Topics in Natural Language Processing

Shay Cohen

Institute for Language, Cognition and Computation

University of Edinburgh

Lecture 7

Administrativia

I received suggested topics from most of you

- If you didn't send a topic yet, please send it as soon as possible
- Next thing: scheduling everybody and allocating brief paper responses
- I will try to allocate brief paper responses on the same topic you present (but different papers)
- It will not always work out

Last class

Semiring inference: CKY and the inside algorithms

$$\alpha(A, i, j) = \bigoplus_{\substack{k \\ A \to B, C}} \alpha(B, i, k) \otimes \alpha(C, k+1, j)$$

$i, j \in \mathbb{N}$, $A \in N$

$A \to B\ C \in R$

$\oplus \to \Sigma$

$\oplus \to \max$

$\otimes \to +$

$\otimes \to \times$

Parsing as Weighted Logic Programming

$\text{constit}(a, i, j) \oplus = \text{constit}(b, i, k) \otimes \text{constit}(c, k+1, j) \otimes \text{rule}(a \to b\, c)$

$\text{constit}(a, i, i) \oplus = \text{rule}(a \to w_i)$

Goal: $\text{constit}(S, \cancel{0}\,1, n)$

Example of a Weighted Logic Programme

We are given a sequence $w_1, \ldots, w_n$ of some symbols.

$$\text{prob}(b, i) \oplus = \text{prob}(a, i-1) \otimes \text{transition}(a \to b) \otimes \text{emission}(b, w_i)$$

$$\text{prob}(a, 1) \oplus = \text{start\_state}(b) \otimes \text{emission}(a, w_1)$$

$$\alpha(B, i) = \sum_{A} \alpha(A, i-1) \times t(A \to B) \times \text{emission}(B \to w_i)$$

state

point in the sentence

"Forward" algorithm

Viterbi

$a \otimes b$	$a \times b$
$a \oplus b$	$\max\{a, b\}$

Example of a Weighted Logic Programme

We are given a sequence $w_1, \ldots, w_n$ of some symbols.

$$\mathrm{prob}(b, i) \oplus = \mathrm{prob}(a, i-1) \otimes \mathrm{transition}(a \rightarrow b) \otimes \mathrm{emission}(b, w_i)$$

$$\mathrm{prob}(a, 1) \oplus = \mathrm{start\_state}(b) \otimes \mathrm{emission}(a, w_1)$$

Hidden Markov models and the forward algorithm:

- $\mathrm{emission}$ are the emission probabilities
- $\mathrm{transition}$ are the transition probabilities
- $\mathrm{start\_state}$ are the initial probabilities
- $\mathrm{prob}(b, i)$ gives the probability $p(w_1, \ldots, w_i, S_i = b)$

Weighted Logic Programming

Once we present the inference algorithm as a weighted logic program, there are general-purpose algorithms to solve it.

