



北京航空航天大学  
BEIHANG UNIVERSITY

# 自然语言处理

人工智能研究院

主讲教师 沙磊

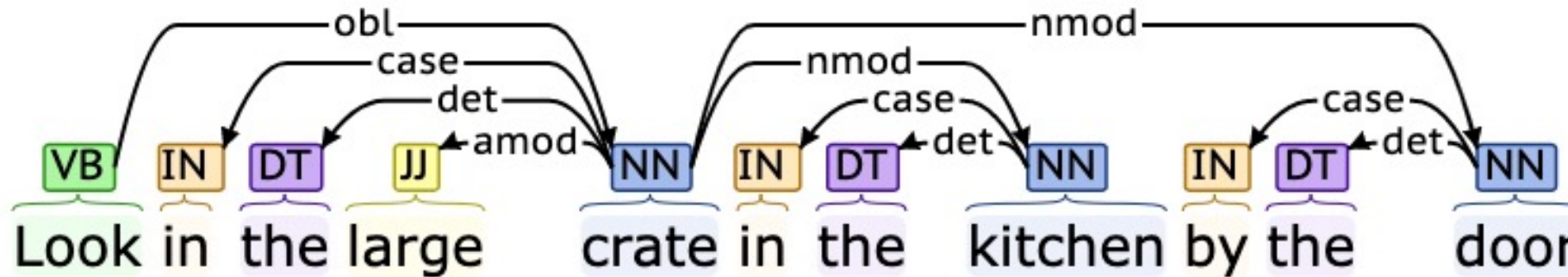


# 依存分析

# Two views of linguistic structure:

## Dependency structure

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



# 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

# Prepositional phrase attachment ambiguity

San Jose cops kill man with knife

Text Paper Translate Listen Close

**San Jose cops kill man with knife**

BBC Sign in News Sport Weather Shop Reel Travel

**NEWS**

Home Video World US & Canada UK Business Tech Science Stories

Science & Environment

San Jose cops kill man with knife

Handwritten annotations: Red arrows and labels (subj, obj, obl, nmod) indicating syntactic structure. A large red arrow points from 'obl' to 'with knife'. Another red arrow points from 'subj' to 'San Jose cops'. A red arrow points from 'obj' to 'kill'. A red arrow points from 'nmod' to 'man'. A red arrow points from 'obl' to 'man'. A red arrow points from 'obl' to 'with knife'.

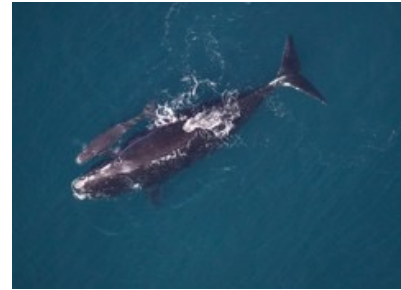
## Scientists count whales from space

By Jonathan Amos  
BBC Science Correspondent



# Prepositional phrase attachment ambiguity

Scientists count whales from space



Scientists count whales from space



# PP attachment ambiguities multiply

- A key parsing decision is how we ‘attach’ various constituents
  - PPs, adverbial or participial phrases, infinitives, coordinations,
- The board approved [its acquisition] [by Royal Trustco Ltd.]  
[of Toronto]  
[for \$27 an share]  
[at its monthly meeting].

# Coordination scope ambiguity

- [Shuttle veteran] and [longtime NASA executive Fred Gregory] appointed to board 2 people
- [Shuttle veteran and longtime NASA executive Fred Gregory] appointed to board 1 people



# Coordination scope ambiguity



# Adjectival/Adverbial Modifier Ambiguity

numbers, including some that featured a bucket and bells brigade of performers who beat rhythms on buckets and trash cans with drums sticks and hammer mallets. PHOTO BY JENNIFER STULTZ

## MENTORING DAY Students get first hand job experience

By Gale Rose  
grose@pratttribune.com

Eager students invaded businesses all over Pratt Tuesday, October 24 as they looked for future job opportunities on Disability Mentoring Day.

The 97 students from 12 schools fanned out across Pratt and got first hand experience what it would be like to work at those 40 businesses. They asked questions and got some hands on experience with various operations.

Paola Luna of Pratt High School, Gina Patton of Kingman High School and America Fernandez of St. John chose the Main Street Small Animal Veterinarian Clinic for their business. Students got a tour of the facility, learned what happens in an examination, got to handle various animals and watched a snake eat a mouse.

Luna said she was interested in animal health and wanted to know more about caring for hurt animals. Patton likes all kinds of animals and said she learned a lot from the experience. Watching the snake eat the mouse impressed her the most.

Fernandez wants to become a veterinarian and enjoyed learning everything that veterinarians

SEE MENTORING, 6

**ing Meyer**  
ty Commissioner

- Hospital Pharmacist for 41 years
- 4 years Commissioner for Pratt Planning and Zoning Board of Appeals
- 3 years Pratt City Commission
- Graduate of Pratt High School and KU School of Pharmacy
- Past Member and President of Civic Groups and Organizations
- Experience and Knowledge of Financial Responsibility and Budgeting
- Supports Family Values, Education, and Business Growth
- Common Sense Approach for the Sustained Progress of Pratt

12 SATURDAY, October 28, 2017 ■ The Pratt Tribune ■ www.pratttribune.com

# Verb Phrase (VP) attachment ambiguity



The screenshot shows the Guardian website's navigation bar with icons for user, search, and menu, followed by the 'theguardian' logo. Below the navigation bar, a breadcrumb trail reads 'home > world > americas', with 'asia' and a menu icon '≡ all' to the right. The article title 'Rio de Janeiro' is displayed in blue. The headline 'Mutilated body washes up on Rio beach to be used for Olympics beach volleyball' is shown in black text. Hand-drawn annotations highlight the ambiguity: a green line under 'Mutilated body' and a red line under 'on Rio beach' both have arrows pointing to the verb phrase 'to be used for Olympics beach volleyball', illustrating how the phrase can attach to either the subject or the location.

