CS224n: NLP with Deep Learning

Lecture 5: Dependency Parsing

Syntactic Structure: Consistency and Dependency

Phrase structure

- Phrase structure:
  Sentences are built out of phrases, that can be nested

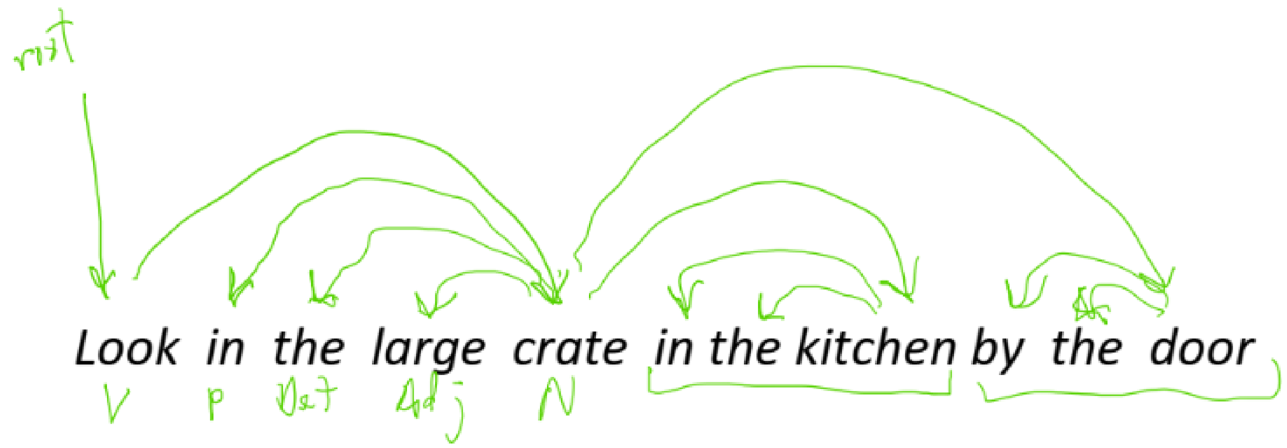
Ex: Name Phrase = Det (Adj) Noun

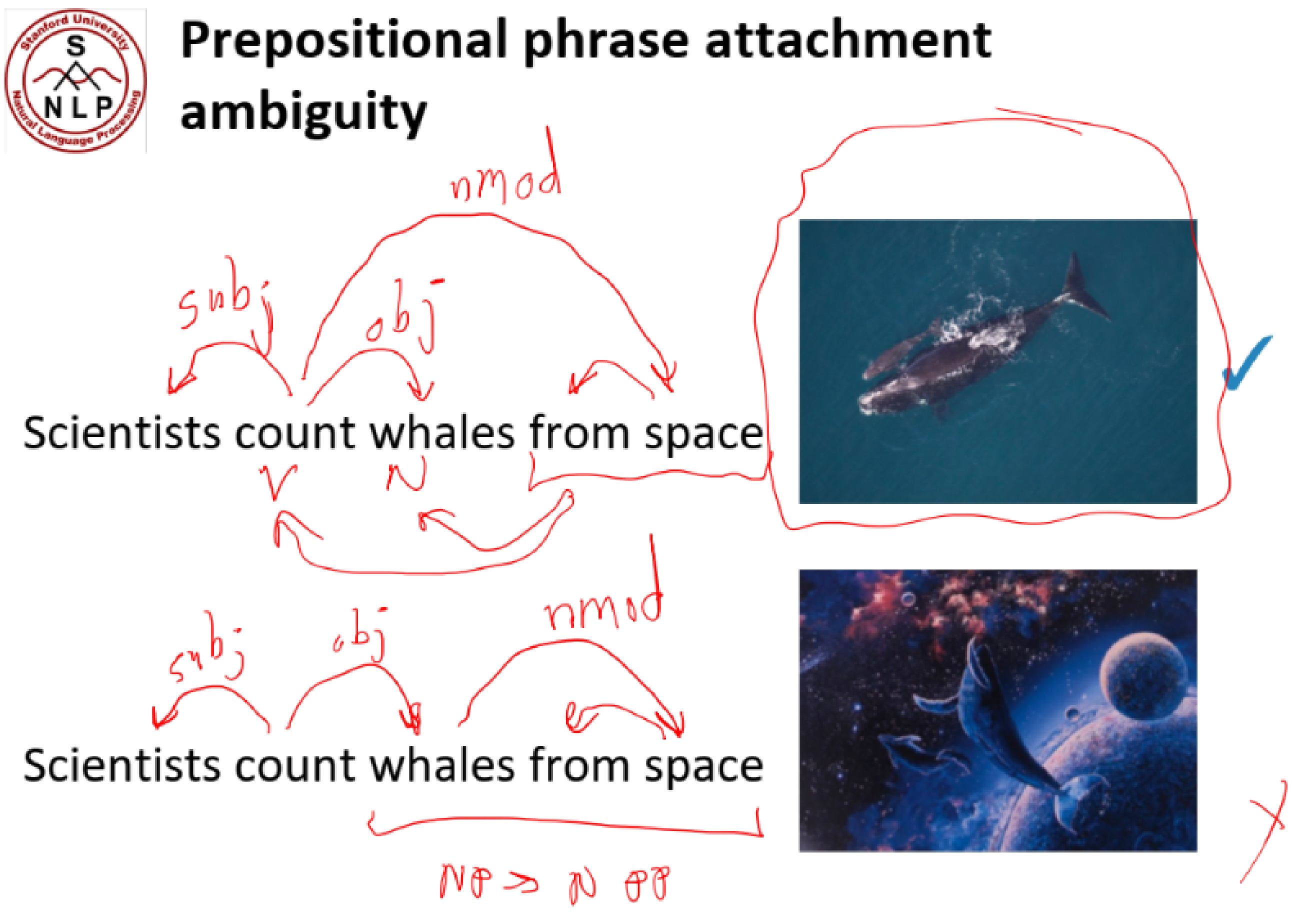
This is the work linguists do

It is the dominant approach to linguistics structure

Dependency Structure

Show which words depend on which other words





