Making Predictions with Dynamic Programming

SPFLODD

October 4, 2011
Inference

- Eventually, you need to run your structured predictor on test data!
- Till now we have focused on learning mostly.
- For sequence labeling and segmentation models with local interactions, decoding was pretty much always the Viterbi algorithm.
- Variations:
  - forward-backward
  - cost-augmented decoding
Notation for Linear Models

- Training data: \( \{(x_1, y_1), (x_2, y_2), ..., (x_N, y_N)\} \)
- Testing data: \( \{(x_{N+1}, y_{N+1}), ..., (x_{N+N'}, y_{N+N'})\} \)
- Feature function: \( g \)
- Weights: \( w \)
- Decoding:
  \[
  \text{decode}(w, x) = \arg \max_y w^\top g(x, y)
  \]
- Learning:
  \[
  \text{learn} \left( \left\{ (x_i, y_i) \right\}_{i=1}^{N} \right) = \arg \max_w \Phi \left( w, \left\{ (x_i, y_i) \right\}_{i=1}^{N} \right)
  \]
- Evaluation:
  \[
  \frac{1}{N'} \sum_{i=1}^{N'} \text{cost} \left( \text{decode} \left( \text{learn} \left( \left\{ (x_i, y_i) \right\}_{i=1}^{N} \right), x_{N+i}, y_{N+i} \right) \right)
  \]
Probabilistic Inference Problems

Given values for some random variables \((X \subset V)\) ...

- **Most Probable Explanation**: what are the *most probable* values of the *rest* of the r.v.s \(V \setminus X\)?

(More generally ...)

- **Maximum A Posteriori (MAP)**: what are the most probable values of *some* other r.v.s, \(Y \subset (V \setminus X)\)?

- Random *sampling* from the posterior over values of \(Y\)
- Full *posterior* over values of \(Y\)
- *Marginal* probabilities from the posterior over \(Y\)

- **Minimum Bayes risk**: What is the \(Y\) with the lowest expected cost?
- **Cost-augmented decoding**: What is the most *dangerous* value of \(Y\), compared to true \(y^*\)?

These do not need to be probabilistic! Change “most probable” to “maximum scoring.”

*Different kinds of decoding.
Approaches to Inference

- **exact**
  - variable elimination
  - ILP
  - dynamic program’ng
  - MCMC
  - Gibbs

- **approximate**
  - randomized
    - importance sampling
    - randomized search
  - variational
    - simulated annealing
    - mean field
  - deterministic
    - loopy belief propagation
    - LP relaxations
    - dual decomp.
    - local search
    - beam search

- today
Dynamic Programming

• Examples you’ve seen:
  – Viterbi, forward, backward algorithms for HMMs
  – Minimum-cost edit distance
  – Dijkstra’s shortest path

• Can we define it more broadly?
General Views of Decoding

• For HMMs, the decoding algorithm we usually think of first is the Viterbi algorithm.
  – This is just one example.

• We will view decoding in a few different ways.
  – Sequence models as a running example.
  – These views are not just for HMMs.
  – Sometimes they will lead us back to Viterbi!
Different Views of Decoding
1. Probabilistic Graphical Models

- View the linguistic structure as a collection of random variables that are interdependent.
- Represent interdependencies as a directed or undirected graphical model.
- Conditional probability tables (BNs) or factors (MNs) encode the probability distribution.
Inference in Graphical Models

• General algorithm for exact MAP inference: variable elimination.
  – Iteratively solve for the best values of each variable conditioned on values of “preceding” neighbors.
  – Then trace back.

The Viterbi algorithm is an instance of max-product variable elimination!
Hidden Markov Model

• $X$ and $Y$ are both sequences of symbols
  – $X$ is a sequence from the vocabulary $\Sigma$
  – $Y$ is a sequence from the state space $\Lambda$

\[
p(X = x, Y = y) = \left( \prod_{i=1}^{n} p(x_i \mid y_i)p(y_i \mid y_{i-1}) \right) p(\text{stop} \mid y_n)\]

• Parameters:
  – Transitions $p(y' \mid y)$
    • including $p(\text{stop} \mid y), p(y \mid \text{start})$
  – Emissions $p(x \mid y)$
Hidden Markov Model

• The joint model’s independence assumptions are easy to capture with a Bayesian network.

\[ p(X = x, Y = y) = \left( \prod_{i=1}^{n} p(x_i | y_i)p(y_i | y_{i-1}) \right) p(stop | y_n) \]
Hidden Markov Model

- The usual inference problem (decoding) is to find the most probable value of $Y$ given $X = x$.
  - Instantiates “most probable explanation” inference and “maximum a posteriori” inference.

\[
\begin{align*}
Y_0 & \rightarrow Y_1 & \rightarrow Y_2 & \rightarrow Y_3 & \rightarrow \cdots & \rightarrow Y_n & \rightarrow \text{stop} \\
X_1 = x_1 & \rightarrow X_2 = x_2 & \rightarrow X_3 = x_3 & \rightarrow \cdots & \rightarrow X_n = x_n
\end{align*}
\]
A Graphical Models View of Viterbi

1. Convert to a factor graph.
2. Reduce factors to respect observation at each position.
3. Do variable elimination from left to right.
   - Calculates the best prefix-path scores into each state at each position.
4. Trace back to recover the labels.
Hidden Markov Model

• The usual inference problem is to find the most probable value of $Y$ given $X = x$.

