Graph-Based Dependency Parsing

Dependency Parsing

- Dynamic programming:
  - Similar to lexicalized PCFG (which we will discuss)
  - Eisner's (1996) algorithm gives an improved run-time
- Transition-based algorithms:
  - Greedy
  - Left-to-right traversal of the text, each choice is done with a classifier
- Graph algorithms:
  - Structured prediction

Not covered here

Next Class

Covered Today

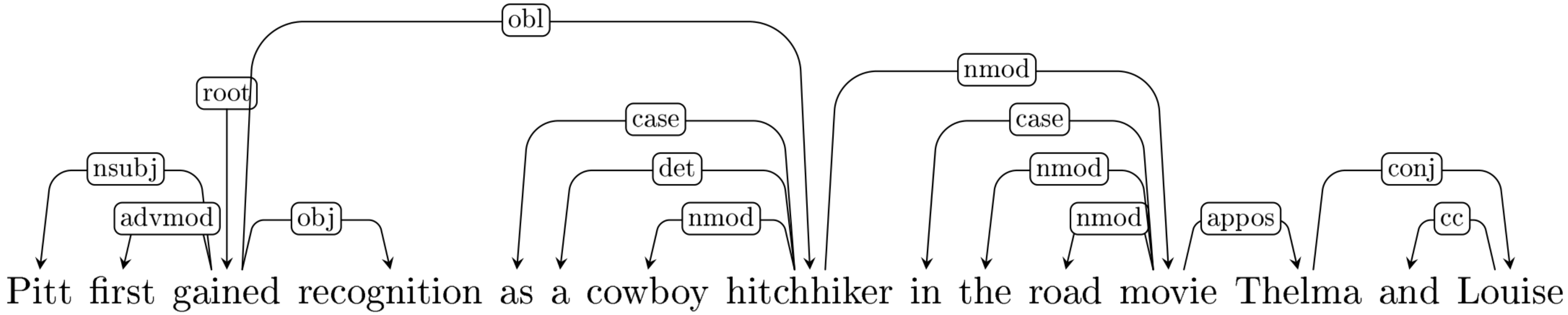
Jason Eisner. *Three new probabilistic models for dependency parsing: An exploration*. In COLING, 1996.

Dependency Parsing

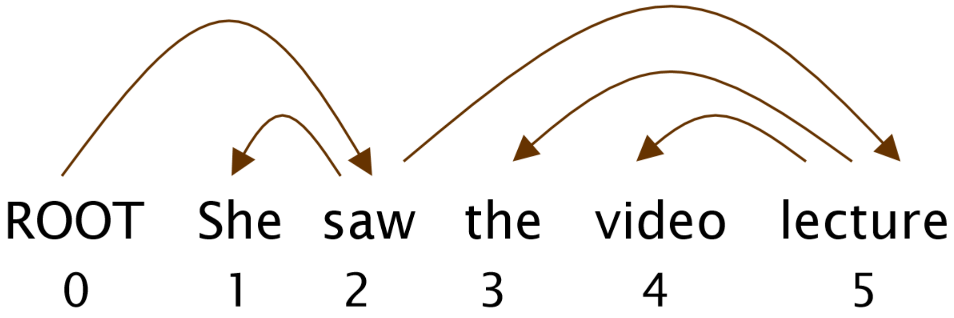
- What are the sources of information for dependency parsing?
  - Bi-lexical affinities

    [gained → Pitt] is plausible, [Pitt → gained] is not
  - Dependency distance

    mostly with nearby words
  - Intervening material: dependencies rarely span intervening verbs or punctuation
  - Valency – how many dependents on which side are usual for a head?



Dependency Parsing Evaluation



$$ACC = \frac{\#CORRECT\ EDGES}{\#NUMBER\ OF\ WORDS}$$

- Accuracy (aka Attachment Score)
- There is Unlabeled Attachment Score and Labeled Attachment Score

