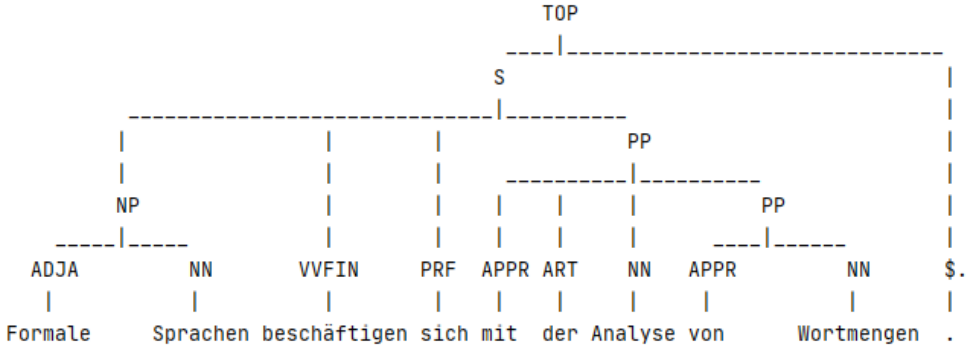
KVASIR-Wörterbuch-Integration in den Berkeley-Parser

Es gibt unter https://github.com/slavpetrov/berkeleyparser einen java-basierten Parser, dessen Fundament in folgenden wissenschaftlichen Arbeiten beschrieben ist:

- Slav Petrov, Leon Barrett, Romain Thibaux, Dan Klein: *Learning Accurate, Compact, and Interpretable Tree Annotation*. In COLING-ACL 2006
- Slav Petrov, Dan Klein: *Improved Inference for Unlexicalized Parsing*. In HLT-NAACL 2007

Das Lexikon und die Grammatik dieses Parsers wurden aus Treebanks gelernt. Sie liegen als Modell für Deutsch im ger_sm5.gr-File vor. Mit dem Parser wird eine Ausgabefunktion mitgeliefert, die aus dem .gr-File 4 lesbare Dateien generiert. Diesen Dateien ist zu entnehmen, dass die genutzte Grammatik zwar in ihrem Kern einem Produktionsregelsystem in Chomsky-Normalform (CNF) entspricht, aber darüber hinaus eine Reihe von künstlich generierten Regeln und Wichtungen enthält, die auf einen ML-Algorithmus zurückzuführen sind. Insgesamt berücksichtigt die Grammatik (mit dem Lexikon) 47895 Token[1]. Aufgrund der Größe des Alphabets erzeugt der Berkeley-Parser für Sätze der Deutschen Sprache gute Syntaxbäume aus Wortarten und Konstituenten (vgl. https://de.wikipedia.org/wiki/Konstituente).

Für den Satz *„Formale Sprachen beschäftigen sich mit der Analyse von Wortmengen.“* entsteht der folgende Syntaxbaum:



Aufgrund fehlender Fachbegriffe (im Beispiel nur „Formale“; für „Wortmengen“ gibt es beide Bestandteile) und der ausschließlichen Modellierung der Konstituenten-Struktur in den Grammatikregeln, besteht der Wunsch, diesen Parser zu erweitern und im Parsing-Prozess das eigene KVASIR-Wörterbuch satzbezogen zu integrieren. Der Parser selbst arbeitet mit Inputstreams in einem Pipeline-Modell. Dazu wird die eingelesene Zeichenkette in einzelne Sätze zerlegt (segmentation) und jeder Satz in seine Token (tokenization) aufgespalten. Für jeden Token wird im Anschluss ein Part-of-Speech-Tag (POS-Tag) bestimmt. Jeder POS-Tag repräsentiert eine andere Wortart, die anhand eines Wörterbuches bestimmt werden kann (tagging). Gleichzeitig entsprechen diese POS-Tags den Nichtterminalsymbolen in der CNF, die auf der rechten Seite zu Terminalsymbolen bzw. den Token führen. Diese Produktionsregeln sind im Lexikon des .gr-Files enthalten. Die Wortarten als Klassifikatoren für eine grammatische Syntaxanalyse sind aber nicht ausreichend. Die Klassen genügen nicht, um die syntaktisch korrekte Nutzung zu be-