The preposition can modify either the verb, or the noun that comes beforehand

A big ambiguity in English language

- We have to understand what it means by ourselves, we don't need to put brackets () around the expression, as we do in python *if* statements

- PP = Prepositional Phrase

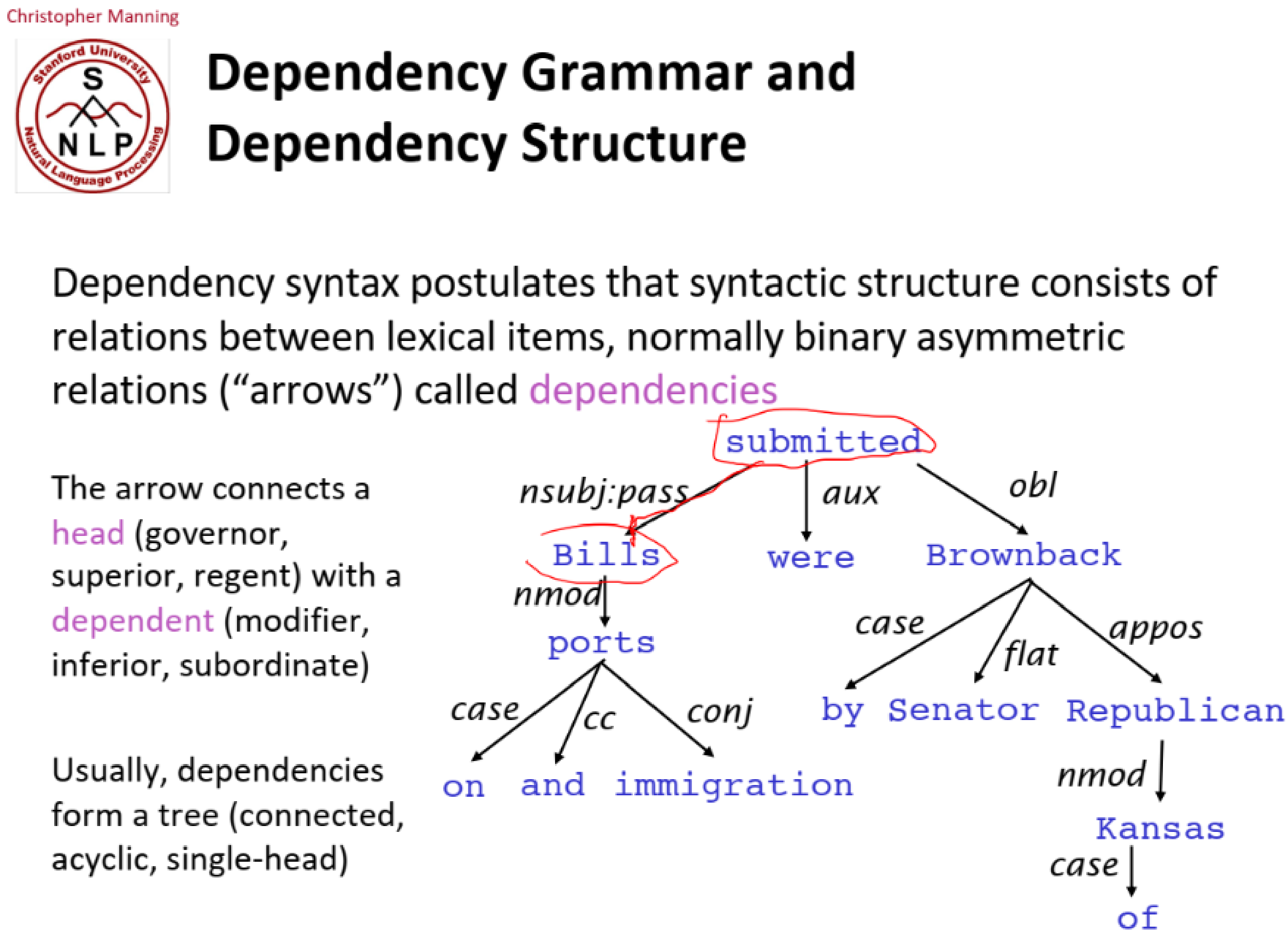
There is an exponential number of possible structures given several Prepositional Phrases
This number is given by the Catalan numbers

