die das Decoding beeinflussen. *Top-k Sampling* beschränkt die Auswahl auf die k wahrscheinlichsten Wörter. *Top-p Sampling* (auch *nucleus sampling* genannt) berücksichtigt dagegen alle Wörter, die zusammen mindestens einen bestimmten Prozentsatz p der Gesamtwahrscheinlichkeit ausmachen – die Anzahl der Wörter variiert dabei also dynamisch je nach Verteilung. *Temperature Sampling* verändert die Wahrscheinlichkeitsverteilung selbst: Eine niedrige Temperatur verstärkt die Auswahl wahrscheinlicher Wörter, während eine höhere Temperatur auch seltene Wörter wahrscheinlicher macht und so kreativere, weniger vorhersehbare Texte ermöglicht [JM09].

Pretraining großer Sprachmodelle

Das Pretraining ist eine zentrale Phase in der Entwicklung großer Sprachmodelle, in der sie aus großen Mengen von Textdaten lernen, bevor sie für spezifische Anwendungen weiter angepasst werden. Das Pretraining basiert auf *Self-Supervised Learning*, einer Methode, bei der sich das Modell selbst korrigiert, indem es beispielsweise versucht, das nächste Wort in einem Satz vorherzusagen (*Next-Token Prediction*). Dies geschieht, indem das Modell in jedem Schritt lernt, das nächste Wort auf Grundlage des vorherigen Kontextes vorherzusagen. Dazu benötigt es keine externen Labels, da die natürliche Wortfolge im Text bereits als Selbstüberwachung dient [JM09].

Die Optimierung erfolgt durch die Minimierung der *Cross-Entropy-Loss-Funktion*, die misst, wie gut das Modell das nächste Wort vorhersagt. Dadurch verbessert sich die Genauigkeit der Vorhersagen mit der Zeit [JM09].

Für das Pretraining werden umfangreiche Textkorpora aus verschiedenen Quellen verwendet, darunter automatisch gesammelte Webdaten (*Common Crawl*), Wikipedia, wissenschaftliche Artikel und Bücher. Da diese Daten von sehr unterschiedlicher Qualität sind, werden sie durch Filtermechanismen bereinigt, um Duplikate, irrelevante Inhalte oder problematische Texte (z. B. urheberrechtlich geschützte oder sensible Informationen) zu entfernen. Trotz dieser Maßnahmen bleiben Herausforderungen in Bezug auf Datenqualität, Verzerrungen, Urheberrecht und Datenschutz, die bei der Verwendung großer Sprachmodelle berücksichtigt werden müssen [JM09].

Durch das Pretraining entwickeln die Sprachmodelle ein allgemeines Sprachverständnis, das es den Modellen ermöglicht, Texte zu generieren, Fragen zu beantworten oder verschiedene NLP-Aufgaben auszuführen. Dieses allgemeine Sprachwissen wird später durch *Finetuning* weiter verfeinert, um die Modelle für spezifische Anwendungsbereiche zu optimieren [JM09].

Finetuning

Finetuning ist der Prozess, bei dem ein bereits vortrainiertes Sprachmodell weiter an spezifische Daten angepasst wird, um es für neue Domänen oder spezielle Aufgaben zu optimieren. Obwohl große Sprachmodelle durch ihr Pretraining ein

breites Sprachverständnis entwickeln, ist es oft notwendig, sie gezielt an Inhalte anzupassen, die im Pretraining nicht ausreichend repräsentiert wurden, z.B. medizinische Texte, bestimmte Sprachen oder spezielle Aufgabenbereiche. In solchen Fällen wird das Modell durch zusätzliches Training mit geeigneten Datensätzen verfeinert [JM09].

Es gibt verschiedene Methoden des Finetunings, die sich in ihrem Ansatz und Rechenaufwand unterscheiden [JM09]:

- **Vollständiges Finetuning:** Hier werden alle Parameter des Modells weitertrainiert, um es vollständig an die neue Aufgabe oder Domäne anzupassen. Dieser Ansatz liefert präzise Anpassungen, ist aber rechenintensiv und erfordert große Datenmengen.
- **Parameter-effizientes Finetuning:** Um den Rechenaufwand zu reduzieren, werden nur bestimmte Parameter aktualisiert, während der Großteil des Modells unverändert bleibt. Dies macht das Training effizienter und ressourcenschonender.
- **Aufgabenbezogenes Finetuning:** Diese Methode erweitert das Modell um eine zusätzliche Klassifikationsschicht, die häufig im Rahmen eines überwachten Lernprozesses auf eine neue Aufgabe - z. B. *Sentiment-Analyse* oder *Textklassifikation* - trainiert wird. Das zugrundeliegende Modell bleibt dabei eingefroren.
- **Supervised Finetuning:** Hier wird das Modell mit spezifischen Anweisungen und erwarteten Antworten trainiert. Dies wird häufig bei dialogbasierten KI-Systemen angewendet, um Modelle gezielt auf Nutzerinteraktionen vorzubereiten, etwa für *Frage-Antwort-Systeme* oder textbasierte Anweisungen.

4

Methodik

4.1 Symbolnamen und Annotationen in Ontologien

Bedeutung von Symbolnamen

Symbolnamen sind zentrale Elemente von Ontologien, da sie Konzepte, Relationen und Entitäten repräsentieren und als semantische Brücke zwischen der abstrakten logischen Struktur und der durch die Ontologie modellierten realen Welt dienen. Wie Heiner Stuckenschmidt in [Stu09] betont, sind Symbolnamen unverzichtbar, da sie die Verbindung zwischen symbolischen Repräsentationen und den Objekten oder Konzepten, auf die sie sich beziehen, herstellen. Diese Verbindung ist sowohl für die Wissensrepräsentation als auch für die maschinelle Interpretation und Verarbeitung von ontologischer Information essentiell.

Ein anschauliches Beispiel liefert eine biologische Ontologie, in der ein Symbolname wie „Photosynthese“ nicht nur einen Prozess bezeichnet, sondern gleichzeitig mit dem dahinterliegenden biologischen Wissen verknüpft ist. Gut gewählte Symbolnamen tragen laut Stuckenschmidt wesentlich zur semantischen Konsistenz und Interoperabilität von Ontologien bei, ein besonders wichtiger Aspekt in hoch standardisierten Domänen wie der Medizin oder der Biologie, in denen die exakte Begriffsverwendung entscheidend ist.

In der Praxis zeigt sich jedoch häufig, dass maschinelle Systeme Symbolnamen lediglich als syntaktische Bezeichner behandeln, ohne ihren semantischen Gehalt auszuwerten. Dadurch geht ein wesentlicher Teil des in ihnen enthaltenen Wissenspotenzials verloren. Um die Leistungsfähigkeit von Inferenzsystemen zu verbessern, ist es daher notwendig, Methoden zu entwickeln, die sowohl syntaktische Strukturen als auch semantische Inhalte von Symbolnamen systematisch berücksichtigen. Symbolnamen sind also weit mehr als nur Bezeichner, sie sind integraler Bestandteil der semantischen Architektur von Ontologien [Stu09].

Annotationen und semantische Informationen

Annotationen erweitern die Funktion von Symbolnamen, indem sie zusätzliche Informationen bereitstellen, die den Kontext und die Bedeutung eines Symbols präzisieren. Lordick et al. zeigen in ihrem Artikel [LBB+16], wie Annotationen

in digitalen Objekten wie Texten, Bildern und Videos eingesetzt werden, um semantische Beziehungen herzustellen und Wissen besser zugänglich zu machen. Sie betonen, dass Annotationen nicht nur maschinell verarbeitet werden können, sondern auch wertvolle Einblicke für die menschliche Interpretation bieten. In Verbindung mit kollaborativen Forschungsumgebungen und semantischen Webtechnologien eröffnen sie neue Perspektiven für die Analyse und Vernetzung von Wissen.

Darüber hinaus ermöglichen Annotationen eine Verbindung zwischen der symbolischen und der textuellen Ebene. Bauer et al. [BVZ22] betonen, dass Annotationen nicht nur Kontext liefern, sondern auch tiefere Bedeutungsschichten erschließen, die sonst verborgen bleiben könnten. Kommentierte Annotationen strukturieren digitale Texte und fördern neue Einsichten in deren Inhalte. Diese Funktionalitäten tragen dazu bei, digitale Wissenssysteme effektiver zu gestalten und innovative Formen der semantischen Analyse zu unterstützen.

Symbolnamen und Annotationen sind zentrale Bestandteile von Ontologien. Während Symbolnamen als semantische Anker dienen, liefern Annotationen Kontextinformationen, die sowohl die maschinelle Verarbeitung als auch das menschliche Verständnis erleichtern. Diese enge Verknüpfung bildet die Grundlage für die Analyse und Weiterentwicklung von Ontologien, wie sie in den folgenden Abschnitten untersucht wird. Der Schwerpunkt liegt dabei auf der Frage, wie Symbolnamen und Annotationen extrahiert, analysiert und für NLP-basierte Verfahren nutzbar gemacht werden können.

4.2 Analyse von Ontologien

Dieses Kapitel widmet sich der detaillierten Analyse ausgewählter Ontologien mit dem Fokus auf der Struktur, den Symbolnamen und deren Annotationen.

Für die Untersuchung wurden vier Ontologien ausgewählt: die **Pizza-Ontologie**, die **CLYH-Ontologie**[1], die **Three-ST-Ontologie**[2] und die **Gene-Ontologie**[3]. Diese wurden gezielt ausgewählt, um ein möglichst breites Spektrum an Anwendungskontexten, Modellierungsstrategien und Komplexitätsgraden abzudecken – von einer didaktisch reduzierten Beispielontologie bis hin zu einem umfangreichen, biologischen Standardvokabular.

