WestlakeNLP

Natural Language Processing

Yue Zhang
Westlake University

西湖大學
WESTLAKE UNIVERSITY

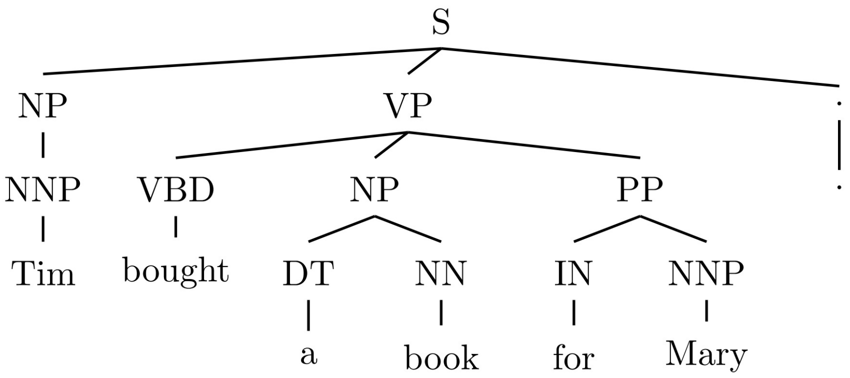
Chapter 7

Generative Sequence Labeling

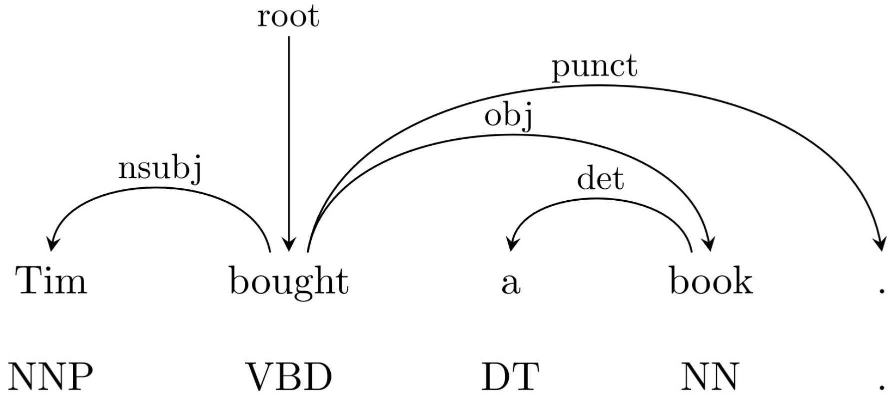
Contents

Contents

Structure Prediction



(a) An example constituent tree.



(b) An example dependency tree.

Task	Input	Output
Morphological Analysis	(English) *walking*	*walk + ing*
	(Arabic) *wktAbnA*	*w + ktAb + nA*
	(German) *Wochenarbeitszeit*	*Wochen + arbeits + zeit*
Tokenisation	*Mr. Smith visited Wendy's new house.*	*Mr. Smith visited Wendy 's new house .*
Word segmentation	其中国外企业	其中 国外 企业
	中国外企业务	中国 外企 业务
	はきものを脱ぐ	はきものを 脱ぐ
	きものを着る	きものを 着る
POS Tagging	I can open this can	PRP MD VB DT NN

- Output inter-dependency

Sequence Labelling

- Part-of-Speech tagging as example
- Input is a sentence $s = W_{1:n} = w_1 w_2 \dots w_n$
- Output is a sequence of POS tags $T_{1:n} = t_1 t_2 \dots t_n$

Sentence	POS tagging sequence
Jame went to the shop yesterday .	NNP VBD TO NN NN .
What would you like to eat ?	WP MD PRP VB TO VB .
Tim is talking with Mary .	NNP VBZ VBG IN NNP .
I really appreciate it .	PRP RB VBP PRP .
John is a famous athlete .	NNP VBZ DT JJ NN .

NNP：Proper Noun VBD：Verb, past tense TO: to NN: Noun

Sequence Labelling

- Part-of-Speech tagging as example
- Input is a sentence $s = W_{1:n} = w_1 w_2 \dots w_n$
- Output is a sequence of POS tags $T_{1:n} = t_1 t_2 \dots t_n$

James went to the shop yesterday .
NNP VBD TO DT NN NN .

James|NNP went|VBD to|TO the|DT park|NN yesterday|NN .|.

NNP：Proper Noun VBD：Verb, past tense TO: to NN: Noun

Local Model

- Treat the assignment of each POS tag as a separate classification task.
- Features : five-word window $[w_{i-2}, w_{i-1}, w_i, w_{i+1}, w_{i+2}] \Longrightarrow t_i$

  e.g. James went to the shop $\Longrightarrow TO$
- Model: Naive Bayes and discriminative classifiers (e.g., SVM, perceptron, log-linear models)

Local Model

- Treat the assignment of each POS tag as a separate classification task.
- Features : five-word window $[w_{i-2}, w_{i-1}, w_i, w_{i+1}, w_{i+2}] \Longrightarrow t_i$

  e.g. James went to the shop $\Longrightarrow TO$
- Model: Naive Bayes and discriminative classifiers (e.g., SVM, perceptron, log-linear models)
- Disadvantage: ignore dependencies between different output POS tags !

DT JJ NN

determiner → noun (NN) or adjective (JJ) , not verb (VB)

adverb (AD) → verb (VB) , not possessive pronoun (PRP$)

Contents

Model Target

- Model target:

$$P(T_{1:n}|W_{1:n})$$

$$w_i \in V, t_i \in L, i \in [1, \dots, n]$$

- Parameterization?

Structured Model

- Use the same techniques as for chapter 2:

  1. Use the Bayes rule:

$$P(T_{1:n}|W_{1:n}) = \frac{P(W_{1:n}|T_{1:n})P(T_{1:n})}{P(W_{1:n})}$$

$$\propto P(W_{1:n}|T_{1:n})P(T_{1:n})$$

$$= P(W_{1:n}, T_{1:n})$$

Structured Model

- Use the same techniques as for chapter 2:

  2. Applying the probability chain rule for $P(T_{1:n})$:

$$P(T_{1:n}) = P(t_1)P(t_2|t_1)P(t_3|t_1t_2) \dots P(t_{n-1}|t_1 \dots t_{n-2})P(t_n|t_1 \dots t_{n-1})$$

(chain rule)

Structured Model

- Use the same techniques as for chapter 2:

  2. Applying the probability chain rule for $P(T_{1:n})$:

  $$P(T_{1:n}) = P(t_1)P(t_2|t_1)P(t_3|t_1t_2) \dots P(t_{n-1}|t_1 \dots t_{n-2})P(t_n|t_1 \dots t_{n-1})$$

  (chain rule)

