Topics in Natural Language Processing

Shay Cohen

Institute for Language, Cognition and Computation

University of Edinburgh

Lecture 8

Administrativia

- The schedule for presentations and brief responses was sent to everybody
- One slot with three presenters
- Please plan to come at 1pm to this lecture
- Neural networks were quite popular
- Presentations: coordinate with me
- Regarding the essay: start thinking about it

Last class

Log-linear models. $p(x, y \mid w) = \dfrac{\exp\left(\sum_{i=1}^{d} w_i \, g_i(x,y)\right)}{Z(w)}$

$$Z(w) = \sum_{x,y} \exp\left(\sum_{i=1}^{d} w_i \, g_i(x,y)\right)$$

$g_i(x,y) =$ x is dog
y is PN

"overlap"

$g_i(x,y) =$ x ends in -ion
y is NN

$g_i(x,y) =$ x ends 'organisation'
y is NN

Gradient:

$$\frac{\partial L}{\partial w_j} = E_{\tilde{p}}[g_j] - E_{p(\cdot \mid w)}[g_j]$$

Log-likelihood maximisation tries to have the model feature expectations and the empirical distribution feature expectations "agree"

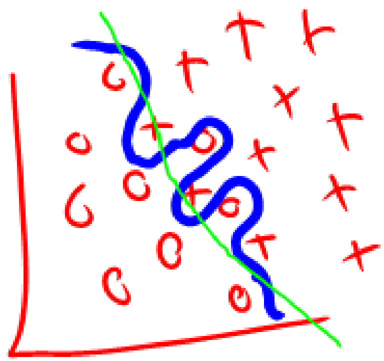
Overfitting

The advantage of log-linear models: can have arbitrary features

The problem: too many features lead to overfitting

Regularisation

What is overfitting?



better generalization

overfitting

L_2 Regularisation

$\boxed{||x_p||_p = \left(\sum x_i^p\right)^{\frac{1}{p}}}$ $\quad x \in \mathbb{R}^d \rightarrow ||x||_p \in \mathbb{R}$

$1 \leq p \leq \infty$

$\boxed{L_p \text{ norm}}$

New objective:

$$G(w|x_1, y_1, \ldots, x_n, y_n) = L(w|x_1, y_1, \ldots, x_n, y_n) - \boxed{\frac{\lambda}{2}||w||_2^2}$$

where $||w||_2^2 = \boxed{\sum_{i=1}^{d} w_i^2}$

Partial derivatives:

$$\frac{\partial G}{\partial w_j} = \frac{\partial L}{\partial w_j} - \frac{\lambda}{2} \cdot 2 \cdot w_j =$$

$$= \frac{\partial L}{\partial w_j} - \lambda w_j$$

$$\boxed{\begin{array}{c} f(x) = x^2 \\ \Downarrow \\ f'(x) = 2x \end{array}}$$

L_1 Regularisation

$p(w_i) \propto \exp(-\lambda|w_i|)$

Laplacian distribution

New objective:

$$G(w|x_1, y_1, \ldots, x_n, y_n) = L(w|x_1, y_1, \ldots, x_n, y_n) - \lambda||w||_1^2$$

where $||w||_1^2 = \sum_{i=1}^{d} |w_i|$

Encourages sparse solutions, such that most of w_i are exactly 0

"feature selection"

Bayesian interpretation to regularlisation

$\arg\max_w \log p(x|w) + \log p(w)$

$$G(w|x_1, y_1, \ldots, x_n, y_n) = L(w|x_1, y_1, \ldots, x_n, y_n) - \frac{\lambda}{2}||w||_2^2$$

Could the answer be a MAP estimate for some prior?

$$G(w|x_1, y_1, \ldots, x_n, y_n) \propto \log p(x_1, y_1, \ldots, x_n, y_n|w) + \log p(w)$$

$$p(w) \propto \exp\left(-\frac{\lambda}{2}\sum_{i=1}^{d} w_i^2\right) = \exp\left(-\frac{\sum_{i=1}^{d} w_i^2}{2 \times \frac{1}{\lambda}}\right)$$

Bayesian interpretation to regularlisation

$$G(w|x_1, y_1, \ldots, x_n, y_n) = L(w|x_1, y_1, \ldots, x_n, y_n) - \frac{\lambda}{2}||w||_2^2$$

Could the answer be a MAP estimate for some prior?

