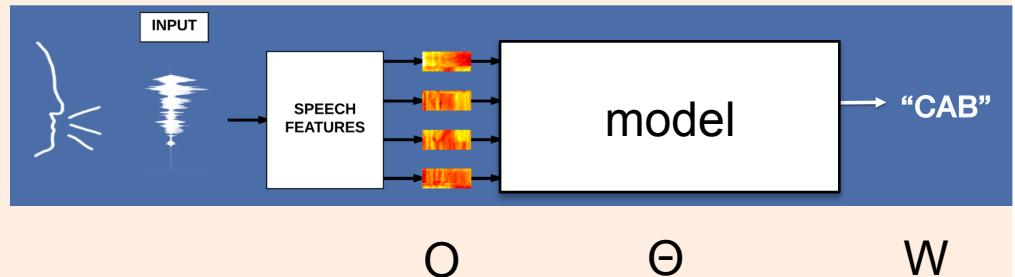


# Algorithms for speech and language processing

#3. ASR: Language modeling

# This hour

- I. Generative Models
- II. Decoding
- III. Learning



## Notations

$\theta$ : model parameters

$O = o_1 \dots o_t$  : observed speech frames

$W = w_1 \dots w_n$ : word transcriptions

**Decoding:** given observed  $O$  and  $\theta$ ,  
find the most likely  $W$ .

$$W^* = \arg \max_w p(O|W, \theta)$$

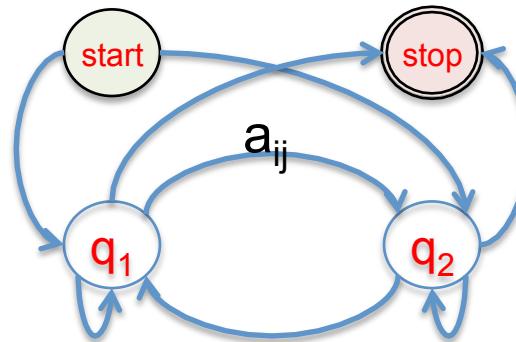
**Learning:** given an observed  $O$   
and model, find the best  $\theta$

$$\theta^* = \arg \max_{\theta} p(O|\theta)$$

# Part I: Generative Models

- Useful finite state models (recap)
- Model decomposition
  - acoustic model
  - pronunciation model
  - language model

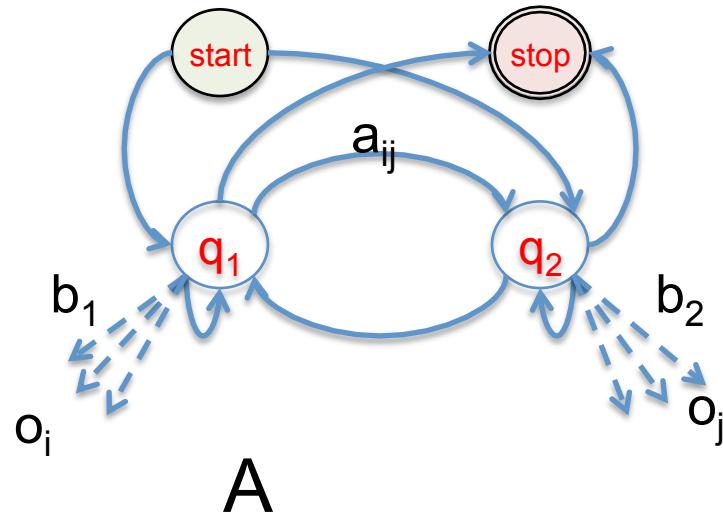
# markov chains



A

	$q_0$ (start)	$q_1$	$q_2$	$q_F$ (stop)
$q_0$ (start)	0	$a_{01}$	$a_{02}$	0
$q_1$	0	$a_{11}$	$a_{12}$	$a_{1F}$
$q_2$	0	$a_{21}$	$a_{22}$	$a_{2F}$
$q_F$ (stop)	0	0	0	0

# hidden markov models (HMMs)



B

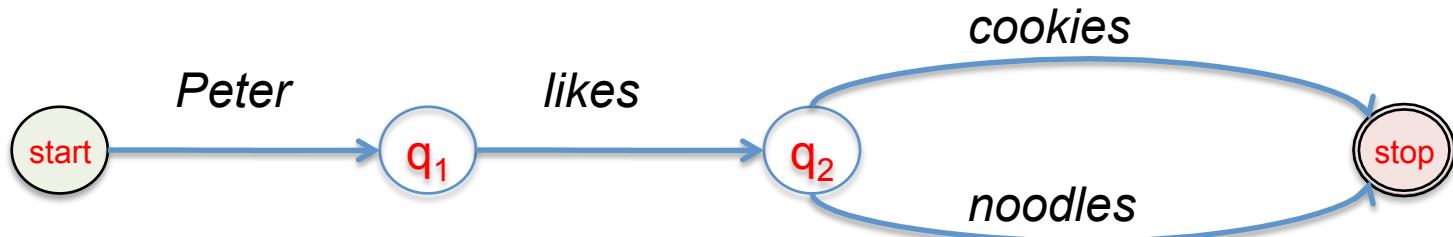
	$q_1$	$q_2$
$o_1$	$b_{11}$	$b_{21}$
$o_2$	$b_{12}$	$a_{22}$
$o_3$	$b_{13}$	$a_{23}$

A

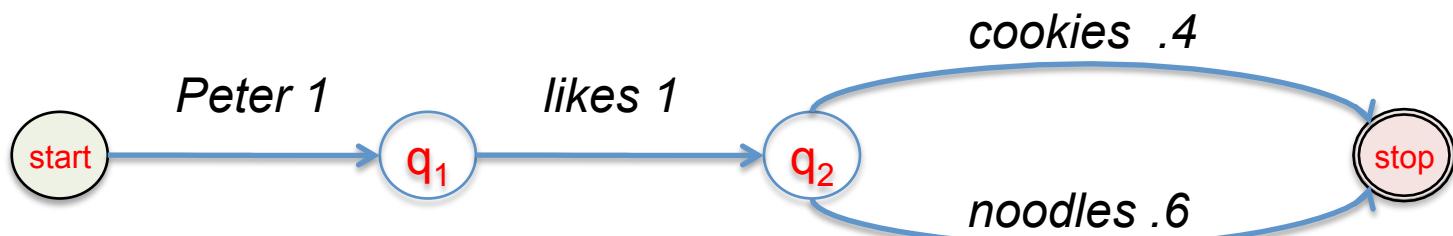
	$q_0$ (start)	$q_1$	$q_2$	$q_F$ (stop)
$q_0$ (start)	0	$a_{01}$	$a_{02}$	0
$q_1$	0	$a_{11}$	$a_{12}$	$a_{1F}$
$q_2$	0	$a_{21}$	$a_{22}$	$a_{2F}$
$q_F$ (stop)	0	0	0	0

# (weighted) finite state transducers

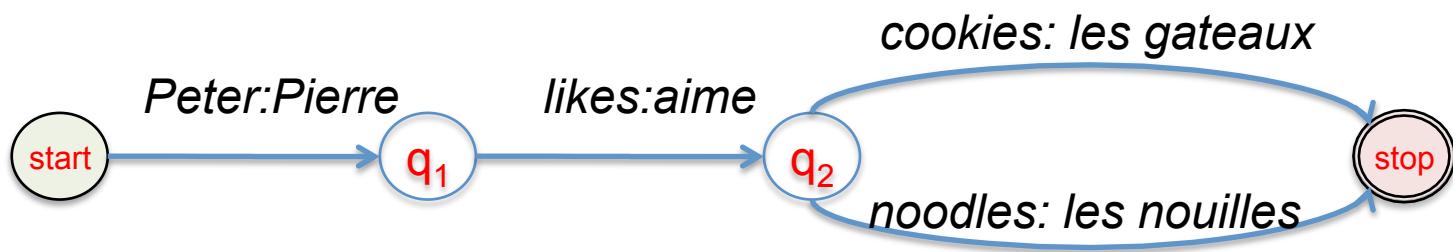
Finite State  
Acceptor (FSA)



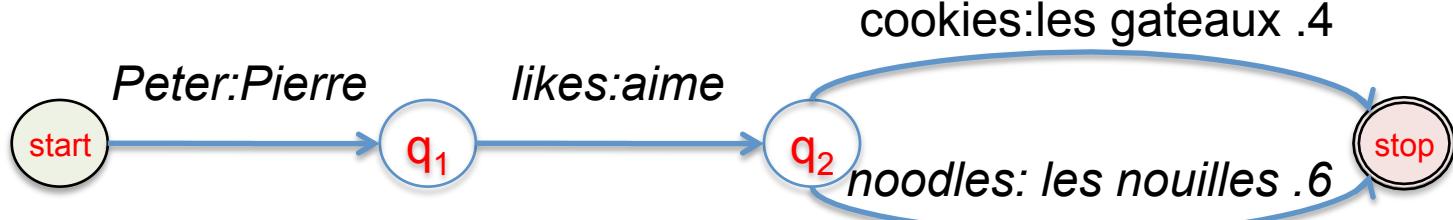
Weighted Finite  
State Acceptor  
(WFSA)



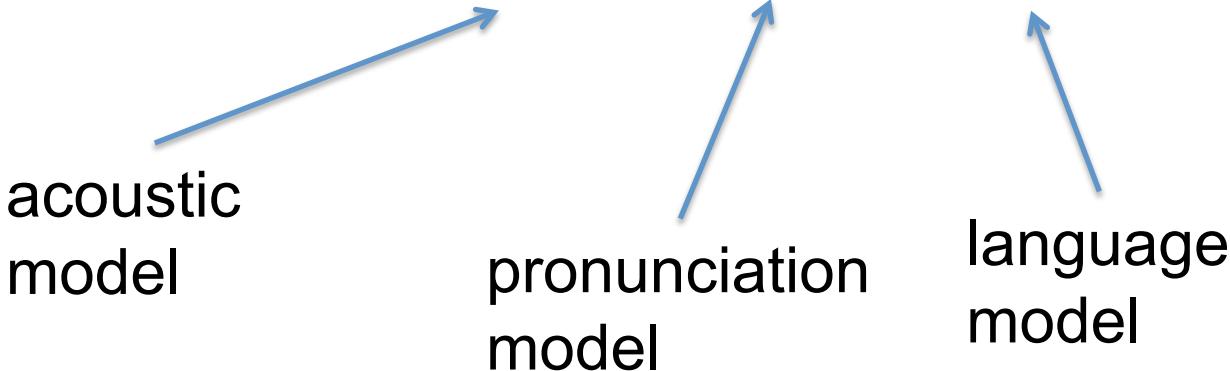
Finite State  
Transducer  
(FST)