  3. Make independence assumptions for $P(T_{1:n})$

     - First-order Markov assumption:

     $$P(T_{1:n}) \approx P(t_1)P(t_2|t_1) \dots P(t_n|t_{n-1})$$

     - Second-order Markov assumptions:

     $$P(T_{1:n}) \approx P(t_1)P(t_2|t_1)P(t_3|t_1t_2) \dots P(t_{n-1}|t_{n-3}t_{n-2})P(t_n|t_{n-2}t_{n-1})$$

Structured Model

- Use the same techniques for $P(W_{1:n}|T_{1:n})$:

  1. Use the Bayes rule:

$$P(T_{1:n}|W_{1:n}) = \frac{P(W_{1:n}|T_{1:n})P(T_{1:n})}{P(W_{1:n})}$$

$$\propto P(W_{1:n}|T_{1:n})P(T_{1:n})$$

Structured Model

- Use the same techniques for $P(W_{1:n}|T_{1:n})$:

2. Applying the probability chain rule for $P(W_{1:n}|T_{1:n})$:

$$P(\mathrm{W}_{1:\mathrm{n}}|T_{1:n}) = P(w_1|T_{1:n})P(w_2|w_1,T_{1:n})P(w_3|w_{1:2},T_{1:n}) \dots P(w_n|w_{1:\mathrm{n}-1},T_{1:n})$$

Structured Model

- Use the same techniques for $P(W_{1:n}|T_{1:n})$:

2. Applying the probability chain rule for $P(W_{1:n}|T_{1:n})$:

$$P(\mathrm{W}_{1:\mathrm{n}}|T_{1:n}) = P(w_1|T_{1:n})P(w_2|w_1, T_{1:n})P(w_3|w_{1:2}, T_{1:n}) \dots P(w_n|w_{1:\mathrm{n}-1}, T_{1:n})$$

3. Make independence assumptions for $P(W_{1:n}|T_{1:n})$

$$P(W_{1:n}|T_{1:n}) = P(\mathrm{w}_1|t_1)P(w_2|t_2) \dots P(w_n|t_n)$$

Structured Model

- Generate Stories

$$P(T_{1:n}|W_{1:n}) \propto P(W_{1:n}|T_{1:n})P(T_{1:n})$$

  - Generate tags

  First-order

$$P(T_{1:n}) \approx P(t_1)P(t_2|t_1)\ldots P(t_n|t_{n-1})$$

  Second-order

$$P(T_{1:n}) \approx P(t_1)P(t_2|t_1)P(t_3|t_1t_2)\ldots P(t_{n-1}|t_{n-3}t_{n-2})P(t_n|t_{n-2}t_{n-1})$$

  - Generate words

$$P(W_{1:n}|T_{1:n}) \approx P(w_1|t_1)P(w_2|t_2)\ldots P(w_n|t_n) \text{ (independence assumption)}$$

Structured Model

- Summary
  - First-order Model:

$$P(T_{1:n}|W_{1:n}) \propto P(W_{1:n}, T_{1:n}) = P(T_{1:n})P(W_{1:n}|T_{1:n})$$

$$\approx \prod_{i=1}^{n} P\,(t_i|t_{i-1}) \cdot \prod_{i=1}^{n} P\,(w_i|t_i)$$

$$= \prod_{i=1}^{n} P\,(t_i|t_{i-1}) \cdot P(w_i|t_i)$$

  - Second-order Model:

$$P(T_{1:n}|W_{1:n}) \propto P(W_{1:n}, T_{1:n}) = P(T_{1:n})P(W_{1:n}|T_{1:n})$$

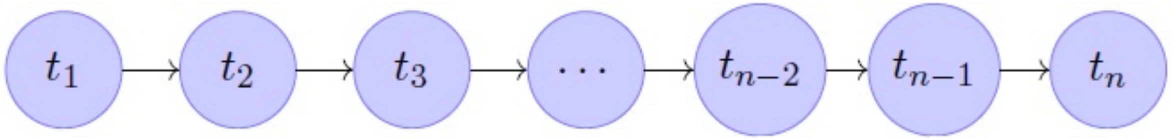
$$\approx \prod_{i=1}^{n} P\,(t_i|t_{i-2}\,t_{i-1}) \cdot \prod_{i=1}^{n} P\,(w_i|t_i).$$

$$= \prod_{i=1}^{n} P\,(t_i|t_{i-2}\,t_{i-1}) \cdot P(w_i|t_i).$$

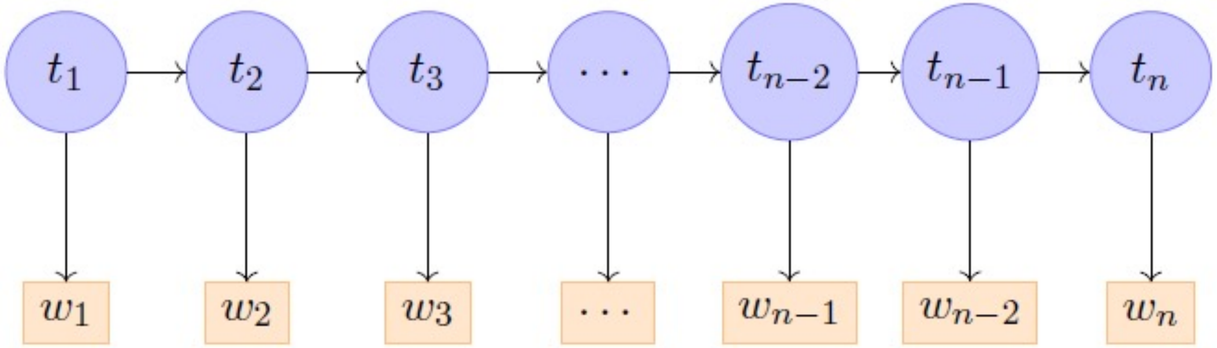
Structured Model

- A generative model. Use first-order HMM as an example.

- First generate **tags** such as "NNP (proper noun) VBZ (verb third-person singular) NN(noun)"

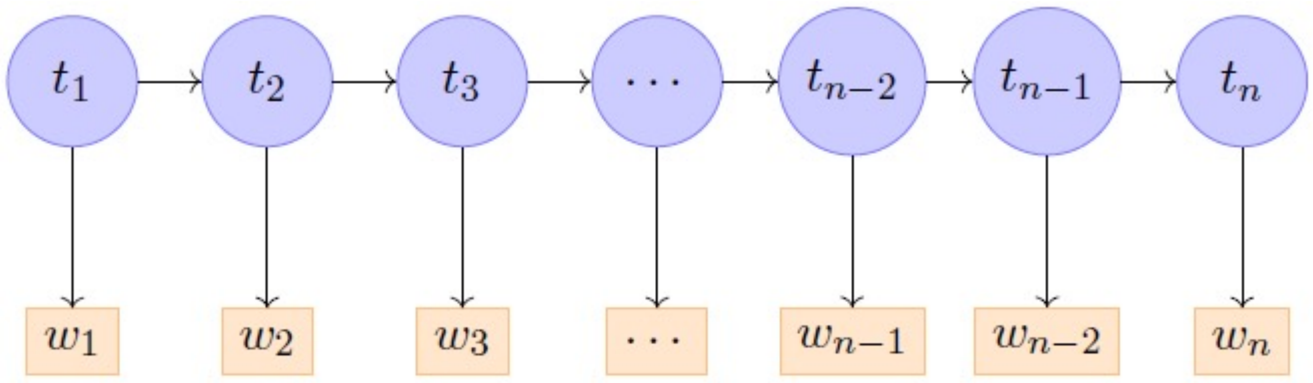


- Then filling the **words** such as "Jim reads thrillers".