Die Analyse wurde in mehreren Schritten durchgeführt: Zunächst wurden mit Hilfe eines implementierten Python-Skripts alle Symbolnamen aus den jeweiligen OWL-Dateien extrahiert. Anschließend wurden für jede Ontologie statistische Kennzahlen wie die Anzahl der Symboltypen (Klassen, Properties, Individuen) sowie die Verteilung und inhaltliche Tiefe verschiedener Annotationstypen berechnet. Besonderes Augenmerk wurde auf semantisch relevante Annotationen

[1] Clytia hemisphaerica Development and Anatomy Ontology, verfügbar unter: `https://bioportal.bioontology.org/ontologies/CLYH`, abgerufen am 15.04.2025.

[2] 3-Step Theory of suicide Ontology, verfügbar unter: `https://bioportal.bioontology.org/ontologies/THREE-ST`, abgerufen am 15.04.2025.

[3] Gene Ontology, verfügbar unter: `https://bioportal.bioontology.org/ontologies/GO`, abgerufen am 15.04.2025.

wie `rdfs:label`, `rdfs:comment`, `skos:definition` und `IAO_0000115`[4] gelegt, die zur Information Artifact Ontology (IAO) gehört und die offizielle Definition zur Erklärung der Bedeutung einer Klasse oder Property bereitstellt, sowie auf synonymbezogene Annotationen. Die Ergebnisse wurden nicht nur tabellarisch zusammengefasst, sondern auch in Form von Balkendiagrammen visualisiert, um zentrale Muster auf einen Blick erkennbar zu machen. Abschließend wird auch die formale Struktur der Symbolnamen analysiert und ihre sprachliche Gestaltung kommentiert.

Pizza Ontologie

Die Pizza-Ontologie wurde bewusst als Einstieg in die Analyse gewählt, da sie eine geringe Komplexität aufweist. Durch ihre überschaubare Größe und klare Struktur eignet sie sich besonders gut, um das Analyseverfahren zunächst zu erproben und exemplarisch darzustellen. Die Symbolnamen und Annotationen in dieser Ontologie sind leicht verständlich und können manuell mit der Darstellung in Protégé[5] abgeglichen werden. Zudem enthält sie keine domänenspezifischen Fachbegriffe, was die Auswertung insbesondere bei der Betrachtung sprachlicher Eigenschaften erleichtert.

Im ersten Schritt wurde ermittelt, welche Arten von Symbolen in der Ontologie vorhanden sind. In OWL-Ontologien werden verschiedene Typen von Properties unterschieden: Objekt-Properties verbinden zwei Individuen miteinander, Daten-Properties verbinden ein Individuum mit einem konkreten Datenwert und Annotations-Properties dienen der Anreicherung von Ontologieelementen mit zusätzlichen Informationen wie Labels oder Kommentaren [HKR10].

Die folgende Übersicht zeigt die Gesamtzahl der extrahierten Symbolnamen:

Symboltyp	Anzahl
`Klassen`	99
`Objekt-Properties`	8
`Daten-Properties`	0
`Annotation-Properties`	8
`Individuen`	5

Tabelle 4.1: Übersicht über Symboltypen in der Pizza-Ontologie

Abgleich der Extraktionsergebnisse mit Protégé

Der manuelle Vergleich der extrahierten Symboltypen mit der Darstellung in Protégé bestätigte die Korrektheit aller automatisiert ermittelten Werte – mit einer

[4] `http://purl.obolibrary.org/obo/IAO_0000115` abgerufen am 15.04.2025.
[5] Protégé ist ein Ontologie-Editor zur grafischen Bearbeitung und Analyse von Ontologien.

Ausnahme: Während Protégé insgesamt 17 Annotation-Properties anzeigt, wurden bei der automatisierten Auswertung nur 8 Annotation-Properties extrahiert. Die Ursache liegt vermutlich in der Behandlung von Namespaces. Annotation-Properties aus den Namespaces `rdfs:` und `owl:` wurden von Owlready2[6] nicht als Annotation-Properties erkannt. Annotationen aus anderen Namespaces wie `dc:`, `skos:` oder ohne explizites Präfix (z.B. *contributor*, *license*, *provenance*) wurden hingegen extrahiert.

Analyse der Annotationen

Ein zentrales Ziel dieser Analyse war es, festzustellen, wie viele Symbole mit erklärenden Annotationen - insbesondere `rdfs:label`, `rdfs:comment` und `skos:definition` - versehen sind. Die folgende Tabelle gibt einen Überblick über die Anzahl der Symboltypen, die mit diesen Annotationen versehen sind.

	Klassen	**Objekt-Prop**	**Annotation-Prop**	**Individuen**
Gesamt	99	8	8	5
Mit `rdfs:label`	99	0	0	0
Mit `rdfs:comment`	11	5	0	0
Mit `skos:definition`	8	0	0	0

Tabelle 4.2: Anzahl annotierter Symbole in der Pizza-Ontologie[7]

Die Tabelle zeigt, dass alle Klassen mit einem `rdfs:label` versehen sind, jedoch nur ein kleiner Teil auch über `rdfs:comment`- oder `skos:definition`-Annotationen verfügt. Bei den Objekt-Properties hingegen liegt ein hoher Anteil an Kommentaren vor, während andere Annotationstypen fehlen. Ein Beispiel für eine Klasse in dieser Ontologie mit mehreren Annotationen ist `NonVegetarianPizza`. Diese Klasse besitzt die Annotation `rdfs:label` mit dem Wert *NonVegetarianPizza*. Die zugehörige `skos:definition` beschreibt das Konzept inhaltlich als *„Any Pizza that is not a VegetarianPizza“*.

Um die Verteilung der Annotationen noch übersichtlicher zu machen, wurden die prozentualen Anteile in einem Balkendiagramm visualisiert.

[6] Python-Bibliothek zur Verarbeitung von OWL-Ontologien.
[7] Objekt-Prop = Objekt-Properties, Annotation-Prop = Annotation-Properties

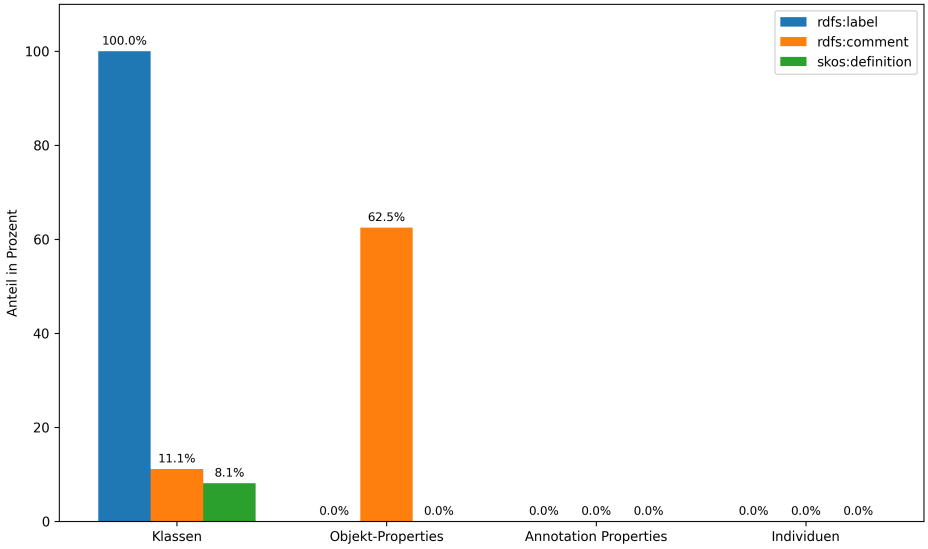

Abbildung 4.1: Prozentuale Annotationen pro Symboltyp in der Pizza-Ontologie

Neben der Betrachtung der Annotationsverteilung wurde auch die inhaltliche Ausführlichkeit der Annotationen untersucht. Dazu wurde die durchschnittliche Wortanzahl für die Annotationstypen `rdfs:label`, `rdfs:comment` und `skos:definition` berechnet. Abbildung 4.2 veranschaulicht diese Unterschiede grafisch.

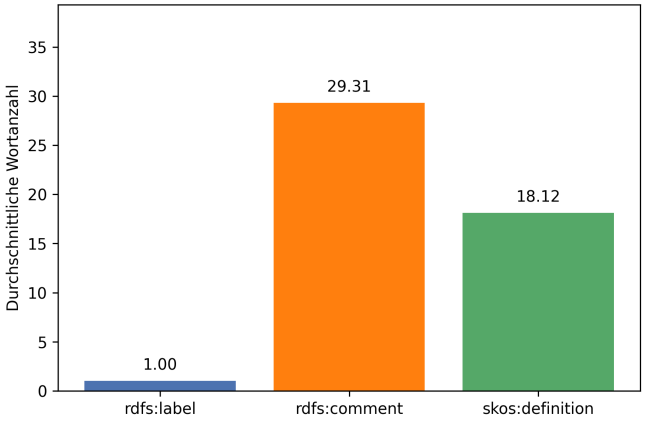

Abbildung 4.2: Durchschnittliche Wortanzahl von Kommentaren und Definitionen

Während Labels (`rdfs:label`) mit durchschnittlich nur 1,00 Wörtern sehr knapp formuliert sind, entspricht dies ihrem Zweck als Kurzbezeichnungen und ist daher nicht ungewöhnlich. Kommentare (`rdfs:comment`) hingegen sind mit durchschnittlich 29,31 Wörtern deutlich ausführlicher und dienen der inhaltlichen Erläuterung. Definitionen (`skos:definition`) liegen mit durchschnittlich 18,12 Wörtern zwischen diesen beiden Extremen und liefern kompakte, aber dennoch informative Beschreibungen.

Zusätzlich wurden sprachbezogene Annotationen wie `skos:prefLabel` und `skos:altLabel` identifiziert. Diese enthalten bevorzugte oder alternative Bezeichnungen für Symbole. Da ausschließlich Klassen diese Annotationen aufweisen, bezieht sich die nachfolgende Auswertung nur auf diese Symbolkategorie.