$$G(w|x_1, y_1, \ldots, x_n, y_n) \propto \log p(x_1, y_1, \ldots, x_n, y_n|w) + \log p(w)$$

likelihood

Gaussian

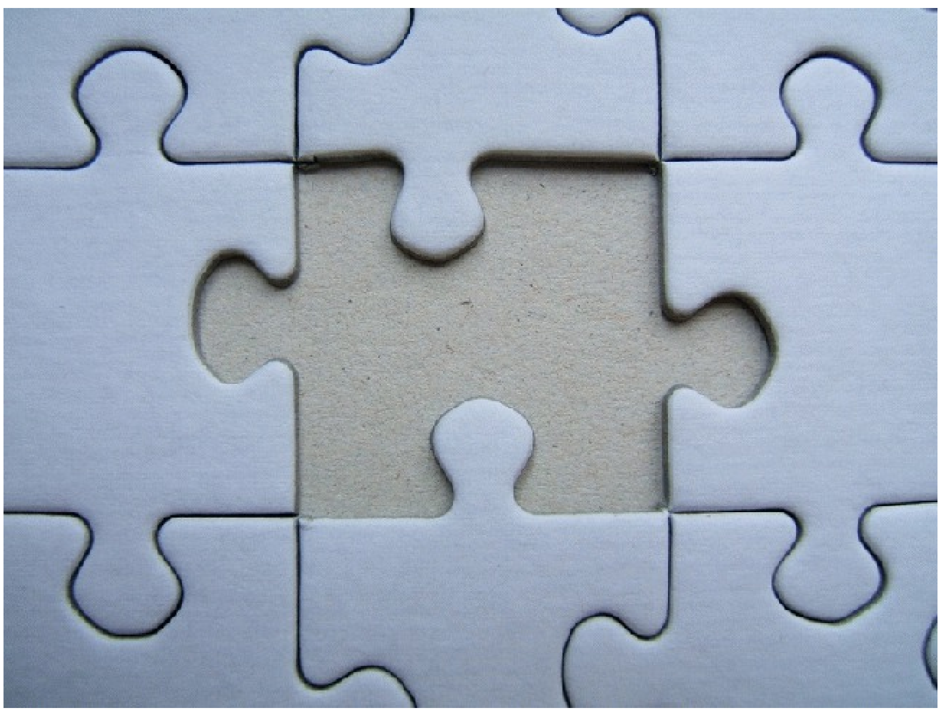
$p(w) \propto$

This means that $p(w)$ is a Gaussian distribution with mean 0 and variance $1/\lambda$

MLE with L_2-regularisation is MAP estimate with Gaussian prior

Today's class

Learning from incomplete data



Learning from Incomplete Data

- Semi-supervised learning

  Small amounts of labelled data and "large" amounts of unlabelled data.

- Latent variable learning

  Add extra information to the model

- Unsupervised learning

  Hard: Given only input examples, learn a decoder

How to estimate a PCFG?

We learned how to estimate a PCFG from treebank

Reminder:

count and normalize

Unsupervised learning: PCFGs

How to estimate a PCFG from strings?

(Assuming we have grammar)

General case: Viterbi (or “hard”) EM

Model: $p(x, y \mid \theta)$ (this means the grammar structure/rules are known)

Observed Data: $x_1, \ldots x_n$

Step 0:

Guess some θ

Step 1:

Parse $x_1 \ldots x_n$ using $\theta \rightarrow y_1 \ldots y_n$

Step 2: Estimate θ using $x_1 \ldots x_n$, $y_1 \ldots y_n$

Repeat step 1

Maximum likelihood estimation

General principle: write down the likelihood of **whatever** you observe, and then maximise with respect to parameters

Model: $p(x, y \mid \theta)$

Observed: $x_1, \ldots, x_n$

Likelihood: $L(x_1 \ldots x_n \mid \theta)$

$$L(x_1, \ldots, x_n \mid \theta) = \prod_{i=1}^{n} p(x_i \mid \theta) = \prod_{i=1}^{n} \left(\sum_{y} p(x_i, y \mid \theta) \right)$$

$$\max_{\theta} \sum_{i=1}^{n} \log \left(\sum_{y} p(x_i, y \mid \theta) \right)$$

The EM Algorithm

- A softer version of hard EM
- Instead of identifying a single tree per sentence, identify a distribution over trees (E-step)
- Then re-estimate the parameters, with each tree for each sentence “voting” according to its probability (M-step)
- Semiring parsing: instead of CKY use the inside algorithm

EM: Main Disadvantage

Sensitivity to initialisation (finds local maximum)

