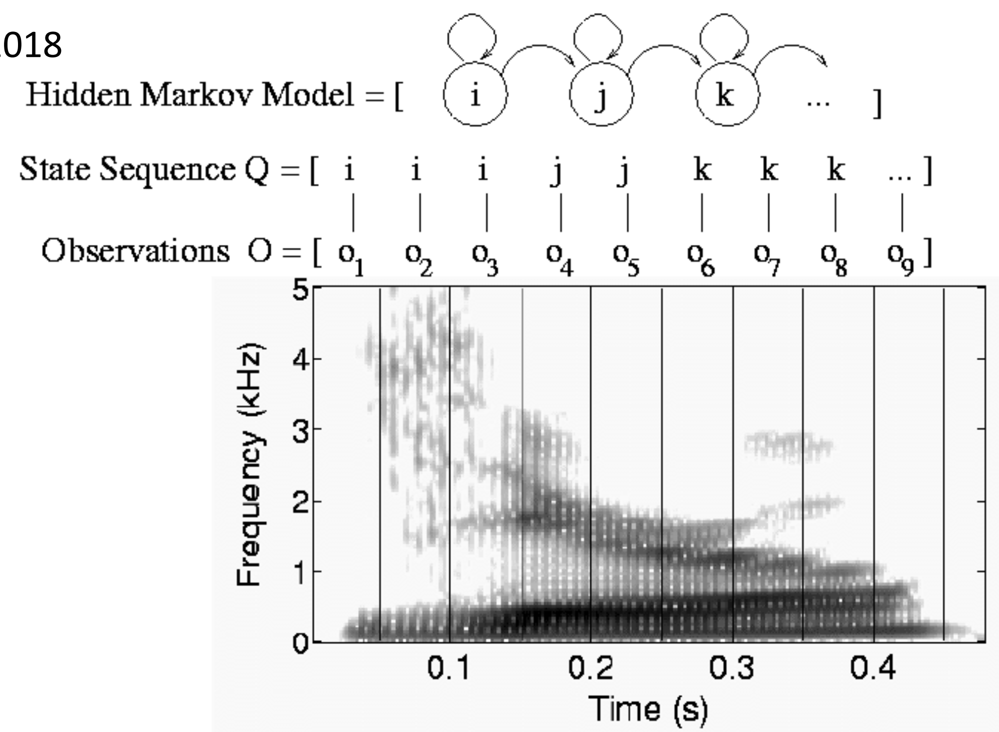
CS440/ECE448 Lecture 19: Hidden Markov Models

Slides by Svetlana Lazebnik, 11/2016

Modified by Mark Hasegawa-Johnson, 4/2018

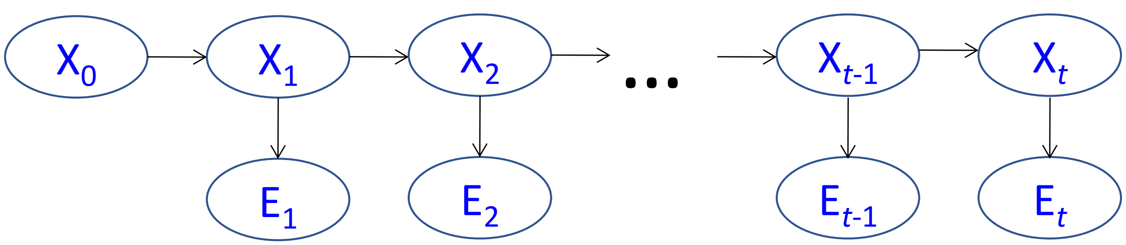


Probabilistic reasoning over time

- So far, we've mostly dealt with *episodic* environments
  - Exceptions: games with multiple moves, planning
- In particular, the Bayesian networks we've seen so far describe static situations
  - Each random variable gets a single fixed value in a single problem instance
- Now we consider the problem of describing probabilistic environments that evolve over time
  - Examples: robot localization, human activity detection, tracking, speech recognition, machine translation,

Hidden Markov Models

- At each time slice t, the state of the world is described by an unobservable variable X_t and an observable *evidence* variable E_t
- **Transition model:** distribution over the current state given the whole past history:
  $P(X_t \mid X_0, \ldots, X_{t-1}) = P(X_t \mid \mathbf{X}_{0:t-1})$
- **Observation model:** $P(E_t \mid \mathbf{X}_{0:t}, \mathbf{E}_{1:t-1})$

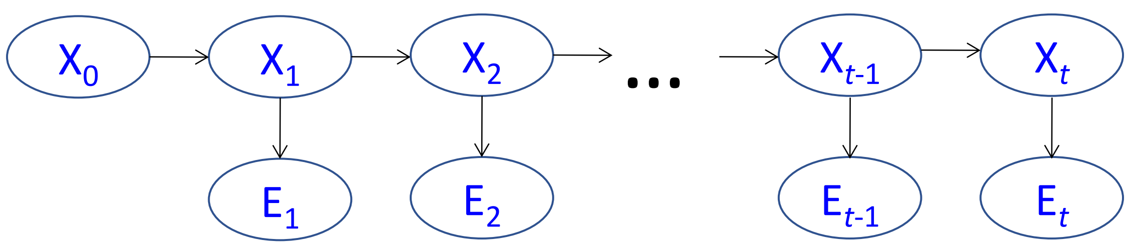


Hidden Markov Models

- **Markov assumption** (first order)
  - The current state is conditionally independent of all the other states given the state in the previous time step
  - What does $P(X_t \mid \mathbf{X}_{0:t-1})$ simplify to?

    $P(X_t \mid \mathbf{X}_{0:t-1}) = P(X_t \mid X_{t-1})$
- Markov assumption for observations
  - The evidence at time *t* depends only on the state at time *t*
  - What does $P(E_t \mid \mathbf{X}_{0:t}, \mathbf{E}_{1:t-1})$ simplify to?