	Klassen
Gesamt	99
Mit `skos:prefLabel`	97
Mit `skos:altLabel`	22

Tabelle 4.3: Verteilung sprachbezogener Annotationen in der Pizza-Ontologie

Insgesamt zeigt sich, dass skos:prefLabel in fast allen Klassen vorkommt, während skos:altLabel nur vereinzelt vergeben wurde. Eine manuelle Überprüfung zeigt, dass die sprachliche Qualität dieser Annotationen variiert: Neben echten Synonymen finden sich auch technische Varianten oder wörtliche Wiederholungen der Klassennamen. Ein Beispiel hierfür ist die Klasse `NonVegetarianPizza`, die neben einem `rdfs:label` in Englisch auch über ein `skos:prefLabel` *Non Vegetarian Pizza* verfügt – eine besser lesbare Bezeichnung mit Leerzeichen.

Ein weiteres Beispiel ist die Klasse `Soho`, der zwei `skos:altLabel`-Annotationen zugewiesen wurden: *Soho* und *Soho Pizza*. Letzteres ist eine alternative Schreibweise mit einem erklärenden Zusatz, ohne den semantischen Inhalt wesentlich zu verändern.

Schreibweise und Sprache der Symbolnamen

Bei der Analyse der Symbolnamen fällt auf, dass das Schema *UpperCamelCase* (oder *PascalCase*) verwendet wurde. Dabei werden zusammengesetzte Wörter ohne Leerzeichen geschrieben, wobei jedes Wort mit einem Großbuchstaben beginnt (z. B. `PizzaTopping`, `VegetarianPizza`, `ThinAndCrispyBase`).

Auch sprachlich zeigen sich deutliche Muster: Die meisten Symbolnamen sind in Englisch, während die `rdfs:label`-Annotationen teilweise in Portugiesisch ergänzt wurden. Dies deutet auf ein gewisses Maß an Mehrsprachigkeit hin, auch wenn diese nicht flächendeckend umgesetzt ist. Ein Beispiel hierfür ist die Klasse `NonVegetarianPizza`, die neben dem englischen Label *NonVegetarianPizza* auch das portugiesische `rdfs:label` *PizzaNaoVegetariana* enthält.

Clytia hemisphaerica Development and Anatomy Ontologie

Die Clytia hemisphaerica Development and Anatomy Ontologie (Clyh-Ontologie) beschreibt die Entwicklung und Anatomie des Süßwasserpolypen Clytia hemis-

phaerica, einem Modellorganismus aus der Gruppe der Nesseltiere[8]. Die Ontologie ist fachlich deutlich spezifischer als die didaktisch ausgerichtete Pizza-Ontologie und orientiert sich an biologischen Prozessen und anatomischen Strukturen.

Nach der Extraktion der Symbolnamen aus der CLYH-Ontologie ergeben sich die in Tabelle 4.4 dargestellten Gesamtzahlen.

Symboltyp	**Anzahl**
Klassen	265
Objekt-Properties	5
Daten-Properties	0
Annotation-Properties	4
Individuen	0

Tabelle 4.4: Übersicht über Symboltypen in der CLYH-Ontologie

Abgleich der Extraktionsergebnisse mit Protégé:

Ein manueller Abgleich mit der Anzeige in Protégé ergab, dass dort insgesamt 13 Annotation-Properties für die Ontologie CLYH aufgelistet sind. Bei der automatisierten Auswertung wurden jedoch nur 4 Annotation Properties extrahiert. Die Differenz ergibt sich daraus, dass systeminterne Annotationen mit Namespaces `rdfs:` und `owl:` - wie z.B. `owl:deprecated` - von `owlready2` nicht als Annotation Properties erkannt werden, obwohl sie in Protégé als solche dargestellt werden.

Analyse der Annotationen

Auffällig ist, dass die Ontologie keine `rdfs:comment`-Annotationen enthält. Stattdessen erfolgt die Beschreibung der Klassen im Wesentlichen über die Annotation `IAO_0000115`, die laut zugehörigem `rdfs:label` als *definition* gekennzeichnet ist.

Wie in der Tabelle 4.5 zu sehen ist, verfügen alle Klassen der Ontologie über ein `rdfs:label`, was angesichts der numerisch kodierten Symbolnamen von besonderer Bedeutung ist. Diese Labels (z. B. `CLYH_0000027`) dienen in Kombination mit der Definition als primäre Quelle semantischer Informationen.

[8] `https://bioportal.bioontology.org/ontologies/CLYH`, abgerufen 15.04.2025.

	Klassen	Objekt-Prop	Annotation-Prop
Gesamt	265	5	4
Mit `rdfs:label`	265	5	2
Mit `IAO_0000115`	212	0	0
Mit `hasExactSynonym`	17	0	0
Mit `IAO_0000301`	7	0	0
Mit `deprecated`	3	0	0

Tabelle 4.5: Anzahl annotierter Symbole in der CLYH-Ontologie

Darüber hinaus kommen auch Annotationen wie `hasExactSynonym`, `IAO_00003-01` (label: *citation*) und `owl:deprecated` vor. Während `hasExactSynonym` alternative Bezeichnungen für einige Klassen liefert, verweist `IAO_0000301` auf Literaturquellen oder den kontextuellen Ursprung der Konzepte. Die seltene Verwendung von `owl:deprecated` deutet darauf hin, dass einzelne Klassen als veraltet markiert wurden.

Ein Beispiel für die Kombination verschiedener Annotationen ist die Klasse `CLYH_0000066`. Sie ist mit einem `rdfs:label` *12-bulb medusa stage* versehen, das eine kurze technische Beschreibung liefert. Ergänzend beschreibt die `IAO_0000115`-Annotation den Begriff ausführlicher als *„period of medusa growth when only 12 tentacles are present“*. Zusätzlich wird ein `hasExactSynonym` mit dem Ausdruck *young medusa stage* angegeben, der eine verständlichere Alternativbezeichnung darstellt.

Um die Verteilung der Annotationen über die verschiedenen Symboltypen besser nachvollziehen zu können, veranschaulicht Abbildung 4.3 den prozentualen Anteil ausgewählter Annotationen für Klassen, Properties und Individuen.

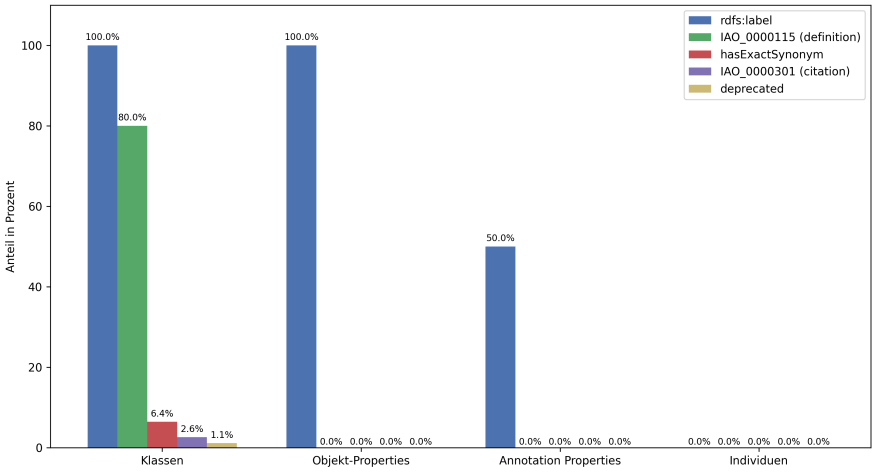

Abbildung 4.3: Prozentuale Annotationen pro Symboltyp in der CLYH-Ontologie

Neben der reinen Häufigkeit wurde auch die inhaltliche Tiefe der Annotationen untersucht. Abbildung 4.4 zeigt die durchschnittliche Wortanzahl für die Annotationstypen `rdfs:label` und `IAO_0000115`.

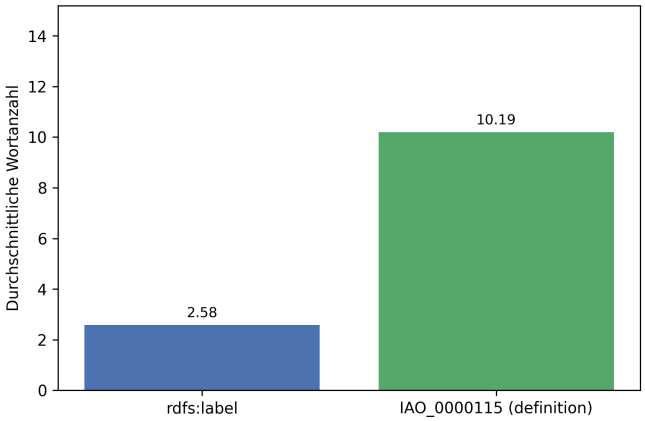

Abbildung 4.4: Durchschnittliche Wortanzahl von Labels und Definitionen

Interessant ist, dass die Labels mit durchschnittlich 2,58 Wörtern etwas ausführlicher sind, als man es von typischen Labels, die oft nur aus einem Wort bestehen, erwarten würde. Dies könnte damit zusammenhängen, dass viele Begriffe aus dem Bereich der Biologie stammen und daher eine genauere Beschreibung erfordern.Die Definitionen hingegen enthalten im Durchschnitt 10,19 Wörter und liefern damit deutlich mehr semantischen Inhalt. Dies bestätigt ihre Funktion als beschreibende Ergänzung zu den eher benennenden Labels.

Schreibweise und Sprache der Symbolnamen

Bei der Analyse der Symbolnamen in der CLYH-Ontologie fällt auf, dass alle Namen einem schematischen Code-Muster folgen: Sie bestehen aus einem technischen Kürzel wie `CLYH` oder `UBERON`, gefolgt von einer siebenstelligen Ziffernfolge (z. B. `CLYH_0000053`, `UBERON_0000105`). Die eigentliche Bedeutung der Symbole wird erst durch begleitende Annotationen wie `rdfs:label` oder `IAO_0000115` erkennbar gemacht. Sprachlich sind alle Labels und Definitionen durchgängig in Englisch formuliert. Im Gegensatz zur Pizza-Ontologie, in der auch portugiesische Bezeichnungen vorkamen, weist die CLYH-Ontologie keine Anzeichen von Mehrsprachigkeit auf. Diese klare sprachliche Fokussierung unterstützt die Konsistenz, verringert aber zugleich die sprachliche Zugänglichkeit für nicht-englischsprachige Nutzer*innen.