Weighted Finite  
State Transducer  
(WFST)



# Model decomposition

- $W^* = \arg \max_W P(W|O, \Theta)$
- $P(W|O, \Theta) \propto P(O|W, \Theta)P(W|\Theta)$
- $$= P(O|Q)P(Q|W)P(W)$$


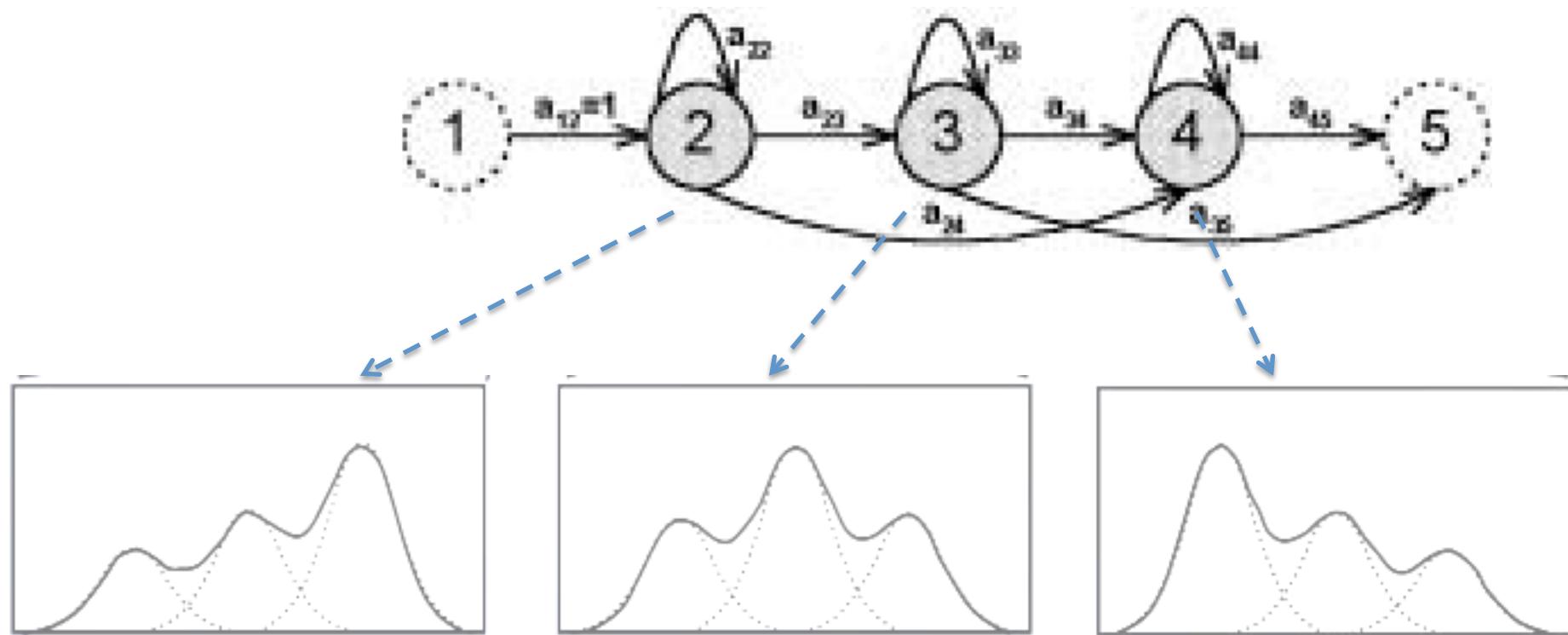
acoustic model      pronunciation model      language model

# Acoustic models

$P(O|Q)$

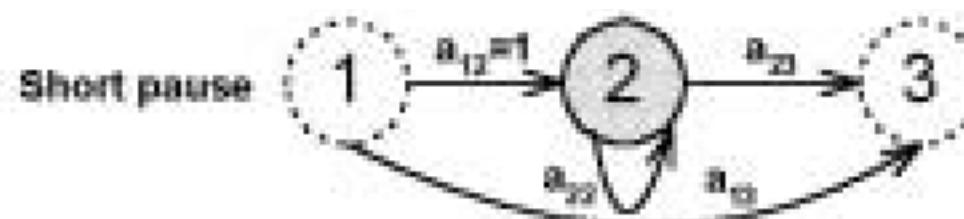
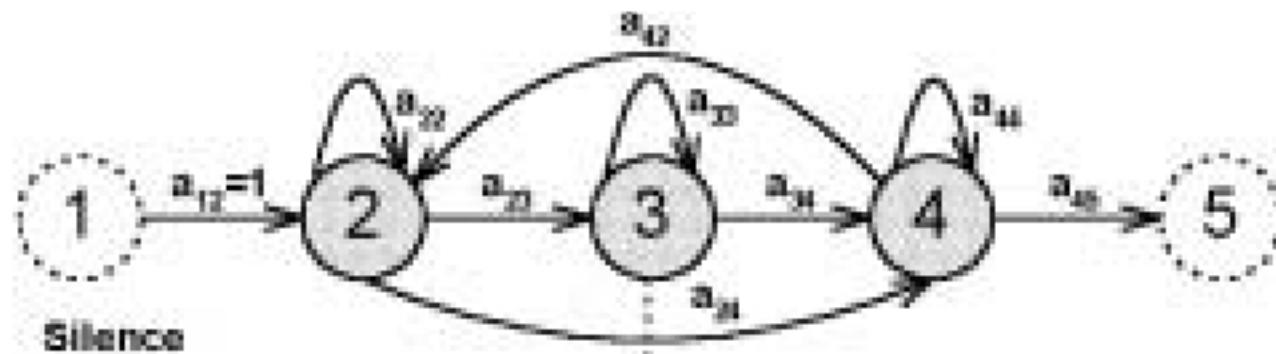
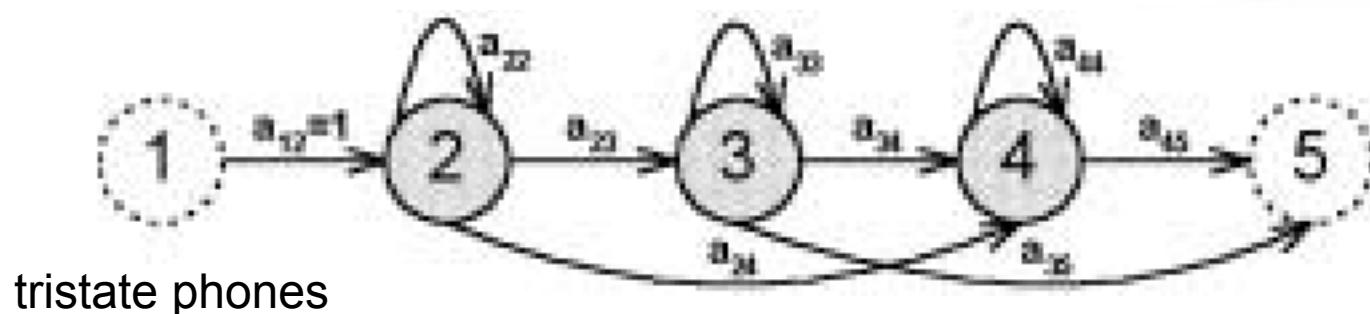


- tri-state phones



# Acoustic models

$P(O|Q)$



# Pronunciation models

$P(Q|W)$

Prononciations of the word  
'eleven' in the state of  
Pennsylvania, USA

ə'lɛvən

ə'lɛvər

ə'lɛvər

ə'lɛvən

ə'lɛvər

ə'lɛvən

ə'lɛvən

lɛvən

lɛvən

ə'lɛvən

...

(total= 220 variants)



<https://www.youtube.com/watch?v=J3YLphzAnw>

Phonological/Phonetic transformations

What are you doing?

ʰwʌtʃə'duɪn

I can inquire.

ʰaɪkŋ'kwaiər

Did you eat yet?

ʰdʒitjɛ?

I don't believe him.

ai'doðbə'lɪvəm

We ought to have come.

ʰwi'ɔfɪ'kʌm

# Pronunciation models

## 1. Using an existing phonetic dictionary

orthography	SAMPA	IPA
counting	kA_wntlN	kəwntiŋ
counting(1)	kA_wnlN	kəwniŋ
amortization	@mOrt@zeS@n	əmɔrtəzeʃən
amortization(1)	{ mOrt@zeS@n	æmɔrtəzeʃən
amortization(2)	@mOrtA_jzeS@n	əmɔrtaizeʃən
..etc...		

## 2. Using G2P to extend the dictionary

- eg: g2p-seq2seq (CMU), phonetisaurus
  - align graphemes and phonemes
  - build an ngram model
  - make a WFST from it
  - iterate

“speaking” = s p ea k ing  
[spi:kɪŋ] [s] [p] [i:] [k] [ɪŋ]

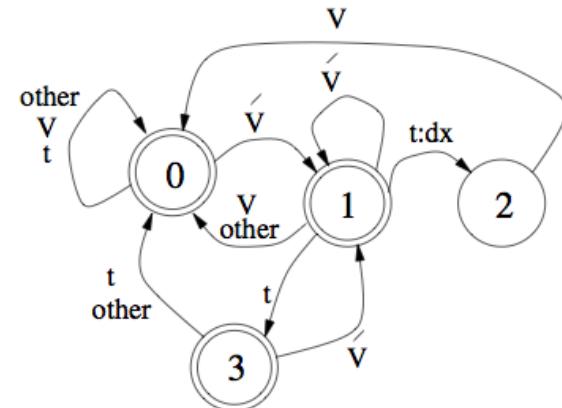
ଶାସ୍ ଶାନ୍ତିକୁମାର୍

ପ୍ରଯୋଗ୍ୟାନ୍ତିକୁମାର୍

### 3. phonological rules (hand crafted)

- ex: english flapping

$/t/ \rightarrow [dx] / [+vowel] \xrightarrow{[-stress]}$



Input symbols: ax t eh n y uw ey t ax d

State sequence: 0 → 0 → 0 → 1 → 0 → 0 → 0 → 1 → 2 → 0 → 0

Output symbols: ax t eh n y uw ey dx ax d

Name	Rule	Example	Prob
<b>Syllabic Rules*</b>			
Syllabic n	[ax ix] n → en	button	.35
Syllabic m	[ax ix] m → em	bottom	.32
Syllabic l	[ax ix] l → el	bottle	.72
Syllabic r	[ax ix] r → axr	butter	.77
Flapping	[tcl dcl] [t d] → dx /V ____ [ax ix axr]	button	.87
Flapping-r	[tcl dcl] [t d] → dx /V r ____ [ax ix axr]	barter	.92
H-voicing	hh → hv / [+voice] ____ [+voice]	ahead	.92
L-deletion	l → Ø/ ____ y [ax ix axr]	million	n/a
Gliding	iy → y / ____ [ax ix axr]	colonial	n/a
Nasal-deletion	[n m ng] → Ø/ ____ [-voice -consonant]	rant	n/a
Function words			
h-deletion	h → Ø/ # ____	he, him	n/a
w-deletion	w → Ø/ # ____	will, would	n/a
dh-deletion	dh → Ø/ # ____	this, those	n/a
Dental-deletion	[tcl dcl] [t d] → Ø/ [+vowel] ____ [th dh]	breadth	n/a
Final dental-deletion	([tcl dcl]) [t d] → Ø/ [+cons +continuant] ____ # soft (as)	n/a	n/a
Slur	ax → Ø/ [+consonant] ____ [r l n] [+vowel]	camera	n/a
Stressed slur	[+vowel +stress] r → er	warts	n/a
Pre-stress contraction	ax → Ø/ [+cons] ____ [+cons] [+vowel +stress]	senility	n/a
Ruh-reduction	r ax → er / [-word bdry] ____ [-word bdry]	separable	n/a
Transitional stops			
t-introduction	Ø → tcl / [+dental +nasal] ____ [+fricative]	prin[t]ce	n/a
t-deletion	[tcl] → Ø/ [+dental +nasal] ____ [+fricative]	prints	n/a