Structured Model

- First-order Model:



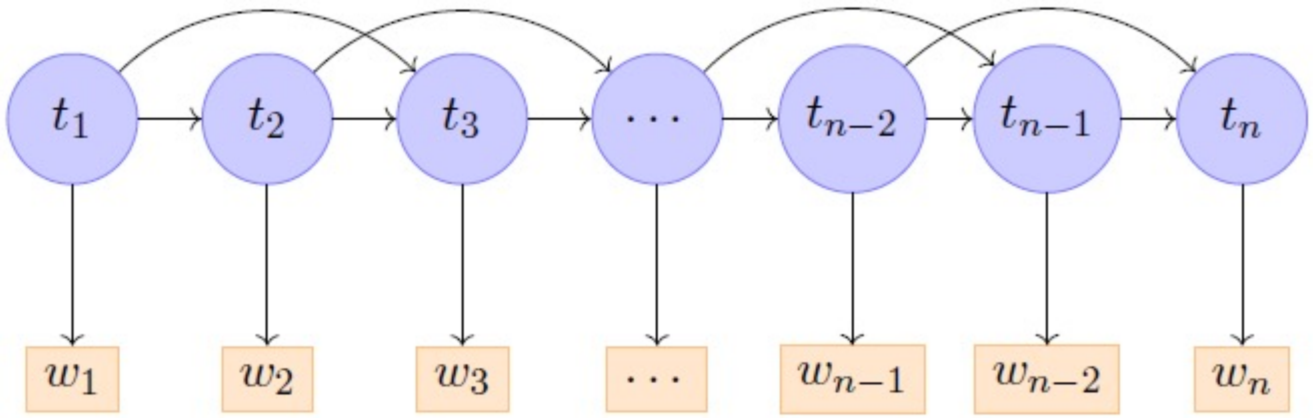
Emission probability:

$$P(w_i|t_i)$$

Transition probability:

$$P(t_i|t_{i-1}\dots t_{i-k})$$

- Second-order Model:



Contents

Hidden Markov Model

- Model parameterization

$P(T_{1:n}|W_{1:n}) \approx \prod_{i=1}^{n} P\,(t_i|t_{i-1}) \cdot \prod_{i=1}^{n} P\,(w_i|t_i\,)$ (first order)

$P(T_{1:n}|W_{1:n}) \approx \prod_{i=1}^{n} P\,(t_i|t_{i-2}\,t_{i-1}) \cdot \prod_{i=1}^{n} P\,(w_i|t_i)$ (second order)

- Parameters: $P(w_i|t_i), P(t_2|t_1)\ or\ P(t_3|t_1, \mathrm{t}_2)$
- Training (by using MLE)
  - emission probabilities can be estimated as:

$$P(w_{\mathrm{i}}|t_{\mathrm{i}}) = \frac{\#(w_{\mathrm{i}}t_{\mathrm{i}})}{\sum_w \#(t_{\mathrm{i}}\ w)},$$

  - transition probability can be estimated as:

$$P(t_2|t_1) = \frac{\#(t_1 t_2)}{\sum_t \#(t_1\ t)} \quad \text{or} \quad P(t_3|t_1 t_2) = \frac{\#(t_1 t_2 t_3)}{\sum_t \#(t_1 t_2\ t)}$$

Contents

Hidden Markov Model

- Classification --- **enumerate** tags.
- Structured prediction --- Decoding
  - **Enumerating** all tag sequences has exponential computational complexity, which is intractable.
  - **Dynamic programming** is possible for the decoding task.
  - Find the **optimal sub problem using first-order HMM** :

$$P(W_{1:n}, T_{1:n}) = \prod_{i=1}^{n} P\,(t_i | t_{i-1}) P(w_i | t_i)$$

Hidden Markov Model

$$P(W_{1:n}, T_{1:n}) = \prod_{i=1}^{n} P\,(t_i|t_{i-1})P(w_i|t_i)$$

$$P(W_{1:n}, T_{1:n}) = P(W_{1:\mathrm{n}-1}, T_{1:\mathrm{n}-1}) \cdot (P(t_n|t_{\mathrm{n}-1})P(w_n|t_n))$$

$$P(W_{1:n-1}, T_{1:n-1}) = P(W_{1:\mathrm{n}-2}, T_{1:\mathrm{n}-2}) \cdot (P(t_{n-1}|t_{\mathrm{n}-2})P(w_{n-1}|t_{n-1}))$$

…

$$P(W_{1:i}, T_{1:i}) = P(W_{1:i-1}, T_{1:i-1}) \cdot (P(t_i|t_{i-1})P(w_i|t_i))$$

…

$$P(W_{1:1}, T_{1:1}) = P(t_1|t_0)P(w_1|t_1)$$

Hidden Markov Model

- Denote

  $\hat{T}_{1:i}$ as the highest-scored tag sequence among $T_{1:i}$.

- Denote

  $T_{1:i}(t_i = t)$ as a tag sequence $T_{1:i}$ where $t_i = t$

  $\hat{T}_{1:i}(t_i = t)$ as the highest-scored tag sequence among $T_{1:i}$ where $t_i = t$

- Suppose that in $\hat{T}_{1:i}$, $\hat{t}_i = t$, and $\hat{\mathrm{t}}_{\mathrm{i-1}} = \mathrm{t}'$.