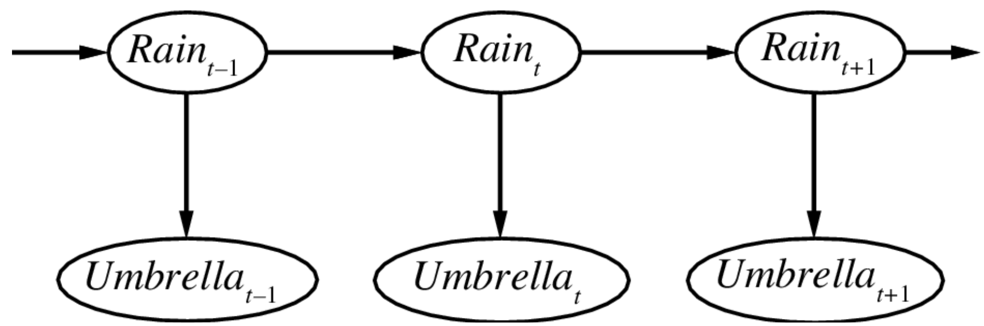
    $P(E_t \mid \mathbf{X}_{0:t}, \mathbf{E}_{1:t-1}) = P(E_t \mid X_t)$



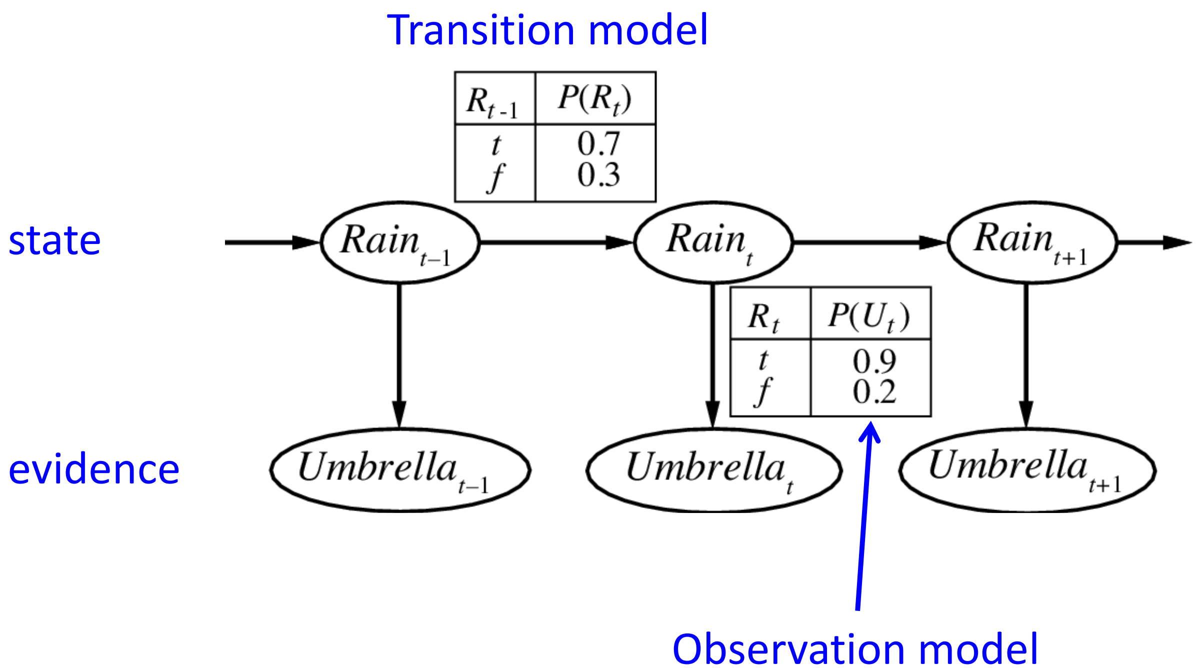
Example

state

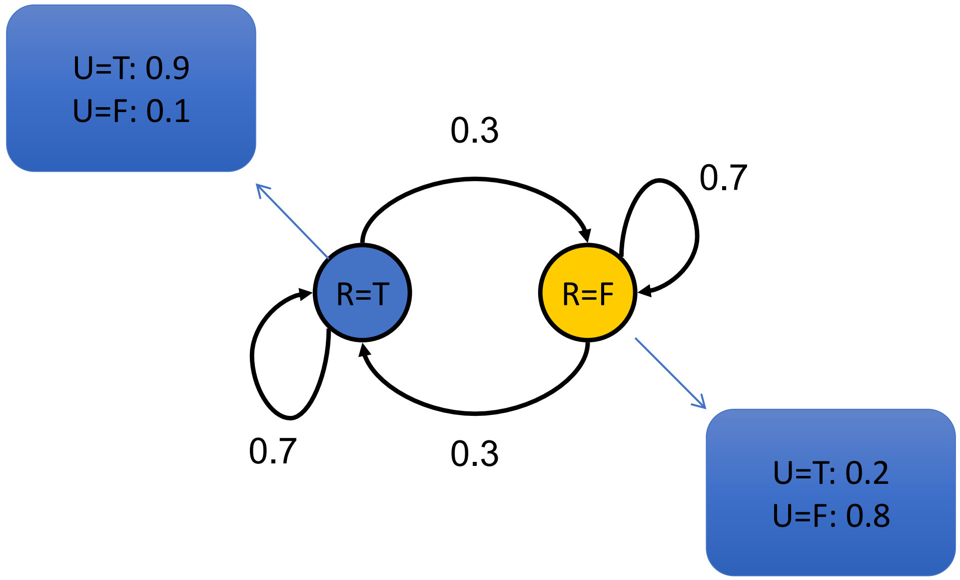
evidence



Example


Transition model
R_{t-1} | $P(R_t)$
t | 0.7
f | 0.3
state
$Rain_{t-1}$
$Rain_t$
$Rain_{t+1}$
R_t | $P(U_t)$
t | 0.9
f | 0.2
evidence
$Umbrella_{t-1}$
$Umbrella_t$
$Umbrella_{t+1}$
Observation model