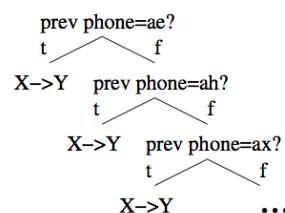
## 4. phonological rules (data driven)

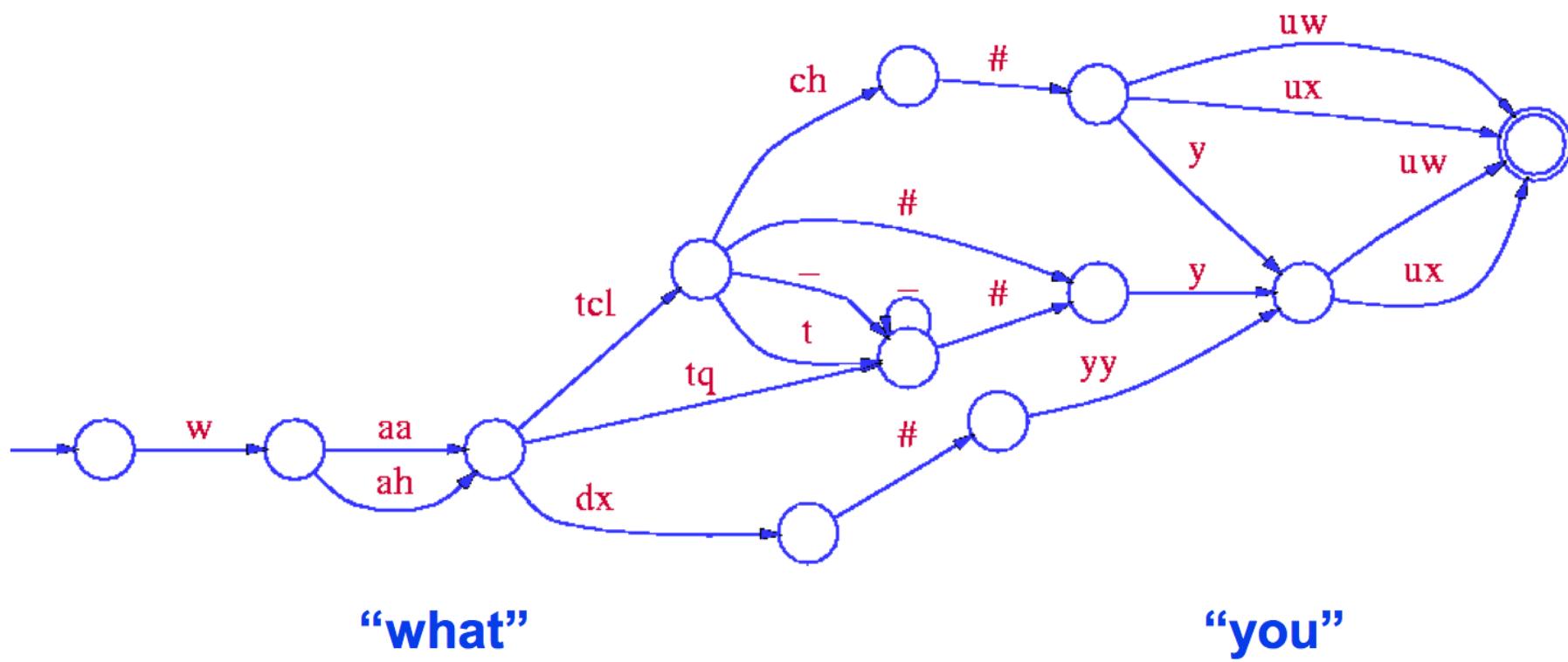
- using a phone recognizer to learn a pronunciation dictionary

a	baseball	game	<b>word sequence</b>
ax	b ey s b ao l g ey m		<b>canonical phone sequence</b>
ax	b eh ey s b el g eh m		<b>aligned surface phones</b>
ax	b eh ey s b el g eh m		<b>proposed new pronunciations</b>

- inducing pronunciation rules
  - trigram models
  - decision trees

$$P(S^i) = \prod_{j=1}^n (P(s_j^i | c_{j-1}, c_j, c_{j+1})) .$$





# Language models

P(W)



$$p(w_1..w_n) = p(w_1)p(w_2|w_1) \dots p(w_n|w_1..w_{n-1})$$

- N-gram models

- truncate  $p(w_i|w_1.. w_{i-N+1} .. w_{i-1})$  to  $p(w_i|w_{i-N+1}..w_{i-1})$

- 1-gram:  $p(w_i)$

N-1

- 2-gram:  $p(w_i|w_{i-1})$

- 3-gram:  $p(w_i|w_{i-2}w_{i-1})$

- etc.

1

Gram

- To him swallowed confess hear both. Which. Of save on trail for are ay device and rote life have
- Hill he late speaks; or! a more to leg less first you enter

2

Gram

- Why dost stand forth thy canopy, forsooth; he is this palpable hit the King Henry. Live king. Follow.

- What means, sir. I confess she? then all sorts, he is trim, captain.

3

Gram

- Fly, and will rid me these news of price. Therefore the sadness of parting, as they say, 'tis done.
- This shall forbid it should be branded, if renown made it empty.

4

Gram

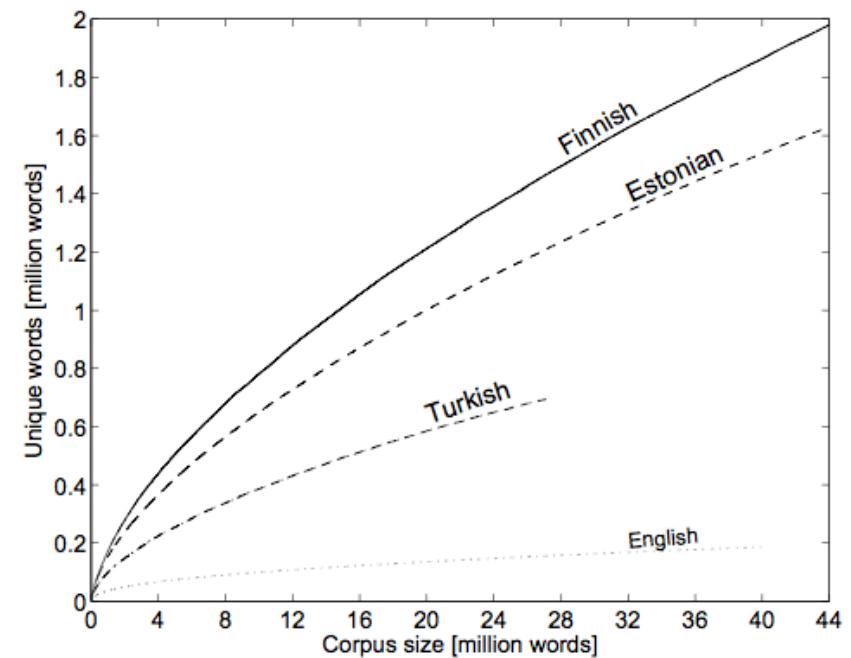
- King Henry. What! I will go seek the traitor Gloucester. Exeunt some of the watch. A great banquet serv'd in;
- It cannot be but so.

- estimating Ngram models from data

- counts

$$P(w_i) = \frac{c_i}{N}$$

- unknown words (OOVs)
  - the lexicon is infinite!
  - a new corpus will always have new words
  - a fast and dirty solution: recoding with <UNK>



	<b>i</b>	<b>want</b>	<b>to</b>	<b>eat</b>	<b>chinese</b>	<b>food</b>	<b>lunch</b>	<b>spend</b>
<b>i</b>	5	827	0	9	0	0	0	2
<b>want</b>	2	0	608	1	6	6	5	1
<b>to</b>	2	0	4	686	2	0	6	211
<b>eat</b>	0	0	2	0	16	2	42	0
<b>chinese</b>	1	0	0	0	0	82	1	0
<b>food</b>	15	0	15	0	1	4	0	0
<b>lunch</b>	2	0	0	0	0	1	0	0
<b>spend</b>	1	0	1	0	0	0	0	0

- unknown ngrams
  - most ngrams have been seen 0 time
  - problem grows exponentially with n
  - true 0s versus sampling
  - solutions:
    - smoothing
    - backoff, interpolation, clustering

$$P_{\text{Laplace}}(w_i) = \frac{c_i + 1}{N + V}$$

$$c_i^* = (c_i + 1) \frac{N}{N + V}$$

$$\hat{P}(w_n | w_{n-2} w_{n-1}) = \lambda_1 P(w_n | w_{n-2} w_{n-1}) + \lambda_2 P(w_n | w_{n-1}) + \lambda_3 P(w_n)$$

see Jurafsky & Martin 2017

- evaluating language models
  - extrinsic: improved WER
  - intrinsic: perplexity