Gold

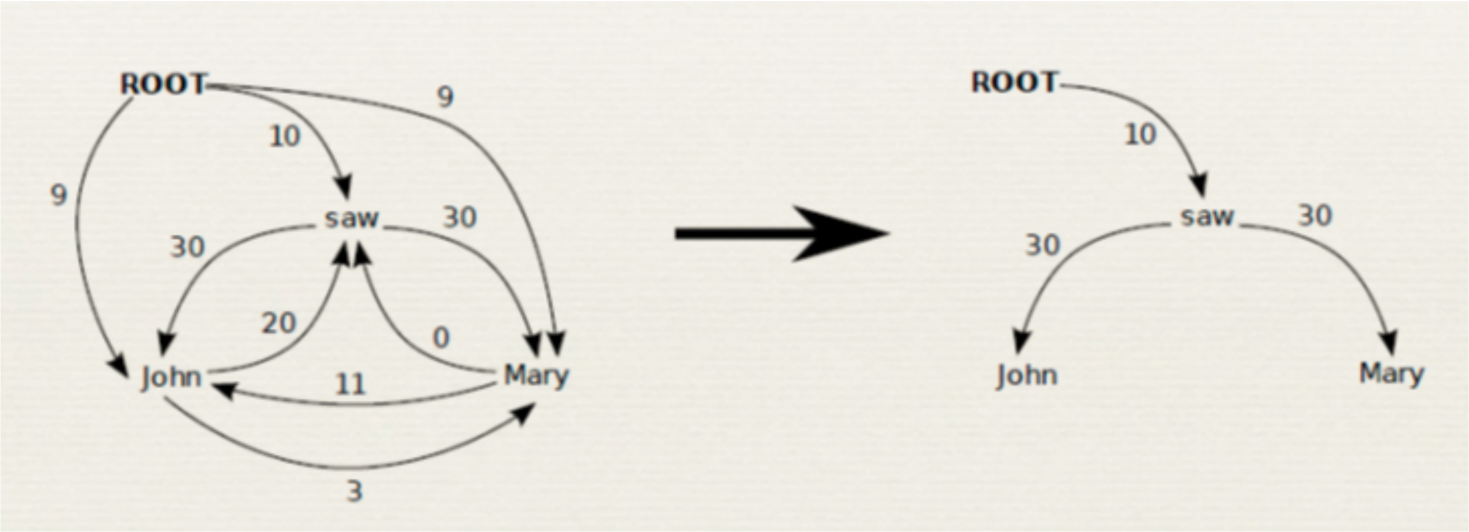
1	2	She	nsubj
2	0	saw	root
3	5	the	det
4	5	video	nn
5	2	lecture	dobj

Parsed

1	2	She	nsubj
2	0	saw	root
3	4	the	det
4	5	video	nsubj
5	2	lecture	ccomp
index	head index	word	edge label

Graph-based Parsing

- Graph-based parsing addresses it as a structured prediction problem
- MST Parser:
    1. Score the arcs independently, based on how likely they are to appear in a parse
    2. Find the maximum directed spanning tree over the resulting weighted graph



Online Large-Margin Training of Dependency Parsers
R. McDonald, K. Crammer, and F. Pereira, *ACL 2005*

MST Parser

Define a scoring function over all possible directed trees over $V = \{w_1, ..., w_n, ROOT\}$ where $ROOT$ is the root of the tree. Let $\Phi : V^2 \times L \times S \rightarrow \{0,1\}^d$, where L is the label set and S is the set of sentences (feature values can also be real numbers if needed), be a feature function over possible edges.

Let θ be the weight vector (the parameters of the model):

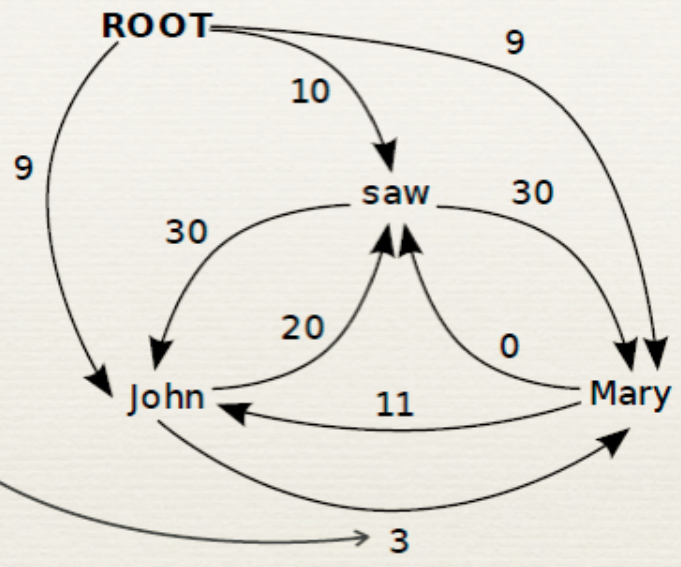
$$score_\theta(v_1, v_2, l \,|x_{1:n}) = \theta^t \cdot \Phi(v_1, v_2, l, x_{1:n})$$

For a directed tree T define:

$$score_\theta(T \,|x_{1:n}) = \sum_{(v_1, v_2, l) \in T} score_\theta(v_1, v_2, l \,|x_{1:n})$$

MST Parser

$$\text{score}(\text{John}\rightarrow\text{Mary}) = \mathbf{w}\cdot\boxed{\mathbf{f}(\text{John}\rightarrow\text{Mary})}$$



Binary Features

Head-POS=NOUN
Mod-POS=NOUN
Head-Word=John
Mod-Word=Mary
In-Between=VERB
Distance=2
Direction=Right
etc.