  $\hat{T}_{1:i-1}(t_{i-1} = t')$ must be the highest-scored among all $T_{1:i-1}(t_{i-1} = t')$

  (proof by contradiction)

Hidden Markov Model

- Solving the optimal sub-sequence problem:

$$\hat{T}_{1:i}(t_i = t) = \ argmax_{t' \in L} P(W_{1:i-1}, \hat{T}_{1:i-1}(t_{i-1} = t'))(P(t|t')P(w_i|t))$$

- Incrementally find $\hat{T}_{1:i}(t_i = t)$ for $i = 1,2, \dots, n$
- Maintain two tables
  - tb --- n columns, $|L|$ rows, storing $\hat{T}_{1:i}(t_i = t)$
  - bp --- n columns, $|L|$ rows, storing

$$\max_{\mathrm{t}' \in L} P(W_{1:i-1}, \hat{T}_{1:i-1}(t_{i-1} = t'))(P(t|t')P(w_i|t))$$

Hidden Markov Model

- An example (adding <B> in the beginning)

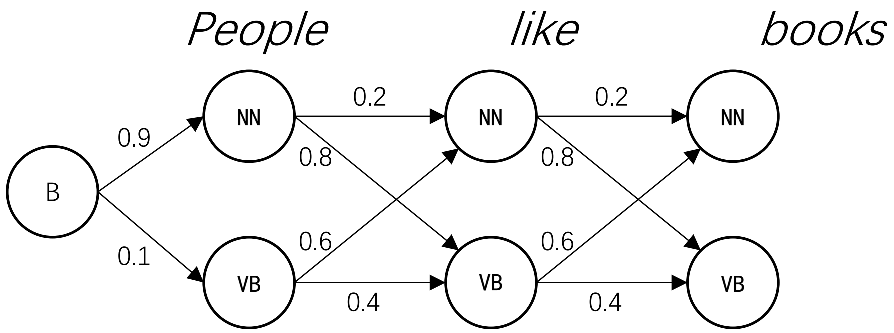
  Find the path with the highest probability.

tb table

	0	1	2	3
$\hat{T}_{1:i}(t_i = NN)$	1	0	0	0
$\hat{T}_{1:i}(t_i = VB)$	1	0	0	0

bp table

NUL L			
NUL L			



- Transition probabilities

  $P(NN|B) = 0.9$

  $P(VB|NN) = 0.8$

  …
- Omit emission probabilities

Hidden Markov Model

- An example (adding <B> in the beginning)

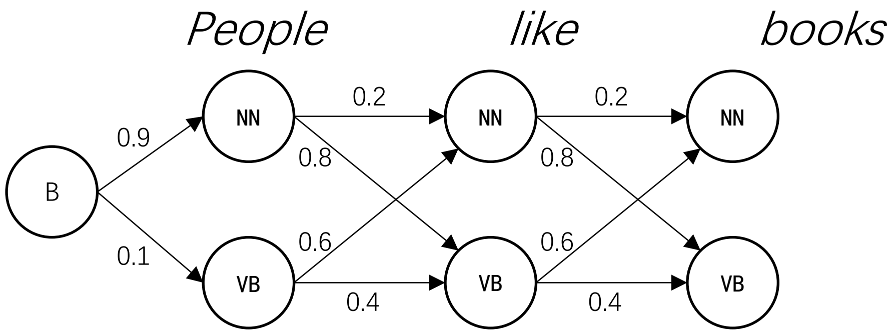
  Find the path with the highest probability.

tb table

1	0	0	0
1	0	0	0

bp table

NUL L			
NUL L			



1	0.9	0	0
1	0.1	0	0

NULL	B		
NULL	B		

$\hat{T}_{1:1}(t_i = NN)$=0.9

$\hat{T}_{1:1}(t_i = VB)$=0.1

Hidden Markov Model

- An example (adding <B> in the beginning)

  Find the path with the highest probability.

tb table

1	0	0	0
1	0	0	0

bp table

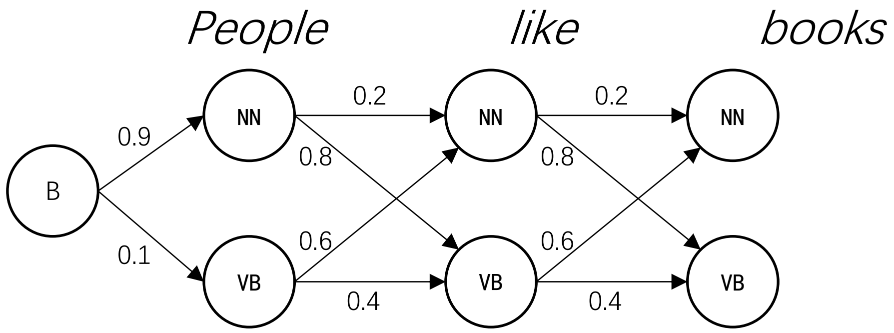
NUL L			
NUL L			

1	0.9	0	0
1	0.1	0	0

NUL L	B		
NUL L	B		

1	0.9	0.9*0.2 0.1*0.6	0
1	0.1	0.9*0.8 0.1*0.4	0

NUL L	B	NN	
NUL L	B	NN	



$\hat{T}_{1:2}(t_2 = NN)$=0.9*0.2=0.18 (NN NN > VB NN)

$\hat{T}_{1:2}(t_2 = VB)$=0.9*0.8=0.72 (NN VB > VB VB)

Hidden Markov Model

- An example (adding <B> in the beginning)

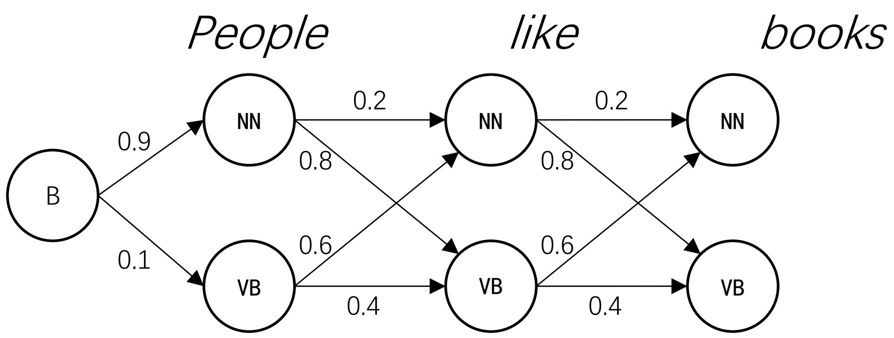
  Find the path with the highest probability.

tb table

1	0	0	0
1	0	0	0

bp table

NULL			
NULL			



1	0.9	0	0
1	0.1	0	0

NULL	B		
NULL	VB		

1	0.9	0.9*0.2 0.1*0.6	0
1	0.1	0.9*0.8 0.1*0.4	0

NULL	B	NN	
NULL	B	NN	

1	0.9	0.18	0.18*0.2 0.72*0.6
1	0.1	0.72	0.18*0.8 0.72*0.4

NULL	B	NN	VB
NULL	B	NN	VB

(NNN NN NN < NN VB NN)
(NN NN VB < NN VB VB)

Hidden Markov Model

- An example (adding <B> in the beginning)

Find the path with the highest probability.