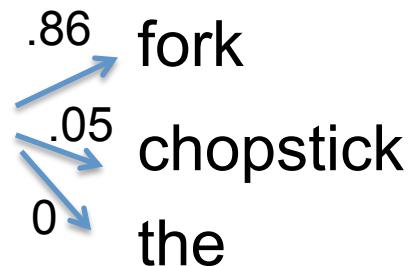
$$PP(W) = P(W)^{-1/N}$$

$$PP(W) = \prod_1^N P(w_i | w_1..w_{i-1})^{-1/N}$$

$$PP(W) = 2^{H(W)}$$

with  $H(W) = -1/N \log_2 P(W)$

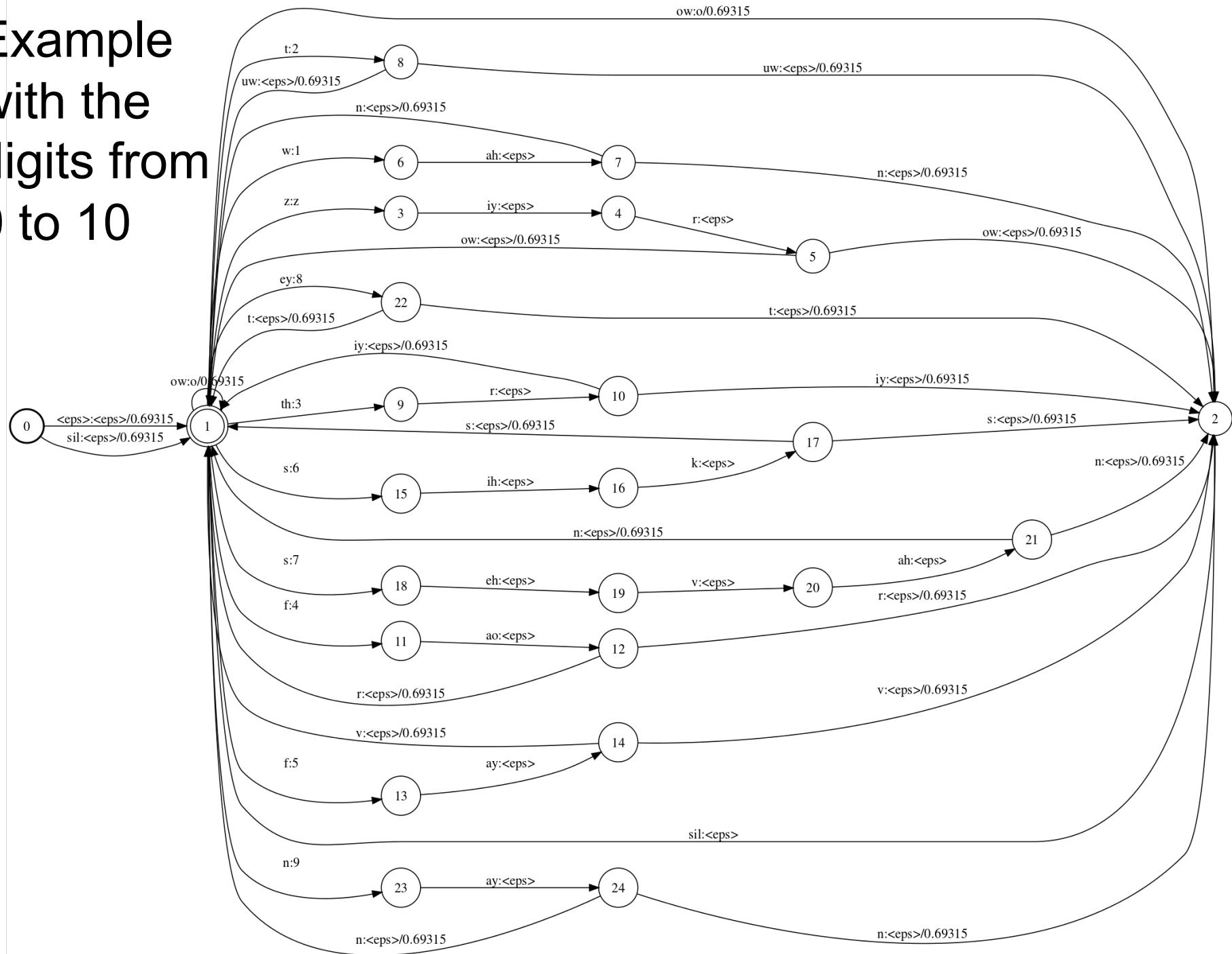
He is eating a steak with a knife and a ...



# Putting all together

- acoustic model+speech input: a decoding graph
- pronunciation+language model: an FST  
→ a gigantic decoding graph

# Example with the digits from 0 to 10



# Putting all together

- acoustic model+speech input: a decoding graph
- pronunciation+language model: an FST  
→ a gigantic decoding graph
- in practice:
  - FST tools: composing & simplifying FSTs
  - decomposing processing in two stages: a first decoding lattice, followed by lattice rescoring with a large LM

# fast forward: RNN language models

- character language models

PANDARUS:

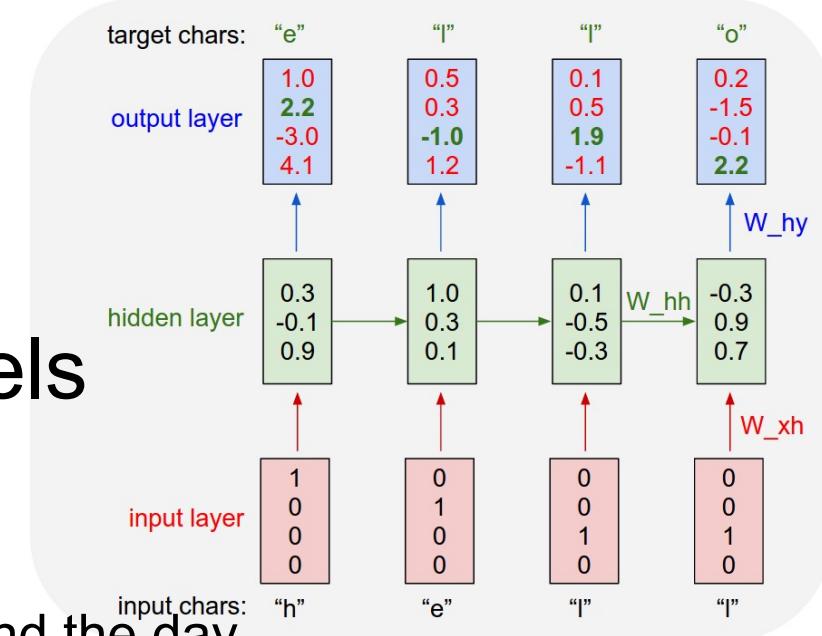
Alas, I think he shall be come approached and the day  
When little strain would be attain'd into being never fed,  
And who is but a chain and subjects of his death,  
I should not sleep.

Second Senator:

They are away this miseries, produced upon my soul,  
Breaking and strongly should be buried, when I perish  
The earth and thoughts of many states.

DUKE VINCENTIO:

Well, your wit is in the care of side and that.



- in practice word RNNs work better, but grow very large
- wordpiece RNNs

# Part II. decoding

$$Q^* = \operatorname{argmax}_Q p(Q|O)$$

$$Q^* = \operatorname{argmax}_Q p(O|Q) \cdot p(Q)$$

- Dynamic programming (a quick tour)
- Dynamic time warping
- HMM Decoding

# Dynamic programming

1. define subproblems
2. identify recurrence
3. set the initial conditions

- Problem: given  $n$ , find the number of different ways to write  $n$  as the sum of 1, 3, 4
  - Example: for  $n = 5$ , the answer is 6

$$5 = 1+1+1+1+1$$

$$= 1+1+3$$

$$= 1+3+1$$

$$= 3+1+1$$

$$= 1+4$$

$$= 4+1$$

## 1. Define subproblems

- Let  $D_n$  be the number of ways to write  $n$  as the sum of 1,3,4

## 2. Find the recurrence

- Consider one possible solution  $n=x_1+x_2+\cdots+x_m$
  - If  $x_m = 1$ , the rest of the terms must sum to  $n-1$
  - Thus, the number of sums that end with  $x_m = 1$  is equal to  $D_{n-1}$
  - if  $x_m=3$ , it is equal to  $D_{n-3}$
  - if  $x_m=4$ , it is equal to  $D_{n-4}$
- $D_n = D_{n-1} + D_{n-3} + D_{n-4}$

### 3. Set the initial conditions

- $D_0 = 1$
- $D_n = 0$  for all negative  $n$

Done!

$$D[0] = D[1] = D[2] = 1$$

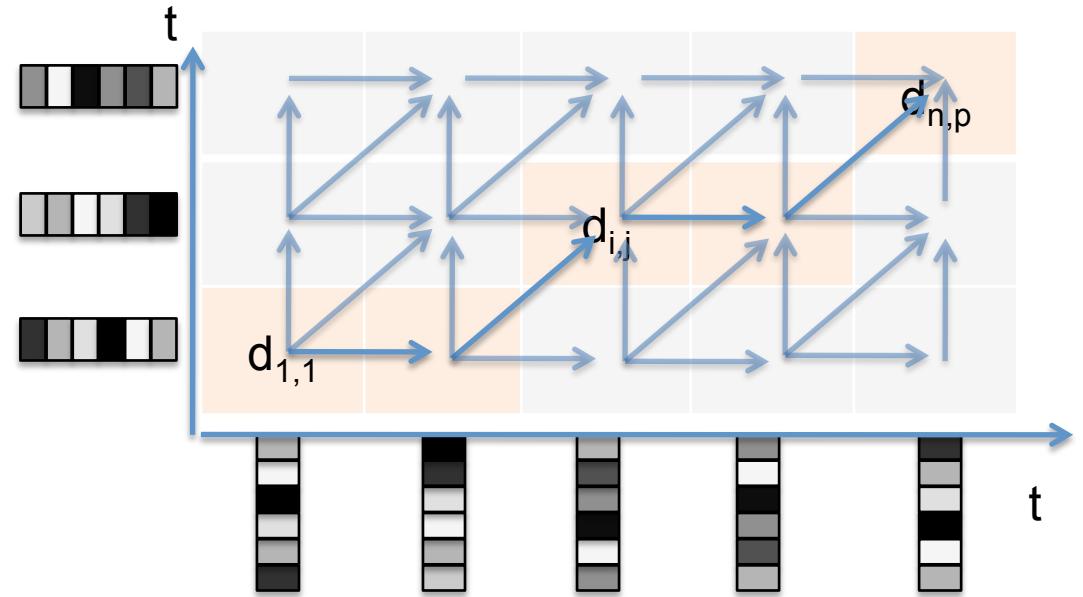
$$D[3] = 2$$

for( $i = 4; i \leq n; i++$ )

$$D[i] = D[i-1] + D[i-3] + D[i-4]$$

# Application: Dynamic Time Warping

Find the *path* which minimizes  $\sum_{\text{path}} d_{i,j}$  (where the path can only go through increasing and adjacent cells)