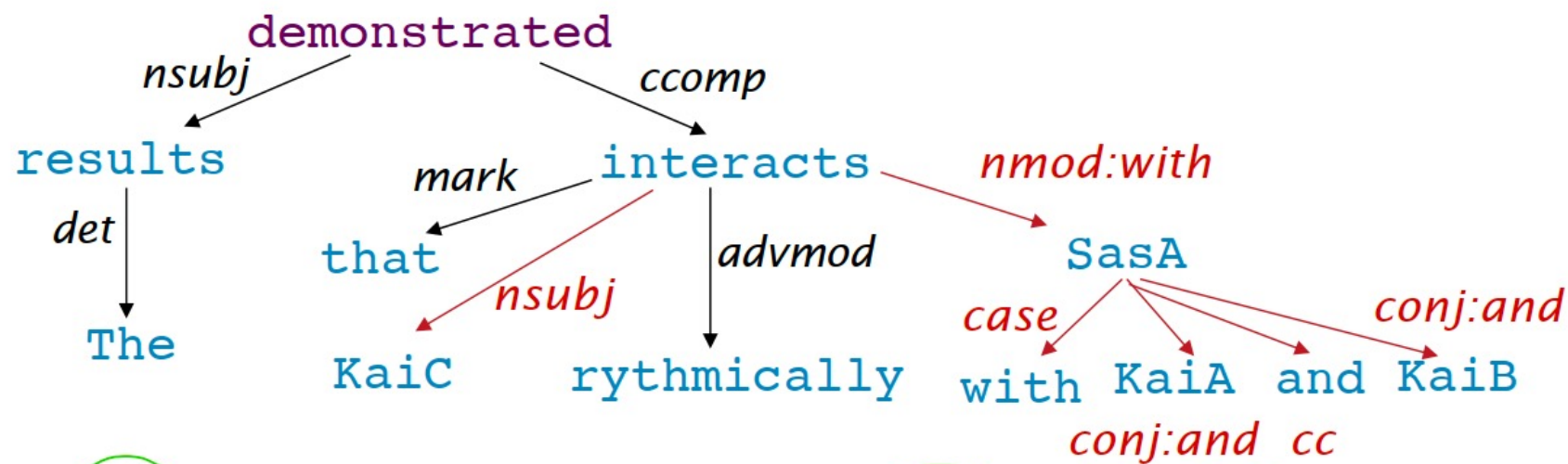
home > world > americas    asia    ≡ all

**Rio de Janeiro**

Mutilated body washes up  
on Rio beach to be used for  
Olympics beach volleyball

6/29/16, 1:48 PM

Dependency paths help extract semantic interpretation – simple practical example: extracting protein-protein interaction

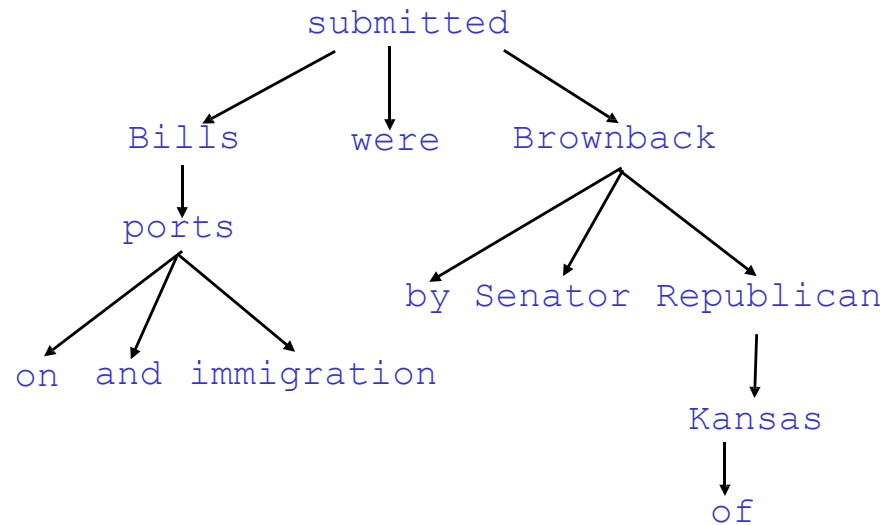


KaiC ← *nsubj* interacts *nmod:with* → SasA  
KaiC ← *nsubj* interacts *nmod:with* → SasA *conj:and* → KaiA  
KaiC ← *nsubj* interacts *nmod:with* → SasA *conj:and* → KaiB

[Erkan et al. EMNLP 07, Fundel et al. 2007, etc.]

## 2. Dependency Grammar and Dependency Structure

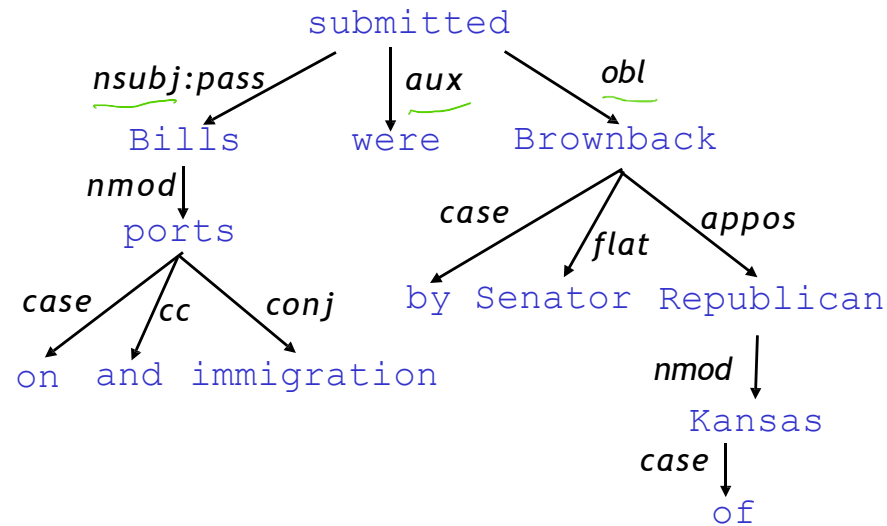
- Dependency syntax postulates that syntactic structure consists of relations between lexical items, normally binary asymmetric relations (“arrows”) called **dependencies**



# Dependency Grammar and Dependency Structure

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

The arrows are commonly **typed** with the name of grammatical relations (subject, prepositional object, apposition, etc.)



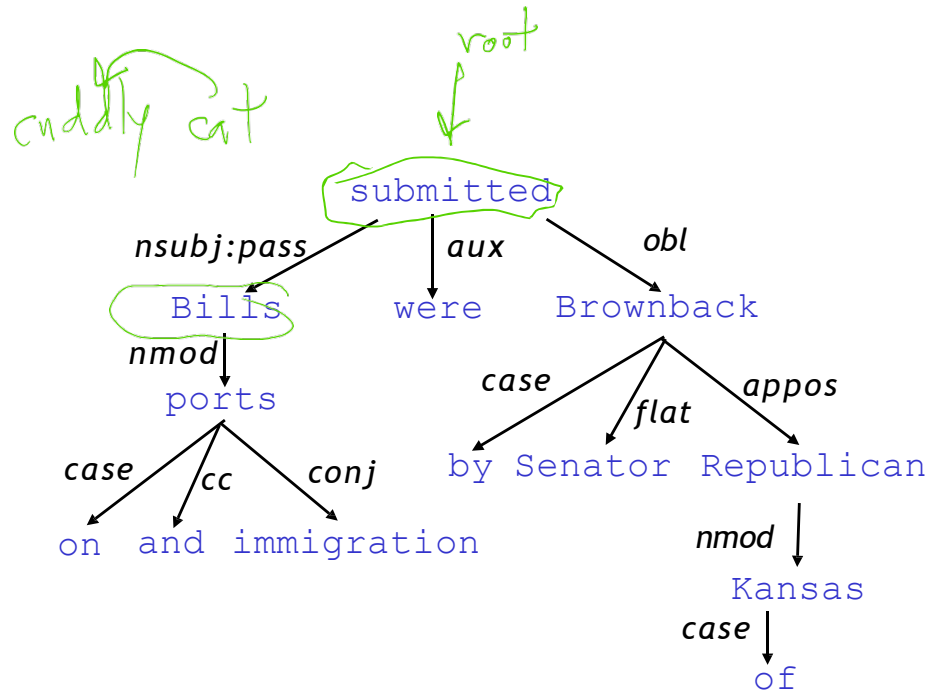


# Dependency Grammar and Dependency Structure

- 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, regent) with a **dependent** (modifier, inferior, subordinate)