tb table

1	0	0	0
1	0	0	0

bp table

NULL			
NULL			

1	0.9	0	0
1	0.1	0	0

NULL	B		
NULL	B		

1	0.9	0.9*0.2 0.1*0.6	0
1	0.1	0.9*0.8 0.1*0.4	0

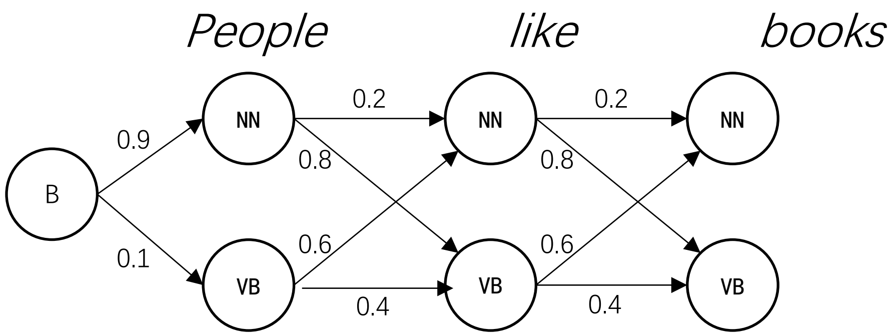
NULL	B	NN	
NULL	B	NN	

1	0.9	0.18	0.18*0.2 0.72*0.6
1	0.1	0.72	0.18*0.8 0.72*0.4

NULL	B	NN	VB
NULL	B	NN	VB

1	0.9	0.18	0.432
1	0.1	0.72	0.288

NULL	B	NN	VB
NULL	B	NN	VB



(NN VB NN > NN VB VB)

Hidden Markov Model

- An example (adding <B> in the beginning)

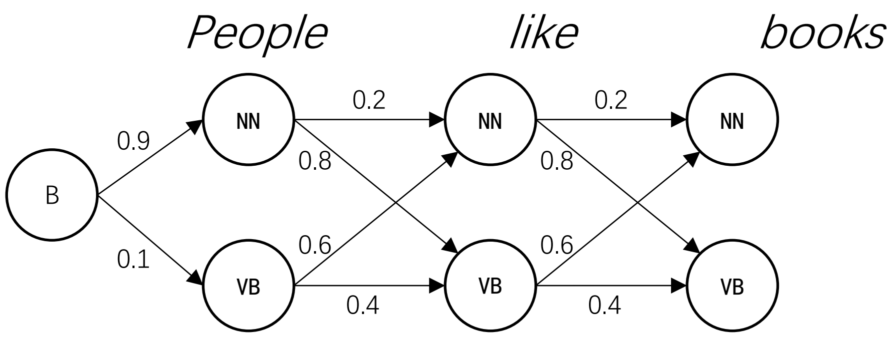
Find the path with the highest probability.

tb table

1	0	0	0
1	0	0	0

bp table

NULL			
NULL			



1	0.9	0	0
1	0.1	0	0

NULL	B		
NULL	B		

1	0.9	0.9*0.2 0.1*0.6	0
1	0.1	0.9*0.8 0.1*0.4	0

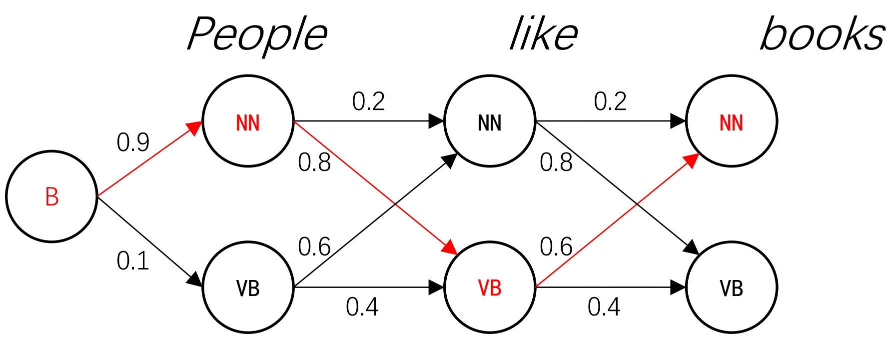
NULL	B	NN	
NULL	B	NN	

1	0.9	0.18	0.18*0.2 0.72*0.6
1	0.1	0.72	0.18*0.8 0.72*0.4

NULL	B	NN	VB
NULL	B	NN	VB

1	0.9	0.18	0.432
1	0.1	0.72	0.288

NULL	B	NN	VB
NULL	B	NN	VB



Hidden Markov Model

- An example (adding <B> in the beginning)

Find the path with the highest probability.

tb table

1	0	0	0
1	0	0	0

1	0.9	0	0
1	0.1	0	0

1	0.9	0.9*0.2 0.1*0.6	0
1	0.1	0.9*0.8 0.1*0.4	0

1	0.9	0.18	0.18*0.2 0.72*0.6
1	0.1	0.72	0.18*0.8 0.72*0.4

1	0.9	0.18	0.432
1	0.1	0.72	0.288

bp table

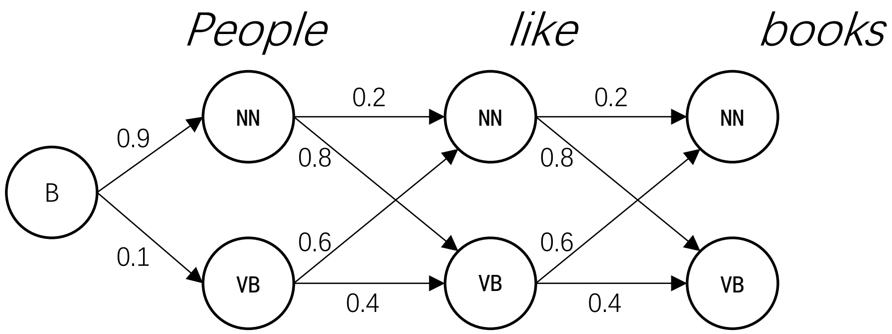
NULL			
NULL			

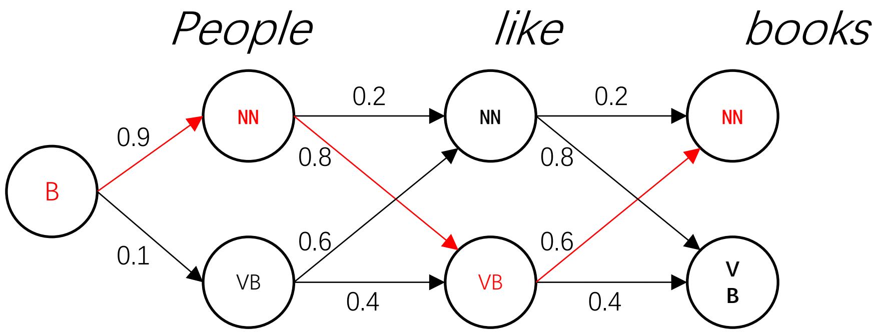
NULL	B		
NULL	B		

NULL	B	NN	
NULL	B	NN	

NULL	B	NN	VB
NULL	B	NN	VB

NULL	B	NN	VB
NULL	B	NN	VB





Hidden Markov Model

- Time：

  $O(|L|^n) \rightarrow O(n|L|)$

- Space (two $L*n$ tables):

  tb: the probability table

  bp: the “back pointer”

- Algorithms

  -- building table (Viterbi)

  -- finding tag sequence (back tracking)

