TECHNISCHE UNIVERSITÄT DRESDEN

SEMANTIC COMPUTING

Lecture 2: Introduction to Linguistics and Natural Language Processing

Dagmar Gromann

International Center For Computational Logic

TU Dresden, 26 October 2018

Overview

- Brief Overview of Formal Semantics
- Introduction to Natural Language Processing
- Syntactic Parsing

Formal Semantics

Formal?

Natural Language

- evolves naturally and historically in a community of human speakers
- rules (grammar) discovered through empirical investigation (inherently complex)
- defined by syntax, semantics, and pragmatics
- complexity of NL understanding: AI-complete

Formal Language

- constructed, artificial language (exist as mathematical abstractions)
- properties and rules are stipulated mathematically (defined)
- defined by syntax and semantics (applications could be considered pragmatics)
- complexity theory (depending on problem)

Semantics and Not Pragmatics

The boundary between semantics and pragmatics is open for debate. In this lecture, we say that semantics looks at the literal meaning of a sentence, while pragmatics investigates the meaning of an utterance, that is, the use of the sentence.

Example sentences

a. Stand up, Bob! (imperative; command)
b. Could you please stand up, Bob? (interrogative; question)
c. Bob stands up. (assertive; statement)

Do all speech acts (a, b and c) have the same meaning? Formal semanticists are interested in evaluating whether sentences convey the same proposition, that is, the same semantic "value".

Formal Semantics

Main assumptions:

- speakers of a language have the same internalized rules for pairing lexical items (words, phrases, etc.) with non-linguistic elements in the world
- the meaning of a phrase is a function of the meanings of its immediate syntactic constituents and the way they are combined (Frege's Principles of Compositionality).

Formal semantics investigates the **construction of precise mathematical models** of the principles that speakers use to define the relationship between language and real-world.

Source: Lappin, S. (2008). An Introduction to Formal Semantics. In: The Handbook of Linguistics, Wiley, pp. 369-393, http://www.blackwellpublishing.com/content/BPL_Images/Content_store/WWW_Content/9780631204978/15.pdf

Linguistic Example

To understand statement a. we need to understand the entity "John" and the property "finished his paper'.' If speaker b. is able to understand a. and b. fully, it follows that asserting b. is incompatible with statement a (if b == true, a != true).

Example from Source

a. John has finished his paper.
b. No one in the class has finished his/her paper.

For a competent speaker of English, this is clear. However, developing a complete theory of this semantic competence which renders our **tacit knowledge of linguistic meaning formal and explicit** rather than presupposing it is not so easy, especially a theory that covers for all syntactically well-formed sentences.

Meaning and Denotation

Frege's idea: semantic theory = specification of a correspondence between syntactic categories and real-world entities (first-order logic types):

- individual terms (proper names or terms occurring in their position); e.g. "John" or "the Prime Minister"
- predicates (terms of properties and relations); e.g. "sings"
- connectives (and, or, if, then, not, ...)
- quantifiers (binding variables); e.g. "every" and "some"

Denotational meaning is defined in the relation of an expression to a real-world object and in this relation the **truth value** of assertions (declarative sentences) can be identified.

Example: "John sings" is true if and only if (iff) the individual which "John" denotes is an element of the set that "sings" denotes.

English to propositional logic

Truth-functional propositional logic that studies logical operators and their connectives to produce complex statements whose truth-values depends on the truth-values of their parts.

Conjunction

P = John sings.

Q = Mary dances.
Hypothesis: John signs or Mary dances.

P	Q	P ∨ Q	Q ∨ P
T	T	T	T
T	F	T	T
F	T	T	T
F	F	F	F

English to proposition logic

If P is true, then Q is true.

Implication

P = It is raining.

Q = It is cold.
Hypothesis: If it is raining, it is cold.

P	Q	$P \Longrightarrow Q$
T	T	T
T	F	F
F	T	T
F	F	T

A conditional statement is true unless the false when the antecedent (P) is satisfied, but the consequent (Q) is not.

