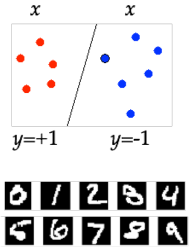
Structure Learning

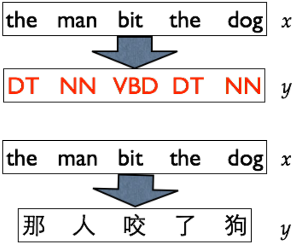
Jordan Boyd-Graber
University of Colorado Boulder
8. DECEMBER 2014

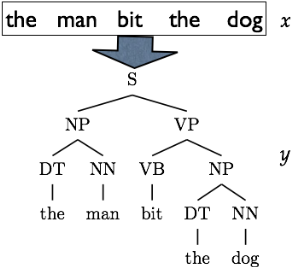
Slides adapted from Liang Huang

Roadmap

- Structured learning
- Alternative to generative models


x
x
y=+1
y=-1
0 1 2 3 4
5 6 7 8 9


the man bit the dog
x
DT NN VBD DT NN
y
the man bit the dog
x
那 人 咬 了 狗
y

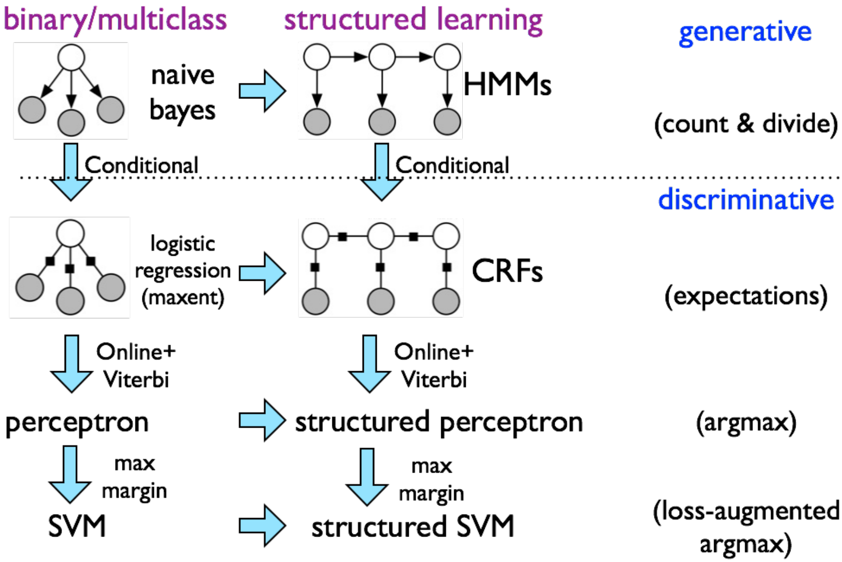

the man bit the dog
x
S
NP
VP
DT
NN
VB
NP
the
man
bit
DT
NN
the
dog
y


小心滑落
Slip carefully

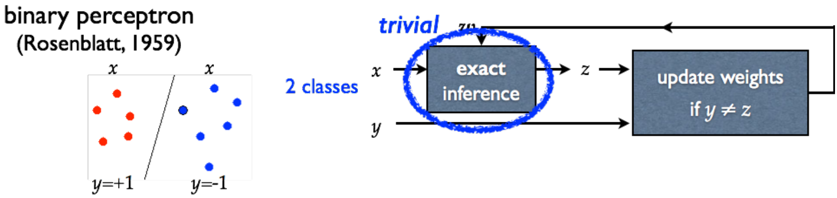

小心溺水
CAREFUL DROWNING


小心碰头
LOOKOUTKNOCKHEAD
Archived at www.ChineseEnglish.com


小心滑倒
Slip and fall down carefully

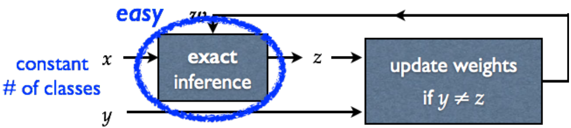

binary/multiclass
structured learning
generative
naive bayes
HMMs
(count & divide)
Conditional
Conditional
discriminative
logistic regression (maxent)
CRFs
(expectations)
Online+ Viterbi
Online+ Viterbi
perceptron
structured perceptron
(argmax)
max margin
max margin
SVM
structured SVM
(loss-augmented argmax)

Binary to Structure

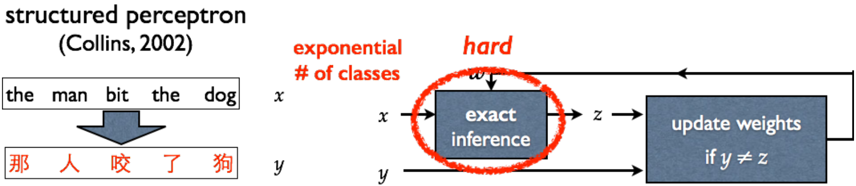


Binary to Structure

multiclass perceptron
(Freund/Schapire, 1999)