Global log-likelihood optimisation in general is “hard”

Latent-variable learning

"Structure" is present

Some information is missing from model

Model: $p(x, y, h \mid \theta)$

Observed: $(x_1, y_1), \ldots, (x_n, y_n)$

Log-likelihood:

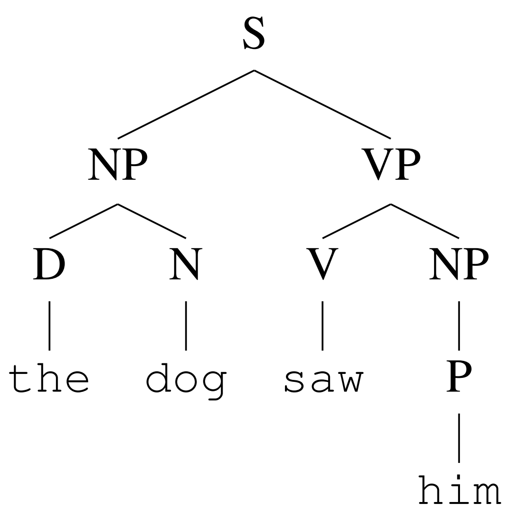
$$\underline{L(x_1, \ldots, x_n, y_1, \ldots, y_n|\theta)} = \prod_{i=1}^{n} p(x_i, y_i | \theta) =$$

$$= \prod_{i=1}^{n} \sum_{h} p(x_i, y_i, h | \theta)$$

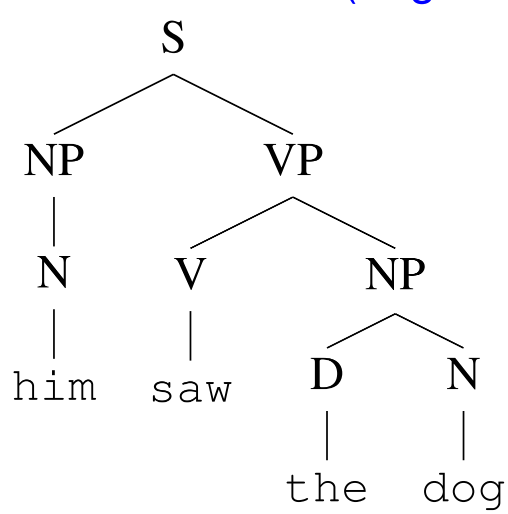
Example of Latent-Variable Use in PCFGs

“Context-freeness” can lead to over-generalization:

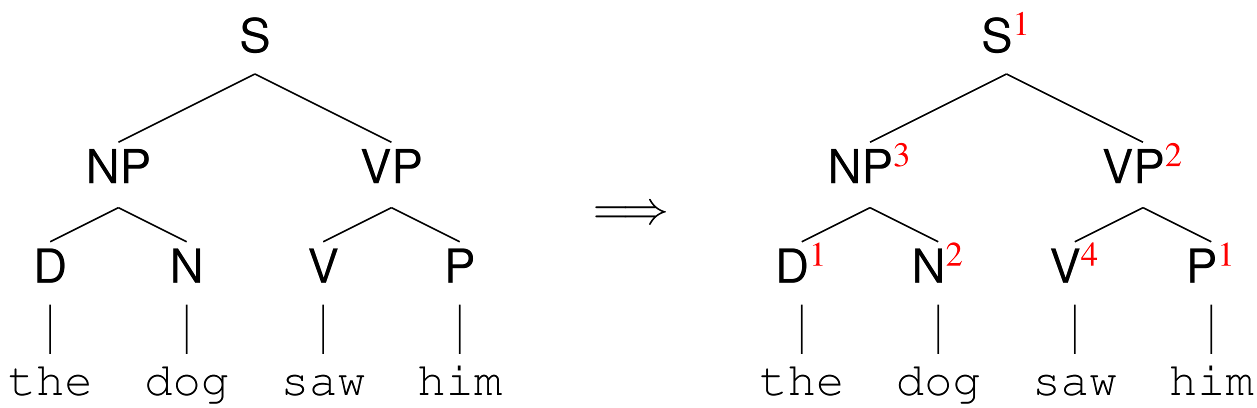
Seen in data:



Unseen in data (ungrammatical):



Latent-Variable PCFGs



The **latent states** for each node are never observed

How to learn with latent variables?

- Expectation-Maximisation (EM)
- Current surging interest: method of moments and spectral learning
- Revival of old methods: Neural networks
- Other methods