English to predicate logic

Predicate logic extends propositional logic by quantification (some, every) and quantifiers range over a domain of discourse (e.g. all dogs).

Example

"All dogs are mamals."

$\forall\, x\ (\text{dog}(x) \implies \text{mammal}(x))$

This means all individuals with property P also have property Q.

$\forall\, x\ (\text{P}(x) \implies \text{Q}(x))$

"Some dogs are brown."

$\exists\, x\ (\text{dog}(x) \wedge \text{brown}(x))$

Montague grammar

- Richard Montague: Natural and formal languages can be treated alike (both are sets of strings); natural language **is** a formal language
- his background: set theory and modal logic
- Montague grammar: categorial grammar with a systematic correspondence between syntactic categories and semantic types; model-theoretical semantics: reference to a class of models has to be made in formalizing language, and therefore the interpretation of a language will be defined with respect to a set of (suitable) model
- proper introduction to Montague grammar goes beyond the scope of this lecture[1]

[1] If interested see
`https://plato.stanford.edu/entries/montague-semantics/`

Discourse Representation Theory

A theory introduced by Hans Kamp and Irene Heim independent of each other that has two main components:

- discourse referents (objects under discussion)
- Discourse Representation Structure (DRS)-conditions that encode the information gathered about the object

$[x, y : farmer(x), donkey(y), chased(x, y)]$
referents: x, y
DRS condition set: $farmer(x), donkey(y), chased(x, y)$
A farmer chased a donkey.

Comparison

Montagovian grammar

- focus on sentence and truth-value of sentence
- works well on quantification and coordination

Both

- model-theoretic tool

Discourse Representation Theory

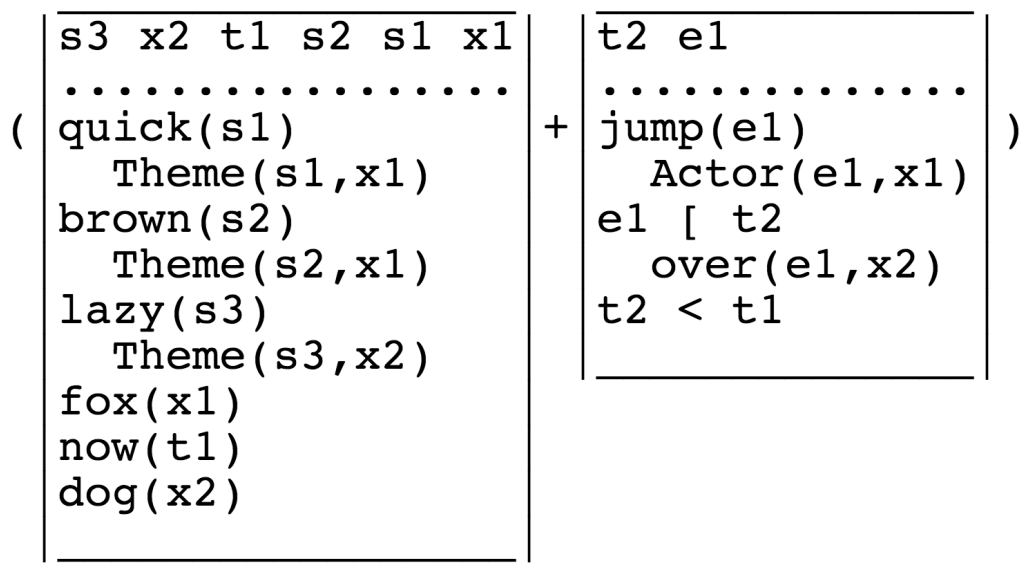
- focus on discourse rather than sentences (cross-sentential)
- works well on anaphora and tenses

Both

- model-theoretic tool

Text to DRS

Boxer was a Combinatory Categorial Grammar (CCG) parser that transformed sentences to a DRT notation. Which sentence is represented here?



Applications

- Usually with controlled natural language
- Question-answering
- Textual entailment (“Mark Twain wrote Huckleberry Finn." => “Mark Twain is a writer.”)
- Inferences
- ...