Dynamic programming analysis

subp:  $C_{ij}$

recur:  $C_{i,j} = d_{i,j} + \min(C_{i,j-1}, C_{i-1,j-1}, C_{i-1,j})$

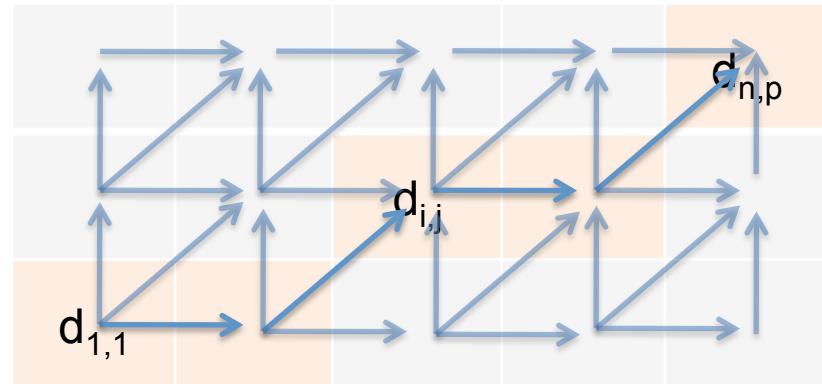
init:  $C_{0,0} = 0, C_{i,0} = \infty, C_{0,j} = \infty$

( $i > 0, j > 0$ )

$\infty$						
$\infty$						
$\infty$						
0	$\infty$	$\infty$	$\infty$	$\infty$	$\infty$	$\infty$

Diagram illustrating the dynamic programming table for DTW. The table has columns labeled  $i=0, 1, 2, 3, 4, 5, 6$  and rows labeled  $j=0, 1, 2, 3, 4, 5, 6$ . The entry  $C_{i,j}$  is shown with arrows indicating its computation from  $C_{i,j-1}$ ,  $C_{i-1,j-1}$ , and  $C_{i-1,j}$ . The value 0 is in the top-left corner ( $i=0, j=0$ ), and all other entries are  $\infty$ .

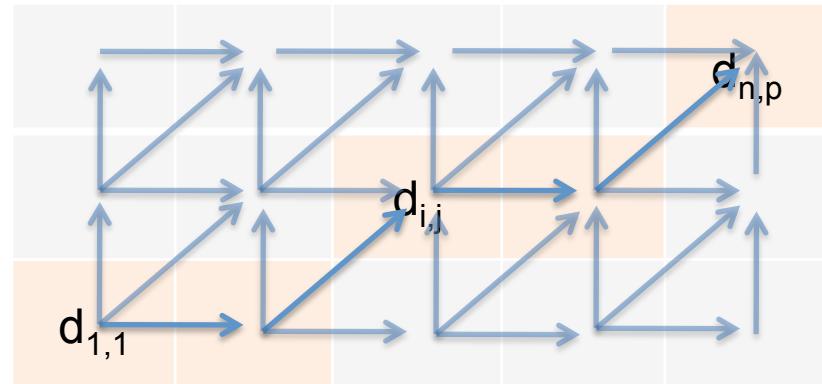
# DTW algorithm



- Initialize

$\infty$	0	0	0	0	0
$\infty$	0	0	0	0	0
$\infty$	0	0	0	0	0
0	$\infty$	$\infty$	$\infty$	$\infty$	$\infty$

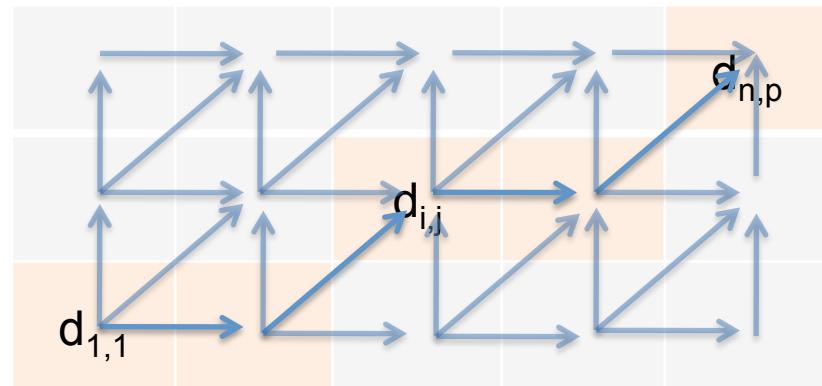
# DTW algorithm



- Apply the recurrence formula  $C_{i,j}=d_{i,j}+\min(C_{i,j-1}, C_{i-1,j-1}, C_{i-1,j})$
- Leave a pointer to which cell you are coming from

$\infty$	0	0	0	0	0
$\infty$	0	0	0	0	0
$\infty$	$d_{1,1}$	0	0	0	0
0	$\infty$	$\infty$	$\infty$	$\infty$	$\infty$

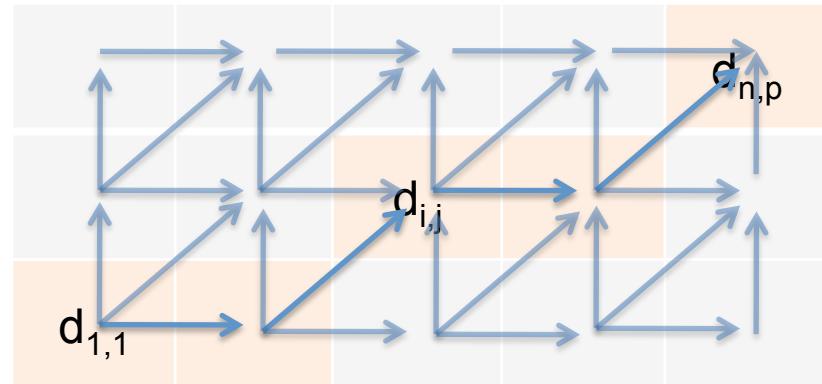
# DTW algorithm



- Apply the recurrence formula  
 $C_{i,j}=d_{i,j}+\min(C_{i,j-1}, C_{i-1,j-1}, C_{i-1,j})$
- Leave a pointer to which cell you are coming from

$\infty$	0	0	0	0	0
$\infty$	$d_{1,1} + d_{2,1}$	0	0	0	0
$\infty$	$d_{1,1}$	0	0	0	0
0	$\infty$	$\infty$	$\infty$	$\infty$	$\infty$

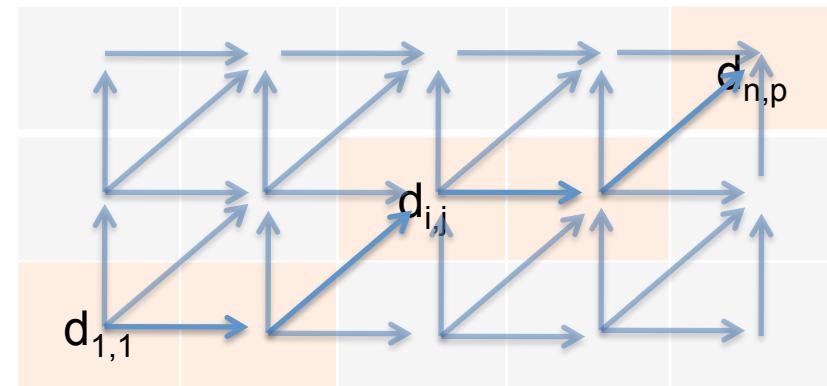
# DTW algorithm



- Apply the recurrence formula  $C_{i,j} = d_{i,j} + \min(C_{i,j-1}, C_{i-1,j-1}, C_{i-1,j})$
- Leave a pointer to which cell you are coming from

$\infty$	$d_{1,1}$ + $d_{2,1}$ + $d_{3,1}$	0	0	0	0
$\infty$	$d_{1,1}$ + $d_{2,1}$	0	0	0	0
$\infty$	$d_{1,1}$	0	0	0	0
0	$\infty$	$\infty$	$\infty$	$\infty$	$\infty$

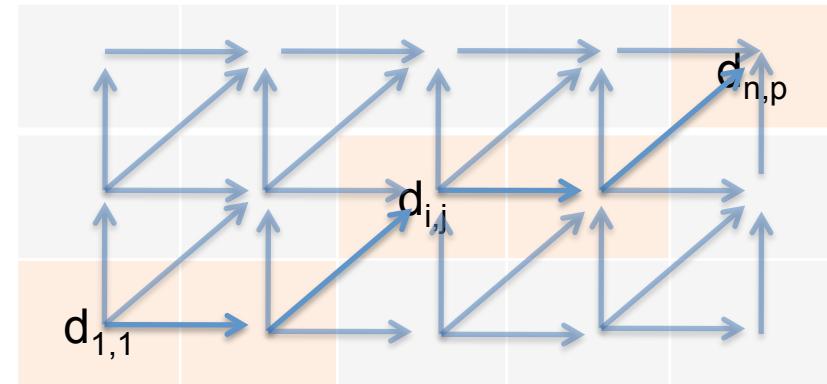
# DTW algorithm



- Apply the recurrence formula  
 $C_{i,j} = d_{i,j} + \min(C_{i,j-1}, C_{i-1,j-1}, C_{i-1,j})$
- Leave a link to which cell you are coming from

$\infty$	$d_{1,1}$ + $d_{2,1}$ + $d_{3,1}$	0	0	0	0
$\infty$	$d_{1,1}$ + $d_{2,1}$	0	0	0	0
$\infty$	$d_{1,1}$	$d_{1,1}$ + $d_{1,2}$	0	0	0
0	$\infty$	$\infty$	$\infty$	$\infty$	$\infty$

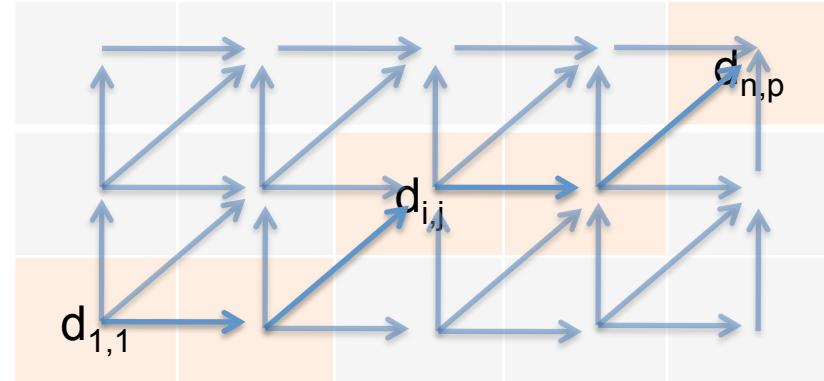
# DTW algorithm



etc...