+conjunctions

MST Parser: Inference and Learning

- Note that inference is simply finding the maximum directed spanning tree
  - We can score each edge based on its features and
  - This is done by the Chu-Liu Edmonds algorithm (not necessarily projective)
- It is possible to define this model as log-linear:

$$Pr(T) = \frac{exp(\sum_{(v_1,v_2,l)\in T} \theta^t \cdot \Phi(v_1, v_2, l))}{Z(V, \theta)}$$

- The gradient of the log-likelihood is given by:

$$\frac{\partial LL}{\partial \theta} = \sum_{i=1}^{N} \Big[\sum_{(v_1,v_2,l)\in T_i} \Phi(v_1, v_2, l) - \mathbf{E}_T \Big(\sum_{(v_1,v_2,l)\in T} \Phi(v_1, v_2, l) \Big) \Big]$$

MST Parser: Inference and Learning

$$\frac{\partial LL}{\partial \theta} = \sum_{i=1}^{N} \Big[\sum_{(v_1,v_2,l)\in T_i} \Phi(v_1, v_2, l) - \mathbf{E}_T\big(\sum_{(v_1,v_2,l)\in T} \Phi(v_1, v_2, l)\big)\Big]$$

- It is possible to compute the second term exactly, but the algorithm is not simple
- The Averaged Perceptron algorithm provides a simple and useful alternative, by replacing the expectation with a maximum →

Structured Prediction Models via the Matrix-Tree Theorem
Koo et al., 2007

MST Parser: Inference and Learning

1. $\theta^{(0)} \leftarrow 0$
2. **for** $r = 1 \ldots N_{iterations}$
3. **for** $i = 1 \ldots N$
4. $T' \leftarrow argmax_T \sum_{(v_1,v_2,l)\in T} score_\theta(v_1, v_2, l)$
5. $\theta^{((r-1)N+i)} \leftarrow \theta^{((r-1)N+i-1)} + \eta \cdot \left(\sum_{(v_1,v_2,l)\in T_i} \Phi(v_1, v_2, l) - \sum_{(v_1,v_2,l)\in T'} \Phi(v_1, v_2, l) \right)$
6. **return** $\frac{1}{N \cdot N_{iterations}} \sum_k \theta^{(k)}$

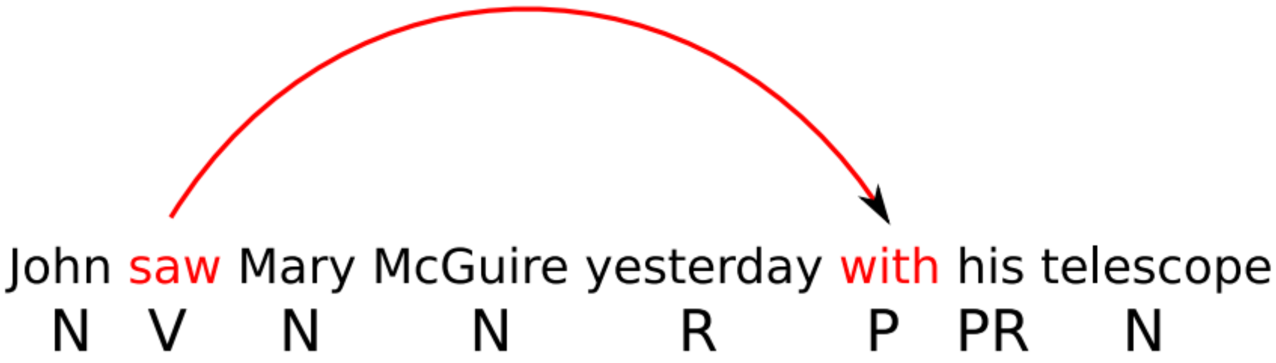
T_i is the gold standard tree

Learning Rate

Feature function of T' instead of expectation over T!