An alternative visualization



Transition probabilities

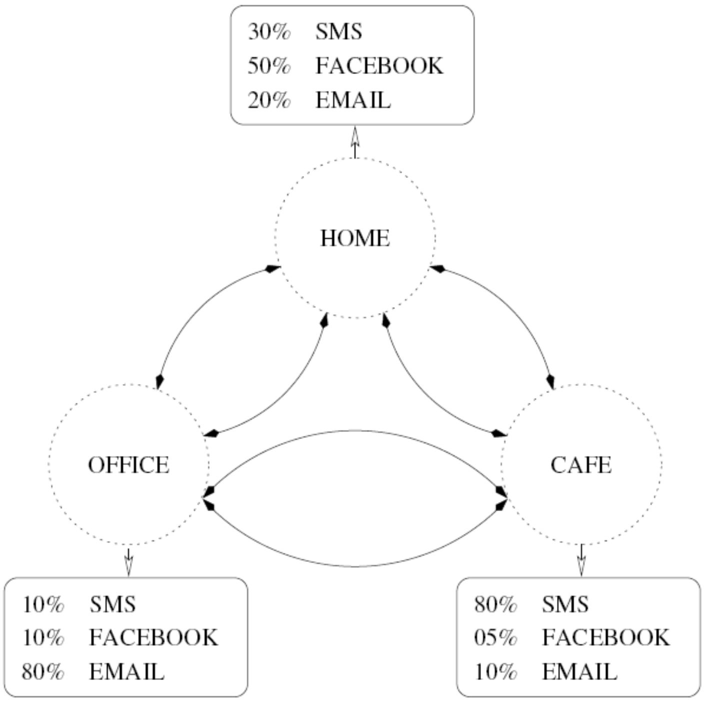
	R_t = T	R_t = F
R_{t-1} = T	0.7	0.3
R_{t-1} = F	0.3	0.7

Observation (emission) probabilities

	U_t = T	U_t = F
R_t = T	0.9	0.1
R_t = F	0.2	0.8

Another example

- **States:** X = {home, office, cafe}
- **Observations:** E = {sms, facebook, email}



Transition Probabilities

	home	office	cafe
home	0.2	0.6	0.2
office	0.5	0.2	0.3
cafe	0.2	0.8	0.0

Emission Probabilities

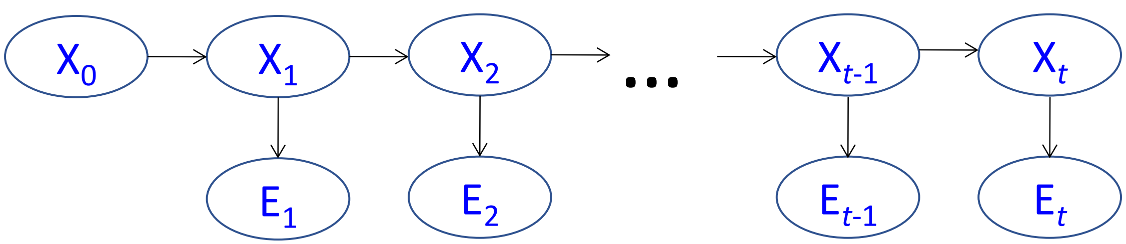
	sms	facebook	email
home	0.3	0.5	0.2
office	0.1	0.1	0.8
cafe	0.8	0.1	0.1

Slide credit: Andy White

The Joint Distribution

- Transition model: $P(X_t \mid \mathbf{X}_{0:t\text{-}1}) = P(X_t \mid X_{t\text{-}1})$
- Observation model: $P(E_t \mid \mathbf{X}_{0:t}, \mathbf{E}_{1:t\text{-}1}) = P(E_t \mid X_t)$
- How do we compute the full joint $P(\mathbf{X}_{0:t}, \mathbf{E}_{1:t})$?

$$P(\boldsymbol{X}_{0:t}, \boldsymbol{E}_{1:t}) = P(X_0) \prod_{i=1}^{t} P(X_i | X_{i-1}) P(E_i | X_i)$$

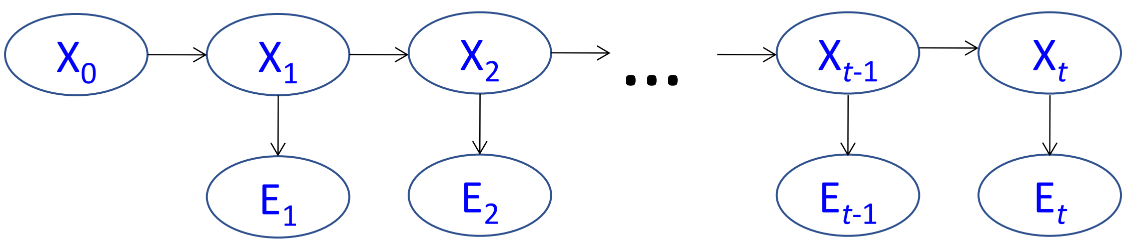


Review: Bayes net inference

- Inference:
  - Trees: Sum-Product Algorithm (Textbook: “Variable Elimination” Algorithm)
  - Other Nets: Junction Tree Algorithm (Textbook: ”Join Tree” Algorithm)
  - In General: NP-Complete, because clique size = graph size in general
- Parameter learning
  - Fully observed: Count # times each event occurs
  - Partially observed: Expectation-Maximization algorithm
    - Estimate Probability of each event at each time
    - E[# times event occurs] = sum_t(Probability event occurs at time t)