$\infty$	$d_{1,1}$ + $d_{2,1}$ + $d_{3,1}$	$d_{1,1}$ + $d_{2,1}$ + $d_{3,2}$	$d_{1,1} + d_{2,1}$ + $d_{2,2}$ + $d_{3,3}$	$d_{1,1} + d_{2,1}$ + $d_{2,2}$ + $d_{3,3}$ + $d_{3,4}$	$d_{1,1} + d_{1,2}$ + $d_{2,3}$ + $d_{2,4}$ + $d_{3,5}$
$\infty$	$d_{1,1}$ + $d_{2,1}$	$d_{1,1} + d_{2,1}$ + $d_{2,2}$	$d_{1,1} + d_{1,2}$ + $d_{2,3}$	$d_{1,1} + d_{1,2}$ + $d_{2,3}$ + $d_{2,4}$	$d_{1,1} + d_{1,2}$ + $d_{2,3}$ + $d_{2,4}$ + $d_{2,5}$
$\infty$		$d_{1,1}$ + $d_{1,2}$	$d_{1,1} + d_{1,2}$ + $d_{1,3}$	$d_{1,1} + d_{1,2}$ + $d_{1,3}$ + $d_{1,4}$	$d_{1,1} + d_{1,2}$ + $d_{1,3}$ + $d_{1,4}$ + $d_{1,5}$
0		$\infty$	$\infty$	$\infty$	$\infty$

# DTW algorithm

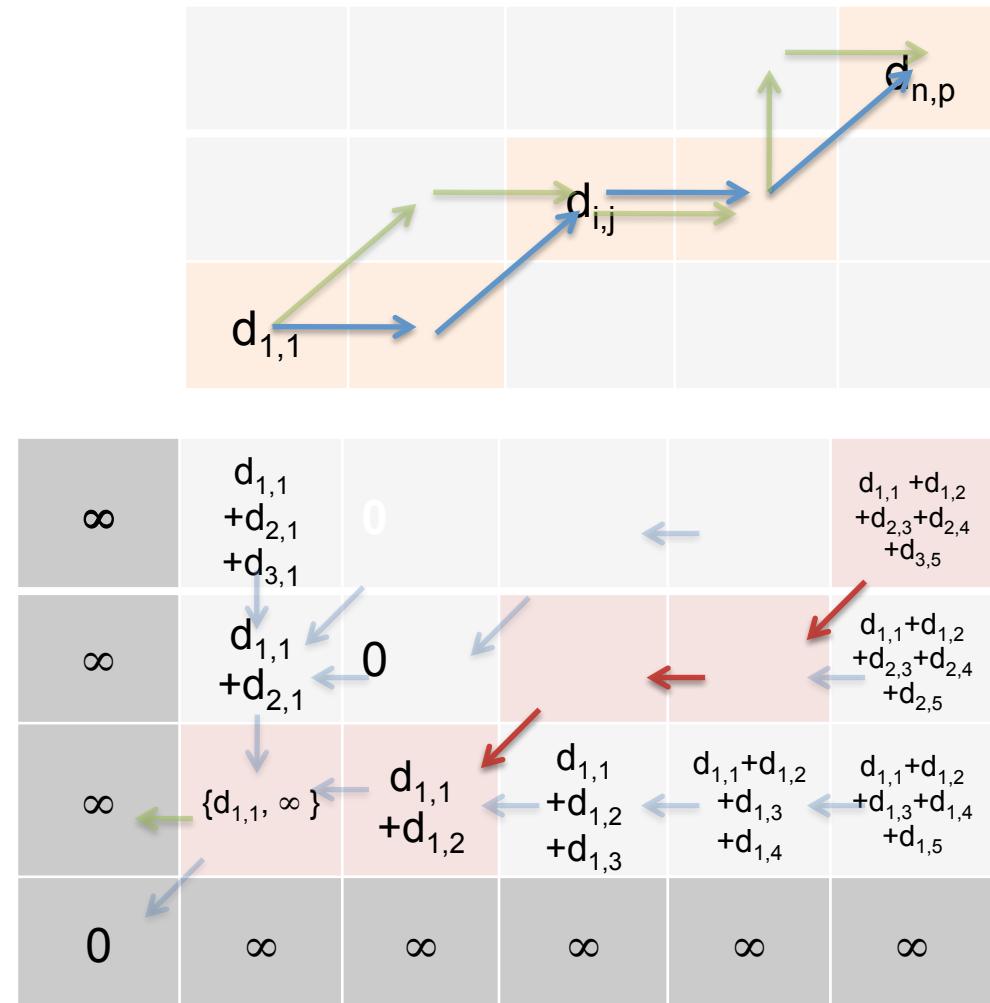


$\infty$	$d_{1,1} + d_{2,1} + d_{3,1}$	$d_{1,1} + d_{2,1} + d_{3,2}$	$d_{1,1} + d_{2,1} + d_{3,3}$	$d_{1,1} + d_{2,1} + d_{3,4}$	$d_{1,1} + d_{2,1} + d_{3,5}$
$\infty$	$d_{1,1} + d_{2,1}$	$d_{1,1} + d_{2,1} + d_{2,2}$	$d_{1,1} + d_{2,1} + d_{2,3}$	$d_{1,1} + d_{2,1} + d_{2,4}$	$d_{1,1} + d_{2,1} + d_{2,5}$
$\infty$	$d_{1,1}$	$d_{1,1} + d_{1,2}$	$d_{1,1} + d_{1,2} + d_{1,3}$	$d_{1,1} + d_{1,2} + d_{1,4}$	$d_{1,1} + d_{1,2} + d_{1,5}$
0	$\infty$	$\infty$	$\infty$	$\infty$	$\infty$

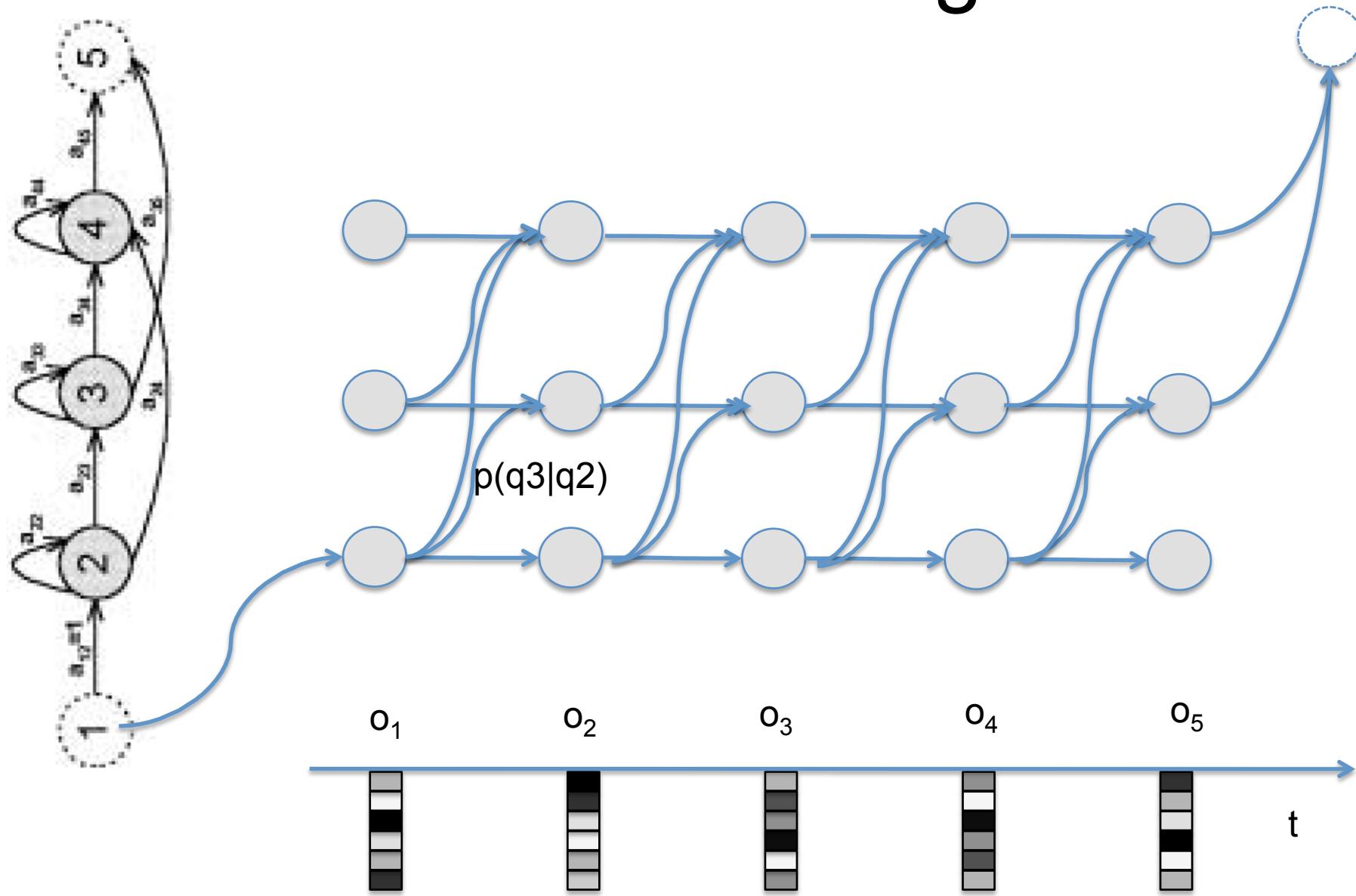
- Start from the final point
- Follow the path backward