Discriminative Training Methods for Hidden Markov Models: Theory and Experiments with Perceptron Algorithms
Collins, 2002

Features Used in Graph-based Dep Parsing



Features from McDonald et al.

- Identities of the words w_i and w_j and the label l_k

head=saw & dependent=with

Features Used in Graph-based Dep Parsing

Features from McDonald et al.

- Part-of-speech tags of the words w_i and w_j and the label l_k

head-pos=Verb & dependent-pos=Preposition

Features Used in Graph-based Dep Parsing

John saw Mary McGuire yesterday with his telescope

N V N N R P PR N

Features from McDonald et al.

- Part-of-speech of words surrounding and between w_i and w_j

inbetween-pos=Noun
inbetween-pos=Adverb
dependent-pos-right=Pronoun
head-pos-left=Noun
...

Again conjoined with the label (omitted from now on for brevity)

Features Used in Graph-based Dep Parsing

Features from McDonald et al.

- Number of words between w_i and w_j, and their orientation

arc-distance=3

arc-direction=right

Features Used in Graph-based Dep Parsing

L

John saw Mary McGuire yesterday with his telescope

N V N N R P PR N

Label features

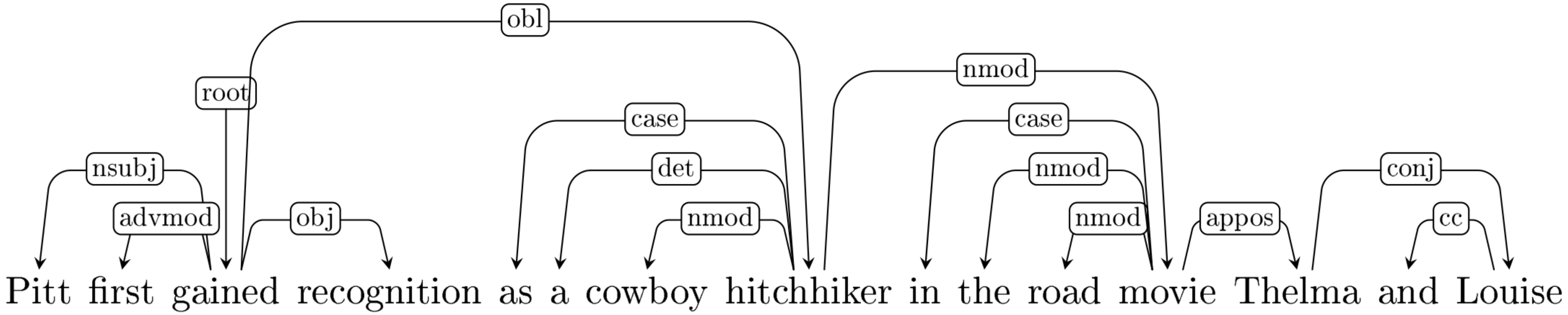
arc-label= **L**

And combinations of all these features...

Graph Based Parsing – Higher Order Features

- MST inference works well when the model factorizes over edges

- However, it has been proven useful to add features that involve several edges, which turns inference into intractable



Higher-order Parsing Models

- Higher-order parsing models don't assume that the score of a tree factorizes over edges, but over sets of edges
- For instance, the "grandchild model" assumes that

$$score(T) = \sum_{\{(i,j,k) \in V^3 \mid (i,j) \in T \ \& \ (j,k) \in T\}} score(i,j,k)$$

$$score(i,j,k) = \theta^t \cdot \Phi(i,j,k,x_{1:n})$$

Higher-order Parsing Models

- The sibling model with labels would look like this:

$$score(T) = \sum_{(i,j,k) \in V^3 | (i,j) \in T \& (i,k) \in T} score(i,j,k)$$

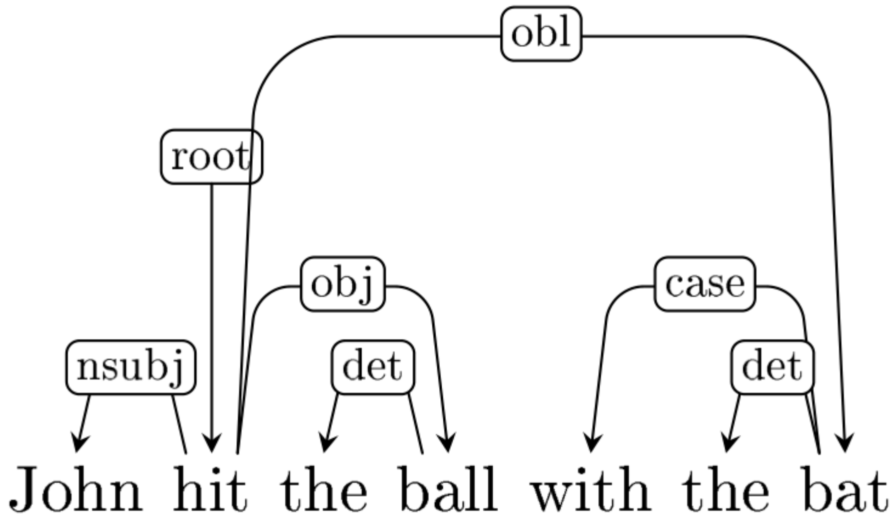
$$score(i,j,k) = \theta^t \cdot \Phi(i,j,k,x_{1:n})$$

