Istituto Dalle Molle di studi sull'intelligenza artificiale
USI/SUPSI
IDSIA

Hidden Markov Models

and related topics

Part 2

Machine Learning 2019

Michael Wand, Jürgen Schmidhuber, Cesare Alippi

TAs: Robert Csordas, Krsto Prorokovic, Xingdong Zou, Francesco Faccio, Louis Kirsch

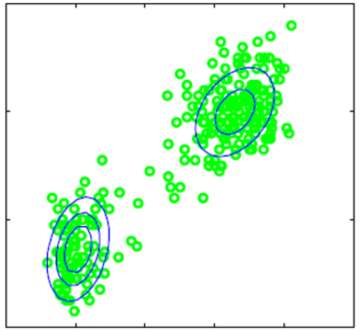
Overview

- Quick Recap of last time: Gaussian Mixtures, Latent Variables, Markov Models
- The HMM formalism
- Example: Speech Recognition

Recap: Gaussian Mixtures

- **Idea:** model *arbitrary* data with mixtures of Gaussians
  - much more flexible than single Gaussians
  - straightforward probability density formula

$$p(x) = \sum_k \pi_k \mathcal{N}(x|\mu_k, \sigma_k)$$



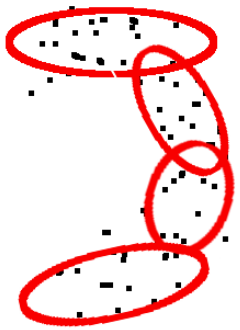
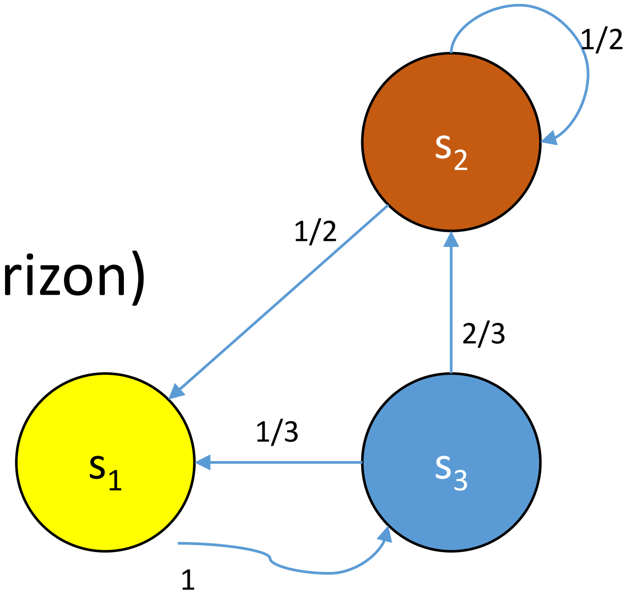


Image source (left): Bishop, PRML, fig. 2.21, modified

- ML estimation of parameters: *Expectation Maximization* algorithm
  - closed-form estimation is impossible because both class assignments and parameters must be optimized
  - class assignments as *hidden variables* which are probabilistically modeled

Recap: Markov Systems

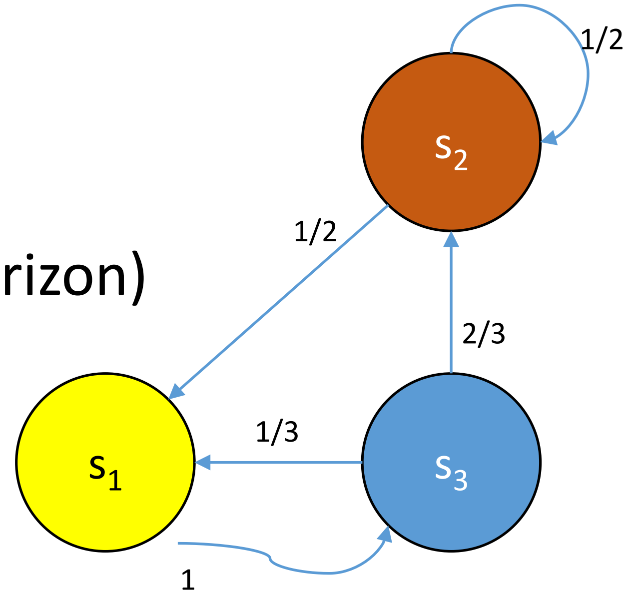
- Sequential processes: system transits probabilistically between states
  - probabilities for next states depend *only on current state* (equivalently: on the last few states, but always with fixed horizon)
  - “memory-less” process



Recap: Markov Systems

Istituto Dalle Molle di studi sull'intelligenza artificiale
USI/SUPSI
IDSIA

- Sequential processes: system transits probabilistically between states
  - probabilities for next states depend *only on current state* (equivalently: on the last few states, but always with fixed horizon)
  - "memory-less" process
- Question: What is $P(q_t=s)$ (the probability of being in state *s* at time *t*?
  - Need to sum over a huge number of possible *paths*
  - Efficient solution: *dynamic programming (DP)*