• Factor graph:

\[
\begin{align*}
Y_0 & \quad Y_1 & \quad Y_2 & \quad Y_3 & \quad \ldots & \quad Y_n & \quad \text{stop} \\
X_1 = x_1 & \quad X_2 = x_2 & \quad X_3 = x_3 & \quad & \quad & \quad X_n = x_n
\end{align*}
\]
Hidden Markov Model

• The usual inference problem is to find the most probable value of $Y$ given $X = x$.

• Factor graph after reducing factors to respect evidence:
Hidden Markov Model

• The usual inference problem is to find the most probable value of $Y$ given $X = x$.

• Clever ordering should be apparent!
Hidden Markov Model

• When we eliminate $Y_1$, we take a product of three relevant factors.
  • $p(Y_1 \mid \text{start})$
  • $\eta(Y_1) = \text{reduced } p(x_1 \mid Y_1)$
  • $p(Y_2 \mid Y_1)$
Hidden Markov Model

• When we eliminate $Y_1$, we first take a product of two factors that only involve $Y_1$.
Hidden Markov Model

• When we eliminate $Y_1$, we first take a product of two factors that only involve $Y_1$.

• This is the Viterbi probability vector for $Y_1$. 
Hidden Markov Model

- When we eliminate \(Y_1\), we first take a product of two factors that only involve \(Y_1\).
- This is the Viterbi probability vector for \(Y_1\).
- Eliminating \(Y_1\) equates to solving the Viterbi probabilities for \(Y_2\).
Hidden Markov Model

• Product of all factors involving $Y_1$, then reduce.
  • $\Phi_2(Y_2) = \max_{y \in \text{Val}(Y_1)} (\Phi_1(y) \times p(Y_2 \mid y))$
  • This factor holds Viterbi probabilities for $Y_2$. 
Hidden Markov Model

- When we eliminate $Y_2$, we take a product of the analogous two relevant factors.
- Then reduce.
  - $\phi_3(Y_3) = \max_{y \in \text{Val}(Y_2)} (\phi_2(y) \times p(Y_3 | y))$
Hidden Markov Model

• At the end, we have one final factor with one row, $\phi_{n+1}$.
• This is the score of the best sequence.
• Use backtrace to recover values.
Why Think This Way?

• Easy to see how to generalize HMMs.
  – More evidence
  – More factors
  – More hidden structure
  – More dependencies

• Probabilistic interpretation of factors is *not* central to finding the “best” $Y$ ...
  – Many factors are not conditional probability tables.
Generalization Example 1

- Each word also depends on previous state.
Generalization Example 2

- “Trigram” HMM
Generalization Example 3

- Aggregate bigram model (Saul and Pereira, 1997)
General Decoding Problem

• Two structured random variables, $X$ and $Y$.
  – Sometimes described as \textit{collections} of random variables.

• “Decode” observed value $X = x$ into some value of $Y$.

• Usually, we seek to maximize some score.
  – Might be probability, might not.
MAP = Optimizing a Linear Score

• Bayesian network:
  \[ \sum_i \log p(x_i \mid \text{parents}(X_i)) \]
  \[ + \sum_j \log p(y_j \mid \text{parents}(Y_j)) \]

• Markov network:
  \[ \sum_C \log \phi_C (\{x_i\}_{i \in C}, \{y_j\}_{j \in C}) \]

• This only works if every variable is in \( X \) or \( Y \).
Inference in Graphical Models

• Remember: more edges make inference more expensive.
  – Fewer edges means stronger independence.

• Really pleasant:
Inference in Graphical Models

• Remember: more edges make inference more expensive.
  – Fewer edges means stronger independence.

• Really unpleasant:
2. Weighted Parsing
Grammars

• Grammars are often associated with natural language parsing, but they are extremely powerful for imposing constraints.

• We can add weights to them.
  – HMMs are a kind of weighted regular grammar (closely connected to WFSAs)
  – PCFGs are a kind of weighted CFG
  – Many, many more.

• Weighted parsing: find the maximum-weighted derivation for a string $x$. 
Decoding as Weighted Parsing

• Every valid \( y \) is a grammatical derivation (parse) for \( x \).
  – HMM: sequence of “grammatical” states is one allowed by the transition table.