Hidden Markov Model

- Decoding

Input: $s = W_{1:n}$, first-order HMM model with $P(t|t')$ for $t, t' \in L$, and $P(w|t)$ for $w \in V, t \in L$;
Variables: tb, bp;
Initialisation:
    $tb[\langle \text{B} \rangle][0] \leftarrow 1$;
    $tb[t][i] \leftarrow 0$, $bp[t][i] \leftarrow \textsc{Null}$ **for** $t \in L, i \in [1, \ldots, n]$;
for $t \in L$ **do**
    $tb[t][1] \leftarrow tb[\langle \text{B} \rangle][0] \times P(t|\langle \text{B} \rangle) \times P(w_i|t)$
for $i \in [2, \ldots, n]$ **do**
    for $t \in L$ **do**
        for $t' \in L$ **do**
            if $tb[t][i] < tb[t'][i-1] \times P(t|t') \times P(w_i|t)$ **then**
                $tb[t][i] \leftarrow tb[t'][i-1] \times P(t|t') \times P(w_i|t)$;
                $bp[t][i] \leftarrow t'$;
$y_n \leftarrow \arg\max_t tb[t][n]$;
for $i \in [n, \ldots, 2]$ **do**
    $y_{i-1} \leftarrow bp[y_i][i]$;
Output: $y_1, \ldots, y_n$;

Contents

Hidden Markov Model

Three basic problems summary：

1. *Scoring*

   given a model and input/output pair, find the probability

2. *Training*

   Given labeled sentences, estimate the parameters of model

3. *Decoding*

   Given a model and input, find the tag sequence

Finding Marginal Probabilities

- Goal – find $P(t_i = t|W_{1:n}), i \in [1, \dots, n]$
- The modeling target form $P(T_{1:n}, W_{1:n})$
- Marginalization

$$P(t_i = t|W_{1:n}) =$$

$$\sum_{t_1 \in L} \sum_{t_2 \in L} \sum_{t_{i-1} \in L} \sum_{t_{i+1} \in L} \dots \sum_{t_n \in L} P(T_{1:n}(t_i = t)|W_{1:n})$$

$$\propto \sum_{t_1 \in L} \sum_{t_2 \in L} \sum_{t_{i-1} \in L} \sum_{t_{i+1} \in L} \dots \sum_{t_n \in L} P(W_{1:n}, T_{1:n}(t_i = t))$$

-- exponential sum, intractable

- Dynamic program again

Finding Marginal Probabilities

$$
\begin{aligned}
P(t_i = t | W_{1:n}) &= \frac{P(t_i = t, W_{1:n})}{P(W_{1:n})} \qquad \text{(Bayes rule conditioned on } W_{1:i}\text{)} \\
&= \frac{P(t_i = t, W_{1:i}, W_{i+1:n})}{P(W_{1:n})} \\
&= \frac{P(W_{1:i}, t_i = t) P(W_{i+1:n} | t_i = t, W_{1:i})}{P(W_{1:n})} \\
&= \frac{P(W_{1:i}, t_i = t) P(W_{i+1:n} | t_i = t)}{P(W_{1:n})} \\
&\qquad (W_{i+1:n} \text{ is conditionally independent of } W_{1:i} \text{ given } t_i) \\
&\propto P(W_{1:i}, t_i = t) P(W_{i+1:n} | t_i = t) \\
&\qquad (P(W_{1:n}) \text{ is constant for all } t).
\end{aligned}
$$

Finding Marginal Probabilities

$$P(t_i = t | W_{1:n}) = \frac{P(t_i = t, W_{1:n})}{P(W_{1:n})} \qquad \text{(Bayes rule conditioned on } W_{1:i}\text{)}$$

$$= \frac{P(t_i = t, W_{1:i}, W_{i+1:n})}{P(W_{1:n})}$$

$$= \frac{P(W_{1:i}, t_i = t) P(W_{i+1:n} | t_i = t, W_{1:i})}{P(W_{1:n})}$$

$$= \frac{P(W_{1:i}, t_i = t) P(W_{i+1:n} | t_i = t)}{P(W_{1:n})}$$

($W_{i+1:n}$ is conditionally independent of $W_{1:i}$ given t_i)

$$\propto \underline{P(W_{1:i}, t_i = t)} \, \underline{P(W_{i+1:n} | t_i = t)}$$

($P(W_{1:n})$ is constant for all t).

Forward algorithm   Backward algorithm

The forward algorithm

- $\alpha(t, i) = P(W_{1:i}, t_i = t)$

  $= \sum_{t_1 \in L} \dots \sum_{t_{i-1} \in L} P\big(W_{1:i}, T_{1:i}(t_i = t)\big)$

  $= \sum_{t_1 \in L} \dots \sum_{t_{i-1} \in L} P\big(W_{1:(i-1)}, T_{1:(i-1)}\big) \cdot P(w_i | t_i = t) \cdot P(t_i = t | t_{i-1})$

  $= \sum_{t_{i-1} \in L}\Big(\sum_{t_1 \in L} \dots \sum_{t_{i-1} \in L} P\big(W_{1:(i-1)}, T_{1:(i-1)}(t_{i-1})\big)\Big)$

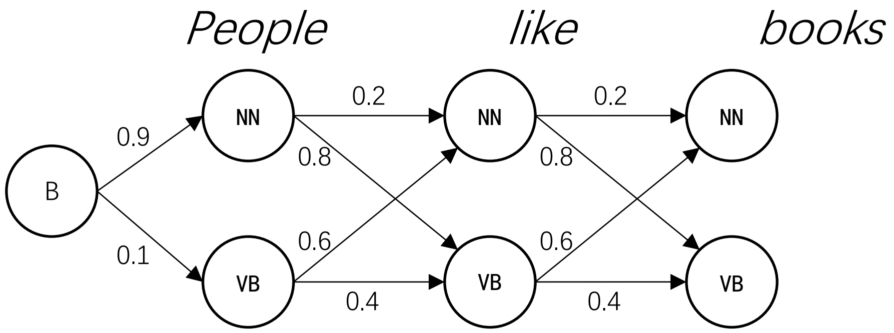
  $\cdot P(w_i | t_i = t) \cdot P(t_i = t | t_{i-1})$

  $= \sum_{t' \in L} \alpha(t', i - 1) \cdot P(w_i | t_i = t) \cdot P(t_i = t | t')$

- A dynamic program is feasible by incrementally building a table $\alpha[t][i]$ with n columns and $|L|$ rows.