[1] entspricht den Elementen des Alphabets in formalen Grammatiken

schreiben. Deshalb wird in der ML-basierten Berkeley-Grammatik mit Untergruppen gearbeitet, die durchnummeriert sind, z.B. für das obige Verb „beschäftigen“ gibt es 22 mit VVINF_0 bis VVINF_21 bezeichnete Untergruppen, die in den Produktionsregeln genutzt werden. Jeder Token erhält zu jeder Untergruppe einen Zugehörigkeitswert, der über die Analyse der Treebank berechnet wurde und in dem Lexikon des.gr-Files steht. Die genutzten Zahlen sind jedoch nicht sehr aussagekräftig, so dass langfristig echte Untergruppen mit syntaktischer Bedeutung wünschenswert sind. Zusätzlich soll mit dem Zugriff auf das KVASIR-Wörterbuch jeder Satz um Feature angereichert werden. Dies sind neben PER (Person), NUM (Numerus), MOD (Modus), TEM (Tempus), RFX (Reflexiv), LEM (Lemma) und PFX (Präfix) bei Verben auch Links zu Wissenselementen wie Frames entsprechend der Framesemantik. Folglich soll das Part-of-Speech-Tagging des Berkeley-Parsers ausgetauscht und der Parser ggf. so angepasst werden, dass er die Feature-Anreicherung als für ihn nicht relevanten Teil der Part-of-Speech-Tags ignoriert.

Leider liegt diese Version des Berkeley-Parsers nur in Java vor, während das KVASIR-Wörterbuch mit einer Python-Schnittstelle ausgestattet ist. Das Gesamtsystem soll primär Python nutzen, so dass der Java-Parser in das Gesamtsystem einzubetten ist. Das Ergebnis der Syntax-Analyse ist relevant, um einen zuverlässigen Chat-Bot für Inhalte der Theoretischen Informatik aufzubauen. Einerseits benötigt der Chat-Bot seine eigene Wissensbasis. Dazu soll er natürlich-sprachliche Sätze einlesen, analysieren und in eine interne Wissensrepräsentation überführen. Andererseits soll er mit einem Nutzer kommunizieren, ihm Fragen beantworten oder bei einer Problemlösung helfen. Auch dazu muss er die natürlich-sprachliche Eingabe des Nutzers syntaktisch analysieren können. Es wird erwartet, dass nur auf Basis einer internen symbolischen umfangreichen Wissensrepräsentation, der Chat-Bot angemessen reagieren kann. Deshalb wird die Fortsetzung das IT-Projekt als Bachelor-Arbeit empfohlen. Zu einer solchen Fortsetzung kann gehören, dass die im .gr-File enthaltenen Informationen in den nltk-Parser (vgl. https://www.nltk.org/) migriert werden müssen. Alternativ ist eine Auseinandersetzung mit dem Berkeley Neural Parser (vgl. https://github.com/nikitakit/self-attentive-parser) möglich. Dieser kann gemeinsam mit nltk genutzt oder in SpaCy als Komponente eingebunden werden. SpaCy (vgl. https://course.spacy.io/de/chapter1) stellt dabei ein sehr mächtiges State-of-the-Art ML-Framework zur natürlichen Sprachverarbeitung dar. Auch hier wäre der Untersuchungsgegenstand entweder der Ausbau der deutschsprachigen Modelle zur Unterstützung der Theoretischen Informatik-Terminologie oder die Einbettung von fundiertem symbolischen Fachwissen aus der internen Wissensrepräsentation in die SpaCy-Pipeline. Zu den konkreten Aufgaben des IT-Projektes gehören aber zunächst:

- Analyse des java-basierten Berkeley-Parsers (generelle Arbeitsweise & posTags-Erzeugung)
- Schaffung einer Schnittstelle zur Anbindung eines externen POS-Taggers
- Anbindung des KVASIR-Wörterbuchs als externer POS-Tagger

Die Lösung der 3 Teilaufgaben ist in geeigneter Form zu dokumentieren. Eine Powerpoint-Präsentation ist als äußere Form der Dokumentation geeignet. Zur Projektumsetzung werden eine fundierte Projekt-planung sowie regelmäßige Konsultationstermine unter Einhaltung aktueller Hygiene-Auflagen mit der betreuenden Hochschullehrerin oder Frau Tanja Schramm empfohlen.

Betreuende Hochschullehrerin: Prof. Dr. Oksana Arnold Mai 2021