Dependency Grammar and Treebanks

Dependency Structure

- Syntactic structure consists of relations between lexical items
- These relations, called **dependencies** are:

- binary
- asymmetric
- Dependencies are represented by arrows



Chomsky: the most well-known linguist

In the rest of the class, we will have the arrows:

- start from the head
- point to the dependent

- Usually, we can add a fake ROOT word, so that every word has a dependent

Treebanks

Goal of Universal Dependencies:
Have a consistent annotation of grammar across all languages

Usages

- Reusable:
  Before, everyone just wrote their own parsers, with their own grammar rules
  No sharing, nor reuse
- Build models that find the right structure, even when there can be multiple interpretations
  This is not possible with only grammar rules

Transition-based dependency parsing

- It is what Google uses when parsing web pages

Example

- We can either shift the word to the stack, or "reduce" it by assigning an arc to a word already in the stack

Left-arc reduction:
Treat the 2nd word on top of the stack as a dependent of the word on top of the stack

Right-arc reduction:
Same, but we put a dependency the other way

We stop when our buffer is empty

Machine Learning !

Build a Machine Learning classifier that tells us which action to do next :

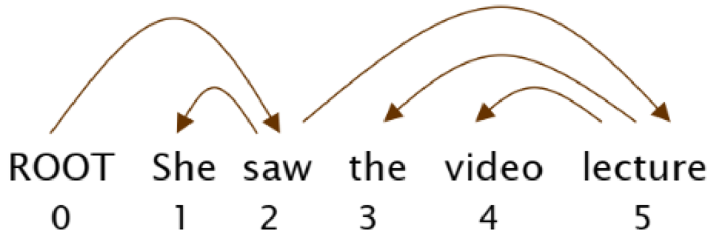
- Shift
- Left-Arc
- Right-Arc

Evaluation of Dependency Parsing

- Count the number of predicted arcs that are correct

Christopher Manning

Evaluation of Dependency Parsing: (labeled) dependency accuracy



$$\text{Acc} = \frac{\text{\# correct deps}}{\text{\# of deps}}$$

UAS = 4 / 5 = 80%
LAS = 2 / 5 = 40%

Gold

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

Parsed

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

Neural Dependency Parsing

Reasons to use Neural Dependency

- The hand-designed features are very sparse: they match very few things
- The features are incomplete
- Expensive computation: too many features !
  95% of the computation time was spent computing the features !

Indicator features

$$s1.w = \text{good} \wedge s1.t = \text{JJ}$$
$$s2.w = \text{has} \wedge s2.t = \text{VBZ} \wedge s1.w = \text{good}$$
$$lc(s_2).t = \text{PRP} \wedge s_2.t = \text{VBZ} \wedge s_1.t = \text{JJ}$$
$$lc(s_2).w = \text{He} \wedge lc(s_2).l = \text{nsubj} \wedge s_2.w = \text{has}$$

Thus, maybe we could get rid of all these hand-engineered features?