Usually, dependencies form a tree (a connected, acyclic, single-root graph)





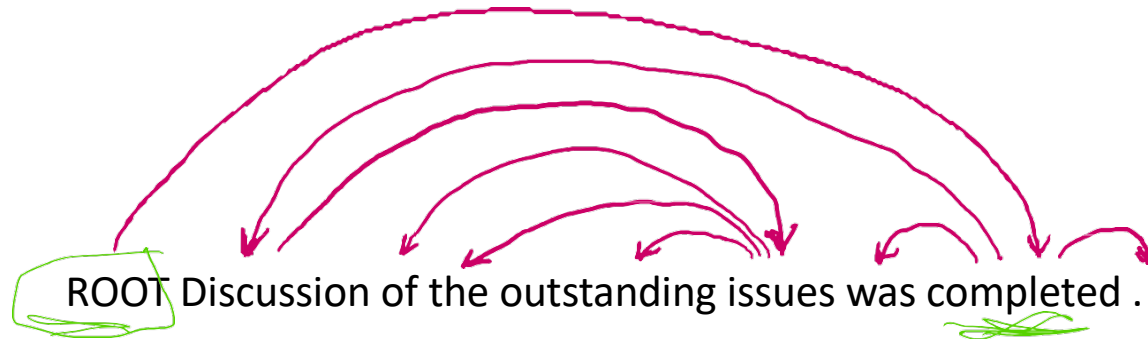
# Pāṇini's grammar (c. 5th century BCE)



# Dependency Grammar/Parsing History

- 依存结构的概念可以追溯到很久以前
  - 源自波尼尼的语法(公元前5世纪左右)
  - 第一个千年的阿拉伯语法学家的基本方法
- constituency/CFG是一个新发明
  - 20世纪的发明(R.S. Wells, 1947;然后是乔姆斯基1953年等)
- 现代依存句法研究通常追溯到吕西安·泰尼尔(1959)
  - 在20世纪的"东方"是主导方法(俄罗斯、中国等)
  - 适用于语序较自由、屈折语言如俄语(或拉丁语!)
- 即使在美国,也被用于最早的一些自然语言处理解析器中:
  - 美国计算语言学的创始人之一大卫·海斯构建了早期(最早?)的依存句法分析器(Hays 1962),并在《语言》杂志上发表了关于依存语法的文章

# Dependency Grammar and Dependency Structure



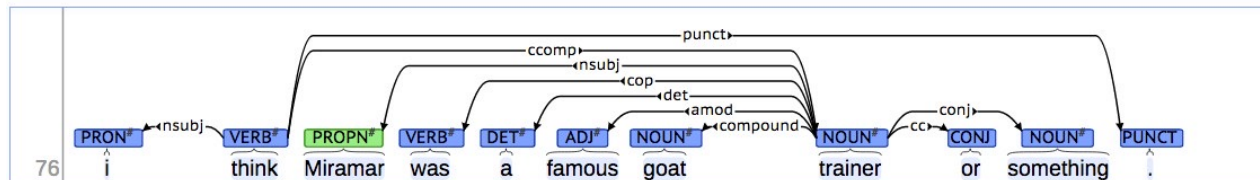
- Some people draw the arrows one way; some the other way!
  - Tesnière had them point from head to dependent – we follow that convention
- We usually add a fake ROOT so every word is a dependent of precisely 1 other node



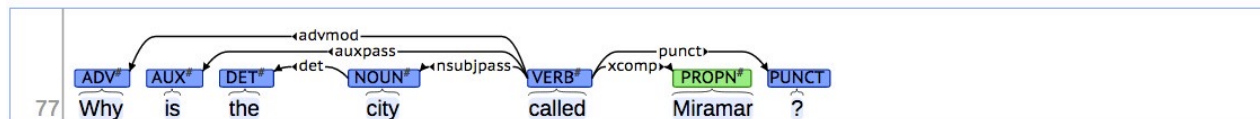
# The rise of annotated data & Universal Dependencies treebanks

- Brown corpus (1967; PoS tagged 1979); Lancaster-IBM Treebank (starting late 1980s); Marcus et al. 1993, The Penn Treebank, *Computational Linguistics*;
- Universal Dependencies: <http://universaldependencies.org/>

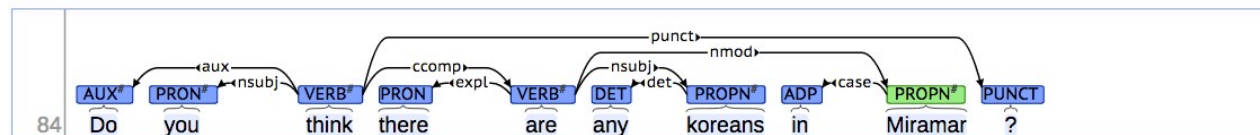
[context] [conllu]



[context] [conllu]



[context] [conllu]



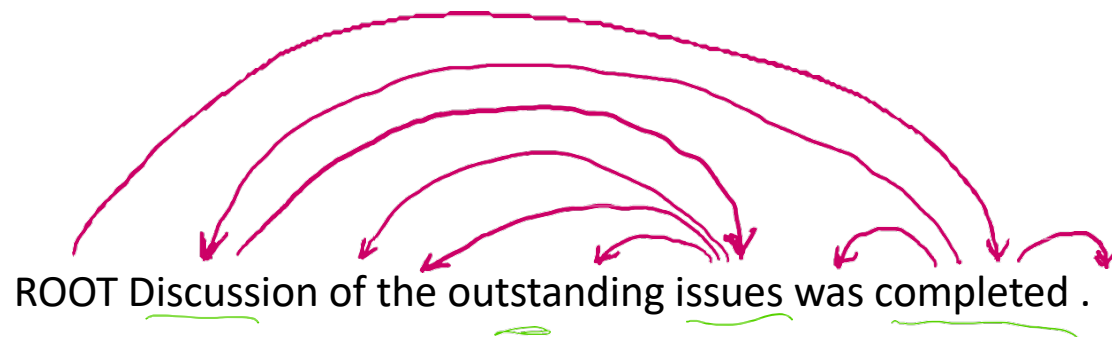
# The rise of annotated data

- 起初,构建树库似乎比(手工)编写语法要慢得多,也没那么有用
- 但是树库给我们带来很多好处
  - 劳动成果的可重用性
    - 可以在其基础上构建许多句法分析器、词性标注器等
    - 语言学研究的宝贵资源
  - 广泛覆盖,而不仅仅是少数直观的语言现象
  - 频率和分布信息
  - 评估自然语言处理系统的一种方式

# Dependency Conditioning Preferences

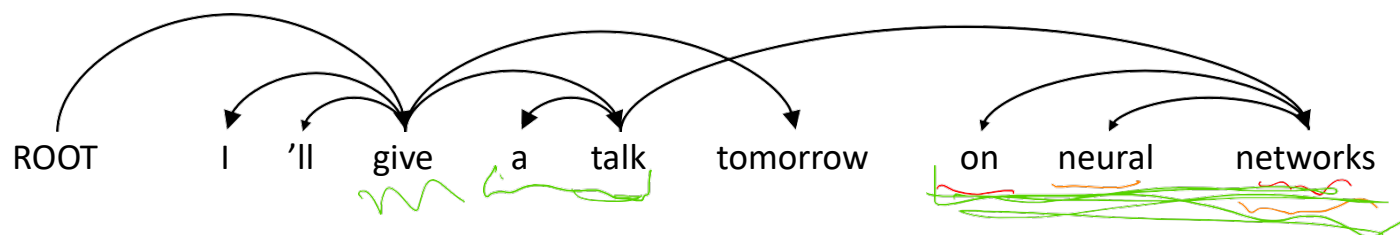
What are the sources of information for dependency parsing?