3-Step Theory of suicide Ontologie

Die *3-Step Theory of Suicide Ontology* (Three-ST-Ontologie) basiert auf dem psychologischen Drei-Stufen-Modell des Suizids und beschreibt Konzepte wie psychisches Leiden, Hoffnungslosigkeit, Bindung und die Fähigkeit, sich das Leben

zu nehmen. Die Ontologie wurde entwickelt, um relevante Begriffe aus klinischen Aufzeichnungen im Bereich der Suizidprävention systematisch zu erfassen[9].

Nach der Extraktion der Symbolnamen aus der Three-ST-Ontologie ergeben sich die in Tabelle 4.6 dargestellten Gesamtzahlen.

Symboltyp	**Anzahl**
Klassen	48
Objekt-Properties	0
Daten-Properties	0
Annotation-Properties	4
Individuen	8484

Tabelle 4.6: Übersicht über Symboltypen in der Three-ST-Ontologie

Die Struktur der Ontologie fällt durch das völlige Fehlen von Objekt- und Daten-Properties auf. Dies deutet darauf hin, dass keine expliziten Beziehungen oder Attributwerte zwischen den Klassen modelliert wurden. Stattdessen scheint der Fokus auf der Repräsentation einzelner Entitäten zu liegen: Die extrem hohe Anzahl an Individuen (8484) im Vergleich zur relativ geringen Anzahl an Klassen (48) deutet darauf hin, dass die Ontologie primär instanzbasiert aufgebaut ist. Die Klassen dienen vermutlich als Kategorien zur Einordnung der zahlreichen Entitäten, ohne dass komplexe semantische Relationen zwischen ihnen definiert wurden.

Abgleich der Extraktionsergebnisse mit Protégé

Wie bei den zuvor analysierten Ontologien zeigt sich auch bei der Three-ST-Ontologie eine Abweichung hinsichtlich der Anzahl der Annotation-Properties. In Protégé werden zusätzlich zu den vier extrahierten Annotationen weitere systeminterne Annotationen mit den Namensräumen `rdfs:` und `owl:` (z. B. `rdfs:label`, `owl:deprecated`) als Annotation Properties aufgeführt.

Ansonsten stimmen die extrahierten Symboltypen vollständig mit der Anzeige in Protégé überein.

Analyse der Annotationen

Im Folgenden wird die Verteilung der Annotationen auf die verschiedenen Symboltypen der Three-ST-Ontologie dargestellt. Der Fokus liegt dabei auf den tatsächlich verwendeten Annotation Properties und deren Zuordnung zu Klassen, Annotation Properties und Individuen.

[9] `https://bioportal.bioontology.org/ontologies/THREE-ST`, abgerufen am 15.04.2025.

	Klassen	Annotation-Prop	Individuen
Gesamt	48	4	8484
Mit `rdfs:label`	48	0	8484
Mit `instance_type_ID`	0	0	8484
Mit `polarity`	0	0	8484
Mit `relationship`	0	0	8484
Mit `term_unique_identifier`	0	0	8484

Tabelle 4.7: Anzahl annotierter Symbole in der Three-ST-Ontologie

Die Tabelle 4.7 zeigt, dass alle Klassen sowie alle Individuen über eine `rdfs:label`-Annotation verfügen. Die vier zusätzlich verwendeten Annotation-Properties (`instance_type_ID`, `polarity`, `relationship`, `term_unique_identifier`)[10] kommen ausschließlich bei Individuen vor. Dies unterstreicht die zentrale Rolle der Individuen in dieser Ontologie, während die Klassen primär durch ihre Labels beschrieben werden. Kommentare oder Definitionen zur inhaltlichen Beschreibung fehlen gänzlich. Ein exemplarischer Vergleich verdeutlicht diese Struktur: Die Klasse `DispositionalCapacityForSuicide` ist mit dem `rdfs:Label` *Dispositional capacity for suicide* versehen - weitere Annotationen sind nicht vorhanden. Im Gegensatz dazu ist das Individuum `abandonMe` nicht nur mit dem `rdfs:Label` *abandon me* versehen, sondern enthält auch die Annotationen `instance_type_ID` mit dem Wert *310201*, `polarity` mit *indicates_presence*, `relationship` mit *is_a_state* sowie `term_unique_identifier` mit dem Wert *40215*.

Zur besseren Visualisierung der Verteilung der annotierten Symbole werden in Abbildung 4.5 die prozentualen Häufigkeiten ausgewählter Annotationen für Klassen, Annotation Properties und Individuen gegenübergestellt. Dadurch können Unterschiede in der Verwendung bestimmter Annotationstypen auf einen Blick erfasst werden.

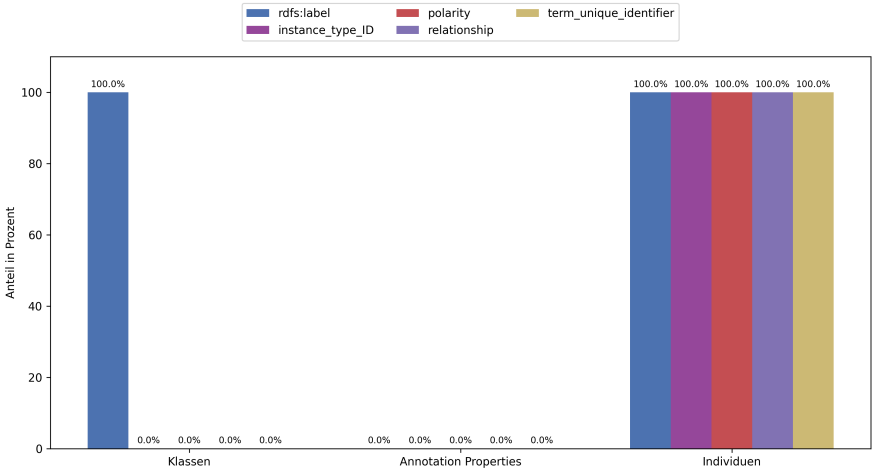

Abbildung 4.5: Anzahl annotierter Symbole in der Three-ST-Ontologie

[10] Diese Annotationen stammen aus dem lokalen Namespace der Ontologie.

Da in der Three-ST-Ontologie keine `rdfs:comment`- oder `definition`-Annotationen verwendet werden, beschränkt sich die Analyse der durchschnittlichen Wortanzahl auf die vorhandenen `rdfs:Label`. Die übrigen verwendeten Annotationen bestehen nur aus einzelnen Wörtern oder codierten Werten und eignen sich daher nicht für eine aussagekräftige Textlängenanalyse. Abbildung 4.6 gibt einen Überblick über die durchschnittliche Länge der Labels.

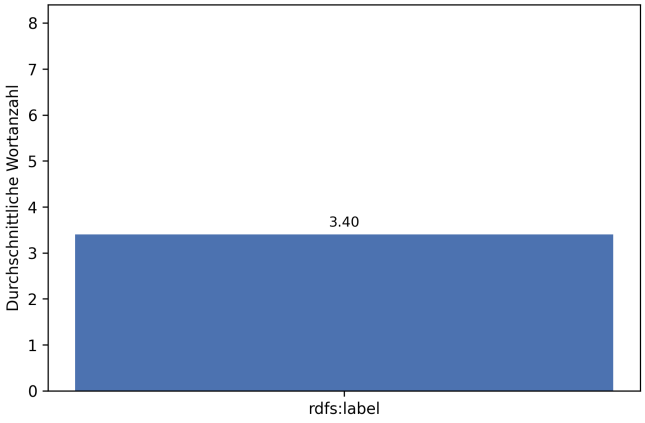

Abbildung 4.6: Durchschnittliche Wortanzahl von Labels

Die durchschnittliche Wortanzahl der `rdfs:label`-Annotationen liegt bei 3,40 Wörtern. Im Vergleich zu den zuvor analysierten Ontologien zeigt sich, dass die Pizza-Ontologie mit durchschnittlich nur 1,0 Wörtern pro Label besonders knapp ist, während die CLYH-Ontologie mit 2,58 Wörtern eine mittlere Position einnimmt. Die Three-ST-Ontologie hebt sich damit leicht ab, was auf den Versuch hindeutet, semantisch gehaltvollere Informationen bereits auf der Ebene der Labels zu vermitteln, ein sinnvoller Ansatz, da die Ontologie keine weiteren Beschreibungen in Form von Kommentaren oder Definitionen bereitstellt.

Schreibweise und Sprache der Symbolnamen

Bei der Analyse der Symbolnamen in der Three-ST-Ontologie fällt auf, dass das Schema *PascalCase* wie in der Pizza-Ontologie für Klassennamen verwendet wird. Beispiele hierfür sind `PsychologicalPain` oder `DispositionalCapacityForSuicide`. Diese zusammengesetzten Begriffe bestehen aus mehreren Wörtern, die jeweils mit einem Großbuchstaben beginnen, und spiegeln komplexe semantische Konzepte wider.

Die Annotation Properties hingegen folgen dem Muster *snake_case*, wie z. B. `instance_type_ID` oder `term_unique_identifier`. Auch bei den Individuen gibt es verschiedene Schreibweisen: Neben *camelCase*, wie z. B. `abandonedByBoyfriend` kommen auch *kebab-case* wie `12-24BeersADay` sowie Mischformen mit Sonderzeichen oder Zahlen vor, wie z. B. `$500WasStolen` oder `60pills)OfCitalopram`.

Insgesamt ist die Sprache aller Labels konsistent englisch, was auf eine klare Zielgruppe der Ontologie schließen lässt. Während die Klassennamen strukturiert

und systematisch aufgebaut sind, weisen die Individuennamen eine hohe formale Varianz auf. Dadurch ergibt sich ein kontrastreiches Bild zwischen formal definierten Klassen und inhaltlich differenzierten, alltagsnahen Individuen.

