

Syntax natürlicher Sprachen

Vorlesung 1: Intro

A. Wisiorek

Centrum für Informations- und Sprachverarbeitung,
Ludwig-Maximilians-Universität München

18.10.2022

Themen der heutigen Vorlesung

- 1 Übersicht
- 2 Programmsysteme
- 3 Syntaxanalyse mit NLTK

Bird, Steven, Ewan Klein und Edward Loper (2009). *Natural language processing with Python: analyzing text with the natural language toolkit*.

URL: <http://www.nltk.org/book/>.

Dürscheid, Christa (2010). *Syntax: Grundlagen und Theorien*. Bd. 3. Vandenhoeck & Ruprecht.

1. Übersicht

1 Übersicht

2 Programmsysteme

3 Syntaxanalyse mit NLTK

- Die Vorlesung behandelt **Grundbegriffe der Grammatik** und wesentliche syntaktische Konstruktionen des Deutschen im Hinblick auf eine **Verwendung in der maschinellen Sprachverarbeitung**
- Dazu werden die in neueren Grammatiktheorien verwendeten **Klassifizierungen von Phrasen**, deren **innere Struktur** sowie deren **relationale Abhängigkeiten** erklärt.
- Außerdem werden in der Computerlinguistik übliche **Grammatikformalismen** (wie Kontextfreie Grammatiken, Unifikationsgrammatiken, PCFGs, Datenbasierte Abhängigkeitsgrammatiken, Partielle Parsingmodelle)
- ebenso wie **syntaktische Annotationsstandards** (z.B. Penn Treebank, Universal Dependencies) vorgestellt und verwendet, um typische oder schwierige syntaktische Konstruktionen genau zu beschreiben.

- Kenntnis und Anwendung **funktionaler und struktureller Begriffe der grammatischen Beschreibung**:
 - Kongruenz
 - Rektion
 - Subkategorisierung
 - Valenz
- Kenntnis **Grammatikformalismus** und Anwendung für Analysen natürlichsprachlicher Sätze:
 - CFG (Kontextfreie Grammatik: *Konstituentenstruktur*)
 - FCFG (Feature-based CFG: *Modellierung grammatischer Merkmale*)
 - PCFG (Probabilistische CFG: *Gewichtung von CFG-Regeln*)
 - DG (Dependency Grammar: *Dependenzstruktur*)
 - **Chunk-Parser** (u.a. RegExpParser: *partielles Parsing, 'Parsing as Tagging'*)
- Kenntnis und Bedienung eines **Programmsystems**, das einen Grammatikformalismus verwendet

2. Programmsysteme

- 1 Übersicht
- 2 Programmsysteme**
- 3 Syntaxanalyse mit NLTK

NLTK: Anwendung regelbasierter Parsing-Grundlagen

- Schreiben von Grammatikregeln und Anwendung in Parsingalgorithmen
- konstituentenbasierte sowie dependenzbasierte Grammatiken
- 'Toy Grammar' für einzelne Beispielsätze

Informationen zum NLTK (Natural Language Toolkit)

- Bündel von Python-Bibliotheken und Programmen für computerlinguistische Anwendungen
- quelloffen, für Lehre entwickelt
- bietet auch Interfaces, z.B. für *Stanford Parser*
- Lehrbuch: <http://www.nltk.org/book>
- Dokumentation: <http://www.nltk.org/howto>
- Daten (Korpora, Grammatiken): <http://www.nltk.org/data.html>

spaCy: NLP-System mit *Dependency-Parser*

- modernes, deep-learning-basiertes NLP-System
- aus Korpora gelernte, umfangreiche Syntax-Modelle
- basierend auf Dependency-Treebanks
 - Beispiel einer **Dependency-Treebank**:
https://github.com/UniversalDependencies/UD_German-GSD/blob/master/de_gsd-ud-dev.conllu
- 'transition-based **Dependency-Parsing**':
<https://spacy.io/api/dependencyparser/>

Stanford-Parser: *NLP-System mit induzierten PCFG-Modellen*

- aus Korpus gelernte, kontextfreie Grammatikregeln mit Gewichtung
- **Konstituentenbasiertes Modell**, basierend auf CFG-Treebank
- Grundlage des englischen Modells ist die **Penn-Treebank**:
 - <https://catalog.ldc.upenn.edu/docs/LDC95T7/c193.html>
 - Sample der Penn-Treebank in NLTK-Data enthalten

Hinweis

Der Stanford-Parser wird im Kurs nur für Beispielanalysen herangezogen, nicht aktiv in der Übung verwendet! (ist als Online-Demo verfügbar)

Dependenzanalysen mit Stanford-Parser

- von CFG-Syntaxbaum abgeleitete Dependenz-Analysen
- **UD-Format** ('Universal Dependencies'):
<https://universaldependencies.org/u/dep/>