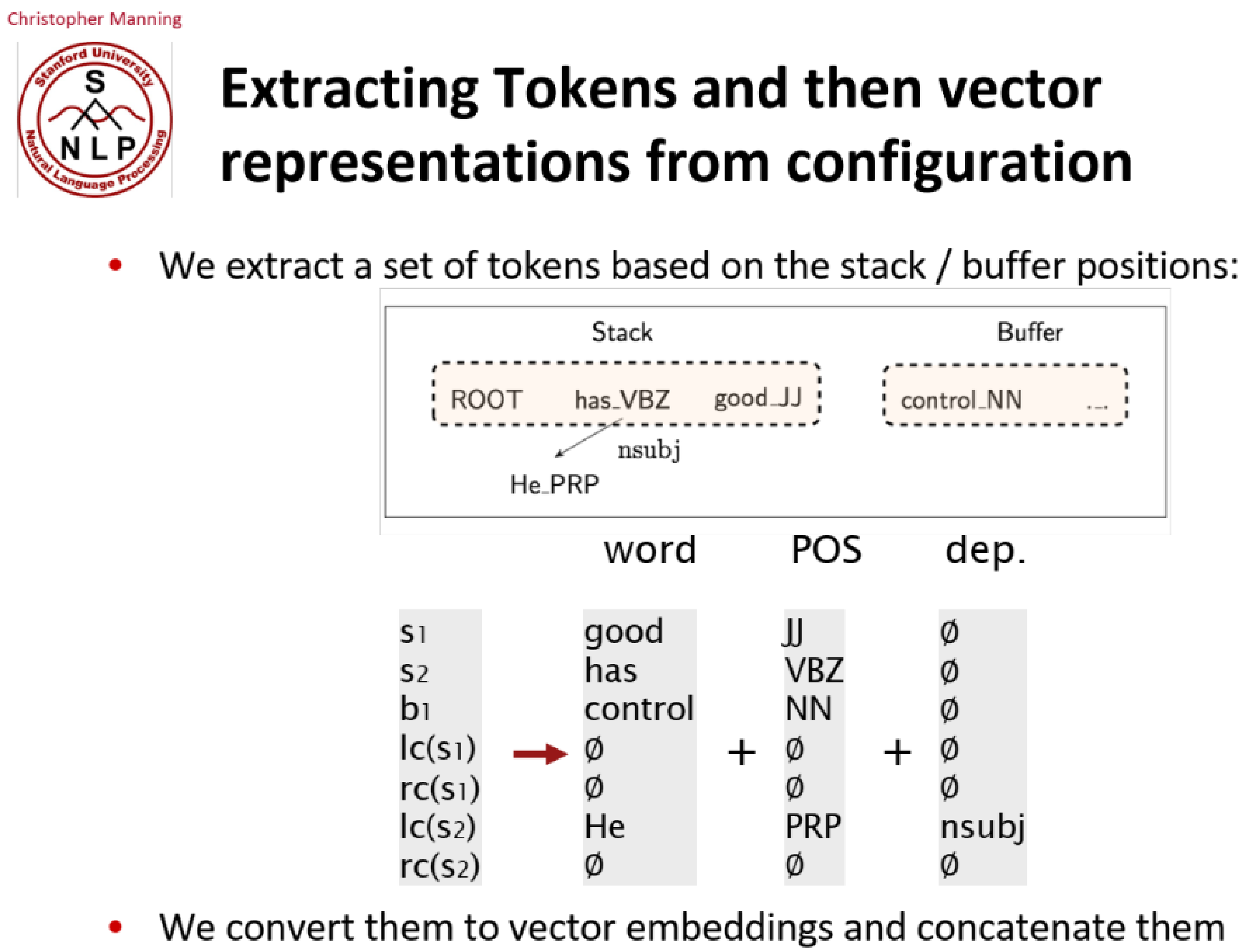
=> Going **Neural** !