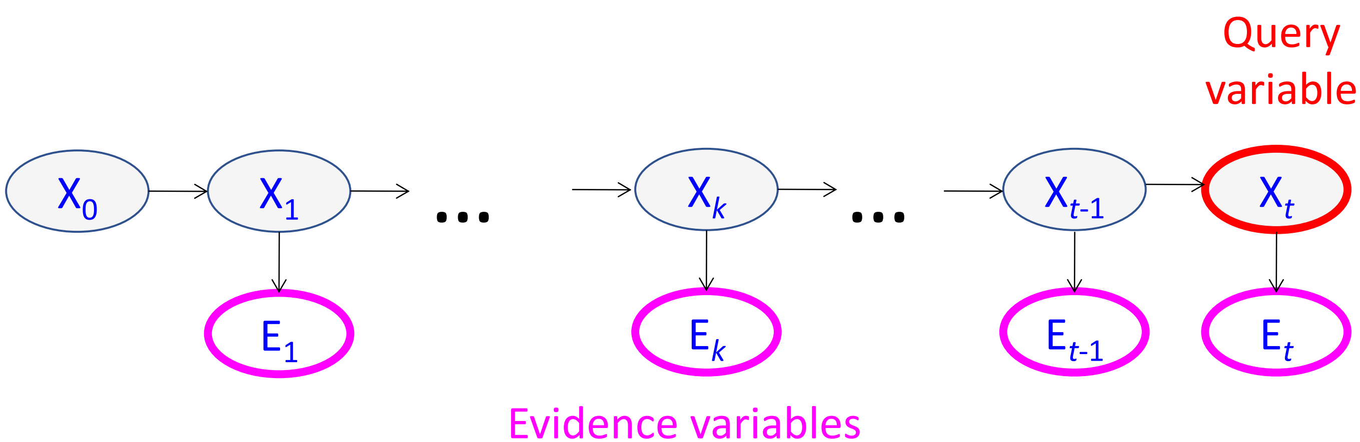
Sum-Product Algorithm for HMMs

- An HMM is a tree!
- For example, suppose we want to find P(X3|E1,E2,E3)
- Product: P(X0,X1,E1)=P(X0)P(X1|X0)P(E1|X1)
- Sum: P(X1|E1)=P(X1,E1)/P(E1)
- Product: P(X1,X2,E2|E1)=P(X1|E1)P(X2|X1)P(E2|X2)
- Sum: P(X2|E1,E2)=P(X2,E2|E1)/P(E2|E1)
- …



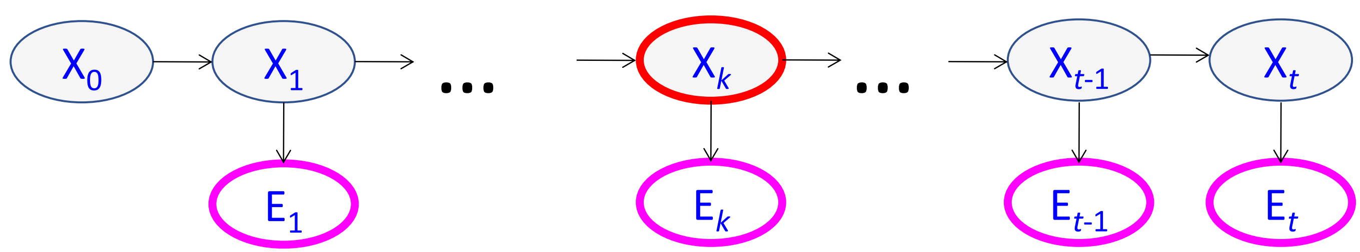
HMM inference tasks

- **Filtering:** what is the distribution over the current state X_t given all the evidence so far, $\mathbf{e}_{1:t}$?
  - The forward algorithm = sum-product algorithm for Xt given e1:t



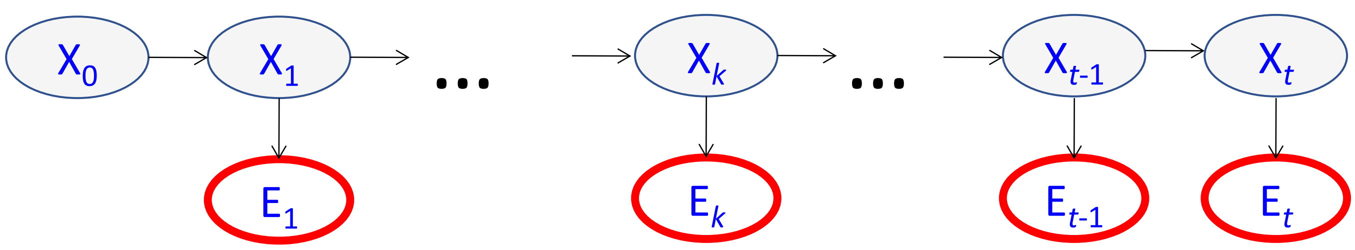
HMM inference tasks

- **Filtering:** what is the distribution over the current state X_t given all the evidence so far, $\mathbf{e}_{1:t}$?
- **Smoothing:** what is the distribution of some state X_k given the entire observation sequence $\mathbf{e}_{1:t}$?
  - The forward-backward algorithm = sum-product algorithm for Xk given e1:t, when 1 < k < t
  - Xk = query variable, unknown, need to consider all its possible values
  - E1:t = evidence variables, known, only need to consider the given values



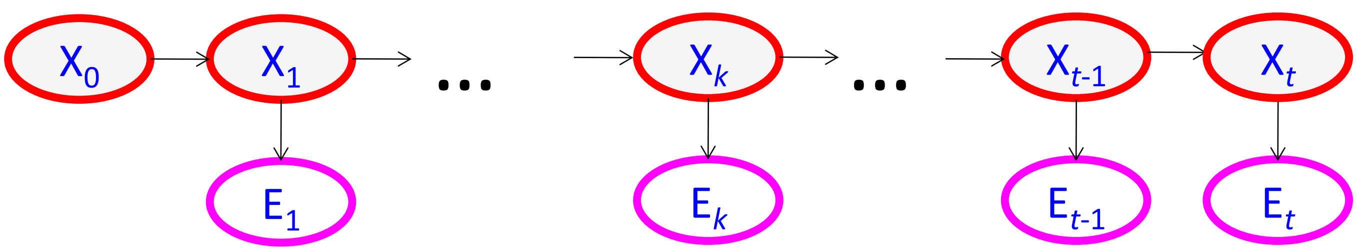
HMM inference tasks

- **Filtering:** what is the distribution over the current state X_t given all the evidence so far, $\mathbf{e}_{1:t}$?
- **Smoothing:** what is the distribution of some state X_k given the entire observation sequence $\mathbf{e}_{1:t}$?
- **Evaluation:** compute the probability of a given observation sequence $\mathbf{e}_{1:t}$



HMM inference tasks

- **Filtering:** what is the distribution over the current state X_t given all the evidence so far, $\mathbf{e}_{1:t}$
- **Smoothing:** what is the distribution of some state X_k given the entire observation sequence $\mathbf{e}_{1:t}$?
- **Evaluation:** compute the probability of a given observation sequence $\mathbf{e}_{1:t}$
- **Decoding:** what is the most likely state sequence $\mathbf{X}_{0:t}$ given the observation sequence $\mathbf{e}_{1:t}$?
  - The Viterbi algorithm