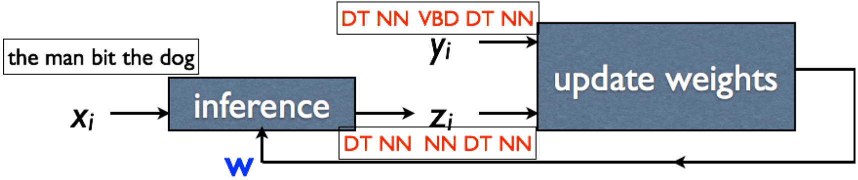
Binary to Structure



Generic Perceptron

- perceptron is the simplest machine learning algorithm
- online-learning: one example at a time
- learning by doing
  - find the best output under the current weights
  - update weights at mistakes

Structured Perceptron



Perceptron Algorithm

Inputs: Training set (x_i, y_i) for $i = 1 \ldots n$

Initialization: $\mathbf{W} = 0$

Define: $F(x) = \operatorname{argmax}_{y \in \mathbf{GEN}(x)} \mathbf{\Phi}(x, y) \cdot \mathbf{W}$

Algorithm: For $t = 1 \ldots T, i = 1 \ldots n$

$z_i = F(x_i)$

If $(z_i \neq y_i)$ $\quad \mathbf{W} \longleftarrow \mathbf{W} + \mathbf{\Phi}(x_i, y_i) - \mathbf{\Phi}(x_i, z_i)$

Output: Parameters $\mathbf{W}$

POS Example

- gold-standard: DT NN VBD DT NN y
  - the man bit the dog x

$\Phi(x, y)$

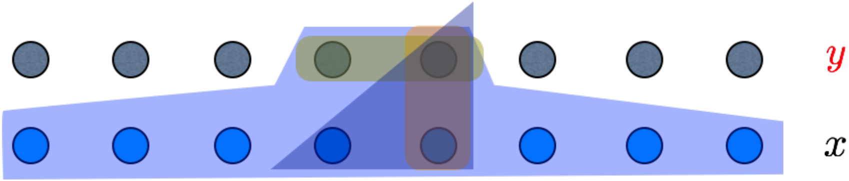
- current output: DT NN NN DT NN z
  - the man bit the dog x

$\Phi(x, z)$

- assume only two feature classes
  - tag bigrams t_{i-1} t_i
  - word/tag pairs w_i
- weights ++: (NN,VBD) (VBD, DT) (VBD $\rightarrow$ bit)
- weights −−: (NN, NN) (NN, DT) (NN $\rightarrow$ bit)

What must be true?

- Finding highest scoring structure must be really fast (you'll do it often)
- Requires some sort of dynamic programming algorithm
- For tagging: features must be local to y (but can be global to x)



Averaging is Good

Inputs: Training set (x_i, y_i) for $i = 1 \dots n$

Initialization: $\mathbf{W}_0 = 0$

Define: $F(x) = \text{argmax}_{y \in \mathbf{GEN}(x)} \mathbf{\Phi}(x, y) \cdot \mathbf{W}$

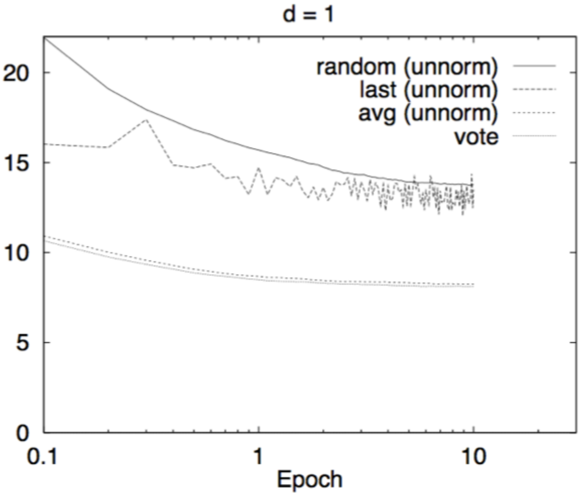
Algorithm: For $t = 1 \dots T, i = 1 \dots n$

$z_i = F(x_i)$

If $(z_i \neq y_i)$ $\mathbf{W}_{j+1} \leftarrow \mathbf{W}_j + \mathbf{\Phi}(x_i, y_i) - \mathbf{\Phi}(x_i, z_i)$

Output: Parameters $\mathbf{W} = \sum_j \mathbf{W}_j$

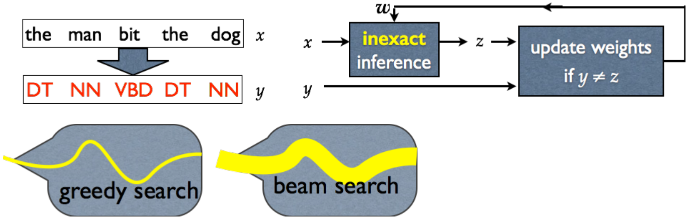
Averaging is Good



Smoothing

- Must include subset templates for features
- For example, if you have feature (t_0, w_0, w_{-1}), you must also have
  - (t_0, w_0); (t_0, w_{-1}); (w_0, w_{-1})

Inexact Search?



- Sometimes search is too hard
- So we use beam search instead
- How to create algorithms that respect this relaxation: track when right answer falls off the beam

Structured Learning

- Sometimes you want discriminative method for complex y
- Learning those models are difficult
- Need to be scalable