Distributed Representations

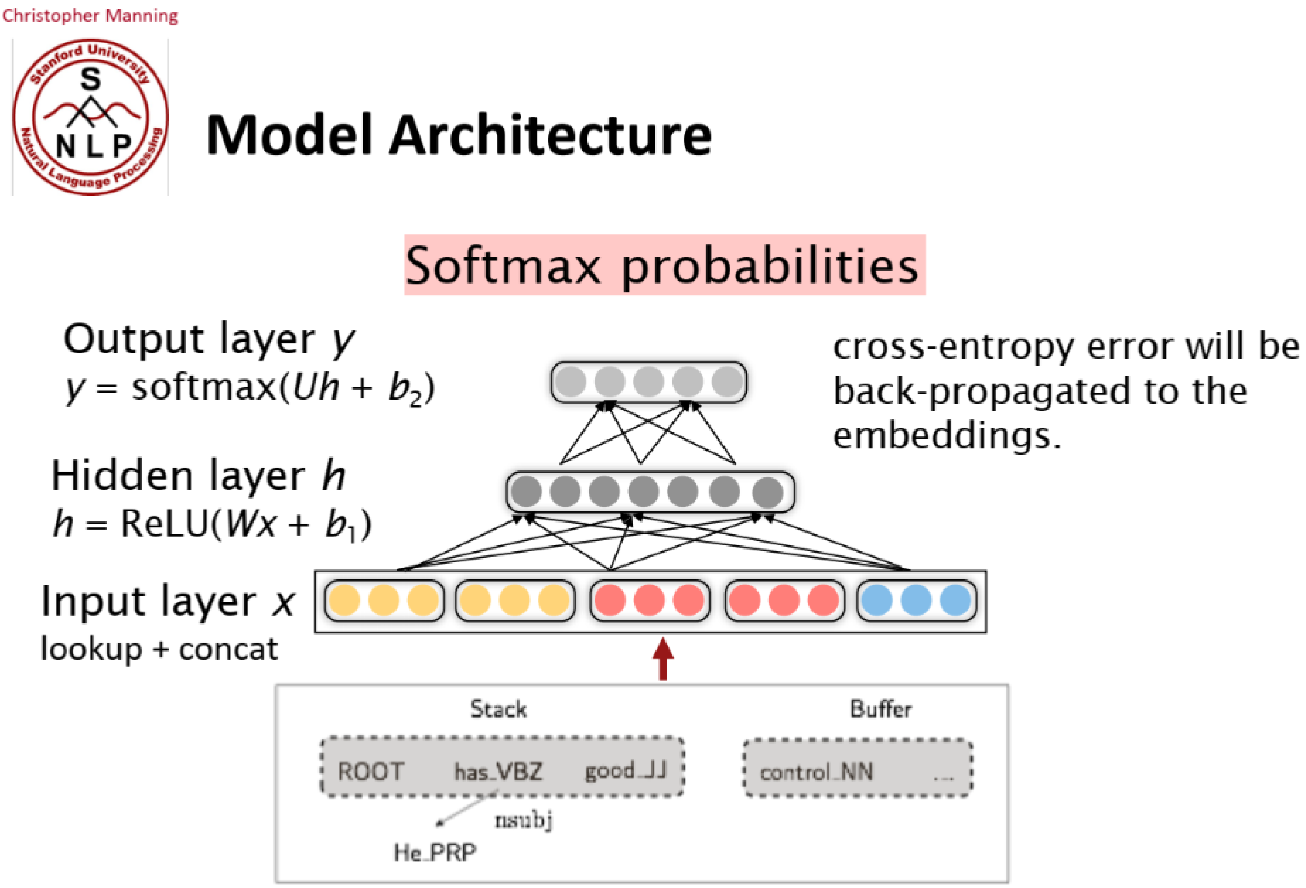
- Words are represented as d-dimensional dense vectors
- Part-of-speech tags (POS) too, to keep an idea of distance:
  - NNS (plural noun) is quite close to NN (singular noun)
- Dependency labels as well !

For each position in the nstack or the buffer, we have a triplet:

- word
- POS
- dependency label



Model Architecture



- Our softmax gives us a probability of whether (Shift, Left-Arc or Right-Arc) is the right action
- We use a cross-entropy loss to see how well we've performed compared to the Treebank parse of the sentence

Beam search:
Instead of deciding which action is the best, allow our selves to decide a bit later:

- Pick a few plausible hypothesis for the possible actions
- Explore their consequences a bit
- Then discard them when we have several other higher-ranked hypotheses

Additional: Google's SyntaxNet article

https://ai.googleblog.com/2016/05/announcing-syntaxnet-worlds-most.html
(https://ai.googleblog.com/2016/05/announcing-syntaxnet-worlds-most.html)