Gene Ontologie

Die Gene-Ontologie ist eine der bekanntesten und am weitesten verbreiteten Ontologien in den Biowissenschaften. Sie bietet strukturierte und kontrollierte Vokabulare zur Beschreibung von Genprodukten hinsichtlich ihrer molekularen Funktion, ihrer zellulären Komponente und ihrer biologischen Rolle[11]. Nach der automatisierten Extraktion der Symbolnamen ergeben sich die in Tabelle 4.8 dargestellten Gesamtzahlen.

Symboltyp	**Anzahl**
Klassen	51641
Objekt-Properties	4
Daten-Properties	0
Annotation-Properties	51
Individuen	0

Tabelle 4.8: Übersicht über Symboltypen Gene-Ontologie

Die Tabelle verdeutlicht die enorme Größe der Gene Ontology im Vergleich zu den zuvor analysierten Ontologien: Mit über 51000 Klassen stellt sie die bei weitem umfangreichste Struktur in dieser Arbeit dar. Gleichzeitig fällt auf, dass die Ontologie keine Individuen enthält und auch nur sehr wenige Objekt-Properties verwendet. Die semantische Beschreibung erfolgt also primär auf Klassenebene, unterstützt durch eine vergleichsweise hohe Anzahl von Annotation-Properties.

Abgleich der Extraktionsergebnisse mit Protégé

Auch bei der Gene Ontology zeigt sich - wie bei den zuvor analysierten Ontologien - ein Unterschied in der Anzahl der Annotation-Properties. Während Protégé insgesamt 51 Annotation-Properties ausweist, wurden bei der automatisierten Extraktion mit `owlready2` einige systeminterne Annotationen mit den Namensräumen `rdfs:` und `owl:` nicht als Annotation-Properties erfasst.

Abgesehen von dieser vorhersehbaren Abweichung stimmen alle anderen Symboltypen und deren Anzahl vollständig mit der Anzeige in Protégé überein.

Analyse der Annotationen

In diesem Abschnitt wird die Verteilung der Annotationen innerhalb der Gene Ontology untersucht. Insbesondere wird untersucht, welche Annotation-Properties

[11] `https://bioportal.bioontology.org/ontologies/GO`, abgerufen am 15.04.2025.

verwendet werden und wie diese den verschiedenen Symboltypen - Klassen, Objekt-Properties, Annotation-Properties - zugeordnet sind. Ziel ist es, einen Überblick über die Struktur und die inhaltliche Tiefe der Annotationen zu erhalten. Für die quantitative Analyse wurden im Folgenden die am häufigsten vorkommenden Annotationen aufgeschlüsselt.

	Klassen	**Objekt-Prop**	**Annotation-Prop**
Gesamt	51641	4	51
Mit `rdfs:label`	47995	4	21
Mit `IAO_0000115(definition)`	47995	0	0
Mit `comment`	9973	0	22
Mit `hasExactSynonym`	25278	0	0
Mit `hasNarrowSynonym`	7781	0	0
Mit `hasBroadSynonym`	2673	0	0
Mit `hasRelatedSynonym`	9026	0	0
Mit `hasDbXref`	8511	4	2
Mit `hasOBONamespace`	47995	4	2
Mit `id`	47995	4	2
Mit `IAO_0100001(term replaced by)`	5345	0	0
Mit `consider`	1849	0	0
Mit `hasAlternativeId`	2436	0	0
Mit `deprecated`	11374	0	0

Tabelle 4.9: Anzahl annotierter Symbole in der Gene Ontology

Auffällig ist, dass fast alle Klassen mit `rdfs:label` und `IAO_0000115` versehen sind, was Ausdruck einer hohen semantischen Beschreibungstiefe ist. `comment`-Annotationen sind seltener. Auffallend ist auch die relativ häufige Verwendung von synonymbezogenen Annotationen wie `hasExactSynonym`, `hasNarrowSynonym` und `hasRelatedSynonym`, was auf ein besonderes Augenmerk auf terminologische Vielfalt und Durchsuchbarkeit schließen lässt. Technische Annotationen wie `hasOBONamespace`[12], `id` oder `hasDbXref`[13] finden sich überwiegend zusätzlich zu den Objekt-Properties. Ein Beispiel für eine solche reichhaltige Annotation findet sich bei der Klasse `GO_0001400`. Diese ist mit einem `rdfs:label` *mating projection base* versehen und enthält eine `IAO_0000115`, die den biologischen Kontext präzisiert: *„The region where the mating projection meets the bulk of the cell, in unicellular fungi exposed to mating pheromone.“* Darüber hinaus ist die Klasse mit einer `hasOBONamespace` *cellular_component* und zwei `hasNarrowSynonym` Einträgen versehen: *base of shmoo tip* und *conjugation tube base*.

Um die Verwendung der Annotationen innerhalb der Gene Ontology differenzierter darzustellen, zeigt die Abbildung 4.7 deren prozentuale Verteilung auf die Symboltypen Klassen, Objekt-Properties und Annotation-Properties.

[12] `hasOBONamespace` gibt an, zu welchem inhaltlichen Modul oder Themenbereich der OBO-Ontologie ein Konzept gehört, z. B. *biological_process*, *molecular_function* oder *cellular_component*.

[13] `hasDbXref` verweist auf verknüpfte externe Ressourcen, etwa Datenbankeinträge oder Wikipedia-Artikel, z. B. *Wikipedia:Reproduction*.

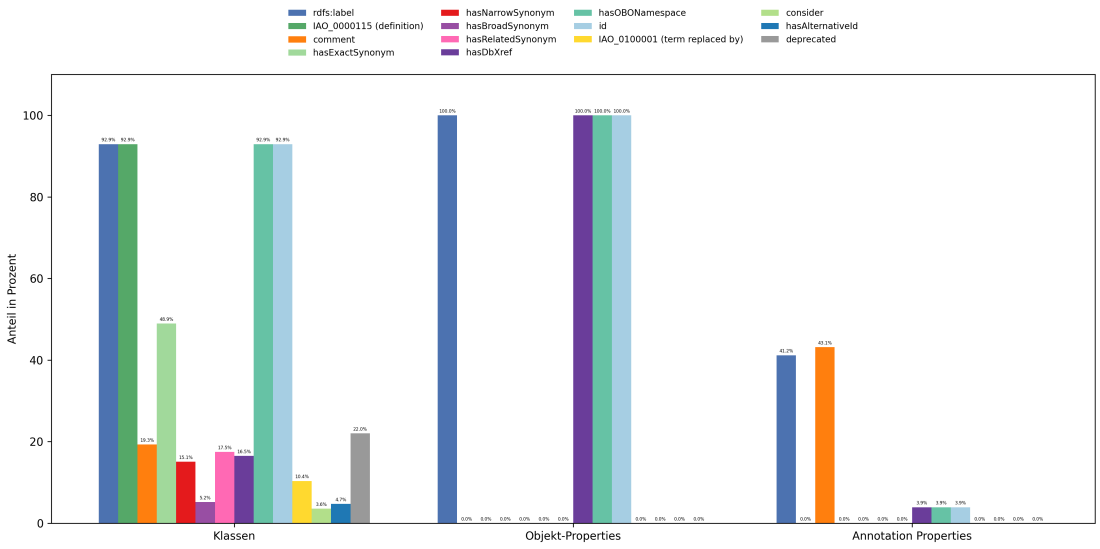

Abbildung 4.7: Anzahl annotierter Symbole in der Gene-Ontologie

Zur genaueren Einschätzung der inhaltlichen Ausführlichkeit wurde die durchschnittliche Wortanzahl verschiedener Annotationstypen untersucht. In Abbildung 4.8 sind nur Annotationen mit einer durchschnittlichen Länge von mehr als einem Wort dargestellt.

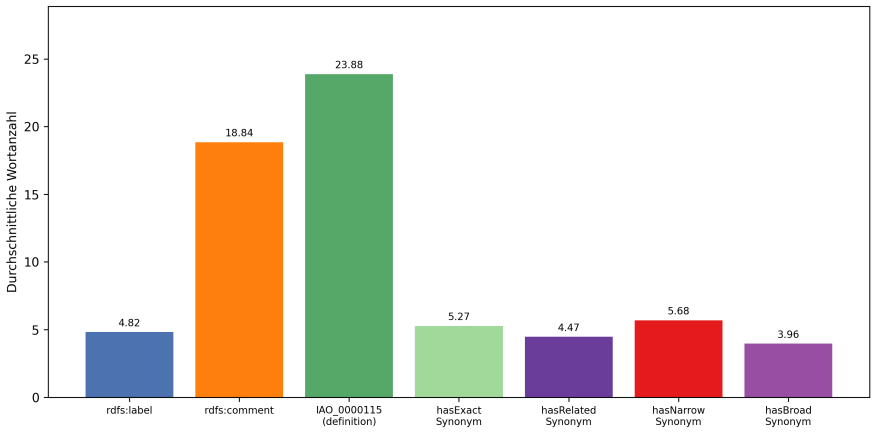

Abbildung 4.8: Durchschnittliche Wortanzahl von Annotationen mit einer durchschnittlichen Länge von mehr als einem Wort

Die Abbildung zeigt, dass insbesondere die `IAO_0000115` Annotationen mit durchschnittlich 23,88 Wörtern sowie die `rdfs:comment` Annotationen mit 18,84 Wörtern eine besonders ausführliche Beschreibung der Konzepte liefern. Labels (`rdfs:label`) sind mit 4,82 Wörtern im Vergleich dazu deutlich kürzer. Auch die synonymbezogenen Annotationen wie `hasExactSynonym`, `hasRelatedSynonym`, `hasNarrowSynonym` und `hasBroadSynonym` weisen eine gewisse inhaltliche Tiefe auf

und tragen damit zur besseren Auffindbarkeit und semantischen Erschließung der Begriffe bei.