Demos

- <http://nlp.stanford.edu:8080/parser/index.jsp>
- <https://corenlp.run/>

3. Syntaxanalyse mit NLTK

1 Übersicht

2 Programmsysteme

3 Syntaxanalyse mit NLTK

Aufgaben eines Parsingalgorithmus

- 1 **Überprüfung** der grammatischen Struktur einer Eingabe auf Wohlgeformtheit
→ ***Suche einer Ableitung** gemäß der Regeln einer formalen Grammatik*
- 2 **Wiedergabe** der gefundenen grammatischen Struktur
→ ***Ableitungsbaum** (auch: Parsebaum, Syntaxbaum)*

Haupttypen

- 1 **CFGs** (kontextfreie Grammatiken)
 - Konstituentenstruktur (Phrasenstrukturgrammatik)
 - Strukturinformationen in den Knoten des Syntaxbaums
- 2 **Dependenzgrammatiken**
 - Abhängigkeitsrelationen zwischen Wörtern
 - Strukturinformationen in den Kanten des Syntaxbaums (Relationslabel)
- 3 **Chunk-Parser** (u.a. reguläre Grammatiken)
 - partielle Syntaxanalyse (nur wichtigste Konstituenten: NPs, VPs, PPs)
 - flache, nicht-hierarchische Strukturanalyse

Erweiterungen von CFGs

- 1 **PCFGs** (Probabilistische CFGs)
- 2 **FCFGs** (Feature-based-CFGs)

Problem

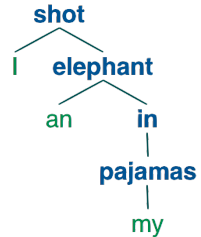
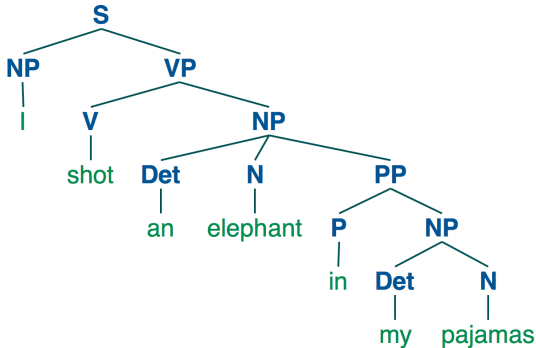
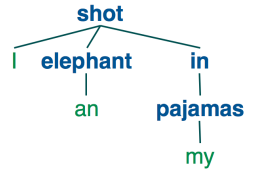
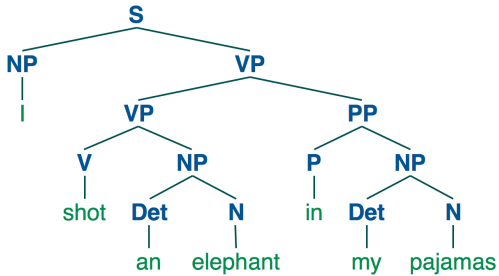
- mehr als eine Position im Syntaxbaum möglich

Lösung

- Disambiguierung u.a. über Probabilistische CFGs (PCFGs)

Beispielsatz für PP-Attachment

- *One morning I shot an elephant in my pajamas.
How he got into my pajamas I don't know.*
(Groucho Marx, Animal Crackers, 1930)



Übergenerierung (en. *overgeneration* / *overproduction*)

Problem

- CFG-Regeln einer Grammatik erzeugen (neben den zu erkennenden Beispielsätzen) auch nicht-wohlgeformte Sätze
- Grund: Nichtberücksichtigung morphologischer Constraints wie:
 - Kasusrektion
 - Kongruenz (Agreement)
 - Subkategorisierung (Art und Anzahl von Argumenten)

Lösung

- Modellierung morphologischer Constraints über Merkmalsstrukturen mit **Feature-based-CFGs** (FCFGs)

Beispiel für Übergenerierung

- CFG-Regeln für: *I shoot, He shoots*
- Übergenerierung: *I shoots, He shoot*

NLTK-Kapitel zu den Parsing-Beispielen

- **NLTK-Kapitel 8:** <https://www.nltk.org/book/ch08.html>
→ *Parsing mit CFGs, Dependenzgrammatiken, PCFGs*
- **NLTK-Kapitel 9:** <https://www.nltk.org/book/ch09.html>
→ *Parsing mit feature-based grammars*
- **NLTK-Kapitel 7:** <https://www.nltk.org/book/ch07.html>
→ *partiell Parsing mit RegexpParser*

Rückblick auf heutige Themen

- 1 Übersicht
- 2 Programmsysteme
- 3 Syntaxanalyse mit NLTK