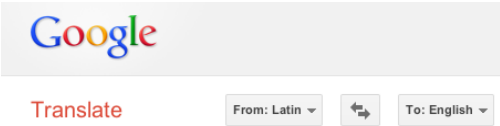
HMM Learning and Inference

- Inference tasks
  - **Filtering:** what is the distribution over the current state X_t given all the evidence so far, $\mathbf{e}_{1:t}$
  - **Smoothing:** what is the distribution of some state X_k given the entire observation sequence $\mathbf{e}_{1:t}$?
  - **Evaluation:** compute the probability of a given observation sequence $\mathbf{e}_{1:t}$
  - **Decoding:** what is the most likely state sequence $\mathbf{X}_{0:t}$ given the observation sequence $\mathbf{e}_{1:t}$?
- Learning
  - Given a training sample of sequences, learn the model parameters (transition and emission probabilities)
    - EM algorithm

Applications of HMMs

- Speech recognition HMMs:
  - Observations are acoustic signals (continuous valued)
  - States are specific positions in specific words (so, tens of thousands)

- Machine translation HMMs:
  - Observations are words (tens of thousands)
  - States are translation options

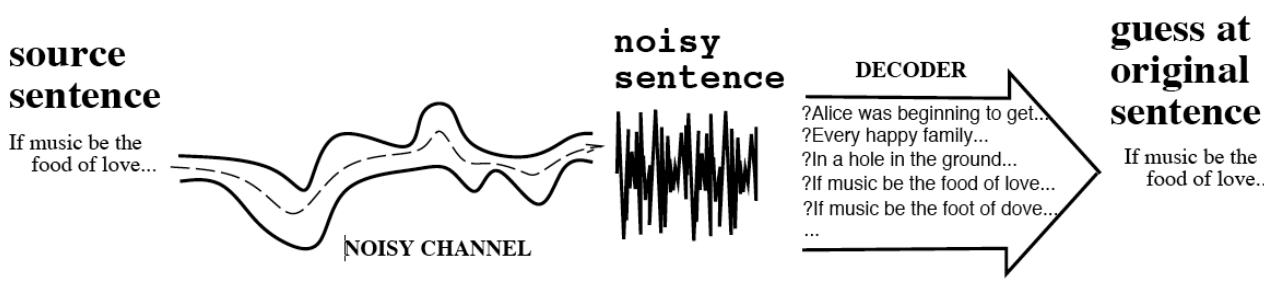


- Robot tracking:
  - Observations are range readings (continuous)
  - States are positions on a map (continuous)

