Structure Learning

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

Slides adapted from Liang Huang
Roadmap

- Structured learning
- Alternative to generative models
the man bit the dog

structure learning

the man bit the dog

the man bit the dog

那个人咬了狗
Slip carefully

CAREFUL DROWNING

LOOKOUT KNOCKHEAD

Slip and fall down carefully
binary/multiclass | structured learning
---|---
naive bayes | HMMs
Conditional | Conditional
logistic regression (maxent) | CRFs
Online+ Viterbi | Online+ Viterbi
perceptron | structured perceptron
max margin | max margin
SVM | structured SVM

generative
(count & divide)
discriminative
(expectations)
argmax
(loss-augmented argmax)
Binary to Structure

binary perceptron
(Rosenblatt, 1959)

2 classes

trivial

exact inference

update weights if $y \neq z$
multiclass perceptron (Freund/Schapire, 1999)
Binary to Structure

structured perceptron (Collins, 2002)

the man bit the dog

那 人 咬 了 狗

exponential # of classes

exact inference

update weights if y ≠ z

hard

update weights if y ≠ z
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

\[ X_i \xrightarrow{\text{inference}} Z_i \xrightarrow{\text{DT NN NN DT NN}} y_i \xrightarrow{\text{DT NN VBD DT NN}} \]

the man bit the dog
Perceptron Algorithm

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

**Initialization:** \(W = 0\)

**Define:** \(F(x) = \arg\max_{y \in \text{GEN}(x)} \Phi(x, y) \cdot W\)

**Algorithm:** For \(t = 1 \ldots T, i = 1 \ldots n\)

\[z_i = F(x_i)\]

If \((z_i \neq y_i)\)

\[W \leftarrow W + \Phi(x_i, y_i) - \Phi(x_i, z_i)\]

**Output:** Parameters \(W\)
POS Example

- gold-standard:  DT  NN  VBD  DT  NN  y  \( \Phi(x, y) \)
- the  man  bit  the  dog  x

- current output:  DT  NN  NN  DT  NN  z  \( \Phi(x, z) \)
- the  man  bit  the  dog  x

- 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 \(i = 1 \ldots n\)

Initialization: \(W_0 = 0\)

Define: 
\[ F(x) = \arg\max_{y \in \text{GEN}(x)} \Phi(x, y) \cdot W \]

Algorithm: For \(t = 1 \ldots T, i = 1 \ldots n\)
\[
\begin{align*}
z_i &= F(x_i) \\
\text{If } (z_i \neq y_i) & \quad W_{j+1} \leftarrow W_j + \Phi(x_i, y_i) - \Phi(x_i, z_i)
\end{align*}
\]

Output: Parameters \(W = \sum_j W_j\)
Averaging is Good

![Graph showing the effect of averaging in the context of structure learning. The graph compares 'random (unnorm)', 'last (unnorm)', 'avg (unnorm)', and 'vote' methods over different epochs, demonstrating the benefits of averaging.]
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