1. Bilexical affinities (双词亲和力) 依存关系 [discussion → issues] 是合理的
2. Dependency distance (依存距离) 大多数依存关系都存在于相邻的词语之间
3. Intervening material (干扰成分) 依存关系很少跨越中间的动词或标点符号
4. Valency of heads (中心词的配价) 一个中心词通常在其左右两侧分别有多少个依存成分?



# Dependency Parsing

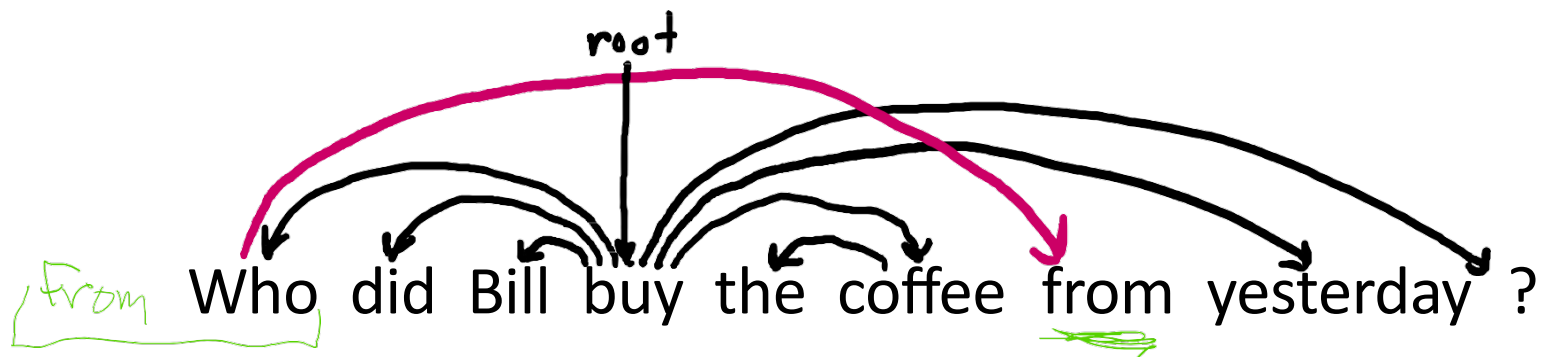
- 通过为每个词选择它依存于哪个其他词(包括ROOT), 来对一个句子进行句法分析
- 通常有一些约束:
  - 只有一个词是ROOT的依存成分
  - 不希望出现循环依存  $A \rightarrow B, B \rightarrow A$
- 这使得依存关系形成一棵树
- 最后一个问题是箭头是否可以交叉(非投射)





# Projectivity

- 投射句法分析的定义: 当词语按其线性顺序排列, 所有依存弧都位于词语上方时, 不存在交叉的依存弧
- 对应于CFG(上下文无关文法)树的依存关系必须是投射的
  - 即, 通过将每个范畴的一个子节点作为中心词来形成依存关系
- 大多数句法结构都是如此投射的, 但是依存理论通常允许非投射结构来解释移位的成分
  - 如果没有这些非投射依存关系, 就不容易正确得到某些结构的语义



# 3. Methods of Dependency Parsing

- 动态规划
  - Eisner (1996) 提出了一个巧妙的算法,其复杂度为 $O(n^3)$ ,通过在两端而不是中间生成带有中心词的句法分析项目
- 图算法
  - 为一个句子创建最小生成树
  - McDonald et al. (2005)的MSTParser使用ML分类器(他使用MIRA进行在线学习,但也可以是其他方法)对依存关系进行独立评分
  - 神经图算法句法分析器: Dozat和Manning (2017)及后续工作——非常成功!
- 约束满足
  - 不满足硬约束的边会被消除。Karlsson (1990)等。
- "基于转移的句法分析"或"确定性依存句法分析"
  - 由良好的机器学习分类器指导依附的贪心选择
  - 例如, MaltParser(Nivre et al. 2008)。事实证明非常有效。

# 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  $\sigma$ , written with top to the right
    - which starts with the ROOT symbol
  - a buffer  $\beta$ , 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

# 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<sub>r</sub>  $\sigma | w_i | w_j, \beta, A \rightarrow \sigma | w_j, \beta, A \cup \{r(w_j, w_i)\}$

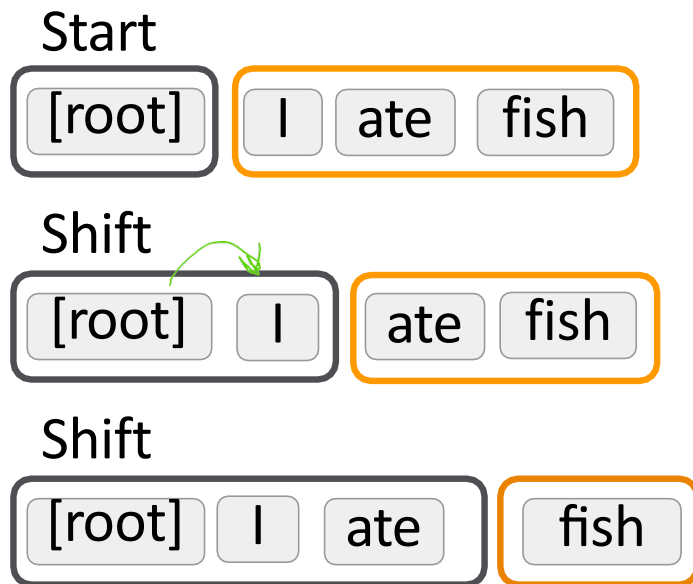
3. Right-Arc<sub>r</sub>  $\sigma | w_i | w_j, \beta, A \rightarrow \sigma | w_i, \beta, A \cup \{r(w_i, w_j)\}$

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

# Arc-standard transition-based parser

(there are other transition schemes ...)