# Variation: nbest

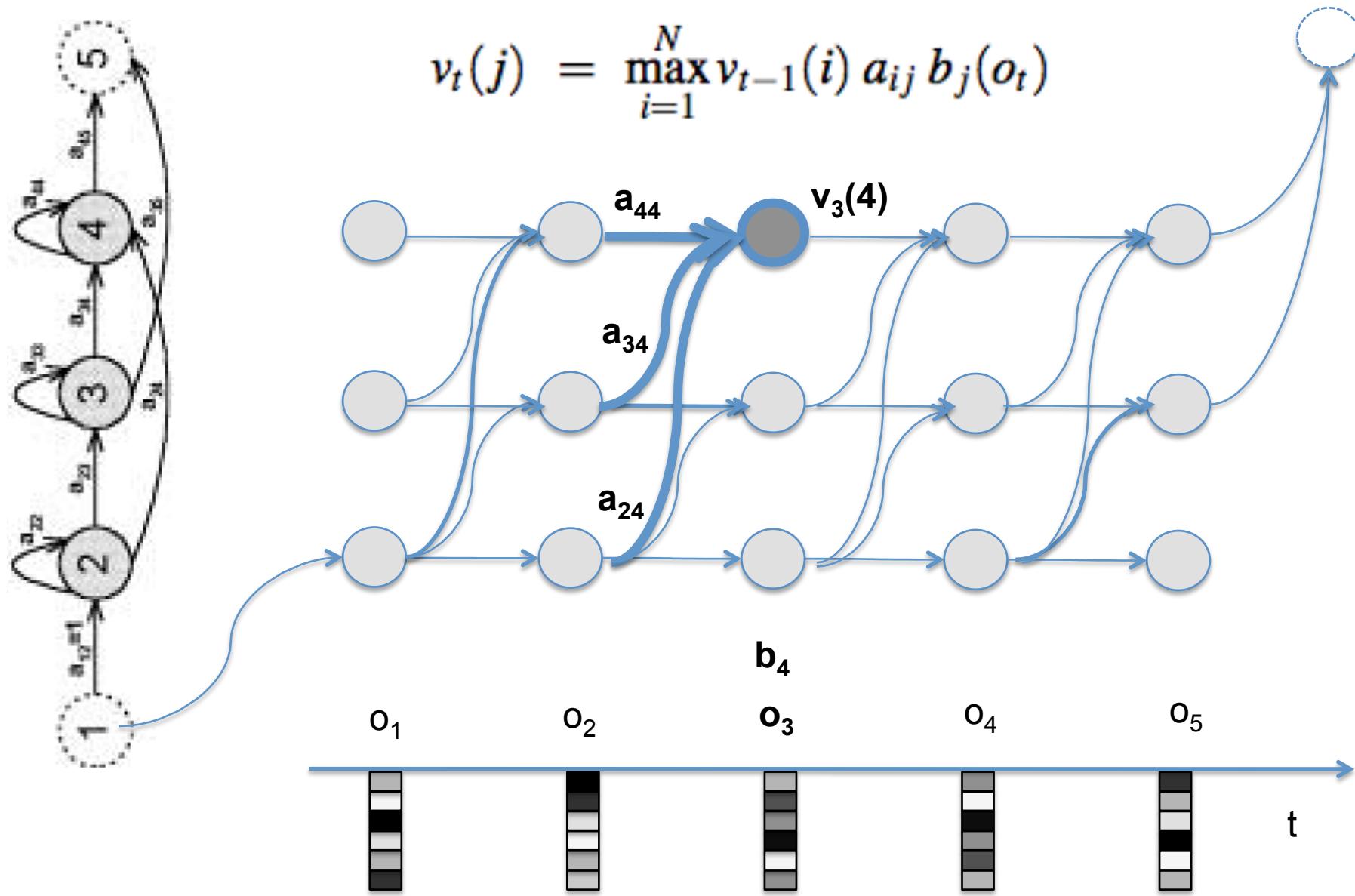
- Change the recurrence formula  
 $C_{i,j} = d_{i,j} + \text{best}(n, C_{i,j-1} \cup C_{i-1,j-1} \cup C_{i-1,j})$   
 where  $\text{best}(n, S)$ =the n smallests elements in set S.
- Memorize n links backward



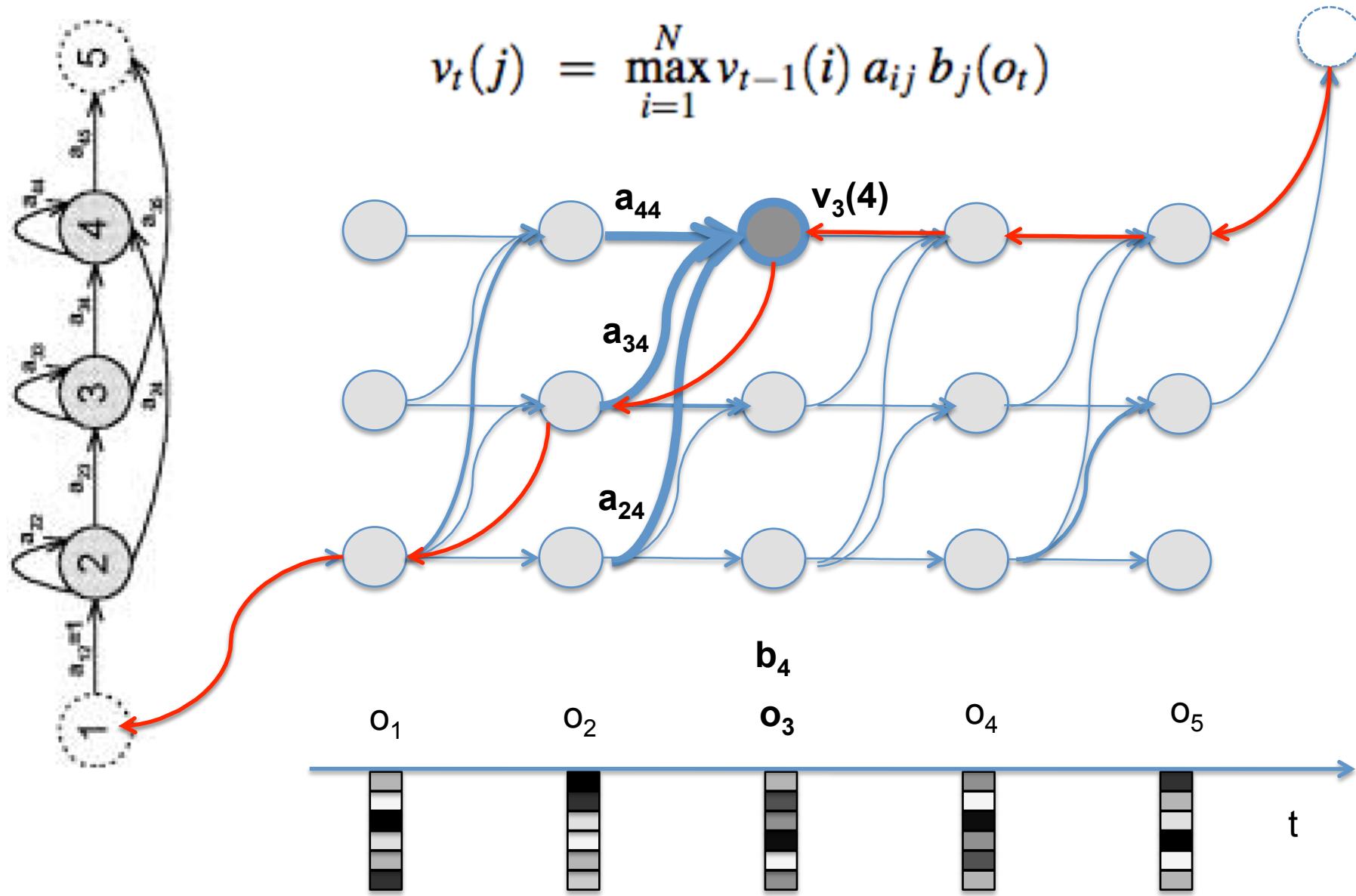
# HMM decoding



# decoding: $Q^* = \operatorname{argmax}_Q P(O|Q)$



# decoding: $Q^* = \operatorname{argmax}_Q P(O|Q)$



# Learning

# Learning: Parameter estimation

- $\Theta^* = \arg \max_{\Theta} p(O|\Theta)$
- EM (Baum Welch, forward backward) algorithm:
  - E step: estimate distribution of hidden variable Q (given O and  $\Theta$ ):  $p(Q|O,\Theta)$
  - M step: maximize  $\Theta$  given the estimate of Q:  
 $\Theta^* = \arg \max_{\Theta} p(O|Q,\Theta)$
  - Iterate

# an example: GMMs

- mixture model

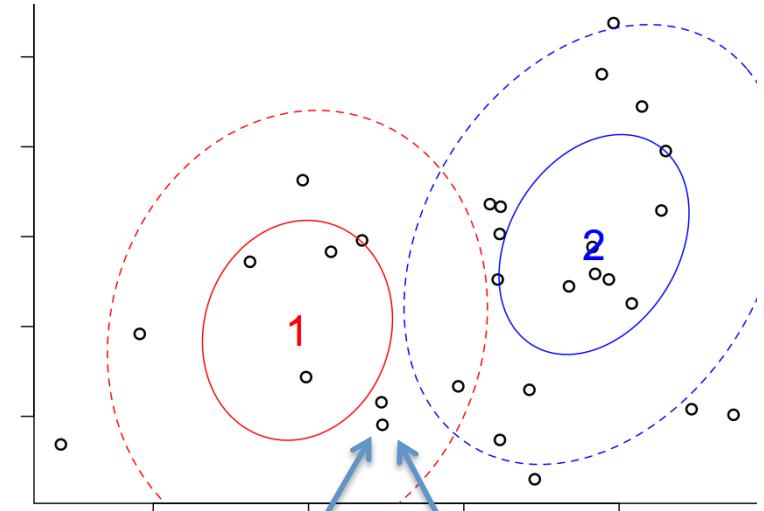
$$p(x) = \sum_{j=1}^K w_j \cdot N(x | \mu_j, \Sigma_j)$$

