Recap: We know how to perform POS tagging with structured perceptron

• An example of sequence labeling tasks
• Requires a predefined set of POS tags
  • Penn Treebank commonly used for English
  • Encodes some distinctions and not others
• Given annotated examples, we can address sequence labeling with multiclass perceptron
  • but computing the argmax naively is expensive
  • constraints on the feature definition make efficient algorithms possible
  • E.g, Viterbi algorithm
Sequence labeling tasks

Beyond POS tagging
Many NLP tasks can be framed as sequence labeling

• Information Extraction: detecting named entities
  • E.g., names of people, organizations, locations

“Brendan Iribe, a co-founder of Oculus VR and a prominent University of Maryland donor, is leaving Facebook four years after it purchased his company.”

Many NLP tasks can be framed as sequence labeling

\[ x = [\text{Brendan, Iribe, ",", a, co-founder, of, Oculus, VR, and, a, prominent, University, of, Maryland, donor, ",", is, leaving, Facebook, four, years, after, it, purchased, his, company, "."] \]


“BIO” labeling scheme for named entity recognition
Many NLP tasks can be framed as sequence labeling

- The same kind of BIO scheme can be used to tag other spans of text
  - Syntactic analysis: detecting noun phrase and verb phrases
  - Semantic roles: detecting semantic roles (who did what to whom)
Many NLP tasks can be framed as sequence labeling

• Other sequence labeling tasks
  • Language identification in code-switched text
    “Ulikuwa ukiongea a lot of nonsense.” (Swahili/English)
  • Metaphor detection
    “he **swam** in a **sea** of diamonds”
    “authority is a **chair**, it needs **legs** to **stand**”
    “in Washington, people change **dance partners** frequently, but not the **dance**”
• ...
Other algorithms for solving the argmax problem
Structured perceptron can be used for other structures than sequences

• The Viterbi algorithm we’ve seen is specific to sequences
  • Other argmax algorithms necessary for other structures (e.g. trees)

• Integer Linear Programming provides a general framework for solving the argmax problem
Argmax problem as an Integer Linear Program

• An integer linear program (ILP) is an optimization problem of the form

\[
\max_{z} \quad a \cdot z \quad \text{subj. to} \quad \text{linear constraints on } z
\]

• For a fixed vector \( a \)
• Example of integer constraint: \( z_3 \in \{0, 1\} \)

• Well-engineered solvers exist
  • e.g., Gurobi
  • Useful for prototyping
  • But general not as efficient as dynamic programming
Casting sequence labeling with Markov features as an ILP

• Step 1: Define variables $z$ as binary indicator variables which encode an output sequence $y$

\[ z_{l,k',k} = 1[\text{label } l \text{ is } k \text{ and label } l - 1 \text{ is } k'] \]

• Step 2: Construct the linear objective function

\[ a_{l,k',k} = \mathbf{w} \cdot \phi_l(x, \langle \ldots, k', k \rangle) \]
Casting sequence labeling with Markov features as an ILP

• Step 3: Define constraints to ensure a well-formed solution
  • Z’s should be binary: for all l, k’, k
    $z_{l,k',k} \in \{0, 1\}$
  • For a given position l, there is exactly one active z
    $\sum_k \sum_{k'} z_{l,k',k} = 1$ for all l
  • The z’s are internally consistent
    $\sum_{k'} z_{l,k',k} = \sum_{k''} z_{l+1,k,k''}$ for all l, k
What you should know

• POS tagging as an example of sequence labeling task
• Requires a predefined set of POS tags
  • Penn Treebank commonly used for English
  • Encodes some distinctions and not others
• How to train and predict with the structured perceptron
  • constraints on feature structure make efficient algorithms possible
  • Unary and markov features => Viterbi algorithm
• Extensions:
  • How to frame other problems as sequence labeling tasks
  • Viterbi is not the only way to solve the argmax: Integer Linear Programming is a more general solution
Syntax, Grammars & Parsing

CMSC 470
Marine Carpuat

Fig credits: Joakim Nivre, Dan Jurafsky & James Martin
This is a simple sentence. But it is an instructive one.
Syntax & Grammar

• **Syntax**
  - From Greek syntaxis, meaning “setting out together”
  - refers to the way words are arranged together.

• **Grammar**
  - Set of structural rules governing composition of clauses, phrases, and words in any given natural language
  - Descriptive, not prescriptive
  - Panini’s grammar of Sanskrit ~2000 years ago
Syntax and Grammar

• Goal of syntactic theory
  • “explain how people combine words to form sentences and how children attain knowledge of sentence structure”

• Grammar
  • implicit knowledge of a native speaker
  • acquired without explicit instruction
  • minimally able to generate all and only the possible sentences of the language

[Philips, 2003]
Two views of syntactic structure

• Constituency (phrase structure)
  • Phrase structure organizes words in nested constituents

• Dependency structure
  • Shows which words depend on (modify or are arguments of) which on other words
Constituency

• Basic idea: groups of words act as a single unit

• Constituents form coherent classes that behave similarly
  • With respect to their internal structure: e.g., at the core of a noun phrase is a noun
  • With respect to other constituents: e.g., noun phrases generally occur before verbs
Constituency: Example

• The following are all noun phrases in English...

<table>
<thead>
<tr>
<th>Harry the Horse</th>
<th>a high-class spot such as Mindy’s</th>
</tr>
</thead>
<tbody>
<tr>
<td>the Broadway coppers</td>
<td>the reason he comes into the Hot Box</td>
</tr>
<tr>
<td>they</td>
<td>three parties from Brooklyn</td>
</tr>
</tbody>
</table>

• Why?
  • They can all precede verbs
  • They can all be preposed/postposed
  • ...
Grammars and Constituency

• For a particular language:
  • What are the “right” set of constituents?
  • What rules govern how they combine?