- Analysis of “I ate fish”



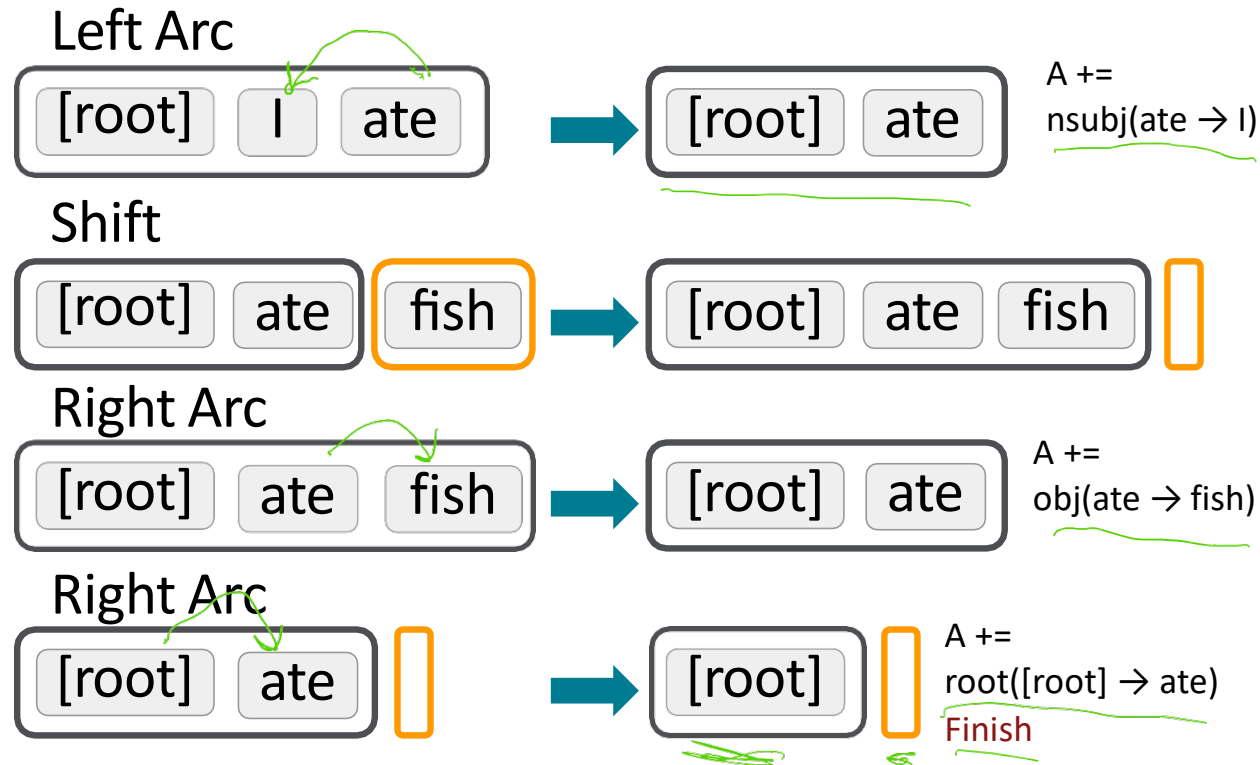
**Start:**  $\sigma = [\text{ROOT}]$ ,  $\beta = w_1, \dots, w_n$ ,  $A = \emptyset$

1. Shift  $\sigma, w_i | \beta, A \rightarrow \sigma | w_i, \beta, A$
2. Left-Arc<sub>r</sub>  $\sigma | w_i | w_j, \beta, A \rightarrow \sigma | w_j, \beta, A \cup \{r(w_j, w_i)\}$
3. Right-Arc<sub>r</sub>  $\sigma | w_i | w_j, \beta, A \rightarrow \sigma | w_i, \beta, A \cup \{r(w_i, w_j)\}$

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

# Arc-standard transition-based parser

- Analysis of “I ate fish”

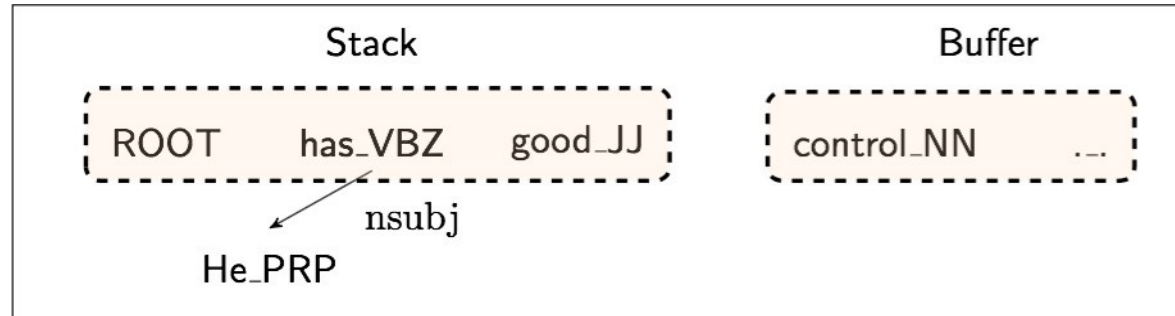


# MaltParser [Nivre and Hall 2005]

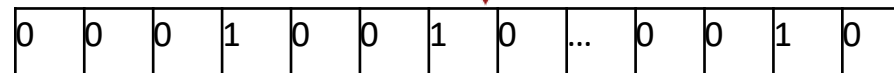
- We have left to explain how we choose the next action 🙋
  - Answer: Stand back, I know machine learning!
- Each action is predicted by a discriminative classifier (e.g., softmax classifier) over each legal move
  - Max of 3 untyped choices; max of  $|R| \times 2 + 1$  when typed
  - Features: top of 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



# Conventional Feature Representation



binary, sparse  
dim =  $10^6 - 10^7$



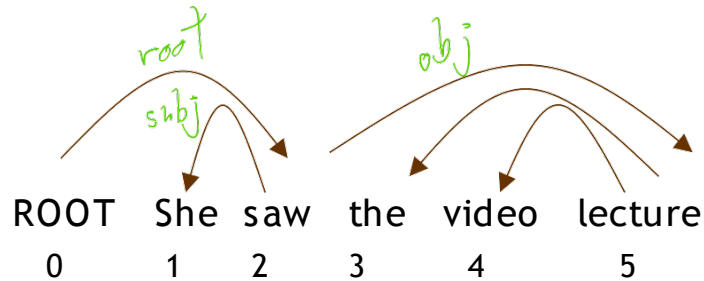
Feature templates: usually a combination of 1–3 elements from the configuration

Indicator features

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

# Evaluation of Dependency Parsing: (labeled) dependency accuracy

- labeled attachment score (LAS)
- unlabeled attachment score (UAS)



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

$$\text{UAS} = 4 / 5 = 80\%$$

$$\text{LAS} = 2 / 5 = 40\%$$

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

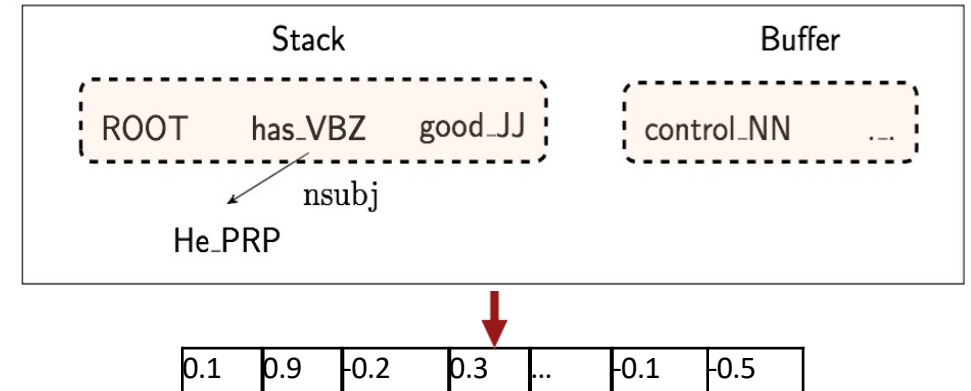
Parsed			
1	2	She	nsubj
2	0	saw	root
3	4	the	det
4	5	video	nsubj
5	2	lecture	ccomp