Schreibweise und Sprache der Symbolnamen

Bei den Symbolnamen in der Gene-Ontology fällt auf, dass alle Begriffe einem technischen Code-Muster folgen: Sie bestehen aus einem festen Präfix wie `GO` (für Gene Ontology), gefolgt von einer siebenstelligen Ziffernfolge[14], z. B. `GO_0003674` oder `GO_0016020`. Diese strukturierte Schreibweise dient der eindeutigen Identifizierung der Konzepte, ist aber semantisch wenig aussagekräftig. Eine vergleichbare Kodierung wurde bereits in der CLYH-Ontologie beobachtet.

Die eigentliche Bedeutung der Begriffe erschließt sich erst durch begleitende Annotationen wie `rdfs:label`, die sprechende Bezeichnungen wie *molecular_function* enthalten, sowie durch präzise formulierte Definitionen und Kommentare.

Sowohl die Labels als auch alle Annotationen sind durchgängig in englischer Sprache verfasst. Insgesamt zeigt sich - ähnlich wie bei der CLYH-Ontologie - eine formal konsistente, aber inhaltlich wenig sprechende Namensvergabe, deren Bedeutung maßgeblich durch Annotationen ergänzt wird.

4.3 Automatische Keyword-Extraktion aus Ontologie-Annotationen

Die Analyse und Nutzung semantischer Informationen aus Ontologien kann - insbesondere bei komplexen oder unstrukturierten Datenbeständen - aufwändige manuelle Auswertungsschritte erfordern. Automatisierte Methoden zur Extraktion relevanter semantischer Informationen können hier eine wesentliche Verbesserung bringen, indem sie eine schnelle, zuverlässige und reproduzierbare Verarbeitung von Ontologien ermöglichen [NAM23].

In diesem Kapitel wird daher eine Methode zur automatischen Extraktion von Keywords aus textuellen Annotationen wie Definitionen oder Kommentaren entwickelt und beschrieben. Ziel dieser Keyword-Extraktion ist es, die wesentlichen semantischen Inhalte einer Ontologie zu erfassen und damit semantische Analysen gezielt zu unterstützen.

Eingesetzte Technologien und Werkzeuge

Für die Umsetzung der automatisierten Keyword-Extraktion wurde die Python-Bibliothek Owlready2 verwendet, um die Ontologie im OWL-Format einzulesen und zu verarbeiten [15]. Owlready2 bietet umfangreiche Funktionen, um Ontologien komfortabel zu durchsuchen und Annotationen gezielt zu extrahieren.

Für die anschließende Keyword-Extraktion aus den ermittelten Annotationen wurde ein Transformer-basiertes Sprachmodell der Plattform Hugging Face verwendet. Konkret wurde das Modell *„ilsilfverskiold/tech-keywords-extractor*[16]“ ver-

[14] `https://geneontology.org/docs/ontology-documentation`, abgerufen am 15.04.2025.
[15] Siehe `https://github.com/PariaBolouki/Bachelorarbeit`
[16] `https://huggingface.co/ilsilfverskiold/tech-keywords-extractor`, abgerufen am 15.04.2025.

wendet, das eine fine-tuned Version des Modells *„facebook/bart-large“* ist. Laut der Beschreibung auf Hugging Face wurde das Modell darauf trainiert, Begriffe wie technische Konzepte, Werkzeuge, Programmiersprachen, Plattformen oder Firmennamen in Texten zu erkennen und zu extrahieren.

Verarbeitung und Extraktion der Ontologie-Annotation

Um relevante Keywords aus der Ontologie zu extrahieren, wurden zunächst alle Annotationen der Ontologie mit Hilfe der Python-Bibliothek Owlready2 systematisch extrahiert. Konkret wurden dabei Annotationen nicht nur auf Ebene der Klassen betrachtet, sondern ebenfalls bei Properties (Object-Properties, Data-Properties, Annotation-Properties) sowie Individuen.

Bei der Extraktion wurden insbesondere Annotationen berücksichtigt, die typischerweise semantische Informationen enthalten. Dazu gehören Annotationen wie `definition`, `comment` und `description`. Zusätzlich wurde auch nach `IAO_0000115` Annotationen gesucht. Bereits bei der Analyse der Ontologien in Kapitel 4.2 wurde festgestellt, dass einige Ontologien ihre Definitionsannotationen nicht als `definition`, sondern explizit mit der Annotation `IAO_0000115` (mit dem Label „definition“) versehen. Die so gewonnenen Annotationstexte wurden dann gesammelt, um im nächsten Schritt als Input für die eigentliche Keyword-Extraktion mit dem Sprachmodell verwendet zu werden.

Durchführung der Keyword-Extraktion

Nachdem alle relevanten Annotationen aus der Ontologie extrahiert und gesammelt wurden, erfolgte im nächsten Schritt die eigentliche Keyword-Extraktion mittels des Transformer-basierten Sprachmodells *„ilsilfverskiold/tech-keywords-extractor“*. Dazu wurden die zuvor extrahierten Annotationstexte einzeln als Input an das Modell übergeben.

Das verwendete Modell generiert aus dem jeweiligen Annotationstext automatisch eine Reihe relevanter Keywords. Die Ausgabe erfolgt dabei typischerweise in Form einer durch Kommata getrennten Liste von Begriffen. Für jede Annotation liefert das Modell somit eine Menge von Keywords, die den wesentlichen semantischen Inhalt widerspiegeln. Die resultierenden Keywords wurden dann in einem strukturierten JSON-Format gespeichert, wobei jedem Ontologieelement (Klasse, Property oder Individuum) die extrahierten Keywords zugeordnet wurden.

Beispielhafte Ergebnisse

Im Folgenden werden beispielhaft Ergebnisse der automatisierten Keyword-Extraktion vorgestellt. Die ersten Beispiele stammen aus der CLYH-Ontologie, die bereits in Kapitel 4.2 im Rahmen der Ontologieanalyse betrachtet wurde. Ziel ist es, zu veranschaulichen, welche Arten von Begriffen das Sprachmodell aus den Annotationstexten als Keywords extrahiert. In der folgenden Tabelle sind die Annotationstextd und die vom Modell extrahierten Keywords einander gegenübergestellt.

Symbolname	Annotationstext	Extrahierte Keywords
CLYH_1000009	"Oral part of the outer layer of the gastrula."	{Gastrula, Oral Layer}
CLYH_1000025	"Organism at the cleavage stage after the first cell division."	{Organism, Cleavage, Cell Division}
CLYH_1000023	"Anatomical entity that comprises the organism in the early stages of growth and differentiation that are characterized by cleavage, the laying down of fundamental tissues."	{Anatomical Entity, Anatomy}

Tabelle 4.10: Beispiele der Keyword-Extraktion aus der CLYH-Ontologie

Die in Tabelle 4.10 dargestellten Beispiele zeigen, dass das verwendete Sprachmodell in der Lage ist, sowohl kurze als auch komplexere Annotationstexte zu verarbeiten und daraus relevante Schlüsselbegriffe zu extrahieren. Insbesondere bei prägnanten Annotationen – wie im Fall von CLYH_1000009 – werden die zentralen Begriffe zuverlässig erkannt.

Auffällig ist jedoch, dass das Modell bei längeren, inhaltlich dichteren Beschreibungen – wie bei CLYH_1000023 – vergleichsweise weniger Keywords generiert. Obwohl der Text mehrere potenziell relevante Konzepte enthält, wurden nur zwei relativ generische Begriffe extrahiert. Dies deutet darauf hin, dass das Modell bei längeren oder abstrakter formulierten Annotationen Schwierigkeiten hat, die inhaltliche Vielfalt vollständig abzubilden.

Da die Pizza-Ontologie keine fachspezifischen Begriffe enthält, ist es besonders interessant zu sehen, welche Keywords aus ihren Annotationen extrahiert werden. Im Gegensatz zur CLYH-Ontologie, in der nur Klassen Annotationen enthalten, sind in der Pizza-Ontologie auch bei Objekt-Properties Kommentare vorhanden. Die folgende Tabelle zeigt zwei Beispiele.

Symbolname	Annotationstext	Extrahierte Keywords
hasTopping	"Note that hasTopping is inverse functional because isToppingOf is functional."	{HasTopping, inverse functional}
isIngredientOf	"The inverse property tree to hasIngredient – all subproperties and attributes of the properties should reflect those under hasIngredient."	{Inverse Property Tree, hasIngredient}

Tabelle 4.11: Beispiele der Keyword-Extraktion aus der Pizza-Ontologie

Die Tabelle 4.11 zeigt, dass auch aus den allgemeinen, nicht fachspezifischen Kommentaren der Pizza-Ontologie sinnvolle Keywords extrahiert werden können. Trotz der vergleichsweise einfachen Sprache und dem Fehlen biologischer Fachbegriffe identifiziert das Modell zentrale Begriffe korrekt. Dies spricht für eine robuste Anwendbarkeit der Methode - auch auf Ontologien mit eher alltagssprachlichem Vokabular.

Die Beispiele beider Ontologien zeigen, dass längere Annotationstexte nicht zwangsläufig zu einer höheren Anzahl extrahierter Keywords führen. So wurden bei `CLYH_1000023` und `isIngredientOf` trotz vergleichsweise ausführlicher Beschreibungstexte jeweils nur zwei Keywords extrahiert. Eine systematische Analyse dieses Zusammenhangs wurde im Rahmen dieser Arbeit nicht durchgeführt, könnte aber Gegenstand zukünftiger Untersuchungen sein.

5

Evaluation der Methoden

5.1 Erstellung eines Goldstandards

Für die Evaluation der entwickelten Methode zur automatischen Keyword-Extraktion wurde ein manuell erstellter Goldstandard benötigt. Um eine objektive und unabhängige Bewertung zu ermöglichen, wurde bewusst eine Ontologie gewählt, mit der im weiteren Verlauf der Arbeit nicht gearbeitet oder experimentiert wurde. Ziel war es, eine Ontologie zu finden, die inhaltlich neu ist, gleichzeitig aber über ausreichend textuelle Annotationen verfügt, um eine fundierte manuelle Extraktion von Schlüsselwörtern zu ermöglichen.