A set-theoretic approach (like the ones introduced above) is not so strong for semantic similarity, which is crucial to many NLP problems

Computational Example from NLTK

```
from nltk.sem.drt import *
dexpr = DrtExpression.fromstring
drs1 = dexpr('([x],[man(x),walk(x)])')
```

Output: ([x],[man(x), walk(x)])

```
drs2 = dexpr('([y],[woman(y),stop(y)])')
drs3 = drs1 + drs2
```

Output: (([x],[man(x), walk(x)]) + ([y],[woman(y), stop(y)]))

```
drs3.pretty_format()
  _________     __________
 | x       |   | y        |
(|---------| + |----------|)
 | man(x)  |   | woman(y) |
 | walk(x) |   | stop(y)  |
 |_________|   |__________|
```

```
drs3.fol()
(exists x.(man(x) & walk(x)) & exists y.(woman(y) & stop(y)))
```

Natural Language Processing

What is Natural Language Processing?

NLP Definition

Natural Language Processing (NLP) is the study of computer manipulation of natural (human) language.

Modern application scenarios:

- Search engines (Google, Bing, etc.)
- Machine translation (Google translate, Bing Translator, etc.)
- Question-answering (IBM Watson)
- Virtual assistants (Siri, Alexa, Cortana)
- Sentiment and emotion analysis
- News digest services (Yahoo!)
- Text summarization and generation
- ...

Why NLP is hard?

- Techniques that work for one text might not work for another
- Methods that work for one language might not work for another
- Ambiguity of natural language (e.g. “He fed her cat food.”)
- Referencing problem (anaphora resolution)
- Determining compositionality (which parts belong together in a sentence? What are their dependencies?)
- Complex sentences are difficult to parse
- Humor and sarcasm
- Semantics vs. pragmatics (A: “Could you tell me how to get to Albertplatz?”, B: “Yes, I could.”)

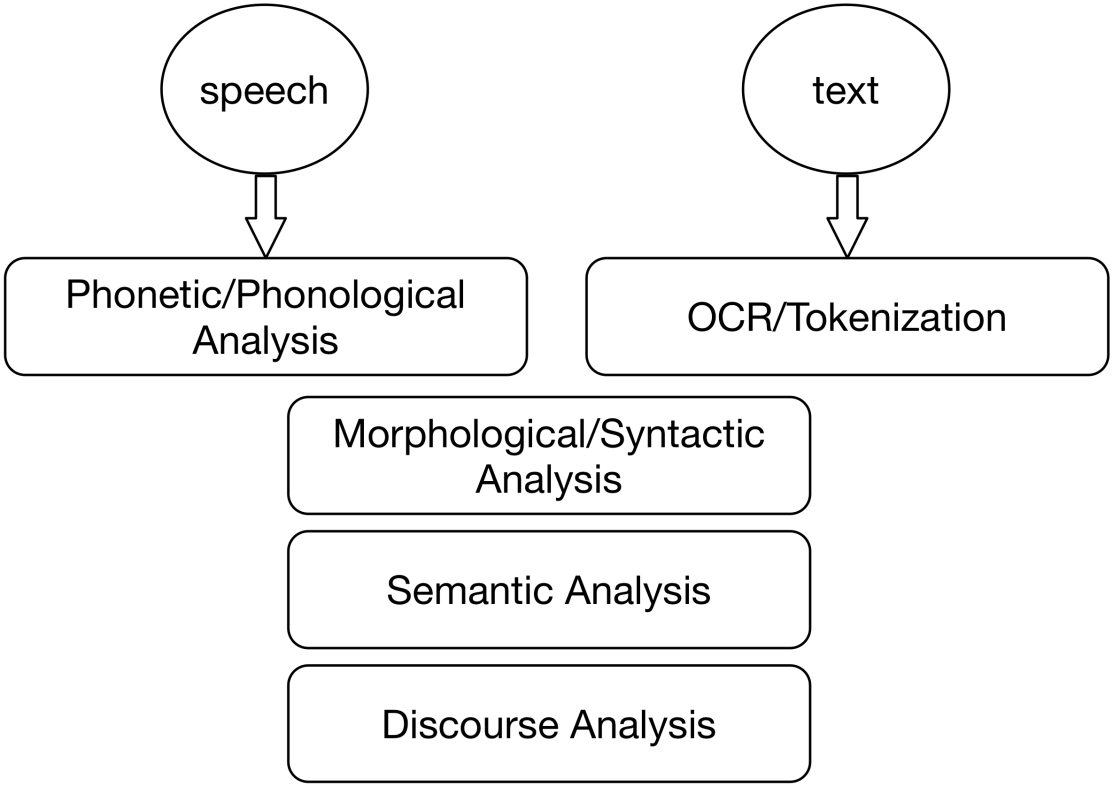
Why NLP is hard?