# Handling non-projectivity

- 我们介绍的基于弧标准的算法只能构建投射依存树
- 指向中心词的可能方向:
  - 对于非投射弧, 直接宣告失败 🙅
  - 使用只有投射表示的依存形式
    - CFG只允许投射结构; 可以提升违反投射性的中心词
  - 使用投射依存分析算法的后处理器来识别和解决非投射链接
  - 添加额外的转换, 以对大多数非投射结构进行建模(例如, 添加一个额外的SWAP转换, 参考冒泡排序)
  - 换成一种不使用或不需要任何投射性约束的分析机制(例如, 基于图的MSTParser或Dozat和Manning(2017))

## 4. Why do we gain from a neural dependency parser? Indicator Features Revisited

- Problem #1: sparse
- Problem #2: incomplete
- Problem #3: expensive computation



- Dense
  - dim =  $\sim 1000$  More than 95% of parsing time is consumed by feature computation
- Neural Approach:
  - learn a dense and compact feature representation

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

# A neural dependency parser [Chen and Manning 2014]

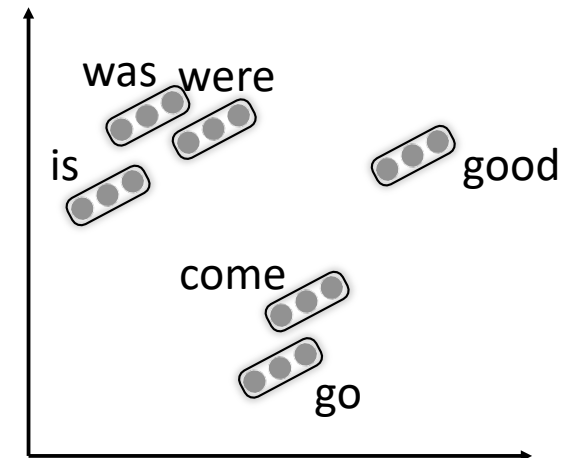
- Results on English parsing to Stanford Dependencies:
  - Unlabeled attachment score (UAS) = head
  - Labeled attachment score (LAS) = head and label

Parser	UAS	LAS	sent. / s
MaltParser	89.8	87.2	469
MSTParser	91.4	88.1	10
TurboParser	<b>92.3</b>	89.6	8
C & M 2014	92.0	<b>89.7</b>	<b>654</b>

# First win: Distributed Representations

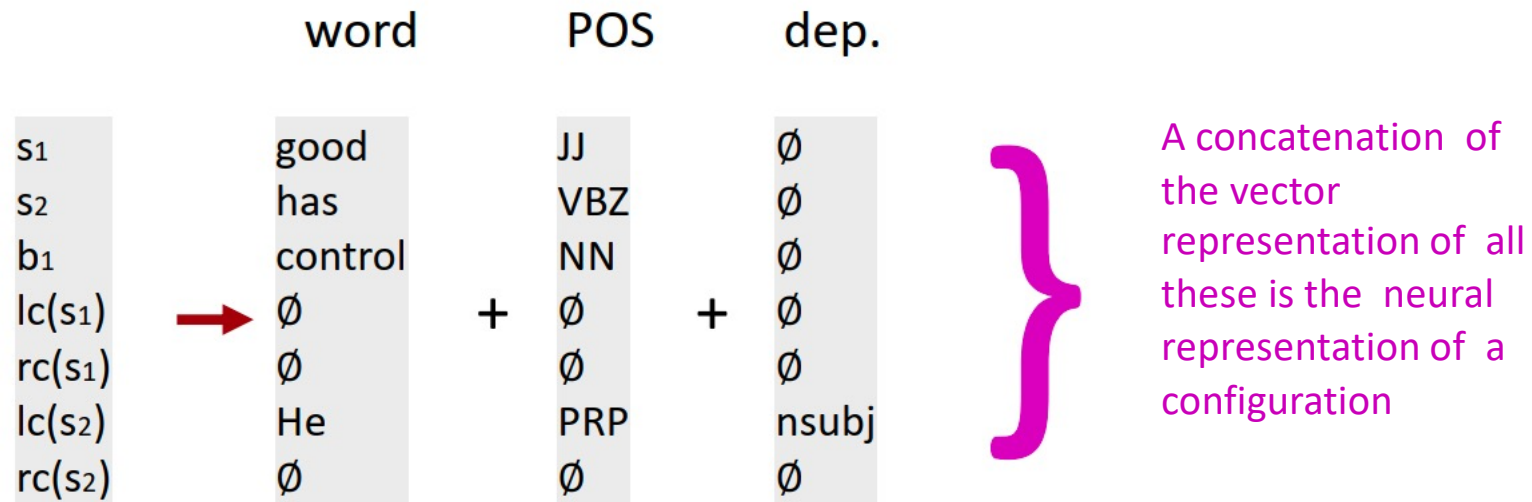
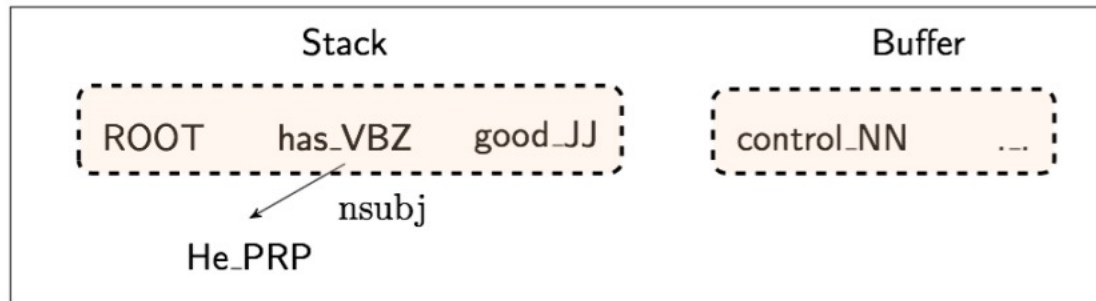
- We represent each word as a  $d$ -dimensional dense vector (i.e., word embedding)
  - Similar words are expected to have close vectors.
- Meanwhile, **part-of-speech tags** (POS) and **dependency labels** are also represented as  $d$ -dimensional vectors.
  - The smaller discrete sets also exhibit many semantical similarities.

NNS (plural noun) should be close to NN (singular noun).  
nummod (numerical modifier) should be close to amod (adjective modifier).



