

HMM Parameter Estimation: the Baum-Welch algorithm

PA154 Jazykové modelování (6.1)

Pavel Rychlý

pary@fi.muni.cz

March 23, 2020

Source: Introduction to Natural Language Processing (600.465)
Jan Hajič, CS Dept., Johns Hopkins Univ.
www.cs.jhu.edu/~hajic

HMM: The Tasks

- HMM(the general case):
 - ▶ five-tuple (S, S_0, Y, P_S, P_Y) , where:
 - ▶ $S = \{s_1, s_2, \dots, s_T\}$ is the set of states, S_0 is the initial state,
 - ▶ $Y = \{y_1, y_2, \dots, y_Y\}$ is the output alphabet,
 - ▶ $P_S(s_j|s_i)$ is the set of prob. distributions of transitions,
 - ▶ $P_Y(y_k|s_i, s_j)$ is the set of output (emission) probability distributions.
- Given an HMM & an output sequence $Y = \{y_1, y_2, \dots, y_k\}$:
 - ▶ (Task 1) compute the probability of Y ;
 - ▶ (Task 2) compute the most likely sequence of states which has generated Y
 - ▶ (Task 3) Estimating the parameters (transition/output distributions)

A variant of Expectation–Maximization

- Idea(≈EM, for another variant see LM smoothing (lect. 3)):
 - ▶ Start with (possibly random) estimates of P_S and P_Y .
 - ▶ Compute (fractional) “counts” of state transitions/emissions taken, from P_S and P_Y , given data Y
 - ▶ Adjust the estimates of P_S and P_Y from these “counts” (using MLE, i.e. relative frequency as the estimate).
- Remarks:
 - ▶ many more parameters than the simple four-way smoothing
 - ▶ no proofs here; see Jelinek Chapter 9

Setting

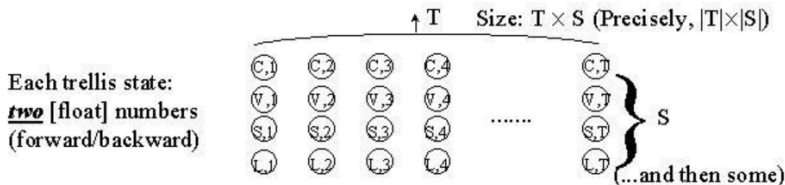
- HMM (without P_S, P_Y)(S, S_0, Y), and data $T = \{y_i \in Y\}_{i=1 \dots |T|}$
 - ▶ will use $T \sim |T|$
- HMM structure is given: (S, S_0)
- P_S : Typically, one wants to allow "fully connected" graph
 - ▶ (i.e. no transitions forbidden \sim no transitions set to hard 0)
 - ▶ why? \rightarrow we better leave it on the learning phase, based on the data!
 - ▶ sometimes possible to remove some transitions ahead of time
- P_Y : should be restricted (if not, we will not get anywhere!)
 - ▶ restricted \sim hard 0 probabilities of $p(y|s, s')$
 - ▶ "Dictionary": states \leftrightarrow words, "m:n" mapping on $S \times Y$ (in general)

Initialization

- For computing the initial expected “counts”
- Important part
 - ▶ EM guaranteed to find a *local* maximum only (albeit a good one in most cases)
- P_Y initialization more important
 - ▶ fortunately, often easy to determine
 - ▶ together with dictionary \leftrightarrow vocabulary mapping, get counts, then MLE
- P_S initialization less important
 - ▶ e.g. uniform distribution for each $p(\cdot|s)$

Data structures

- Will need storage for:
 - ▶ The predetermined structure of the HMM (unless fully connected → need not to keep it!)
 - ▶ The parameters to be estimated (P_S, P_Y)
 - ▶ The expected counts (same size as (P_S, P_Y))
 - ▶ The training data $T = \{y_i \in Y\}_{i=1\dots T}$
 - ▶ The trellis (if f.c.):



The Algorithm Part I

- 1 Initialize P_S, P_Y
- 2 Compute "forward" probabilities:
 - ▶ follow the procedure for trellis (summing), compute $\alpha(s, i)$ everywhere
 - ▶ use the current values of $P_S, P_Y(p(s'|s), p(y_i|s, s'))$:
$$\alpha(s', i) = \sum_{s \rightarrow s'} \alpha(s, i-1) \times p(s'|s) \times p(y_i|s, s')$$
 - ▶ NB: do not throw away the previous stage!
- 3 Compute "backward" probabilities
 - ▶ start at all nodes of the last stage, proceed backwards, $\beta(s, i)$
 - ▶ i.e., probability of the "tail" of data from stage i to the end of data
$$\beta(s', i) = \sum_{s' \leftarrow s} \beta(s, i+1) \times p(s|s') \times p(y_{i+1}|s', s)$$
 - ▶ also, keep the $\beta(s, i)$ at all trellis states

The Algorithm Part II

1 Collect counts:

- ▶ for each output/transition pair compute

$$c(y, s, s') = \sum_{i=0}^{k-1, y=y_{i+1}} \alpha(s, i) \underbrace{p(s'|s) p(y_{i+1}|s, s')}_{\text{this transition prob} \times \text{output prob}} \beta(s', i+1)$$