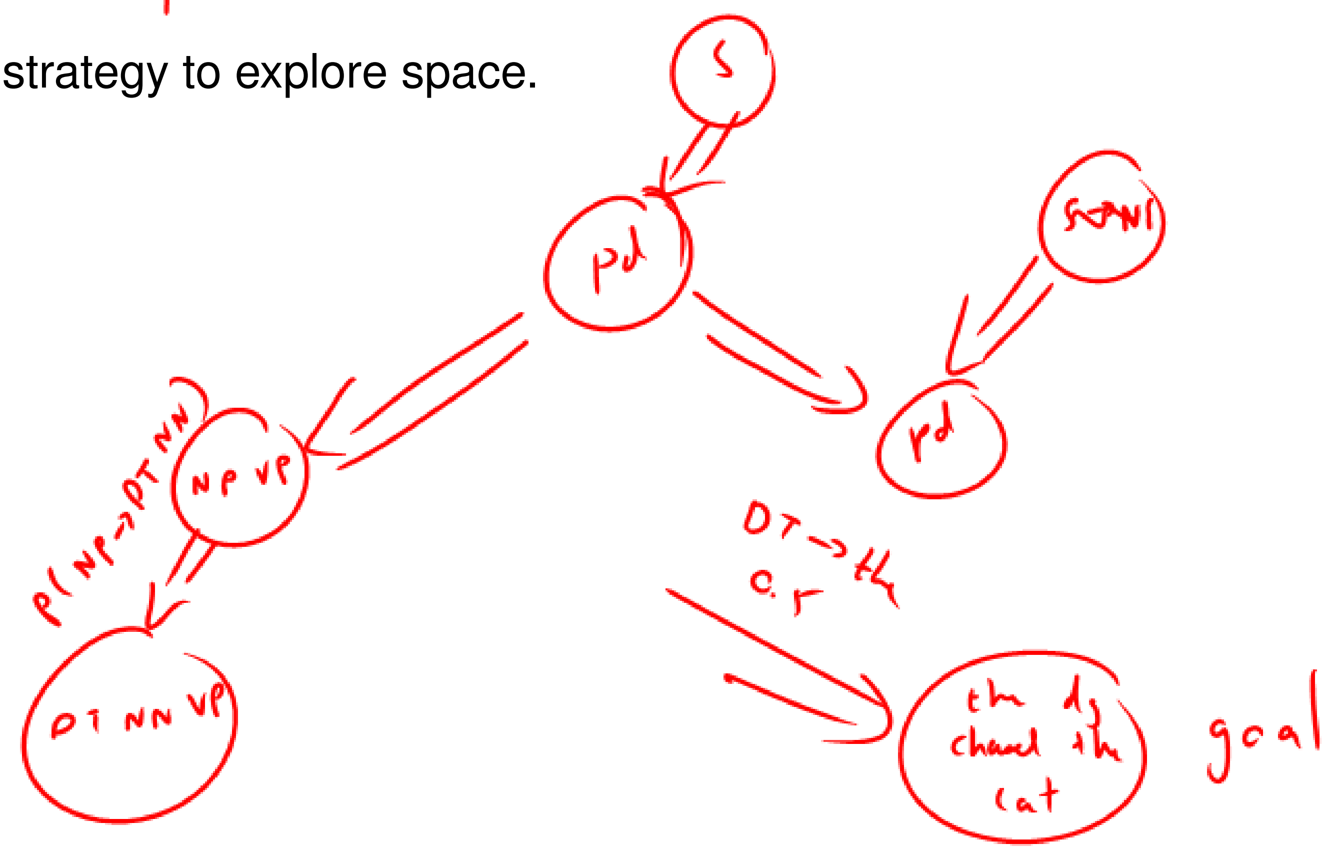
- The “agenda” algorithm
- Search algorithms
- Tabular algorithms

If that interests you, take a look at `http://www.dyna.org`.

Parsing Using Search

State space: partial derivations a a A B c d E A

Need a strategy to explore space.



Inference Using Graphs and Hypergraphs

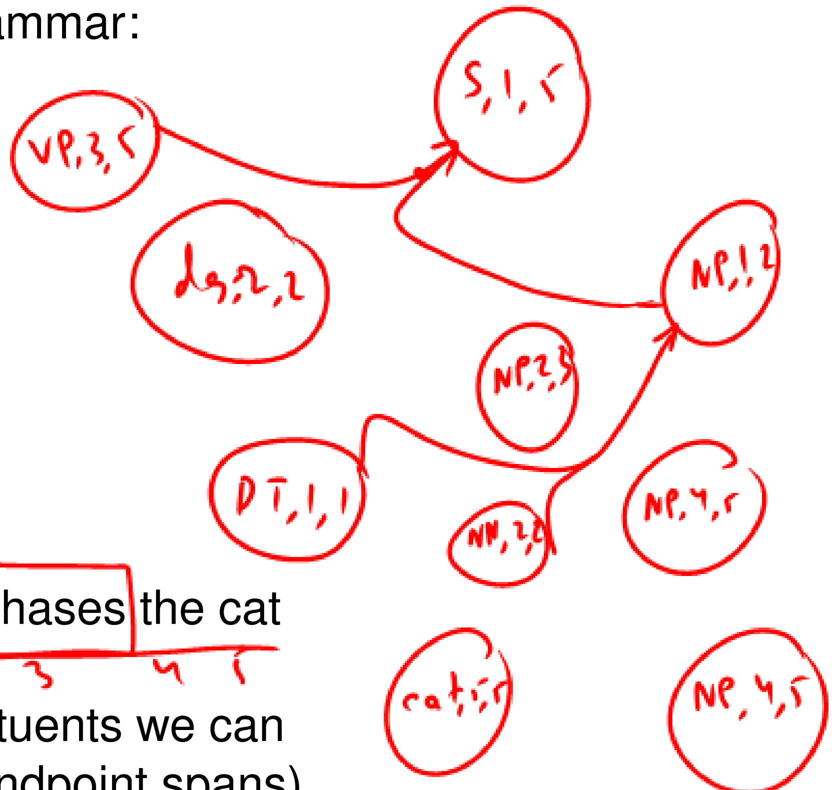
Consider the context-free grammar:

S $\rightarrow$ NP VP
NP $\rightarrow$ DT NN
VP $\rightarrow$ VB NP
VB $\rightarrow$ chases
NN $\rightarrow$ dog | cat | chases
DT $\rightarrow$ the
NP $\rightarrow$ DT NN NN

We want to parse: The dog chases the cat

1 2 3 4 5

What are the potential constituents we can create? (nonterminals with endpoint spans)



Today's class

- Log-linear models and their estimation
- Regularisation

Estimation until now

- Count and normalise
- Corresponds to maximum likelihood estimate for multinomial models

A Type-based POS Tagging Model

- Want to model $p(\overset{y}{\text{tag}} \mid \overset{x}{\text{word}})$
- Can use a simple multinomial model, but...
- What about orthography and morphology?
- What about unseen words?

Problems ① Unseen words

with ~~MLE~~ count & normalize ② Not expoliting linguistic structure

Linear Score for POS Tagging Model

$\Omega = T \times V$ $\quad T$ - tag $\quad V$ - vocabulary

Linear score: $\text{score}(y|x) = \sum_{i=1}^{d} w_i\, g_i(x,y)$

$g_{jk}(x,y) = I(x = \text{dog},\ y = VB)$

$I(y = PN,\ x = \text{capitalized})$

$I(y = VB,\ x \text{ ends in ing})$

Probability model:

$$p(x,y) = \frac{\exp(\text{score}(y|x))}{\sum_{x,y} \exp(\text{score}(y|x))}$$

Estimation

We observe $(x_1, y_1), \ldots, (x_n, y_n)$

What is the likelihood?

$$f(x_1, y_1, \ldots, x_n, y_n \mid w) = \prod_{i=1}^{n} p(x_i, y_i \mid w) =$$

$$= \prod_{i=1}^{n} \frac{\exp\left(\sum_{j=1}^{d} w_j g_j(x_i, y_i)\right)}{Z(w)}$$

$$Z(w) = \sum_{x,y} \exp\left(\sum_{j=1}^{d} w_j g_j(x, y)\right)$$

Estimation

$$Z(w) = \sum_{x,y} \exp\left(w^\top g(x,y)\right)$$

$$f(w|x_1, y_1, \ldots, x_n, y_n) = \prod_{i=1}^{n} \frac{\exp\left(w^\top g(x,y)\right)}{Z(w)}$$

What is the log-likelihood?

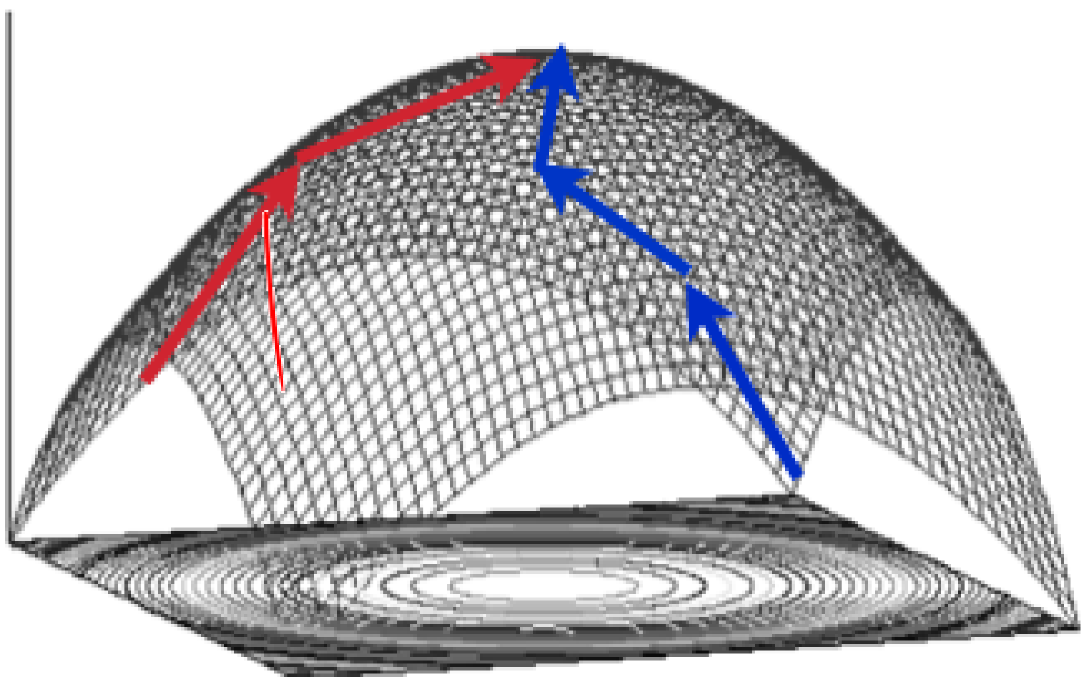
$$L(w|x_1, y_1, \ldots, x_n, y_n) = \frac{1}{n} \sum_{i=1}^{n} \log\left(\frac{\exp\left(w^\top g(x_i,y_i)\right)}{Z(w)}\right) =$$