# Extracting Tokens & vector representations from configuration

- We extract a set of tokens based on the stack / buffer positions:



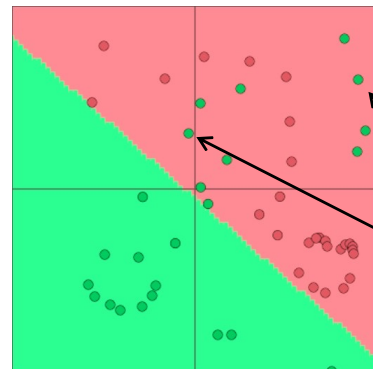


## Second win: Deep Learning classifiers are non-linear classifiers

- A **softmax classifier** assigns classes  $y \in \mathcal{C}$  based on inputs  $x \in \mathbb{R}^d$  via the probability:

$$p(y|x) = \frac{\exp(W_y \cdot x)}{\sum_{c=1}^C \exp(W_c \cdot x)}$$

- We train the weight matrix  $W \in \mathbb{R}^{C \times d}$  to minimize the neg. log loss :  $\sum_i -\log p(y_i|x_i)$
- **Traditional ML classifiers** (including Naïve Bayes, SVMs, logistic regression and softmax classifier) are not very powerful classifiers: they only **give linear decision boundaries**



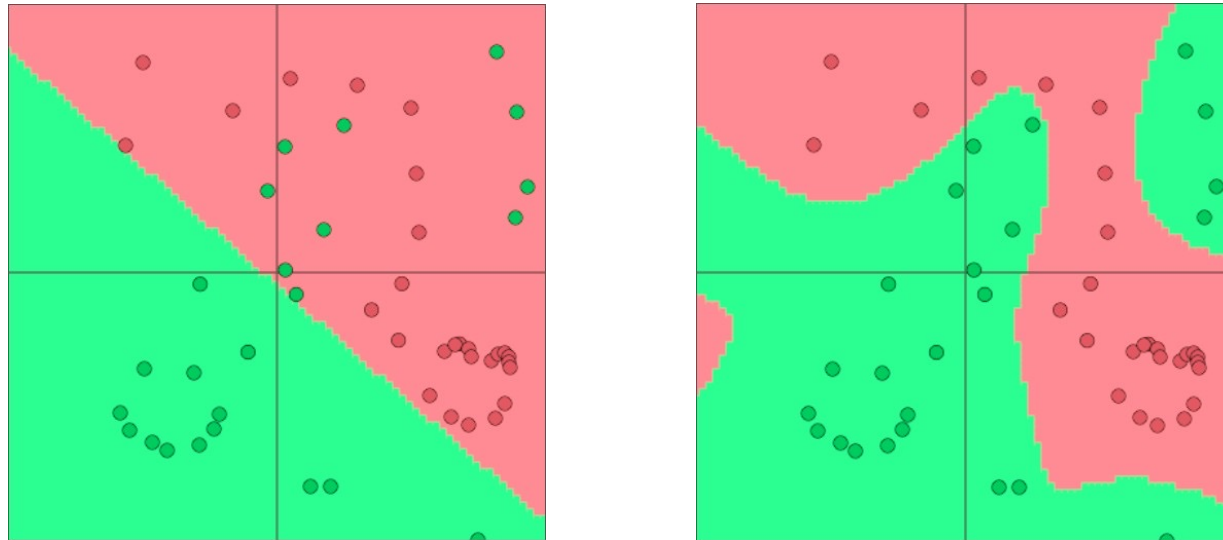
This can be quite limiting

→ Unhelpful when a problem is complex

Wouldn't it be cool to get these correct?

# Neural Networks are more powerful

- Neural networks can learn much more complex functions with nonlinear decision boundaries!
  - Non-linear in the original space, linear for the softmax at the top of the neural network



Visualizations with ConvNetJS by Andrej Karpathy!

<http://cs.stanford.edu/people/karpathy/convnetjs/demo/classify2d.html>

# Simple feed-forward neural network multi-class classifier

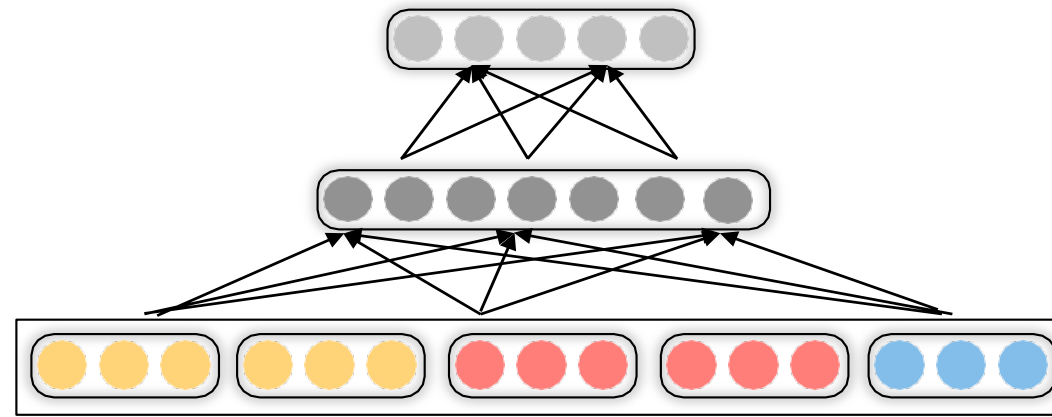
Softmax probabilities

Output layer  $y$   
 $y = \text{softmax}(Uh + b_2)$

Hidden layer  $h$   
 $h = \text{ReLU}(Wx + b_1)$

Input layer  $x$

$x$  is result of lookup  
 $x_{(i,...,i+d)} = Le$   
lookup + concat

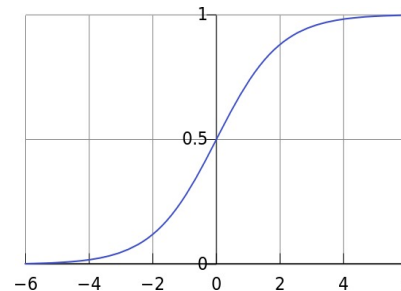


Log loss (cross-entropy error) will be back-propagated to the embeddings

The hidden layer re-represents the input — it moves inputs around in an intermediate layer vector space—so it can be easily classified with a (linear) softmax

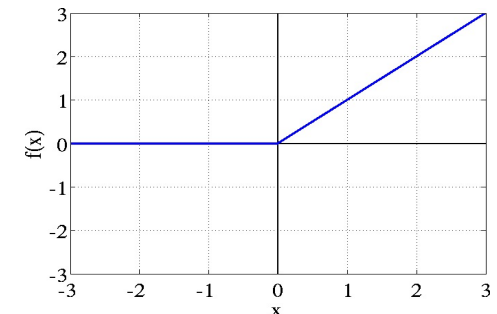
logistic = “sigmoid”