Nach Prüfung verschiedener Kandidaten wurde die Anatomical Entity Ontology[1] (AEO) aus dem BioPortal als geeignet identifiziert. Diese Ontologie erfüllt zwei zentrale Anforderungen: Sie enthält eine moderate Anzahl von Klassen, so dass eine manuelle Sichtung und Bearbeitung realistisch möglich ist, und sie weist eine hohe Dichte an textbasierten Annotationen auf. Um dies quantitativ zu überprüfen, wurde eine systematische Analyse der enthaltenen Annotationen durchgeführt. Von insgesamt 250 Klassen enthielten 235 Klassen eine `IAO_0000115`-Definition, mit einer durchschnittlichen Wortanzahl von 15,04 Wörtern pro Definition. Damit erwies sich die AEO als eine geeignete Basis für die Evaluation.

Die Ontologie wurde in Protégé geladen und alle Klassen mit vorhandenen Definitionen wurden identifiziert. Aus den 235 Klassen mit `IAO_0000115`-Definitionen wurden 50 Klassen manuell und zufällig ausgewählt. Dazu wurden die einzelnen Klassen in Protégé durch manuelles Anklicken aufgerufen. Wenn eine Definition vorhanden war, wurde die Klasse in die Auswahl aufgenommen. Dieser Vorgang wurde wiederholt, bis 50 Klassen mit Definitionen vorhanden waren. Für jede dieser 50 Klassen wurden die enthaltenen Definitionstexte manuell analysiert und basierend darauf eine Liste von Keywords erstellt, die aus menschlicher Sicht die zentralen Begriffe und Konzepte der jeweiligen Definition erfassen.

Zur Veranschaulichung ist im Folgenden ein beispielhafter Eintrag aus dem Goldstandard dargestellt. Die Klasse `CL_0000526` enthält in der Ontologie die folgende Definition und die dazugehörigen Keywords:

[1] `https://bioportal.bioontology.org/ontologies/AEO`, abgerufen am 15.04.2025.

```
"text": "A neuron which conveys sensory information centrally from
         the periphery."

"keywords": ["Neuron",
             "Sensory Information",
             "Periphery"]
```

Um eine vergleichbare Datengrundlage für die Evaluation zu schaffen, wurde anschließend die bestehende Implementierung aus Abschnitt 4.3 auf dieselbe Ontologie angewendet. Dabei wurde darauf geachtet, dass die automatische Keyword-Extraktion nur für die zuvor ausgewählten 50 Klassen durchgeführt wurde. Das Ergebnis dieses Schrittes wurde als JSON-Datei gespeichert.

Parallel dazu wurden die manuell extrahierten Keywords in einer identischen JSON-Struktur dokumentiert, so dass am Ende zwei gleich aufgebaute Datensätze vorlagen: einer mit dem manuell erstellten Goldstandard, und einer mit den automatisch generierten Keywords. Diese dienen im nächsten Abschnitt als Grundlage für den quantitativen Vergleich.

5.2 Methodik des Vergleichs

Um die Qualität der automatisch extrahierten Keywords zu bewerten, wurden diese mit dem zuvor manuell erstellten Goldstandard verglichen. Für jede der 50 ausgewählten Klassen aus der AEO-Ontologie wurden die entsprechenden Keyword-Listen gegenübergestellt.

Für jede Klasse wurde ermittelt, welche Keywords korrekt erkannt wurden (TP[2]), welche fälschlicherweise extrahiert wurden (FP[3]) und welche relevanten Keywords vom System nicht erkannt wurden (FN[4]). Keywords, die weder im Goldstandard noch in der automatisch extrahierten Liste vorkommen (True Negatives), wurden nicht berücksichtigt, da dies eine vollständige Menge aller potenziellen Keywords voraussetzen würde, die in diesem Kontext nicht vorliegt [JM09].

Zur Kategorisierung der Keywords in TP, FP und FN wird eine vereinfachte Confusion-Matrix verwendet (siehe Tabelle 5.1). Sie bildet die Grundlage für die Berechnung der in dieser Arbeit verwendeten Metriken. Ein konkretes Beispiel für die Anwendung dieser Kategorisierung findet sich im Abschnitt 5.3 anhand der Klasse `AEO_0000211`.

[2] True Positives (TP): Keywords, die sowohl im Goldstandard als auch in der automatisch generierten Liste enthalten sind

[3] False Positives (FP): Keywords, die vom System extrahiert wurden, aber nicht im Goldstandard vorkommen.

[4] False Negatives (FN): Keywords, die im Goldstandard vorhanden sind, aber vom System nicht erkannt wurden.

	Gold Positiv	**Gold Negativ**
System Positiv	TP	FP
System Negativ	FN	–

Tabelle 5.1: Darstellung der verwendeten Metriken auf Basis einer vereinfachten Confusion-Matrix [JM09]

Auf der Grundlage dieser Werte wurden die drei üblichen Evaluationsmetriken berechnet: **Precision**, **Recall** und **F1-Score**. Die Berechnung erfolgte pro Klasse: Für jede der 50 Ontologieklassen wurden Precision, Recall und F1-Score separat bestimmt. Diese sind Standardmaße für die Bewertung der Qualität von Klassifikationssystemen und insbesondere im Bereich der Informationsextraktion verbreitet [JM09].

- **Precision** beschreibt den Anteil der korrekt extrahierten Keywords an allen automatisch extrahierten Keywords [JM09]:

$$\text{Precision} = \frac{TP}{TP + FP}$$

  Eine hohe Precision bedeutet, dass nur wenige irrelevante Keywords extrahiert wurden.
- **Recall** misst den Anteil der korrekt extrahierten Keywords an allen relevanten Keywords im Goldstandard [JM09]:

$$\text{Recall} = \frac{TP}{TP + FN}$$

  Eine hohe Recall-Rate bedeutet, dass das System viele relevante Keywords erkennt.
- **F1-Score** ist das harmonische Mittel aus Precision und Recall und kombiniert beide Maße in einer einzigen Bewertungsgröße [JM09]:

$$\text{F1} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

5.3 Evaluationsergebnisse

Die beschriebenen Metriken wurden anschließend auf die automatisch extrahierten und manuell erstellten Keyword-Listen angewendet. Die Berechnung erfolgte für jede der 50 ausgewählten Klassen separat, wobei jeweils die Werte für TP, FP und FN ermittelt wurden. Zusätzlich wurden die Werte über alle Klassen hinweg aufsummiert, um auf dieser Basis eine Gesamtbewertung der Extraktionsergebnisse durch Mittelwerte für Precision, Recall und F1-Score zu ermöglichen.

Die Auswertung erfolgte mit Hilfe eines in Python implementierten Skripts, das die beiden JSON-Dateien – eine mit den automatisch extrahierten Keywords und eine mit dem Goldstandard – vergleicht. Es handelte sich um einen exakten,

wortbasierten Vergleich, bei dem zwei Begriffe nur dann als gleichwertig betrachtet wurden, wenn sie zeichengenau (mit Ausnahme der Groß- und Kleinschreibung) übereinstimmten.

Ein exemplarisches Beispiel ist die Klasse `AEO_0000211`. Die zugehörige Definition lautet:

```
"text": "The sum of all the anlagen and primordia that will
        develop into a single functional system."
```

Der manuell erstellte Goldstandard enthält für diese Klasse folgende Keywords:

```
"keywords": [
    "Anlagen",
    "Primordia",
    "Develop",
    "Single Functional System"
]
```

Die systematisch extrahierten Keywords lauteten dagegen:

```
"keywords": [
    "Anlagen",
    "Primordia",
    "Functional System"
]
```

Der Vergleich dieser beiden Listen ergibt die in Tabelle 5.2 und Tabelle 5.3 dargestellten Ergebnissen.

	Gold Positiv	**Gold Negativ**
System Positiv	2 (TP)	1 (FP)
System Negativ	2 (FN)	–

Tabelle 5.2: Confusion-Matrix für die Klasse `AEO_0000211`

Zwei Keywords, “Anlagen” und “Primordia”, wurden korrekt aus dem Goldstandard erkannt (TP = 2). Ein weiteres Keyword, „Functional System“, wurde zusätzlich extrahiert, obwohl es inhaltlich dem im Goldstandard enthaltenen „Single Functional System“ ähnelt. Da jedoch keine exakte Übereinstimmung vorlag, wurde es als falsch positiv (FP = 1) gewertet. Darüber hinaus wurden zwei Keywords aus dem Goldstandard, nämlich “Develop” und “Single Functional System”, nicht erkannt (FN = 2). Daraus ergeben sich die in Tabelle 5.3 dargestellten Metriken.

Metrik	Wert
Precision	0,67
Recall	0,50
F1-Score	0,57

Tabelle 5.3: Berechnete Metriken für die Klasse `AEO_0000211`

Dieses Ergebnis zeigt, dass das System zwar einige wichtige Keywords richtig erkannt hat, aber auch relevante Keywords übersehen und ein zusätzliches Keyword extrahiert hat.

Gesamtergebnisse

Neben der detaillierten Auswertung der einzelnen Klassen wurden auch aggregierte Kennzahlen über alle 50 ausgewählten Ontologieklassen hinweg berechnet. Für die Berechnung der Mittelwerte der Metriken wurde der sogenannte Makro-Durchschnitt verwendet. Dabei wurden Precision, Recall und F1-Score jeweils für jede der 50 Klassen einzeln berechnet und anschließend der Mittelwert über alle Klassen hinweg gebildet. Auf diese Weise gingen alle Klassen gleichgewichtet in das Gesamtergebnis ein.

Metrik	Durchschnittlicher Wert
Precision	0,72
Recall	0,62
F1-Score	0,66

Tabelle 5.4: Durchschnittliche Evaluationsmetriken über alle 50 Klassen

Die in Tabelle 5.4 dargestellten Werte zeigen, dass das entwickelte System in der Lage ist, einen Großteil relevanter Keywords korrekt zu erkennen. Gleichzeitig werden aber auch vereinzelt irrelevante Keywords extrahiert oder relevante Keywords übersehen.