- Inference (finding the highest scoring tree given θ) is NP-complete. However, there are approximate algorithms.

Online Learning of Approximate Dependency Parsing Algorithms, McDonald and Pereira, EACL 2006

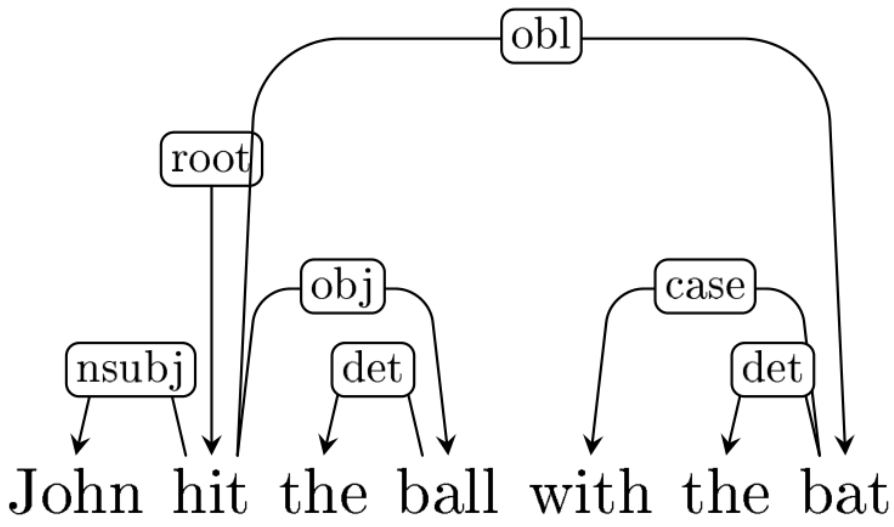
Graph Based Parsing – Higher Order Features

- **Sibling features** are features defined over two edges that have the same parent
- They can be helpful, because once you know “hit → ball” is an edge, “hit → bat” becomes more likely (over “ball → bat”)



Graph Based Parsing – Higher Order Features

- **Grandparent features** are features defined over two edges that form a chain
- They can be helpful, because "hit → bat" becomes much more likely once "bat → with" is included



Subcategorization Frames

- Verbs tend to appear in specific patterns, in terms of their number of arguments, their order and their prepositions
  - Subject gave Object_1 Obect_2
  - Subject gave Obect_2 to Object_1
  - Object_1 was given to Object_2
  - It was Object_2 that was given to Object_1
  - …
- Encoding what sub-categorization frame is associated with each verb can promote subcategorization frames that appeared during training
- This feature is defined over all children of the verb and some of their children (higher than second-order)

Graph-based Parsing – Higher Order Features

- A lot of interesting machine learning research has been done on building learning models for higher-order features
- A few examples:
  - Xavier Carreras, 2007. Experiments with a Higher-Order Projective Dependency Parser. In CoNLL
  - Yuan Zhang, Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2014. Greed is good if randomized: New inference for dependency parsing. In EMNLP
  - Ilan Tchernowitz, Liron Yedidsion and Roi Reichart. 2016. Effective Greedy Inference for Graph-based Non-Projective Dependency Parsing. In EMNLP

Some Results

- The basic MST parser scores about 88% LAS on English (Wall Street Journal articles, in domain setting)
- Recently, using Neural Networks, parsing performance with graph-based has gone up to ~95% (same setting as above)

- Graph-based systems that use higher-order features score a few points higher as well
  - That is, models whose score does not factor over individual edges (node pairs), but also on larger sub-sets of words
  - Still insufficient work on using higher-order features with the new neural machinery

Deep Biaffine Attention for Neural Dependency Parsing
Dozat and Manning, ICLR 2017