$$f(z) = \frac{1}{1 + \exp(-z)}$$

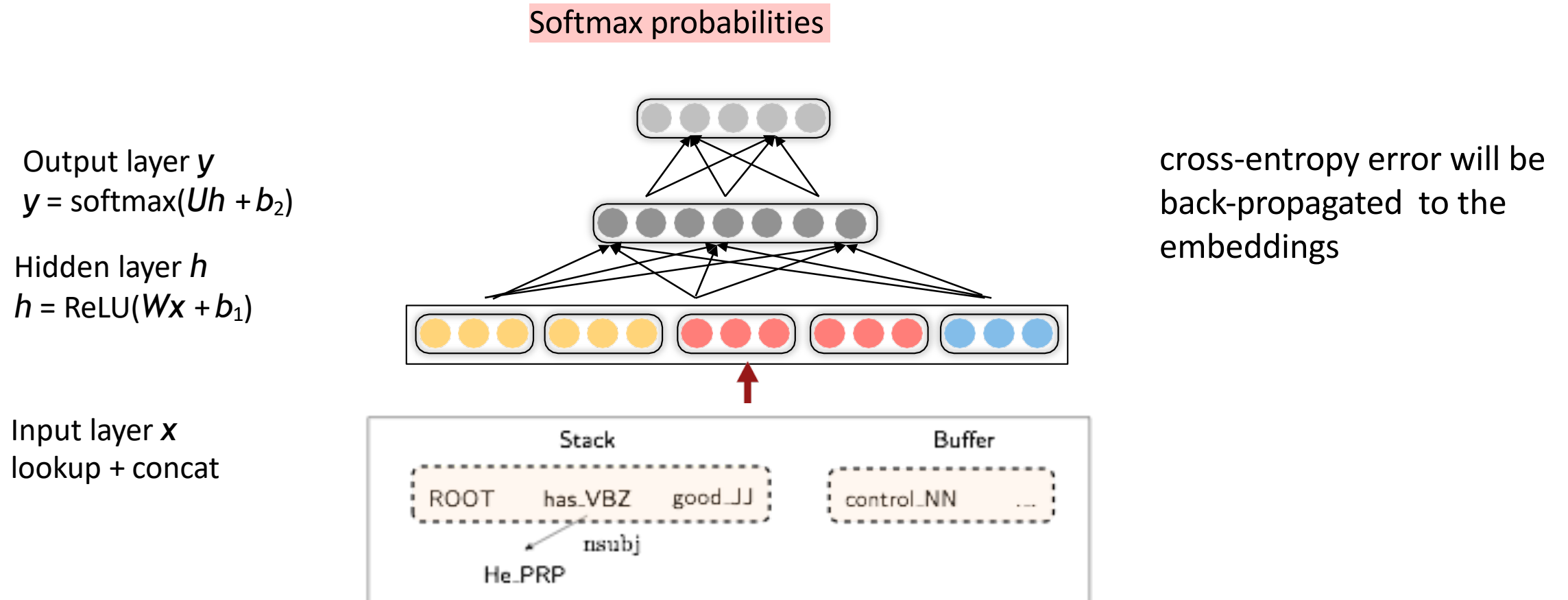


ReLU = Rectified Linear Unit

$$\text{rect}(z) = \max(z, 0)$$

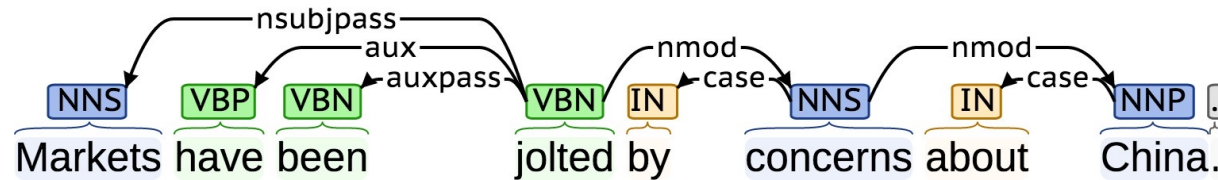


# Neural Dependency Parser Model Architecture



# Dependency parsing for sentence structure

- Neural networks can accurately determine the structure of sentences, supporting interpretation



- Chen and Manning (2014) was the first simple, successful neural dependency parser
- The dense representations (and non-linear classifier) let it outperform other greedy parsers in both accuracy and speed

# Further developments in transition-based neural dependency parsing

This work was further developed and improved by others, including in particular at Google

- Bigger, deeper networks with better tuned hyperparameters
- Beam search
- Global, conditional random field (CRF)-style inference over the decision sequence Leading to SyntaxNet and the Parsey McParseFace model (2016):  
“The World’s Most Accurate Parser”

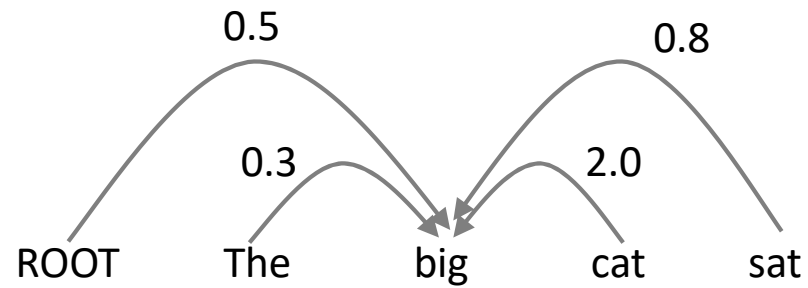
<https://research.googleblog.com/2016/05/announcing-syntaxnet-worlds-most.html>



Method	UAS	LAS (PTB WSJ SD 3.3)
Chen & Manning 2014	92.0	89.7
Weiss et al. 2015	93.99	92.05
Andor et al. 2016	94.61	92.79

# Graph-based dependency parsers

- Compute a score for every possible dependency for each word
  - Doing this well requires good “contextual” representations of each word token,

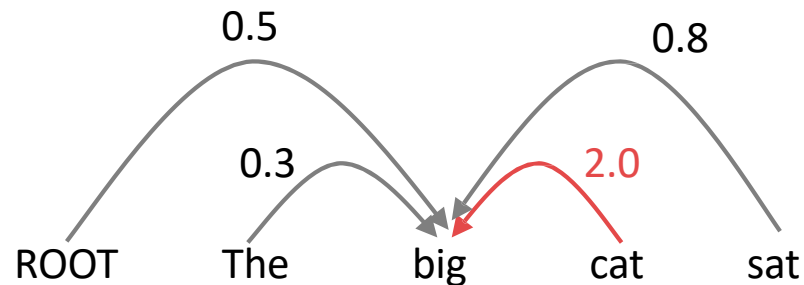


e.g., picking the head for “big”



# Graph-based dependency parsers

- Compute a score for every possible dependency for each word
  - Doing this well requires good “contextual” representations of each word token
  - And repeat the same process for each other word




e.g., picking the head for “big”

# A Neural graph-based dependency parser

[Dozat and Manning 2017; Dozat, Qi, and Manning 2017]

- This paper revived interest in graph-based dependency parsing in a neural world
  - Designed a biaffine scoring model for neural dependency parsing
    - Also crucially uses a neural sequence model
- Really great results!
  - **But slower than the simple neural transition-based parsers**
    - There are  $n^2$  possible dependencies in a sentence of length  $n$



Method	UAS	LAS (PTB WSJ SD 3.3)
Chen & Manning 2014	92.0	89.7
Weiss et al. 2015	93.99	92.05
Andor et al. 2016	94.61	92.79
<b>Dozat &amp; Manning 2017</b>	<b>95.74</b>	<b>94.08</b>

Thank you!