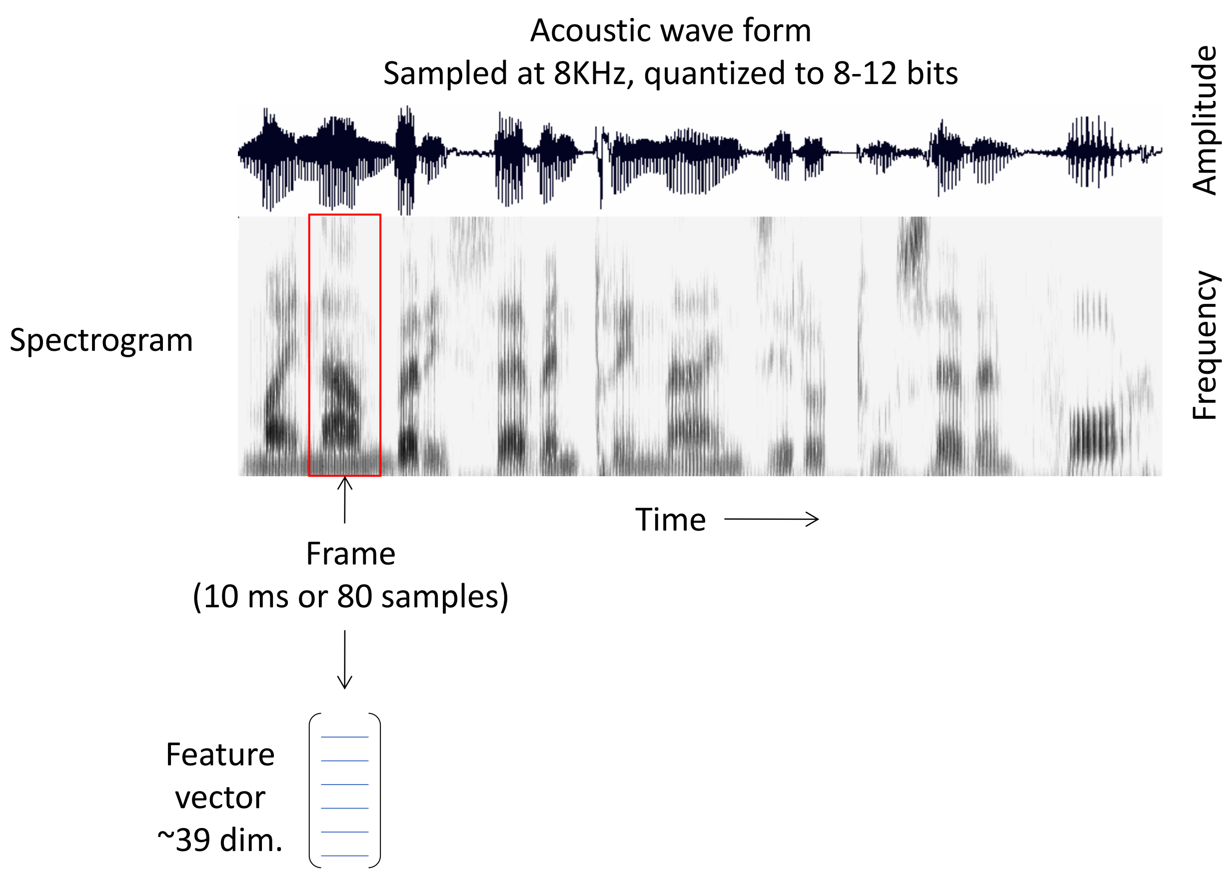
Source: Tamara Berg

Application of HMMs: Speech recognition

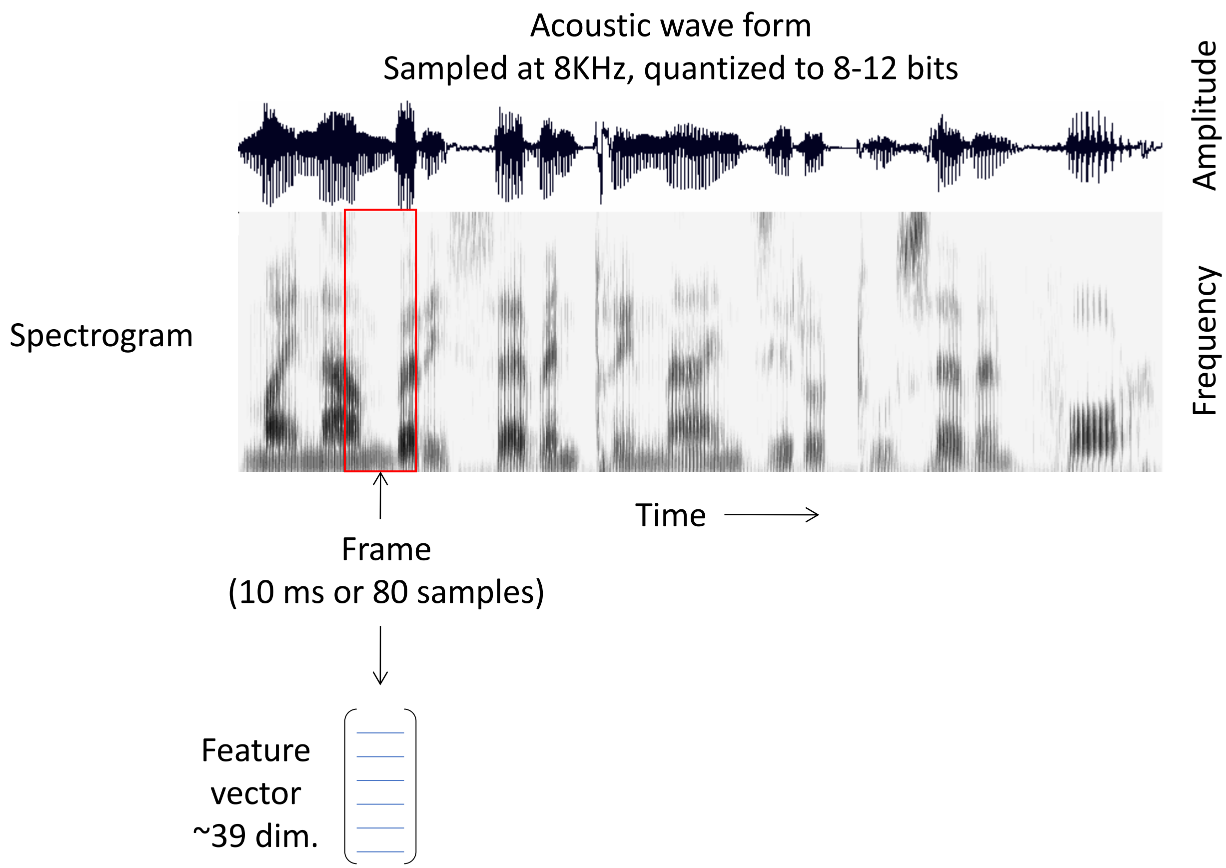
- "Noisy channel" model of speech



Speech feature extraction



Speech feature extraction



Phonetic model

- **Phones:** speech sounds
- **Phonemes:** groups of speech sounds that have a unique meaning/function in a language (e.g., there are several different ways to pronounce “t”)

Phonetic model

IPA Symbol	ARPAbet Symbol	Word	IPA Transcription	ARPAbet Transcription
[p]	[p]	parsley	[ˈparsli]	[p aa r s l iy]
[t]	[t]	tarragon	[ˈtærəgɑn]	[t ae r ax g aa n]
[k]	[k]	catnip	[ˈkætnɨp]	[k ae t n ix p]
[b]	[b]	bay	[beɪ]	[b ey]
[d]	[d]	dill	[dɪl]	[d ih l]
[g]	[g]	garlic	[ˈgɑrlɨk]	[g aa r l ix k]
[m]	[m]	mint	[mɪnt]	[m ih n t]
[n]	[n]	nutmeg	[ˈnʌtmɛg]	[n ah t m eh g
[ŋ]	[ng]	ginseng	[ˈdʒɪnsɨŋ]	[jh ih n s ix ng]
[f]	[f]	fennel	[ˈfɛnl̩]	[f eh n el]
[v]	[v]	clove	[kloʊv]	[k l ow v]
[θ]	[th]	thistle	[ˈθɪsl̩]	[th ih s el]
[ð]	[dh]	heather	[ˈhɛðɚ]	[h eh dh axr]
[s]	[s]	sage	[seɪdʒ]	[s ey jh]
[z]	[z]	hazelnut	[ˈheɪzl̩nʌt]	[h ey z el n ah t]
[ʃ]	[sh]	squash	[skwɑʃ]	[s k w a sh]
[ʒ]	[zh]	ambrosia	[æmˈbroʊʒə]	[ae m b r ow zh ax]
[tʃ]	[ch]	chicory	[ˈtʃɪkɚi]	[ch ih k axr iy]
[dʒ]	[jh]	sage	[seɪdʒ]	[s ey jh]
[l]	[l]	licorice	[ˈlɪkɚɨʃ]	[l ih k axr ix sh]
[w]	[w]	kiwi	[ˈkiwi]	[k iy w iy]
[r]	[r]	parsley	[ˈpɑrsli]	[p aa r s l iy]
[j]	[y]	yew	[yu]	[y uw]
[h]	[h]	horseradish	[ˈhɔrsrædɪʃ]	[h ao r s r ae d ih sh]
[ʔ]	[q]	uh-oh	[ʔʌʔoʊ]	[q ah q ow]
[ɾ]	[dx]	butter	[ˈbʌɾɚ]	[b ah dx axr]
[ɾ̃]	[nx]	wintergreen	[wɪɾ̃ɚgrin]	[w ih nx axr g r i n]
[l̩]	[el]	thistle	[ˈθɪsl̩]	[th ih s el]

Figure 4.1 IPA and ARPAbet symbols for transcription of English consonants.