• Answer: not obvious and difficult
  • There are many different theories of grammar and competing analyses of the same data!
# An Example Context-Free Grammar

<table>
<thead>
<tr>
<th>Grammar Rules</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S \rightarrow NP\ VP$</td>
<td>I + want a morning flight</td>
</tr>
<tr>
<td>$NP \rightarrow$</td>
<td>I</td>
</tr>
<tr>
<td>Proper-Noun</td>
<td>Los Angeles</td>
</tr>
<tr>
<td>Det Nominal</td>
<td>a + flight</td>
</tr>
<tr>
<td>Nominal $\rightarrow$</td>
<td>morning + flight</td>
</tr>
<tr>
<td>Nominal Noun</td>
<td>flights</td>
</tr>
<tr>
<td>$VP \rightarrow$</td>
<td>do</td>
</tr>
<tr>
<td>Verb</td>
<td></td>
</tr>
<tr>
<td>Verb NP</td>
<td>want + a flight</td>
</tr>
<tr>
<td>Verb NP PP</td>
<td>leave + Boston + in the morning</td>
</tr>
<tr>
<td>Verb PP</td>
<td>leaving + on Thursday</td>
</tr>
<tr>
<td>$PP \rightarrow$</td>
<td>from + Los Angeles</td>
</tr>
</tbody>
</table>
Parse Tree: Example

```
S
   / \    
NP   VP   
   / \  /  
Pro Verb NP  
   / \     /  
I prefer Det Nom  
   /   \    /  
a Det Nom Noun  
   /   \    /  
Noun Nom flight  
   /   \    /  
Noun morning
```

Note: equivalence between parse trees and bracket notation.
Dependency Grammars

• Context-Free Grammars focus on constituents
  • Non-terminals don’t actually appear in the sentence

• In dependency grammar, a parse is a graph (usually a tree) where:
  • Nodes represent words
  • Edges represent dependency relations between words
    (typed or untyped, directed or undirected)
Example Dependency Parse

They hid the letter on the shelf

Compare with constituent parse…

What's the relation?
Dependency Grammars

- Syntactic structure = lexical items linked by binary asymmetrical relations called dependencies
Example Dependency Parse

Dependencies (usually) form a tree:
- Connected
- Acyclic
- Single-head

They hid the letter on the shelf

Compare with constituent parse…

What's the relation?
## Dependency Relations

<table>
<thead>
<tr>
<th>Argument Dependencies</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>nsubj</td>
<td>nominal subject</td>
</tr>
<tr>
<td>csubj</td>
<td>clausal subject</td>
</tr>
<tr>
<td>dobj</td>
<td>direct object</td>
</tr>
<tr>
<td>iobj</td>
<td>indirect object</td>
</tr>
<tr>
<td>pobj</td>
<td>object of preposition</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Modifier Dependencies</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tmod</td>
<td>temporal modifier</td>
</tr>
<tr>
<td>appos</td>
<td>appositional modifier</td>
</tr>
<tr>
<td>det</td>
<td>determiner</td>
</tr>
<tr>
<td>prep</td>
<td>prepositional modifier</td>
</tr>
<tr>
<td>Relation</td>
<td>Examples with head and dependent</td>
</tr>
<tr>
<td>------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>NSUBJ</td>
<td>United canceled the flight.</td>
</tr>
<tr>
<td>DOBJ</td>
<td>United diverted the flight to Reno.</td>
</tr>
<tr>
<td></td>
<td>We booked her the first flight to Miami.</td>
</tr>
<tr>
<td>IOBJ</td>
<td>We booked her the flight to Miami.</td>
</tr>
<tr>
<td>NMOD</td>
<td>We took the morning flight.</td>
</tr>
<tr>
<td>AMMOD</td>
<td>Book the cheapest flight.</td>
</tr>
<tr>
<td>NUMMOD</td>
<td>Before the storm JetBlue canceled 1000 flights.</td>
</tr>
<tr>
<td>APPOS</td>
<td>United, a unit of UAL, matched the fares.</td>
</tr>
<tr>
<td>DET</td>
<td>The flight was canceled.</td>
</tr>
<tr>
<td></td>
<td>Which flight was delayed?</td>
</tr>
<tr>
<td>CONJ</td>
<td>We flew to Denver and drove to Steamboat.</td>
</tr>
<tr>
<td>CC</td>
<td>We flew to Denver and drove to Steamboat.</td>
</tr>
<tr>
<td>CASE</td>
<td>Book the flight through Houston.</td>
</tr>
</tbody>
</table>

**Figure 14.3** Examples of core Universal Dependency relations.
Universal Dependencies project

- Set of dependency relations that are
  - Linguistically motivated
  - Computationally useful
  - Cross-linguistically applicable

[Nivre et al. 2016]

universaldependencies.org
Universal Dependencies Illustrated
Parallel examples for English, Bulgarian, Czech & Swedish
What you should know

• Syntax vs. Grammar
• Two views of syntactic structures
  • Context-Free Grammar vs. Dependency grammars
  • Can be used to capture various facts about the structure of language (but not all!)
• Dependency grammars
  • Definition of dependency links: head, dependent
  • Annotate an example given a set of dependency types
• How syntactic analysis can be used to define NLP tasks or features

• Next: how can we predict syntactic parses?