The forward algorithm

- Again using the example



$\alpha[B][0] = 1$

$\alpha[NN][1] = 0.9$
$\alpha[VB][1] = 0.1$

$\alpha[NN][2] = \alpha[NN][1] * 0.2 + \alpha[VB][1] * 0.6$=0.24
$\alpha[VB][2] = \alpha[NN][1] * 0.8 + \alpha[VB][1] * 0.4$=0.76

$\alpha[NN][3] = \alpha[NN][2] * 0.2 + \alpha[VB][2] * 0.6$=0.504
$\alpha[VB][3] = \alpha[NN][2] * 0.8 + \alpha[VB][2] * 0.4$=0.496

The forward algorithm

Inputs: $s = W_{1:n}$, first-order HMM model with $P(t|t')$ for $t, t' \in L$, and $P(w|t)$ where $w \in V$, $t \in L$;

Variables: α;

Initialisation: $\alpha[\langle \mathrm{B} \rangle][0] \leftarrow 1$, $\alpha[t][i] \leftarrow 0$ for $i \in [1, \ldots, n], t \in L$;

for $t \in L$ **do**

  $\alpha[t][1] \leftarrow \alpha[\langle \mathrm{B} \rangle][0] \times P(t|\langle \mathrm{B} \rangle) \times P(w_1|t)$

for $i \in [2, \ldots, n]$ **do**

  for $t \in L$ **do**

    for $t' \in L$ **do**

      $\alpha[t][i] \leftarrow \alpha[t][i] + \alpha[t'][i-1] \times P(t|t') \times P(w_i|t)$;

Output: α;

<B>: beginning of sentence token

An incremental calculation in the forward direction by using table α

The backward algorithm

- $\beta(t,i) = P(W_{i+1:n}|t_i = t)$

$$= \sum_{t_{i+1}\in L} \dots \sum_{t_n \in L} P(W_{i+1:n}, T_{i+1:n}|t_i = t)$$

$$= \sum_{t_{i+1}\in L} \dots \sum_{t_n \in L} P(t_{i+1}|t_i = t) P(w_{i+1}|t_{i+1}) \cdot P(w_{i+2:n}, T_{i+2:n}|t_{i+1})$$

$$= \sum_{t_{i+1}\in L} \left(\sum_{t_{i+2}\in L} \dots \sum_{t_n \in L} P(W_{i+2:n}, T_{i+2:n}|t_{i+1}) \right)$$

$$\cdot P(t_{i+1}|t_i = t) \cdot P(w_{i+1}|t_{i+1})$$

$$= \sum_{t' \in L} \beta(t', i+1) \cdot P(t_{i+1} = t'|t_i = t) \cdot P(w_{i+1}|t')$$

- A dynamic program is feasible by incrementally building a table $\beta[t][i]$ with n columns and $|L|$ rows.

The backward algorithm

Inputs: $s = W_{1:n}$, first-order HMM model with $P(t|t')$ for $t, t' \in L$, and $P(w|t)$ where $w \in V$, $t \in L$;

Variables: β;

Initialisation: $\beta[t][n] \leftarrow 1$ **for** $t \in L$, $\beta[t][i] \leftarrow 0$ **for** $i \in [1, \ldots, n-1], t \in L$;

for $i \in [n-1, \ldots, 1]$ **do**

  for $t' \in L$ **do**

    for $t \in L$ **do**

      $\beta[t'][i] \leftarrow \beta[t'][i] + \beta[t][i+1] \times P(t|t') \times P(w_{i+1}|t)$;

Output: β;

An incremental calculation in the backward direction by using table β

The forward-backward algorithm

$$P(t_i = t | W_{1:n}) \propto P(W_{1:i}, t_i = t) P(W_{i+1:n} | t_i = t)$$

Inputs: $s = W_{1:n}$, first-order HMM model with $P(t|t')$ for $t, t' \in L$, and $P(w|t)$ where $w \in V$, $t \in L$;

Variables: tb, α, β;

$\alpha \leftarrow \text{Forward}(W_{1:n}, model)$;

$\beta \leftarrow \text{Backward}(W_{1:n}, model)$;

for $i \in [1, \ldots, n]$ **do**

    $total \leftarrow 0$;

    for $t \in L$ **do**

        $total \leftarrow total + \alpha[t][i] \times \beta[t][i]$

    for $t \in L$ **do**

        $tb[t][i] \leftarrow \frac{\alpha[t][i] \times \beta[t][i]}{total}$;

Output: tb;

Contents

Baum-Welch algorithm

- **Baum-Welch algorithm**: The particular **EM algorithm** for HMM parameter estimation

- Considering $\log P(W_{1:n}|\Theta) = \log \sum_{T_{1:n}} P\,(W_{1:n}, T_{1:n}|\Theta)$
- Define $E_{P(T_{1:n}|W_{1:n},\Theta')} \log P(W_{1:n}, T_{1:n}|\Theta)$ (Q-function)
- Run standard EM algorithm.

Contents

Recall EM

- EM considers all possible values of hidden variables.

Inputs: data $O = \{o_i\}|_{i=1}^{N}$;
Hidden Variables: $H = \{\mathbf{h}_j\}|_{j=1}^{M}$;
Initialization: model $\Theta^0 \leftarrow \text{RandomModel}()$, $t \leftarrow 0$;
repeat
    Expectation step:
        Compute $P(\mathbf{h}|o_i, \Theta^t)$, $\mathbf{h} \in H$;
        $Q(\Theta, \Theta^t) \leftarrow \sum_{i=1}^{N} \sum_{\mathbf{h} \in H} P(\mathbf{h}|o_i, \Theta^t) \log P(o_i, \mathbf{h}|\Theta)$;
    Maximization step:
        $\Theta^{t+1} \leftarrow \arg\max_{\Theta} Q(\Theta, \Theta^t)$;
    $t \leftarrow t + 1$;
until $\text{Converge}(H, \Theta)$;

- $P(h|o_i, \Theta^t), h \in H$ is the assignment distribution of H.
- $Q(\Theta, \Theta^t)$ is called the Q-function.

Expectation step

Parameterize the expectation function$Q(\Theta, \Theta')$

$$Q(\Theta, \Theta') = \sum_{T_{1:n}} P\,(T_{1:n}|W_{1:n}, \Theta')\log P(W_{1:n}, T_{1:n}|\Theta),$$

$$= \sum_{T_{1:n}} P\,(T_{1:n}|W_{1:n}, \Theta')\log(\prod_{i=1}^{n} P\,(t_i|t_{i-1})P(w_i|t_i))$$

$$= \sum_{T_{1:n}} P\,(T_{1:n}|W_{1:n}, \Theta') \sum_{i=1}^{n}(\,\log P(w_i|t_i) + \log P(t_i|t_{i-1}))$$

$$= \sum_{i=1}^{n}(\sum_{w} \sum_{t} \log P(w|t) \sum_{T_{1:n}} P\,(T_{1:n}|W_{1:n}, \Theta')\delta(t_i, t)\delta(w_i, w))$$

$$+ \sum_{i=1}^{n}(\sum_{t'} \sum_{t} \log P(t|t') \sum_{T_{1:n}} P\,(T_{1:n}|W_{1:n}, \Theta')\delta(t_{i-1}, t')\delta(t_i, t))$$