The famous Winograd Schema Challenges showcases the necessity to combine linguistic and common-sense/world knowledge to really understand the semantics of natural language.

"The trophy doesn't fit into the brown suitcase because it's too [small/large]."

Answer: small = suitcase, large = trophy.
Requires:
A) Anaphora resolution (resolution of "it" to the correct object depending on the adjective)
B) The knowledge that the smaller object can fit into the larger but not vice versa.
C) The knowledge that a suitcase cannot fit into a trophy.

NLP levels


speech
text
Phonetic/Phonological Analysis
OCR/Tokenization
Morphological/Syntactic Analysis
Semantic Analysis
Discourse Analysis

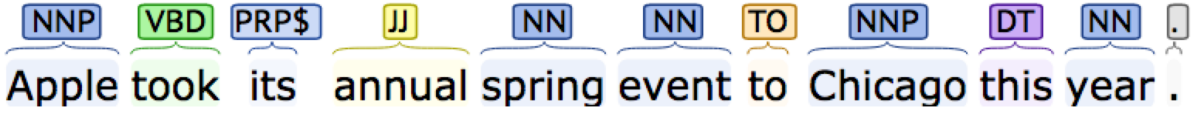
Basic NLP pipeline - Syntactic Analysis

Input: Apple took its annual spring event to Chicago this year.