• Augment parsing algorithms with weights and find the best parse.

The Viterbi algorithm is an instance of recognition by a weighted grammar!
BIO Tagging as a CFG

\[
\begin{align*}
N & \rightarrow B \ R_B \\
N & \rightarrow O \ R_O \\
R_B & \rightarrow B \ R_B \\
R_B & \rightarrow O \ R_O \\
R_B & \rightarrow I \ R_I \\
R_B & \rightarrow \epsilon \\
R_I & \rightarrow B \ R_B \\
R_I & \rightarrow O \ R_O \\
R_I & \rightarrow I \ R_I \\
R_I & \rightarrow \epsilon \\
R_O & \rightarrow B \ R_B \\
R_O & \rightarrow O \ R_O \\
R_O & \rightarrow I \ R_I \\
R_O & \rightarrow \epsilon \\
O & \rightarrow \epsilon
\end{align*}
\]

\[
\forall x \in \Sigma, \quad B \rightarrow x \\
I \rightarrow x \\
O \rightarrow x
\]

- Weighted (or probabilistic) CKY is a dynamic programming algorithm very similar in structure to classical CKY.
3. Paths and Hyperpaths
Best Path

• General idea: take $x$ and build a graph.
• Score of a path factors into the edges.

$$\arg\max_y w^T g(x, y) = \arg\max_y w^T \sum_{e \in \text{Edges}} f(e) 1\{e \text{ is crossed by } y\}'s \text{ path}$$

• Decoding is finding the best path.

The Viterbi algorithm is an instance of finding a best path!
“Lattice” View of Viterbi
Minimum Cost Hyperpath

• General idea: take $x$ and build a hypergraph.
• Score of a hyperpath factors into the hyperedges.
• Decoding is finding the best hyperpath.

• This connection was elucidated by Klein and Manning (2002).
Parsing as a Hypergraph
Parsing as a Hypergraph

cf. “Dean for democracy”
Forced to work on his thesis, sunshine streaming in the window, Mike experienced a ...
Forced to work on his thesis, sunshine streaming in the window, Mike began to...
Why Hypergraphs?

• Useful, compact encoding of the hypothesis space.
  – Build hypothesis space using local features, maybe do some filtering.
  – Pass it off to another module for more fine-grained scoring with richer or more expensive features.
4. Weighted Logic Programming
Logic Programming

• Start with a set of **axioms** and a set of **inference rules**.

\[
\forall A, C, \quad \text{ancestor}(A, C) \iff \text{parent}(A, C)
\]

\[
\forall A, C, \quad \text{ancestor}(A, C) \iff \bigvee_{B} \text{ancestor}(A, B) \land \text{parent}(B, C)
\]

• The goal is to prove a specific theorem, goal.

• Many approaches, but we assume a **deductive** approach.
  – Start with axioms, iteratively produce more theorems.
\[ \forall \ell \in \Lambda, \quad \nu(\ell, 1) = \text{labeled-word}(x_1, \ell) \]

\[ \forall \ell \in \Lambda, \quad \nu(\ell, i) = \bigvee_{\ell' \in \Lambda} \nu(\ell', i - 1) \land \text{label-bigram}(\ell', \ell) \land \text{labeled-word}(x_i, \ell) \]

\[ \text{goal} = \bigvee_{\ell \in \Lambda} \nu(\ell, n) \]
Weighted Logic Programming

• Twist: axioms have weights.

• Want the proof of goal with the best score:

\[
\arg \max_y w^\top g(x, y) = \arg \max_y w^\top \sum_{a \in \text{Axioms}} f(a) \cdot \text{freq}(a; y)
\]

• Note that axioms can be used more than once in a proof (y).
Whence WLP?

- Shieber, Schabes, and Pereira (1995): many parsing algorithms can be understood in the same deductive logic framework.
Dynamic Programming

• Break a problem into slightly smaller problems with optimal substructure.
  – Best path to v depends on best paths to all u such that (u,v) ∈ E.
• Overlapping subproblems: each subproblem gets used repeatedly, and there aren’t too many of them.
• Three main strategies for DP:
  – Memoization
  – Agenda (Dijkstra’s algorithm, A*)
• Things to remember in general:
  – The hypergraph may too big to represent explicitly; exhaustive calculation may be too expensive.
  – The hypergraph may or may not have properties that make “clever” orderings possible.
  – DP does not imply polynomial time and space! Most common approximations when the desired state space is too big: beam search, cube pruning, agendas with early stopping, ...
Summary

• Decoding is the general problem of choosing a complex structure.
  – Linguistic analysis, machine translation, speech recognition, ...
  – Statistical models are usually involved (not necessarily probabilistic).
  – Often a subroutine when learning; almost always when evaluating.

• No perfect general view, but much can be gained through a combination of views.