- problem: given observations  $O = \{o_1..o_N\}$

$$\theta^* = \arg \max_{\theta} \prod_i p(o_i | \theta)$$

- introduce hidden cluster membership  $q_i = 1..k$

$$\theta^* = \arg \max_{\theta} \prod_i p(o_i, q_i | \theta)$$



$$p(1|o_i)$$

$$p(2|o_i)$$

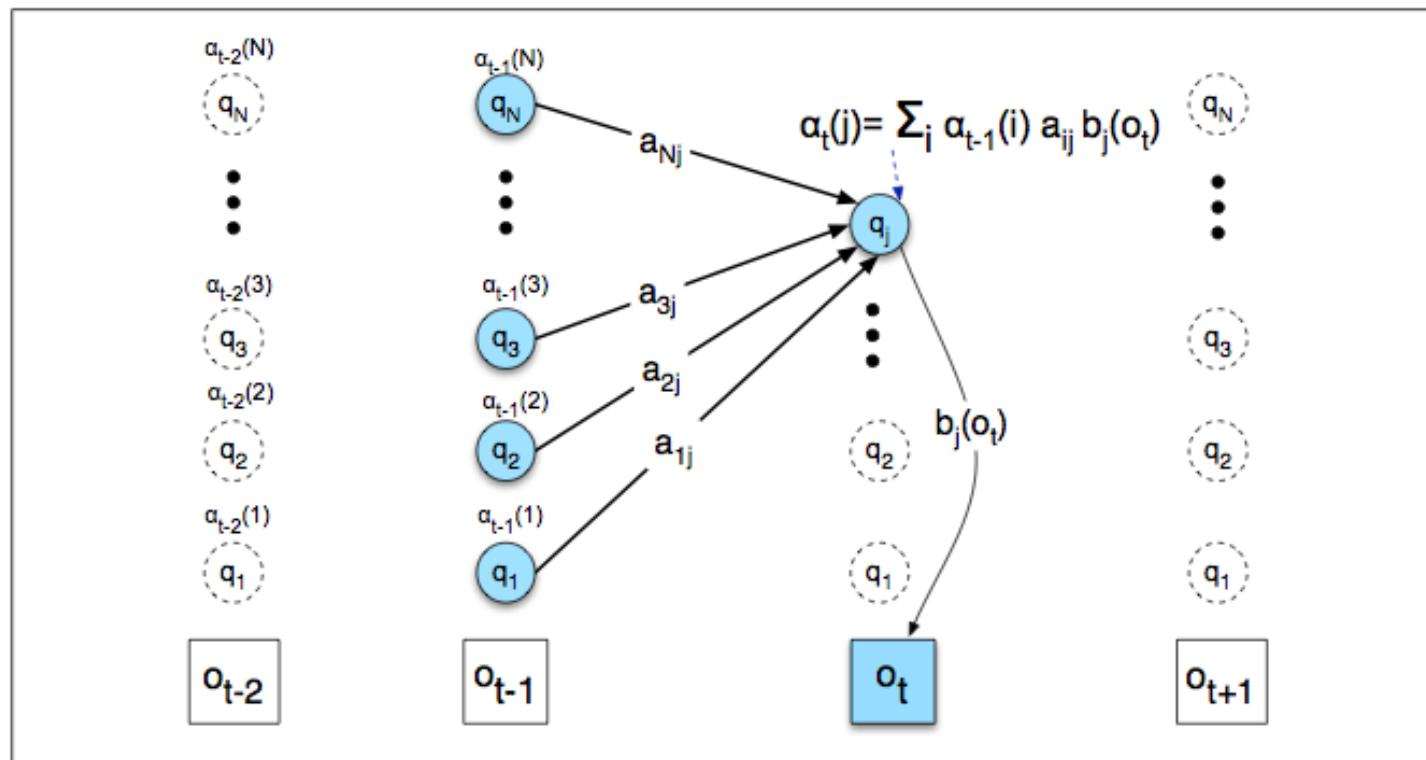
$$\mu_j = \frac{\sum_{i=1}^N p(j | o_i, \theta^t) o_i}{\sum_{i=1}^N p(j | o_i, \theta^t)}$$

idem for  $\Sigma_j$

# going back to HMMs

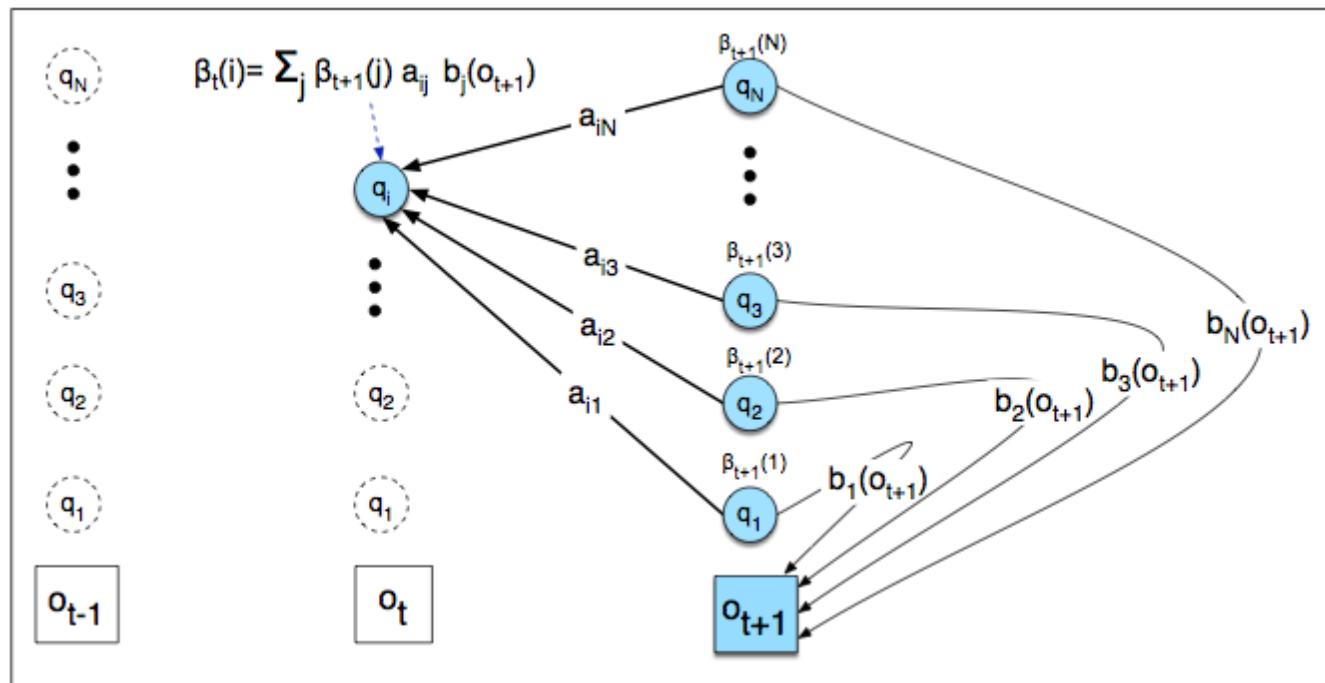
- intuition
  - in a markov chain:
  - in an HMM: iteratively estimate the counts
    - start with an estimate  $a_{ij}$  and  $b_j$ , use them to decode and get better estimates
    - compute these estimates on all paths, weighted by the probability of the paths

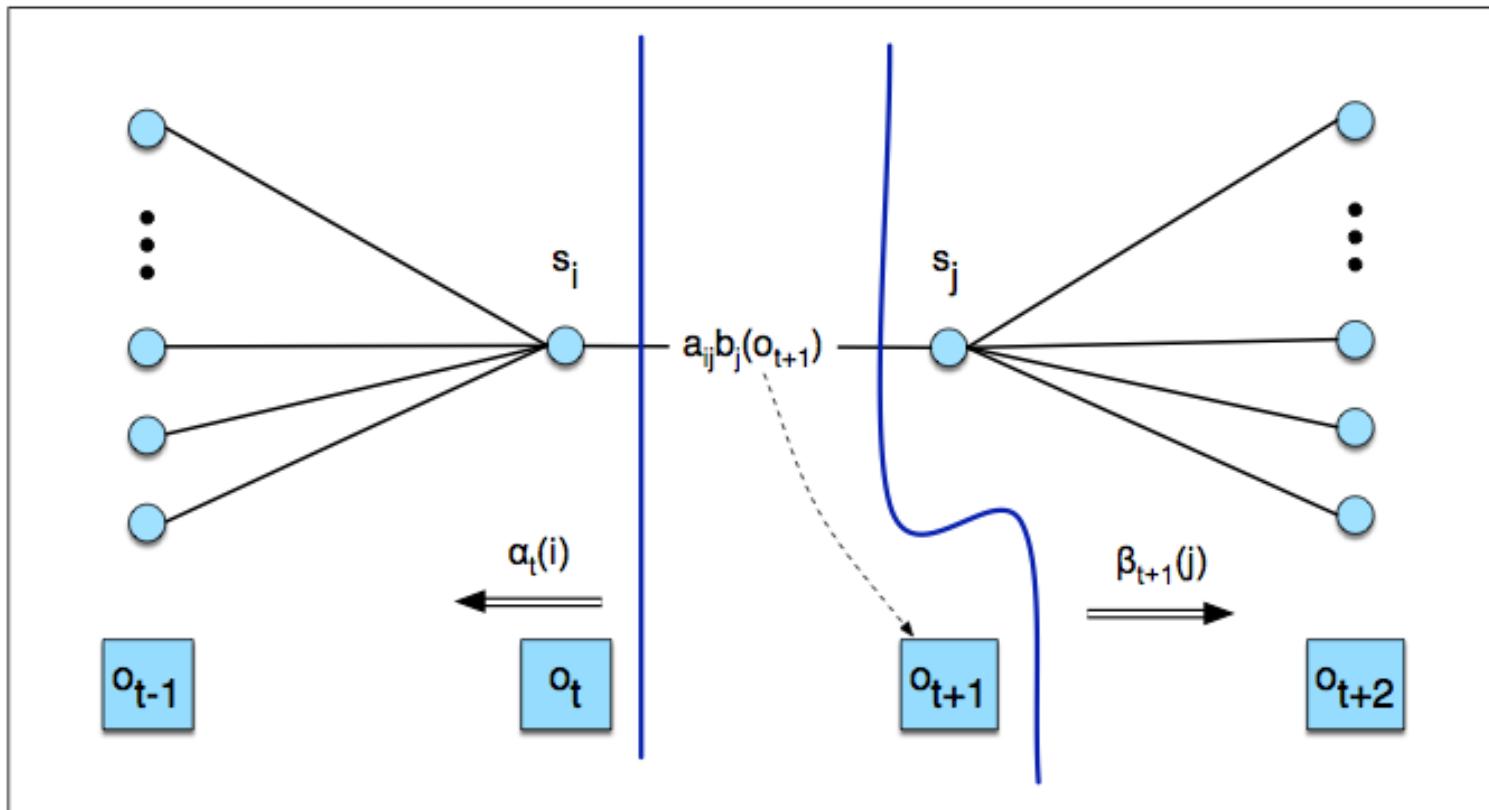
- forward probabilities  $\alpha_t(j) = P(o_1, o_2 \dots o_t, q_t = j | \Theta)$



from J&M (2017)

- backward probabilities  $\beta_t(i) = P(o_{t+1}, o_{t+2} \dots o_T | q_t = i, \Theta)$





$$P(q_t = i, q_{t+1} = j, O | \Theta) = \alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)$$

$$\xi_t(i, j) = P(q_t = i, q_{t+1} = j | O, \Theta) = \frac{\alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}{\alpha_T(q_F)}$$

from J&M (2017)

# Baum-Welch algorithm

- E step

$$\xi_t(i, j) = \frac{\alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}{\alpha_T(q_F)} \quad \gamma_t(j) = \frac{\alpha_t(j) \beta_t(j)}{\alpha_T(q_F)}$$

- M Step

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i, j)}{\sum_{t=1}^{T-1} \sum_{k=1}^N \xi_t(i, k)}$$

$$\hat{b}_j(v_k) = \frac{\sum_{t=1}^T \gamma_t(j) \text{ s.t. } O_t = v_k}{\sum_{t=1}^T \gamma_t(j)}$$

# References

- Language models:
  - Jurasky & Martin (2017). Chapter 4. Language modeling with N-grams
- Dynamic programming:
  - Jaehyun Park (2015). Course materials for CS 97SI, Stanford University
- Viterbi decoding, Baum Welch
  - Jurafsky & Martin (2017). Chapter 9. Hidden Markov Models
- to know more
  - Gales & Young (2007). The application of Hidden Markow Models in Speech Recognition. *Foundations and Trends in Signal Processing*, 1(3), 195-304.