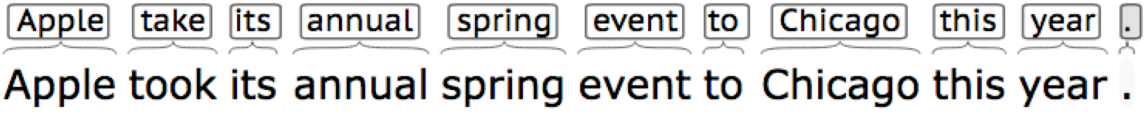
Tokenization

Apple / took / its / annual / spring / event / to / Chicago / this / year

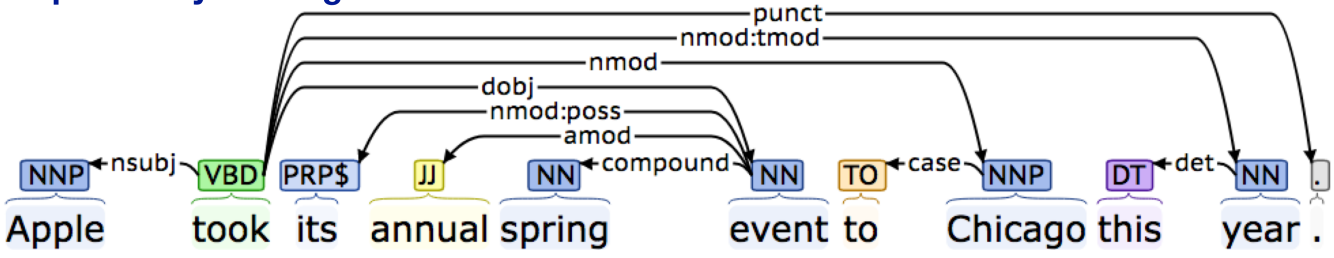
Part-of-Speech Tagging



Lemmatization



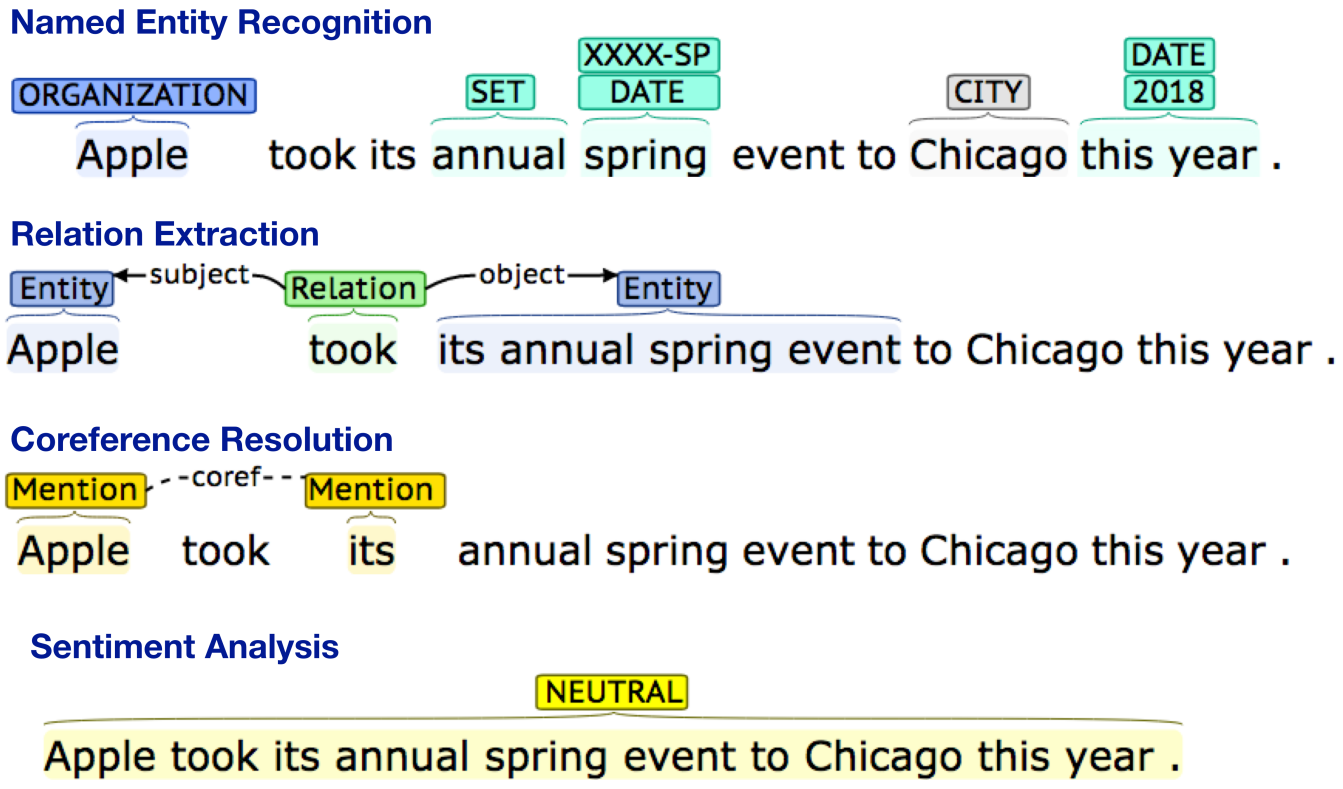
Dependency Parsing



Examples generated with the Stanford Core NLP toolset (`http://corenlp.run/`).

Basic NLP pipeline - Semantic Analysis

Input: Apple took its annual spring event to Chicago this year.



Examples generated with the Stanford Core NLP toolset (`http://corenlp.run/`).

Preprocessing

It is common practice in NLP applications to first preprocess sequences before submitting them to the pipeline.

