

Notes

19 February 2024 07:05

Lecture Outline:

Lecture Plan

- Syntactic Structure and Dependency parsing
- Syntactic Structure: Consistency and Dependency (25 mins)
- Dependency Grammar and Treebanks (15 mins)
- Transition-based dependency parsing (15 mins)
- Neural dependency parsing (20 mins)

Reminders/comments:
In Assignment 3, out on Tuesday, you build a neural dependency parser using PyTorch
Start installing and learning PyTorch (Ass 3 has scaffolding)
Come to the PyTorch tutorial, Friday 10am (under the Zoom tab, not a Webinar)
Final project discussions – come meet with us; focus of Thursday class in week 4

Stanford

Structure of the Language:

- Words: basic component of the language. Can convey limited information by itself.
- Phrases: Combination of words. Adds bit more context to a single word. E.g. the Cuddly cat -> tell you more about a cat than just a single word cat.
- Sentences: Combination of phrases. Convey meaningful information. E.g. the cuddly cat by the door. Combines two different phrases with a preposition 'by'

1. Two views of linguistic structure: Constituency = phrase structure grammar = context-free grammars (CFGs)

Phrase structure organizes words into nested constituents

Starting unit: words

Words combine into phrases

Phrases can combine into bigger phrases

Stanford

Dependency Structure:

A sentence is a combination of different phrases. These phrases are built using several words.
To extract the meaning out of a sentence, we (humans) look at the words, the role they play in a sentence (i.e. Lexicon, Verb, noun, adjective etc) and mainly relationship of the words to other words in the sentence

Dependency Structure: is finding out which word depend other words to find the meaning in the sentence.

- e.g. as shown below – the arrows denote the dependency.
e.g. 1: Look: Where to look? -> in the Crate
e.g. 2: Create: Where is the crate? -> in the Kitchen
e.g. 3: Create: How is the crate? -> Large...

Two views of linguistic structure: Dependency structure

- Dependency structure shows which words depend on (modify, attach to, or are arguments of) which other words.

Stanford

Why to learn this all?

Why do we need sentence structure?

Humans communicate complex ideas by composing words together into bigger units to convey complex meanings
Listeners need to work out what modifies [attaches to] what
A model needs to understand sentence structure in order to be able to interpret language correctly

Stanford

Types of Ambiguities in Dependency Parsing:

Prepositional Phrase attachment:

Cop kills man with knife? Or cop kills a man who had knife?

San Jose cops kill man with knife

Ex-college football player, 23, shot 9 times allegedly charged police at fiancee's home

By Daniel Almaraz and AP Staff
A San Jose police officer shot and killed an unarmed man Saturday night, authorities said. The officer, identified as Phillip Watkins, responded to a 911 call from a woman who was in her home when she heard gunshots. She told officers she was afraid for her life. The officers responded and found a 23-year-old man who was shot multiple times. He was pronounced dead at the scene. The man's mother, Heather Franklin, a 34-year-old woman, told reporters she was at her home when she heard gunshots. She said she witnessed her son being shot and called 911. Franklin said her son had been arrested earlier this month on suspicion of an attempted robbery. She said he had been released on bail and was awaiting trial.

shortly after the incident, police said. At around 11 p.m., police responded to a report of a man in the area. Officers found the man lying on the ground with multiple gunshot wounds. They called for medical help and the man was transported to a hospital. The man died later at the hospital. Police said they are investigating the incident. The man's mother, Heather Franklin, told reporters she was at her home when she heard gunshots. She said she witnessed her son being shot and called 911. Franklin said her son had been arrested earlier this month on suspicion of an attempted robbery. She said he had been released on bail and was awaiting trial.

Stanford

Coordination scope ambiguity:

Shuttle veteran and longtime NASA executive – are these two different persons or single person with two different designations?

Shuttle veteran and longtime NASA executive Fred Gregory appointed to board

Shuttle veteran and longtime NASA executive Fred Gregory appointed to board

Stanford

Adjective Modifier ambiguity:

NSFW: Feel free to find the ambiguity.

MENTORING DAY
Students get first hand job experience

By Gina Stoer
A group of students from various backgrounds and interests gathered at the University of San Francisco on Tuesday, October 24 to participate in the annual Mentoring Day. The event, organized by the University of San Francisco's Office of Career Services, provided students with the opportunity to learn about various career paths and gain valuable experience. The students, who ranged in age from 18 to 22, were paired with mentors from various fields, including business, law, education, and healthcare. The mentors provided guidance and advice on how to succeed in their chosen careers. The event also featured a panel discussion on the importance of networking and building professional relationships. Overall, the event was a success, with many students reporting that they gained valuable insights into their future career goals and how to achieve them.

Stanford

Formal definition of Dependency Grammer:

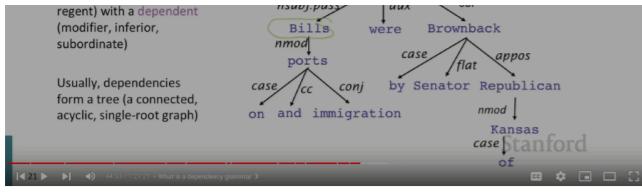
Dependency Grammar and Dependency Structure

Dependency syntax postulates that syntactic structure consists of relations between lexical items, normally binary asymmetric relations ("arrows") called dependencies

An arrow connects a head (governor, superior) to a dependent (dependent, inferior).

submitted

Stanford



Dependency Tree Banks:

Annotating the sentences with dependencies in between the different words.

