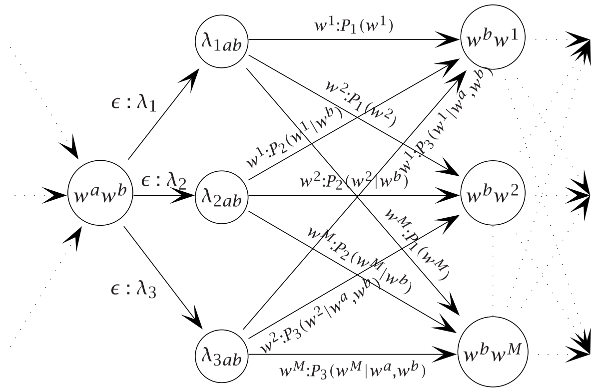
Some of an HMM for an interpolated language model



HMM Training: Baum-Welch reestimation

- Used to automatically estimate parameters of an HMM
- A case of the Expectation Maximization (EM) algorithm
- One starts with initial probability estimates
- One computes expectations of how often each transition is used
- One re-estimates the probabilities based on those expectations
- ... and repeat until convergence

HMM Training: Baum-Welch reestimation

- Needed because the state paths are hidden, and the equations cannot be solved analytically
- Provides a maximum likelihood estimate: attempts to find the model that assigns the training data the highest likelihood
- Hill-climbing algorithm that can get stuck in local maxima
- Not always so effective for POS tagging (the ML re-estimation procedure doesn't know the meaning we have given to the hidden states)

The third problem: Parameter estimation = Parameter learning

- We want the most likely model parameters given the data (using MLE):

$$\arg\max_{\mu} P(O_{\text{training}}|\mu)$$

- This would let us learn model probabilities from raw data
- Can't determine these probabilities analytically.
- Use iterative hill-climbing algorithm: the Baum-Welch or Forward-Backward algorithm – a special case of the EM algorithm.

Intuition

- We don't know what the model is.
- But we can work out the probability of the observation sequence using some (perhaps random) model
- Looking at that calculation, we can see which state transitions and symbol emissions were probably used the most
- By increasing the probability of those, we can choose a revised model which gives a higher probability to the observation sequence

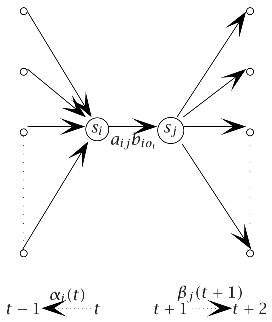
The likelihood of being in state i at time t

Using forward and backward variables, we can calculate $\gamma_i(t)$, the probability of being in state i at time t:

$$\begin{aligned} \gamma_i(t) &= P(X_t = i|O, \mu) \\ &= \frac{P(X_t = i, O|\mu)}{P(O|\mu)} \\ &= \frac{\alpha_i(t)\beta_i(t)}{\sum_{j=1}^{N} \alpha_j(t)\beta_j(t)} \end{aligned}$$

Chance of moving from state i to j at time t

Define $p_t(i,j)$, the probability of traversing an arc $i - j$ at time t given observations O.



Computing probability of traversing arc

$$
\begin{aligned}
p_t(i,j) &= P(X_t = i, X_{t+1} = j | O, \mu) \\
&= \frac{P(X_t=i, X_{t+1}=j, O|\mu)}{P(O|\mu)} \\
&= \frac{\alpha_i(t) a_{ij} b_{io_t} \beta_j(t+1)}{\sum_{m=1}^{N} \alpha_m(t)\beta_m(t)} \\
&= \frac{\alpha_i(t) a_{ij} b_{io_t} \beta_j(t+1)}{\sum_{m=1}^{N}\sum_{n=1}^{N} \alpha_m(t) a_{mn} b_{mo_t} \beta_n(t+1)}
\end{aligned}
$$

Note that $\gamma_i(t) = \sum_{j=1}^{N} p_t(i,j)$.

Expectations

Now, if we sum over the time index, this gives us expectations (counts):

$\sum_{t=1}^{T} \gamma_i(t)$ = expected number of transitions from state i in O

$\sum_{t=1}^{T} p_t(i,j)$ = expected number of transitions from state i to j in O

Parameter reestimation formulae

$$
\begin{aligned}
\hat{\pi}_i &= \text{expected frequency in state } i \text{ at time } t = 1 \\
&= \gamma_i(1) \\
\hat{a}_{ij} &= \frac{\text{expected num. transitions from state } i \text{ to } j}{\text{expected num. transitions from state } i} \\
&= \frac{\sum_{t=1}^{T} p_t(i,j)}{\sum_{t=1}^{T} \gamma_i(t)} \\
\hat{b}_{ik} &= \frac{\text{expected num. times } k \text{ observed in state } i}{\text{expected num. transitions from } i} \\
&= \frac{\sum_{\{t: o_t = k, 1 \le t \le T\}} \gamma_i(t)}{\sum_{t=1}^{T} \gamma_i(t)}
\end{aligned}
$$

Baum-Welch training algorithm

- Begin with some model μ (perhaps random, perhaps preselected)
- Run O through the current model to estimate the expectations of each model parameter
- Change the model to maximize the values of the paths that are used a lot (while still respecting the stochastic constraints)
- Repeat, hoping to converge on optimal values for the model parameters μ.

We're guaranteed to get no worse

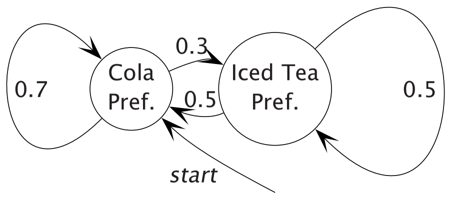
From $\mu = (A, B, \Pi)$, one iteration derives $\hat{\mu} = (\hat{A}, \hat{B}, \hat{\Pi})$.

Baum initially proved for HMMs, and the EM framework in general gives us that:

$$P(O|\hat{\mu}) \ge P(O|\mu)$$

But we may stay in a local optimum.

HMMs: The crazy soft drink machine



Output probability given state

	cola	iced tea (ice_t)	lemonade (lem)
CP	0.6	0.1	0.3
IP	0.1	0.7	0.2

Always starts off in cola-preferring state.

Values for $p_t(i, j)$ on O = (lem, ice_t, cola):

Time (and j)	**1**	CP	IP	γ_1
i	CP	0.3	0.7	1.0
	IP	0.0	0.0	0.0
	2	CP	IP	γ_2
i	CP	0.28	0.02	0.3
	IP	0.6	0.1	0.7
	3	CP	IP	γ_3
i	CP	0.616	0.264	0.88
	IP	0.06	0.06	0.12

The parameters will be reestimated as follows by EM:

		Original			Reestimated		
Π	CP	1.0			1.0		
	IP	0.0			0.0		
		CP	IP		CP	IP	
A	CP	0.7	0.3		0.548	0.451	
	IP	0.5	0.5		0.804	0.195	
		cola	ice_t	lem	cola	ice_t	lem
B	CP	0.6	0.1	0.3	0.403	0.137	0.458
	IP	0.1	0.7	0.2	0.136	0.853	0.0