- remove punctuation
- remove numbers
- handle abbreviations (remove/resolve)
- task-specific preprocessing (e.g. url and user replacing in tweets)

How? Frequently using regular expressions, see Jurafsky Chapter 2.

Jurafsky, D. and Martin, J.H. (2009). Speech and Language Processing: An Introduction to Natural Language Processing, Speech Recognition, and Computational Linguistics. 2nd edition. Prentice-Hall

Tokenization

Tokenization is the process of separating character sequences into smaller pieces, called tokens. In this process certain characters might be omitted, such as punctuation (dependening on the tokenizer).

```
import nltk
sentence = "Apple took its annual spring event to Chicago this year."
tokens = nltk.word_tokenize(sentence)
```

['Apple', 'took', 'its', 'annual', 'spring', 'event', 'to', 'Chicago', 'this', 'year', '.']

```
type(tokens)
len(tokens)
tokens[:3]
```

<class 'list'>
11
['Apple', 'took', 'its']

Part-of-Speech (POS) Tagging

Part-of-speech tagging classifies words into their part-of-speech and labels them according to a specified tagset. Most commonly the Penn Treebank tagset (link) is used.

```
import nltk
sentence = "Apple took its annual spring event to Chicago this year."
nltk.pos_tag(nltk.word_tokenize(sentence))
```

[('Apple', 'NNP'), ('took', 'VBD'), ('its', 'PRP$'), ('annual', 'JJ'), ('spring', 'NN'), ('event', 'NN'), ('to', 'TO'),

('Chicago', 'NNP'), ('this', 'DT'), ('year', 'NN')]

```
nltk.help.upenn_tagset('NNP')
nltk.help.upenn_tagset('VBD')
nltk.help.upenn_tagset('PRP$')
```

NNP: noun, proper, singular
Motown Venneboerger Czestochwa Ranzer Conchita Trumplane Christos, ...
VBD: verb, past tense
dipped pleaded swiped regummed soaked tidied convened halted registered, ...
PRP$: pronoun, possessive
her his mine my our ours their thy your

Lemmatization

- groups words together that have different inflections so that they can be treated as the same item
- reduces a word to its baseform using a online lexicon
- has to be differentiated from stemming which strips the word of its suffixes and prefixes

```
from nltk.stem import WordNetLemmatizer
sentence = "Lemmatization can help identifying different
                                inflections of identical words."
for token in word_tokenize(sentence):
   lemmatizer.lemmatize(token) #(1)
   lemmatizer.lemmatize(token, pos='v') #(2)
```

(1) Lemmatization can help identifying different inflection of identical word.
(2) Lemmatization can help identify different inflections of identical word.

Lemmatization vs. Stemming

```
from nltk.stem import WordNetLemmatizer
from nltk.stem import PorterStemmer

lemmatizer = WordNetLemmatizer()
ps = PorterStemmer()

example_words = ["houses", "organizations", "meanings", "meanness", "mice"]

for word in example_words:
    lemmatizer.lemmatize(word)
    ps.stem(word)
```

Lemmatizer	**Stemmer**
house	hous
organization	organ
meaning	mean
meanness	mean
mouse	mice

Syntactic Similarity Measure

Much of NLP is about finding similarity between words. The so-called edit distance tries to quantify the syntactic distance between two words. For instance, the Levenshtein distance measures the number of operations needed to align two strings (turn one into the other).

Levenshtein edit distance

```
I N T E * N T I O N
| | | | | | | | | |
* E X E C U T I O N
d s s   i s
```

delete, substitute, substitue, -, insert, substitute

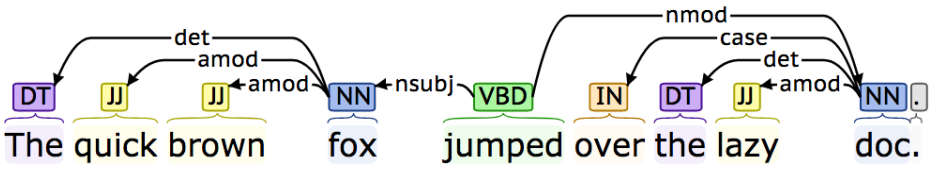
Dependency Parsing

Instead of constituency relations and phrase-structure rules (as in lecture 1), dependency parsing relies on direct binary grammatical relations among words.