IPA Symbol	ARPAbet Symbol	Word	IPA Transcription	ARPAbet Transcription
[i]	[iy]	lily	[ˈlɪli]	[l ih l iy]
[ɪ]	[ih]	lily	[ˈlɪli]	[l ih l iy]
[eɪ]	[ey]	daisy	[ˈdeɪzi]	[d ey z i]
[ɛ]	[eh]	poinsettia	[pɔɪnˈsɛriə]	[p oy n s eh dx iy ax]
[æ]	[ae]	aster	[ˈæstɚ]	[ae s t axr]
[ɑ]	[aa]	poppy	[ˈpɑpi]	[p aa p i]
[ɔ]	[ao]	orchid	[ˈɔrkɨd]	[ao r k ix d]
[ʊ]	[uh]	woodruff	[ˈwʊdrʌf]	[w uh d r ah f]
[oʊ]	[ow]	lotus	[ˈloʊɾəs]	[l ow dx ax s]
[u]	[uw]	tulip	[ˈtulɨp]	[t uw l ix p]
[ʌ]	[uh]	buttercup	[ˈbʌɾɚˌkʌp]	[b uh dx axr k uh p]
[ɝ]	[er]	bird	[ˈbɝd]	[b er d]
[aɪ]	[ay]	iris	[ˈaɪrɨs]	[ay r ix s]
[aʊ]	[aw]	sunflower	[ˈsʌnflaʊɚ]	[s ah n f l aw axr]
[ɔɪ]	[oy]	poinsettia	[pɔɪnˈsɛriə]	[p oy n s eh dx iy ax]
[ju]	[y uw]	feverfew	[fivɚfju]	[f iy v axr f y u]
[ə]	[ax]	woodruff	[ˈwʊdrəf]	[w uh d r ax f]
[ɚ]	[axr]	heather	[ˈhɛðɚ]	[h eh dh axr]
[ɨ]	[ix]	tulip	[ˈtulɨp]	[t uw l ix p]
[ʉ]	[ux]		[]	[]

Figure 4.2 IPA and ARPAbet symbols for transcription of English vowels

HMM models for phones

HMM states in most speech recognition systems correspond to ***subsegments*** *of* ***triphones***

- ***Triphone***: the /b/ in "about" (ax-b+aw) sounds different from the /b/ in "Abdul" (ae-b+d). There are around 60 phones and as many as 60^3 context-dependent *triphones.*
- ***Subsegments***: /b/ has three subsegments: the closure, the silence, and the release. There are 3×60^3 subsegments of triphones.

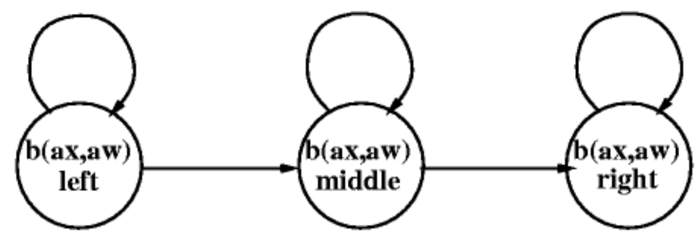


Figure 7.11 An example of the context-dependent triphone b(ax,aw) (the phone [b] preceded by a [ax] and followed by a [aw], as in the beginning of *about*, showing its left, middle, and right subphones.

HMM models for words

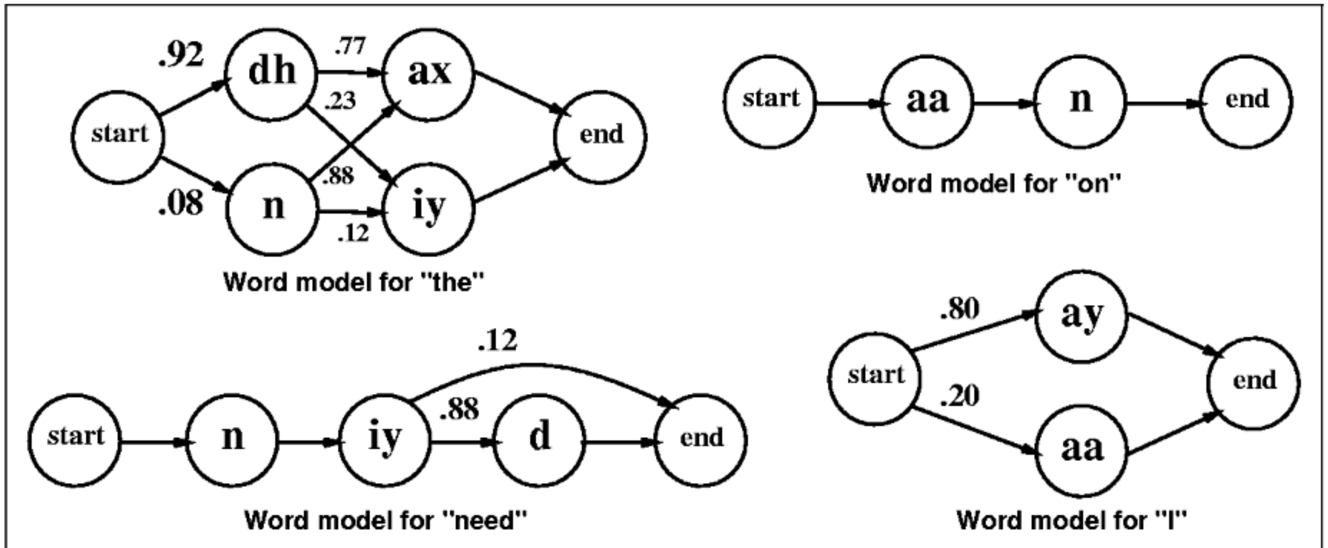
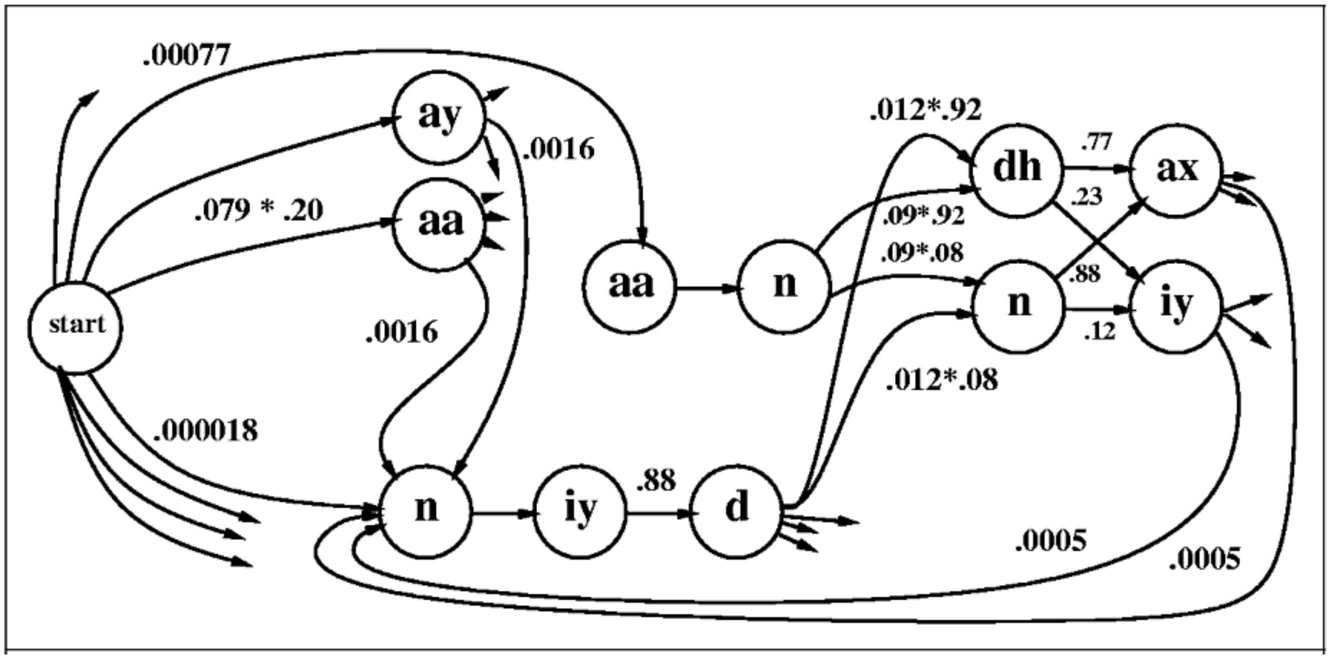