The rise of annotated data

Starting off, building a treebank seems a lot slower and less useful than writing a grammar (by hand)

But a treebank gives us many things

- Reusability of the labor
- Many parsers, part-of-speech taggers, etc. can be built on it
- Valuable resource for linguistics
- Broad coverage, not just a few intuitions
- Frequencies and distributional information
- A way to evaluate NLP systems

Stanford

How to build a Parser once we have dependency trees?

A rule based system:

Dependency should be:
 Plausible (should make sense...)
 Close to the root word
 spans from a ROOT word to a verb or a punctuation
 Have limited words to either side of the ROOT.

Dependency Conditioning Preferences

What are the sources of information for dependency parsing?

1. Bilexical affinities
2. Dependency distance
3. Intervening material
4. Valency of heads

Most dependencies are between nearby words
 Dependencies rarely span intervening verbs or punctuation
 How many dependents on which side are usual for a head?

ROOT Discussion of the outstanding issues was completed . Stanford

Steps:

Choose each word and find what other word it is a dependent of?

and some constraints:

Arrows should be acyclic.
 Arrows should/should not cross other arrows (projectivity)

Dependency Parsing

A sentence is parsed by choosing for each word what other word (including ROOT) it is a dependent of

- Usually some constraints:
 - Only one word is a dependent of ROOT
 - Don't want cycles $A \rightarrow B, B \rightarrow A$
 - This makes the dependencies a tree
- Final issue is whether arrows can cross (be non-projective) or not

ROOT I 'll give a talk tomorrow on neural networks Stanford

Definition of Projectivity:

Projectivity

Definition of a projective parse: There are no crossing dependency arcs when the words are laid out in their linear order, with all arcs above the words

- Dependencies corresponding to a CFG tree must be **projective**
 - i.e., by forming dependencies by taking 1 child of each category as head
- Most syntactic structure is projective like this, but dependency theory normally does allow non-projective structures to account for displaced constituents
- You can't easily get the semantics of certain constructions right without these nonprojective dependencies

Who did Bill buy the coffee from yesterday ? Stanford

3 methods of building Dependency parser:

- Dynamic programming
- Graph algorithms
- Constraint Satisfaction (rule based)
- Transition-based parsing (discussed in detail)

Transition-based Parsing:

Greedy transition-based parsing [Nivre 2003]

A simple form of greedy discriminative dependency parser

- The parser does a sequence of bottom-up actions
- Roughly like "shift" or "reduce" in a shift-reduce parser, but the "reduce" actions are specialized to create dependencies with head on left or right
- The parser has:
 - a stack σ , written with top to the right
 - which starts with the ROOT symbol
 - a buffer β , written with top to the left
 - which starts with the input sentence
 - a set of dependency arcs A
 - which starts off empty
 - a set of actions

Stanford

Algorithm:

Start with a Stack containing root.

Use a buffer β to work (a buffer is a collection of words)

At any point, 3 actions are allowed:

Shift: move next word to the Stack. When only one word on the stack, you always shift.

(Reduce actions) - take top 2 items on the stack and make one of them dependent of the other. End result is one item from stack is removed and a dependency is created

Left-arc: action making left word dependent

Right-arc: action of making right word dependent

Basic transition-based dependency parser

Start: $\sigma = [\text{ROOT}]$, $\beta = w_1, \dots, w_n$, $A = \emptyset$

1. Shift $\sigma, w_i | \beta, A \rightarrow \sigma | w_i, \beta, A$
2. Left-Arc $\sigma | w_j | w_i, \beta, A \rightarrow \sigma | w_j, \beta, A \cup \{w_j, w_i\}$
3. Right-Arc $\sigma | w_i | w_j, \beta, A \rightarrow \sigma | w_i, \beta, A \cup \{w_j, w_i\}$

Finish: $\sigma = [w]$, $\beta = \emptyset$

Stanford

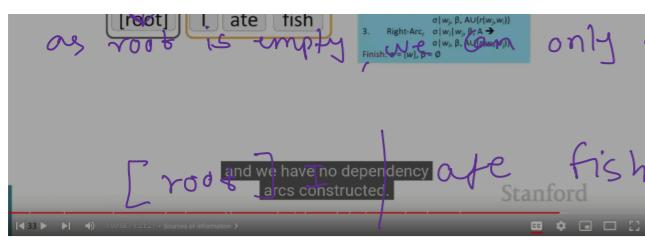
Example:

Arc-standard transition-based parser

(there are other transition schemes ...)

Analysis of "I ate fish"

Start



as [root] is empty, only shift operation

→ here for some reason we don't establish dependency bet'n root & ate.

I.

shifting again,

[root] I ate | fish. → ate → I (left arc)

↓

[root] ate [fish] → once again, we don't establish dependency bet'n root & ate. why?

shift again

[root] ate fish } [empty buffer] → ate ⇒ fish (right arc reduce)

↓

[root] ate } [empty buffer] → root ⇒ ate (right arc)

How to choose operation when Stack has more than one element?

MaltParser [Nivre and Hall 2005]

- We have left to explain how we choose the next action
 - Answer: Stand back, I know machine learning!
- Each action is predicted by a discriminative classifier (e.g., softmax classifier) over each legal move
 - Max of 3 untyped choices; max of $|R| \times 2 + 1$ when typed
 - Features: top of stack word, POS, first in buffer word, POS; etc.
- There is NO search (in the simplest form)
 - But you can profitably do a beam search if you wish (slower but better): You keep k good parse prefixes at each time step
- The model's accuracy is fractionally below the state of the art in dependency parsing, but
- It provides very fast linear time parsing, with high accuracy – great for parsing the web