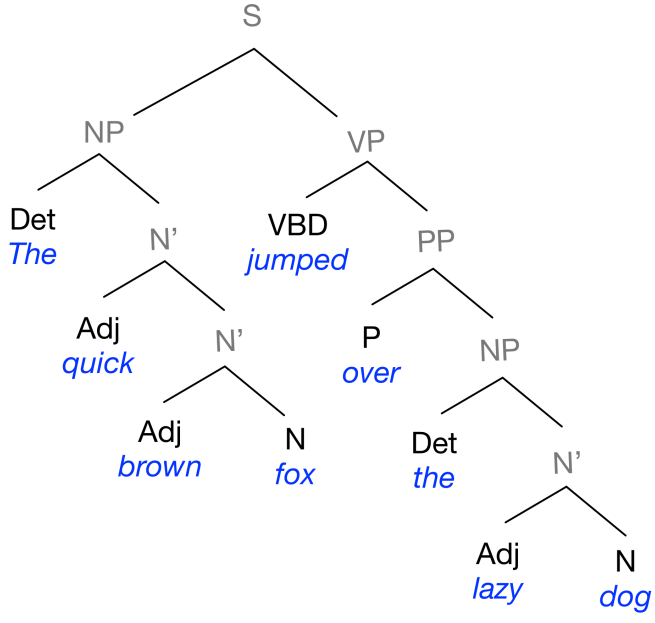
- draws relations from a fixed inventory of grammatical relations
- can handle morphologically rich languages with free word order (abstracts away from direct word order)
- provides an approximation to semantic relations between arguments
- provide simple directed dependency tree

Dependency vs. Constituency Parsing

Dependency Relations



Phrase Structure Tree



Important Dependency Relations

Some selected dependency relations from the Universal Dependency set[1].

Clausal Arugment Relations	**Description**
NSUBJ	Nominal subject
DOBJ	Direct object
IOBJ	Indirect object
Nominal Modifier Relations	**Description**
NMOD	Nominal modifier
AMOD	Adjectival modifier
DET	Determiner
CASE	Preposition, postpositions, other markers
Other Relations	**Description**
CONJ	Conjunct
CC	Coordinating conjunction

[1] De Marneffe, M. C., Dozat, T., Silveira, N., Haverinen, K., Ginter, F., Nivre, J.,& Manning, C. D. (2014). Universal Stanford dependencies: A cross-linguistic typology. In LREC (Vol. 14, pp. 4585-4592).

Code Example

```
from pycorenlp import StanfordCoreNLP

nlp = StanfordCoreNLP('http://localhost:9000/') #this needs to be adapted

sentence = "The big brown fox jumped over the lazy dog."

output = nlp.annotate(text, properties={
  'annotators': 'tokenize,ssplit,pos,depparse,parse',
  'outputFormat': 'text' # json, html
  })
```

Dependency Parse (enhanced plus plus dependencies):

root(ROOT-0, jumped-5)
det(fox-4, The-1)
amod(fox-4, big-2)
amod(fox-4, brown-3)
nsubj(jumped-5, fox-4)
case(dog-9, over-6)
det(dog-9, the-7)
amod(dog-9, lazy-8)
nmod:over(jumped-5, dog-9)

Review of Lecture 2

- What is the difference between a natural and a formal language?
- How do linguists define formal semantics?
- How can we define the principle of compositionality and how does it relate to formal semantics?
- What is the difference between Montague grammar and Discourse Representation Theory?
- What is NLP? What are the main NLP levels?
- What is the difference between phrase structure trees and dependency parsing results?
- Would you rather use lemmatization or stemming for frequency-based analysis of text? Why?