Figure 7.5 Pronunciation networks for the words *I*, *on*, *need*, and *the*. All networks (especially *the*) are significantly simplified.

Putting words together

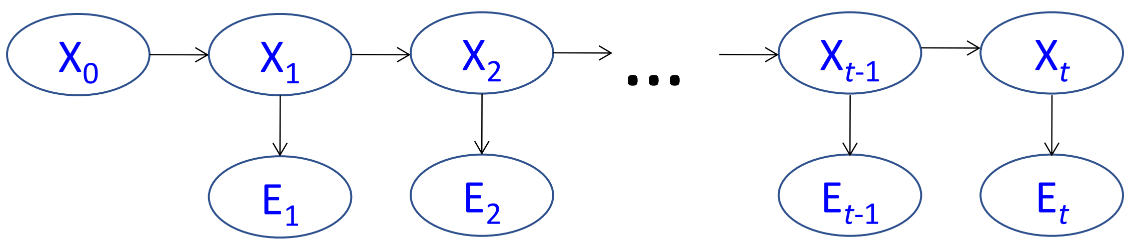


- Given a sequence of acoustic features, how do we find the corresponding word sequence?

The Viterbi Algorithm

$$\max_{\boldsymbol{X}_{0:t}} P(\boldsymbol{X}_{0:t}, \boldsymbol{E}_{0:t})$$
$$= \max_{X_t} P(E_t|X_t) \max_{X_{t-1}} P(X_t|X_{t-1}) P(E_{t-1}|X_{t-1}) \max_{X_{t-2}} \ldots$$

Complexity changes from O{N^T} to O{TN^2}



Decoding with the Viterbi algorithm

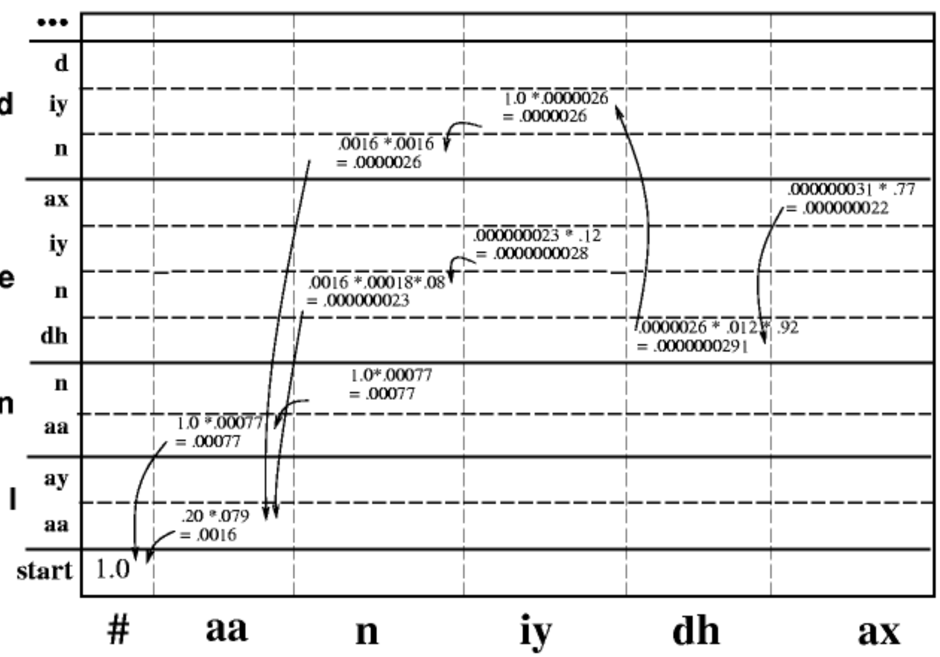


Figure 7.10 The entries in the individual state columns for the Viterbi algorithm. Each cell keeps the probability of the best path so far and a pointer to the previous cell along that path. Backtracing from the successful last word (*the*), we can reconstruct the word sequence *I need the*.

For more information

- CS 447: Natural Language Processing
- ECE 417: Multimedia Signal Processing
- ECE 594: Mathematical Models of Language
- Linguistics 506: Computational Linguistics
- D. Jurafsky and J. Martin, "Speech and Language Processing," 2nd ed., Prentice Hall, 2008