Expectation step

Let $\gamma_i(t) = \sum_{T_{1:n}} P\,(T_{1:n}|W_{1:n}, \Theta')\delta(t_i, t)$ and $\xi_i(t', t) = \sum_{T_{1:n}} P\,(T_{1:n}|W_{1:n}, \Theta')\delta(t_{i-1}, t')\delta(t_i, t))$, both can be computed efficiently.

$$Q(\Theta, \Theta') = \sum_{i=1}^{n}(\sum_{w \in V}\sum_{t \in L} \log P(w|t)\delta(w_i, w)\gamma_i(t)) + \sum_{i=1}^{n}(\sum_{t' \in L}\sum_{t \in L} \log P(t|t')\xi_i(t', t))$$

$$\gamma_i(t) = \frac{\alpha(t_i = t)\beta(t_i = t)}{\sum_{t' \in L} \alpha\,(t_i = t')\beta(t_i = t')},$$

$$\xi_i(t', t) = \frac{\alpha(t_{i-1} = t')P(t|t', \Theta')P(w_i|t, \Theta')\beta(t_i = t)}{\sum_{u \in L} \alpha\,(t_i = u)\beta(t_i = u)}$$

Expectation step

Therefore,

$$Q(\Theta,\Theta') = \sum_{i=1}^{n} \sum_{w} \sum_{t} \log P(w|t)\delta(w_i,w)\gamma_i(t)$$

$$+\sum_{i=1}^{n} \sum_{t'} \sum_{t} \log P(t|t')\xi_i(t',t)$$

$$= \sum_{w} \sum_{t} \log P(w|t) \sum_{i=1}^{n} \delta\,(w_i,w)\gamma_i(t)$$

$$+ \sum_{t'} \sum_{t} \log P(t|t') \sum_{i=1}^{n} \xi_i\,(t',t)$$

Maximization step

- Use Lagrange multipliers to find the constraint optimum.

$$\pi(\Theta, \Lambda) = \sum_{w} \sum_t \log P(w|t) \sum_{i=1}^{n} \delta(w_i, w) \gamma_i(t)$$

$$+ \sum_{t'} \sum_t \log P(t|t') \sum_{i=1}^{n} \xi_i(t', t)$$

from $Q(\Theta, \Theta')$

$$+ \sum_{t} (\lambda_t^1 (1 - \sum_w P(w|t)) + \sum_{t'} \lambda_{t'}^2 (1 - \sum_t P(t|t')))$$

$\sum_w P(w|t) = 1$,
$\sum_t P(t|t') = 1$

- The partial derivative of $\pi(\Theta, \Lambda)$ with respect to $P(w|t)$

$$\frac{\partial \pi(\Theta, \Lambda)}{\partial P(w|t)} = \frac{\sum_{i=1}^{n} \delta(w_i, w) \gamma_i(t)}{P(w|t)} - \lambda_t^1$$

- Let $\frac{\partial \pi(\Theta,\Lambda)}{\partial P(w|t)}$ =0, $\quad P(w|t) = \frac{\sum_{i=1}^{n} \delta(w_i, w) \gamma_i(t)}{\lambda_t^1} = \frac{\sum_{i=1}^{n} \delta(w_i, w) \gamma_i(t)}{\sum_{i=1}^{n} \gamma_i(t)}$
- Similarly,

$$P(t|t') = \frac{\sum_{i=1}^{n} \xi_i(t', t)}{\sum_u \sum_{i=1}^{n} \xi_i(t', u)} = \frac{\sum_{i=1}^{n} \xi_i(t', t)}{\sum_{i=1}^{n} \sum_u \xi_i(t', u)} = \frac{\sum_{i=1}^{n} \xi_i(t', t)}{\sum_{i=1}^{n} \gamma_i(t')}$$

Maximization step

With N observations

$$P(w|t) = \frac{\sum_{k=1}^{N}\sum_{i=1}^{n_k}\delta\,(w_i^k, w)\gamma_i^k(t)}{\sum_{k=1}^{N}\sum_{i=1}^{n_k}\gamma_i^k\,(t)}$$

$$P(t|t') = \frac{\sum_{k=1}^{N}\sum_{i=1}^{n_k}\xi_i^k\,(t', t)}{\sum_{k=1}^{N}\sum_{i=1}^{n_k}\gamma_i^k\,(t')}$$

Baum-Welch algorithm

Inputs: $s = W_{1:n}$;
Initialisations: randomly initialise a first-order HMM model with $P(t|t')$ for $t, t' \in L$, and $P(w|t)$ where $w \in V$, $t \in L$;
Variables: $\alpha, \beta, \gamma, \xi$;
while not CONVERGE $(W_{1:n}, P(t|t'), P(w|t))$ **do**
    $\alpha \leftarrow$ FORWARD$(W_{1:n}, model)$;
    $\beta \leftarrow$ BACKWARD$(W_{1:n}, model)$;
    for $i \in [1, \ldots, n]$ **do**
        $total \leftarrow 0$;
        for $t \in L$ **do**
            $total \leftarrow total + \alpha[t][i] \times \beta[t][i]$;
        for $t \in L$ **do**
            $\gamma[t][i] \leftarrow \frac{\alpha[t][i] \times \beta[t][i]}{total}$;
            for $t' \in L$ **do**
                $\xi[t][t'][i] \leftarrow \frac{\alpha[t'][i-1] P(t|t') P(w_i|t) \beta[t][i]}{total}$;

Calculate $\gamma_i(t)$ and $\xi_i(t', t)$

Baum-Welch algorithm

for $t \in L$ **do**
- $total_t \leftarrow 0$;
- **for** $w \in V$ **do**
  - $count[w] \leftarrow 0$;
- **for** $i \in [1, \ldots, n]$ **do**
  - $total_t \leftarrow total_t + \gamma[t][i]$;
  - $count[w_i] \leftarrow count[w_i] + \gamma[t][i]$;
- **for** $w \in V$ **do**
  - $P(w|t) \leftarrow \frac{count[w]}{total_t}$;

for $t' \in L$ **do**
- $total_{t'} \leftarrow 0$;
- **for** $t \in L$ **do**
  - $count[t] \leftarrow 0$;
- **for** $i \in [1, \ldots, n]$ **do**
  - $total_{t'} \leftarrow total_{t'} + \gamma[t'][i]$;
  - **for** $t \in L$ **do**
    - $count[t] \leftarrow count[t] + \xi[t][t'][i]$;
- **for** $t \in L$ **do**
  - $P(t|t') \leftarrow \frac{count[t]}{total_{t'}}$;

Output: the first-order HMM model $\{P(w|t), P(t|t')\}$ for $w \in V$ and $t, t' \in L$;

Calculate $P(w|t)$ and $P(t|t')$

Summary

- Hidden Markov models (HMM), first order HMMs, second order HMMs
- Viterbi decoding algorithms both for first order HMMs and second order HMMs
- Forward algorithms, backward algorithms, forward-backward algorithms both for first order HMMs and second order HMMs
- EM algorithms for HMMs