$$= \frac{1}{n} \sum_{i=1}^{n} \left(w^\top g(x_i, y_i) - \log Z(w)\right) =$$

$$= \left[\frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{d} w_j g_j(x_i, y_i)\right] - \log Z(w)$$

Maximising the log-likelihood

$w^* = \arg\max_w L(w|x_1, y_1, \ldots, x_n, y_n)$



Maximising the log-likelihood

Many of the maximisation algorithms are a variant of the update:

$$w^{(t+1)} \leftarrow w^{(t)} + \mu v$$

where $v \in \mathbb{R}^d$ and $v_i = \dfrac{\partial L}{\partial w_i}\left(w^{(t)}\right)$.

Estimation

$f(x) = c$
$f'(x) = 0$

$f(x) = ax \Rightarrow$
$\Rightarrow f'(x) = a$

What is the average log-likelihood?

$$L(w|x_1, y_1, \ldots, x_n, y_n) = \frac{1}{n}\sum_{i=1}^{n}\left(\boxed{\sum_{j=1}^{d} w_j g_j(x_i, y_i)} - \log Z(w)\right)$$

$\log f(x)$
$= \frac{1}{f(x)} \cdot f'(x)$

What is the derivative?

$$\frac{\partial L}{\partial w_j} = \frac{1}{n}\sum_{i=1}^{n} 0 + 0 + 0 + \ldots + \underbrace{g_j(x_i, y_i)}_{j} + 0 \ldots + 0$$

$$\underbrace{- \frac{1}{Z(w)} \cdot \frac{\partial Z}{\partial w_j}}_{\text{chain rule}} = \left[\frac{1}{n}\sum_{i=1}^{n} g_j(x_i, y_i)\right] - \frac{1}{Z(w)} \cdot \frac{\partial Z}{\partial w_j}$$

Derivative of $Z(w)$

$[\exp(f(x))]' =$

$= f'(x) \cdot \exp(f(x))$

$$Z(w) = \sum_{x,y} \exp\left(\underbrace{\sum_{j=1}^{d} w_j g_j(x,y)}\right)$$

$$\frac{\partial Z}{\partial w_j}(w) = \sum_{x,y} \exp\left(\sum_{j=1}^{d} w_j g_j(x,y)\right) \cdot g_j(x,y)$$

Gradient of average log-likelihood

$$\frac{\partial L}{\partial w_j} = \left(\frac{1}{n} \sum_{i=1}^{n} g_j(x_i, y_i) \right) - \sum_{x,y} \frac{\exp(\sum_{k=1}^{d} w_k g_k(x,y))}{Z(w)} g_j(x,y)$$

$$\frac{1}{n} \sum_{i=1}^{n} g_j(x_i, y_i) = E_{\tilde{p}}[g_j] \quad \text{where} \quad \tilde{p}(x,y) = \frac{\text{count}(x,y)}{n}$$

$$= \sum_{x,y} \tilde{p}(x,y)\, g_j(x,y)$$

$$\sum_{x,y} \underbrace{\boxed{\frac{\exp(\sum_{k=1}^{d} w_k g_k(x,y))}{Z(w)}}}_{p(x,y|w)} g_j(x,y) = \sum_{x,y} p(x,y|w)\, g_j(x,y) =$$

$$= E_{p(\cdot|w)}[g_j(x,y)]$$

Gradient of average log-likelihood

$$\frac{\partial L}{\partial w_j} = \left(\frac{1}{n} \sum_{i=1}^{n} g_j(x_i, y_i) \right) - \sum_{x,y} \frac{\exp(\sum_{k=1}^{d} w_k g_k(x,y))}{Z(w)} g_j(x,y)$$

$$\frac{1}{n} \sum_{i=1}^{n} g_j(x_i, y_i) =$$

$$\sum_{x,y} \frac{\exp(\sum_{k=1}^{d} w_k g_k(x,y))}{Z(w)} g_j(x,y) =$$

Therefore, the gradient is the difference between empirical expectations and expectations under the model