Die grafische Darstellung zeigt, dass die durchschnittliche Precision mit 0,72 etwas höher ist als der Recall mit 0,62. Dies bedeutet, dass das System tendenziell präziser arbeitet, d.h. es extrahiert eher relevante Keywords, als dass es alle relevanten Keywords vollständig erkennt. Der F1-Score von 0,66 deutet auf eine insgesamt solide, aber noch verbesserungsfähige Extraktionsleistung hin.

Eine Einschränkung der verwendeten Evaluationsmethode besteht darin, dass nur ein exakter, wortbasierter Abgleich zwischen den extrahierten Begriffen und dem Goldstandard durchgeführt wurde. Semantische Ähnlichkeiten oder Teilübereinstimmungen - z.B. wenn ein extrahiertes Keyword nur ein Bestandteil eines längeren Keywords ist - wurden nicht berücksichtigt. Zukünftige Arbeiten könnten diese Einschränkung gezielter adressieren.

6

Diskussion

Diese Bachelorarbeit widmete sich der systematischen Analyse und automatischen Extraktion von Keywords aus textuellen Ontologie-Annotationen. Zunächst wurden Symbolnamen, Annotationen sowie deren Verteilung und sprachliche Eigenschaften in vier verschiedenen Ontologien untersucht. Bereits hier zeigte sich, dass Annotationen sehr unterschiedlich verwendet werden.

Darauf aufbauend wurde eine Methode zur automatischen Keyword-Extraktion entwickelt und angewendet, basierend auf einem Transformer-basierten Sprachmodell der Plattform Hugging Face. Dabei zeigte sich unter anderem, dass längere Definitionen oder Kommentare nicht unbedingt zu einer höheren Anzahl extrahierter Keywords führen - ein Ergebnis, das das Potenzial für weitere Analysen bietet. Eine zukünftige Arbeit könnte im Detail untersuchen, welche Eigenschaften der Annotationstexte tatsächlich einen Einfluss auf die Menge und Qualität der extrahierten Keywords haben und wie sich diese Erkenntnisse zur Optimierung automatischer Extraktionsmethoden genutzt werden können.

Zur Evaluierung der entwickelten Methode wurde ein Goldstandard erstellt, mit dem die Ergebnisse der automatischen Extraktion verglichen wurden. Die durchgeführte Evaluation war jedoch eingeschränkt, da ausschließlich exakte, wortgenaue Übereinstimmungen berücksichtigt wurden. Diese Form der Evaluation ist nur eingeschränkt aussagekräftig. Um die Qualität automatischer Extraktionsverfahren in Zukunft realistischer und differenzierter beurteilen zu können, sollten semantische Ähnlichkeiten und partielle Übereinstimmungen stärker berücksichtigt werden.

Darüber hinaus eröffnet sich die Perspektive, verschiedene Sprachmodelle und Methoden systematisch miteinander zu vergleichen. Solche Vergleiche könnten wertvolle Erkenntnisse darüber liefern, welche Modelle oder Verfahren für bestimmte Ontologietypen oder spezifische Anwendungsfälle besonders geeignet sind - und damit die Forschung im Bereich der semantischen Analyse und der automatisierten Wissensextraktion nachhaltig voranbringen.

Literaturverzeichnis

BCM+03. Baader, Franz, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi und Peter F. Patel-Schneider (Herausgeber): *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003.

BHL+14. Busse, Johannes, Bernhard Humm, Christoph Lübbert, Frank Moelter, Anatol Reibold, Matthias Rewald, Veronika Schlüter, Bernhard Seiler, Erwin Tegtmeier und Thomas Zeh: *Was bedeutet eigentlich Ontologie? - Ein Begriff aus der Philosophie im Licht verschiedener Disziplinen*. Inform. Spektrum, 37(4):286–297, 2014.

BVZ22. Bauer, Matthias, Gabriel Viehhauser und Angelika Zirker: *Zwischenräume. Kommentierende Annotation und hermeneutische Bedeutungserschließung in digitalen Texten*, Seiten 249–279. J.B. Metzler, Stuttgart, 2022.

CDG+24. Chataut, Sandeep, Tuyen Do, Bichar Dip Shrestha Gurung, Shiva Aryal, Anup Khanal, Carol Lushbough und Etienne Z. Gnimpieba: *Comparative Study of Domain Driven Terms Extraction Using Large Language Models*. CoRR, abs/2404.02330, 2024.

Fir57. Firth, J.: *A synopsis of linguistic theory 1930-1955*. Studies in Linguistic Analysis, Philological. Longman, 1957.

FKS19. Furbach, Ulrich, Teresa Krämer und Claudia Schon: *Names Are Not Just Sound and Smoke: Word Embeddings for Axiom Selection*. In: Fontaine, Pascal (Herausgeber): *Automated Deduction - CADE 27 - 27th International Conference on Automated Deduction, Natal, Brazil, August 27-30, 2019, Proceedings*, Band 11716 der Reihe *Lecture Notes in Computer Science*, Seiten 250–268. Springer, 2019.

GDA24. Giglou, Hamed Babaei, Jennifer D'Souza und Sören Auer: *Preface for LLMs4OL 2024: The 1st Large Language Models for Ontology Learning Challenge at the 23rd ISWC*. In: Giglou, Hamed Babaei, Jennifer D'Souza und Sören Auer (Herausgeber): *LLMs4OL 2024: The 1st Large Language Models for Ontology Learning Challenge at the 23rd ISWC,* Co-located with the 23rd International Semantic Web Conference (ISWC 2024)*, Baltimore, Maryland, USA,*

November 11-15, 2024, Band 4 der Reihe *Open Conference Proceedings*, Seiten 1–2. TIB Open Publishing, 2024.

Har81. Harris, Zellig S.: *Distributional Structure*, Seiten 3–22. Springer Netherlands, Dordrecht, 1981.

HKR10. Hitzler, Pascal, Markus Krötzsch und Sebastian Rudolph: *Foundations of Semantic Web Technologies*. Chapman and Hall/CRC Press, 2010.

JM09. Jurafsky, Dan und James H. Martin: *Speech and language processing: an introduction to natural language processing, computational linguistics, and speech recognition, 2nd Edition*. Prentice Hall series in artificial intelligence. Prentice Hall, Pearson Education International, 2009.

JS24. Jakobs, Oliver und Claudia Schon: *Context-Specific Selection of Commonsense Knowledge Using Large Language Models*. In: Hotho, Andreas und Sebastian Rudolph (Herausgeber): *KI 2024: Advances in Artificial Intelligence - 47th German Conference on AI, Würzburg, Germany, September 25-27, 2024, Proceedings*, Band 14992 der Reihe *Lecture Notes in Computer Science*, Seiten 218–231. Springer, 2024.

KSH14. Krötzsch, Markus, Frantisek Simancik und Ian Horrocks: *Description Logics*. IEEE Intell. Syst., 29(1):12–19, 2014.

LBB+16. Lordick, Harald, Rainer Becker, Michael Bender, Luise Borek, Canan Hastik, Thomas Kollatz, Beata Mache, Andrea Rapp, Ruth Reiche und Niels-Oliver Walkowski: *Digitale Annotationen in der geisteswissenschaftlichen Praxis*. Bibliothek Forschung und Praxis, 40(2):186–199, 2016.

LCL+24. Li, Rongbin, Wenbo Chen, Jinbo Li, Hanwen Xing, Hua Xu, Zhao Li und W. Jim Zheng: *GPTON: Generative Pre-trained Transformers enhanced with Ontology Narration for accurate annotation of biological data*. CoRR, abs/2410.10899, 2024.

NAM23. Nadim, Mohammad, David Akopian und Adolfo Matamoros: *A Comparative Assessment of Unsupervised Keyword Extraction Tools*. IEEE Access, 11:144778–144798, 2023.

Sch23. Schon, Claudia: *Associative Reasoning for Commonsense Knowledge*. In: Seipel, Dietmar und Alexander Steen (Herausgeber): *KI 2023: Advances in Artificial Intelligence - 46th German Conference on AI, Berlin, Germany, September 26-29, 2023, Proceedings*, Band 14236 der Reihe *Lecture Notes in Computer Science*, Seiten 170–183. Springer, 2023.

Sch24. Schon, Claudia: *Using the Meaning of Symbol Names to Guide First-Order Logic Reasoning*. In: Özçep, Özgür Lütfü, Nele Russwinkel, Kai Sauerwald und Diedrich Wolter (Herausgeber): *Proceedings of the 10th Workshop on Formal and Cognitive Reasoning colocated with the 47th German Conference on Artificial Intelligence (KI*

2024), Würzburg, Germany, September 23, 2024, Band 3763 der Reihe *CEUR Workshop Proceedings*, Seiten 19–27. CEUR-WS.org, 2024.

SS09. Staab, Steffen und Rudi Studer (Herausgeber): *Handbook on Ontologies*, International Handbooks on Information Systems. Springer, 2009.

Stu09. Stuckenschmidt, Heiner: *Ontologien: Konzepte, Technologien und Anwendungen*. Informatik im Fokus. Springer, 2009.

TP10. Turney, Peter D. und Patrick Pantel: *From Frequency to Meaning: Vector Space Models of Semantics*. J. Artif. Intell. Res., 37:141–188, 2010.

A

Selbstständigkeitserklärung

Hiermit versichere ich, dass ich diese wissenschaftliche Arbeit selbständig verfasst und nur die angegebenen Quellen und Hilfsmittel verwendet habe. Alle von mir aus anderen Veröffentlichungen übernommenen Passagen sind als solche gekennzeichnet.

Zur Optimierung der Python-Implementierung sowie zur sprachlichen Überarbeitung und Verbesserung der Lesbarkeit habe ich unterstützende KI-Tools verwendet, ohne den inhaltlichen Eigenanteil der Arbeit zu beeinflussen.

17.04.2025

Datum

Unterschrift der Kandidatin/des Kandidaten