Annotations for the equation above:
- $\alpha(s, i)$: prefix prob.
- $\beta(s', i+1)$: tail prob.
- $\sum_{i=0}^{k-1, y=y_{i+1}}$: one pass through data, only stop at (output) y

$$c(s, s') = \sum_{y \in Y} c(y, s, s') \quad (\text{assuming all observed } y_i \text{ in } Y)$$
$$c(s) = \sum_{s' \in S} c(s, s')$$

- 2 Reestimate: $p'(s'|s) = c(s, s')/c(s)$ $p'(y|s, s') = c(y, s, s')/c(s, s')$
- 3 Repeat 2-5 until desired convergence limit is reached

Baum-Welch: Tips & Tricks

- Normalization badly needed
 - ▶ long training data \rightarrow extremely small probabilities

- Normalize α, β using the same norm.factor:

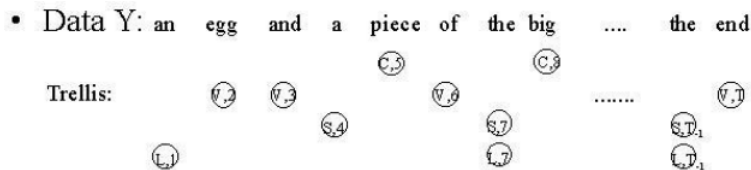
$$N(i) = \sum_{s \in S} \alpha(s, i)$$

as follows:

- ▶ compute $\alpha(s, i)$ as usual (Step 2 of the algorithm), computing the sum $N(i)$ at the given stage i as you go.
- ▶ at the end of each stage, recompute all *alphas*(for each state s):
$$\alpha^*(s, i) = \alpha(s, i) / N(i)$$
- ▶ use the same $N(i)$ for β s at the end of each backward (Step 3) stage:
$$\beta^*(s, i) = \beta(s, i) / N(i)$$

Example

- Task: pronunciation of "the"
- Solution: build HMM, fully connected, 4 states:
 - ▶ S - short article, L - long article, C,V - word starting w/consonant, vowel
 - ▶ thus, only "the" is ambiguous (a, an, the - not members of C,V)
- Output form states only ($p(w|s, s') = p(w|s')$)



Example: Initialization

- Output probabilities:

- ▶ $p_{init}(w|c) = c(c, w)/c(c)$; where $c(S, the) = c(L, the) = c(the)/2$
(other than that, everything is deterministic)

- Transition probabilities:

- ▶ $p_{init}(c'|c) = 1/4$ (uniform)

- Don't forget:

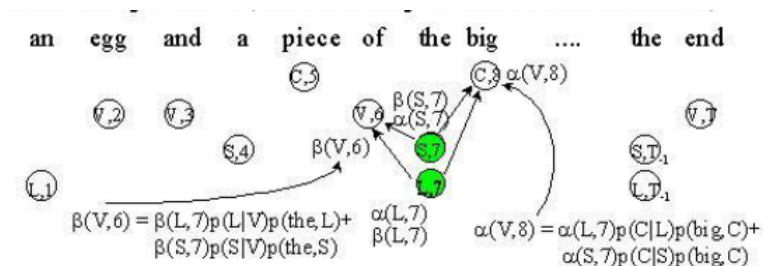
- ▶ about the space needed
- ▶ initialize $\alpha(X, 0) = 1$ (X : the never-occurring front buffer st.)
- ▶ initialize $\beta(s, T) = 1$ for all s (except for $s = X$)

Fill in alpha, beta

- Left to right, alpha:

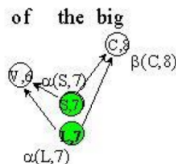
$\alpha(s', i) = \sum_{s \rightarrow s'} \alpha(s, i-1) \times p(s'|s) \times p(w_i|s')$, where s' is the output from states

- Remember normalization (N(i)).
- Similarly, beta (on the way back from the end).



Counts & Reestimation

- One pass through data
- At each position i , go through all pairs (s_i, s_{i+1})
- Increment appropriate counters by frac. counts (Step 4):
 - ▶ $\text{inc}(y_{i+1}, s_i, s_{i+1}) = a(s_i, i)p(s_{i+1}|s_i)p(y_{i+1}|s_{i+1})b(s_{i+1}, i+1)$
 - ▶ $c(y, s_i, s_{i+1}) += \text{inc}$ (for y at pos $i+1$)
 - ▶ $c(s_i, s_{i+1}) += \text{inc}$ (always)
 - ▶ $c(s_i) += \text{inc}$ (always)
- $\text{inc}(\text{big}, L, C) = \alpha(L, 7)p(C|L)p(\text{big}, C)\beta(V, 8)$
- $\text{inc}(\text{big}, S, C) = \alpha(S, 7)p(C|S)p(\text{big}, C)\beta(V, 8)$
- Reestimate $p(s'|s), p(y|s)$
 - ▶ and hope for increase in $p(C|S)$ and $p(V|L) \dots !!$



- Parameter "tying"
 - ▶ keep certain parameters same (\sim just one "counter" for all of them)
 - ▶ any combination in principle possible
 - ▶ ex.: smoothing (just one set of lambdas)
- Real Numbers Output
 - ▶ Y of infinite size (R, R^n)
 - ▶ parametric (typically: few) distribution needed (e.g., "Gaussian")
- "Empty" transitions: do not generate output
 - ▶ \sim vertical areas in trellis; do not use in "counting"