- Computation is simple.
- Just fill in this table in this order:

t	$p_t(1)$	$p_t(2)$	…	$p_t(N)$
0	0	1		0
1				
:				
t_{final}				

Hidden Markov Models (HMMs)

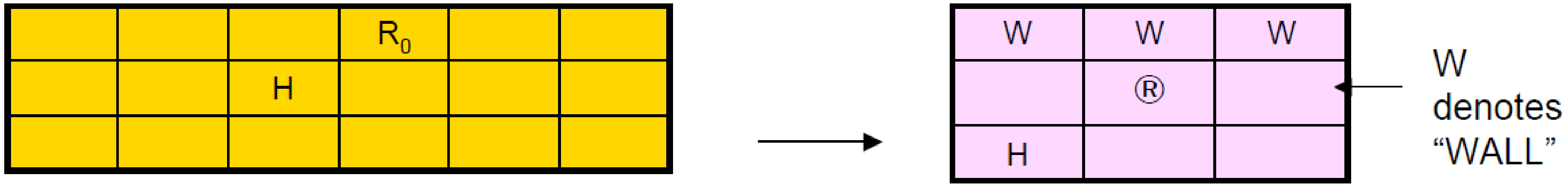
(this part is based on a lecture By Prof. Andrew Moore, Carnegie Mellon University)

Introducing the HMM

- So far, we assumed that we can observe the state evolution of the Markov model directly
  - remember the robot example
  - state = combination of robot position and human position
- But what if we cannot?
  - Assume the robot has sensors, but cannot see arbitrarily far
  - maybe it does not see the human because she is somewhere else
  - and neither does it know its own absolute position

Example: Robot with Proximity Sensors

- example: *proximity sensors* – the robot can observe the contents of the adjacent 8 squares

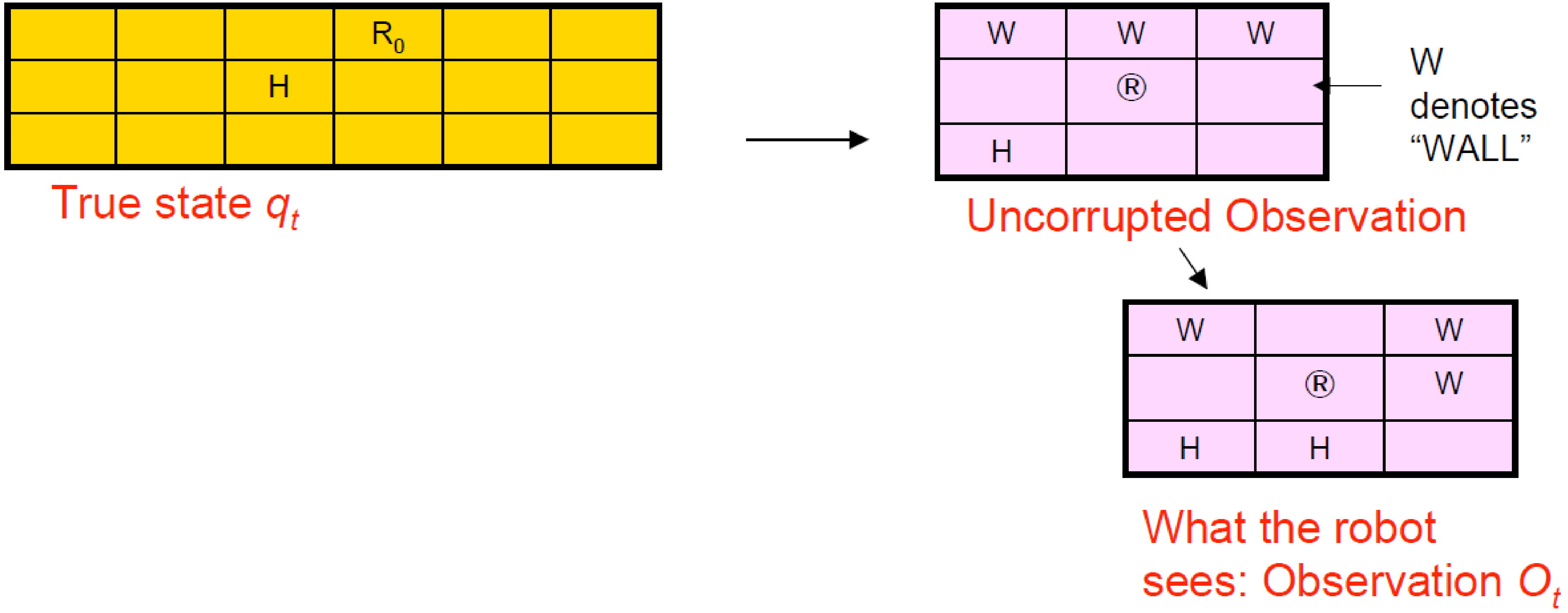


True state q_t

What the robot sees:
Observation O_t

Example: Robot with Proximity Sensors

- Example: *noisy* proximity sensors



Modeling observations

- A *Markov model*
  - Has *n* states, called s_1, s_2, … s_N.
  - state evolution follows the Markov property
- A *Hidden* Markov model
  - Has *n* states, called s_1, s_2, … s_N, whose evolution follows the Markov property
  - At each timestep, has an *observation* O_t which depends probabilistically on the current state (but not on the history)
  - O_t is a continuous or discrete random variable which depends only on current state:

$$P(O_t = x | q_t = s_i) = P(O_t = x | q_t = s_i, \text{any earlier history})$$

  - The observations are frequently modeled with Gaussian mixtures.
  - Other rules are the same as for the standard Markov model.

HMM questions

- The robot with noisy sensors is a good example for HMM modeling
- Question 1: Probability estimation
  - What is $P(O_0,O_1,\ldots,O_t)$?
- Question 2: Most probable path
  - What is the most probable path, given an observation $O_0,O_1,\ldots,O_t$?
  - I.e.

$$\mathrm{argmax}_{q_0,q_1,\ldots,q_t} P(q_0, q_1, \ldots, q_t | O_0, O_1, \ldots, O_t)$$

- Question 3: How to optimize the HMM?
  - by maximum likelihood criterion, given a series of observations

HMM questions

- The robot with noisy sensors is a good example for HMM modeling
- Question 1: Probability estimation
  - What is $P(O_0,O_1,\ldots,O_t)$?
- Question 2: Most probable path
  - What is the most probable path, given an observation $O_0,O_1,\ldots,O_t$?
  - I.e.

$$\mathrm{argmax}_{q_0,q_1,\ldots,q_t} P(q_0, q_1, \ldots, q_t | O_0, O_1, \ldots, O_t)$$

Solved by variants of Dynamic programming, just as last time

- Question 3: How to optimize the HMM?
  - by maximum likelihood criterion, given a series of observations

Solved by the EM algorithm!

HMM Formalization

- States: $s_1, \ldots, s_N$.
- For now, assume discrete possible observations $v_1, \ldots, v_M$.
- For a particular *trial* (i.e. sequence of observations)
  - T is the number of observations / states passed through
  - $O = O_0, O_1, \ldots, O_T$ is the sequence of observations
  - $Q = q_0, q_1, \ldots, q_T$ is a path, which we wish to model probabilistically

HMM Formalization

USI/SUPSI
Istituto Dalle Molle di studi sull'intelligenza artificiale
IDSIA

- An HMM (with discrete observations) is a 5-tuple consisting of:
  - States: $s_1, \ldots, s_N$.
  - Observations: $v_1, \ldots, v_M$.
  - $\{\pi_1, \ldots, \pi_N\}$: Starting state probabilities
  - State transition probabilities a_{ij}: $a_{ij} = P(q_{t+1}=s_j \mid q_t=s_i)$, $1 \le i,j \le N$
  - Observation probabilities $b_i(m) = P(O_t=v_m \mid q_t=s_i)$, $1 \le i \le N$, $1 \le j \le M$
- The latter two are conveniently arranged in matrices

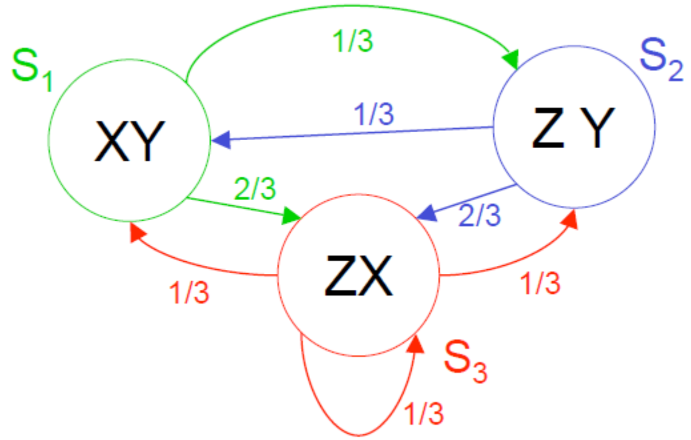
$$A = \begin{pmatrix} a_{11} & \cdots & a_{1N} \\ \vdots & \ddots & \vdots \\ a_{N1} & \cdots & a_{NN} \end{pmatrix}, \qquad B = \begin{pmatrix} b_1(1) & \cdots & b_1(M) \\ \vdots & \ddots & \vdots \\ b_N(1) & \cdots & b_N(M) \end{pmatrix}$$

*L. R. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," Proc. of the IEEE, Vol.77, No.2, pp.257--286, 1989.

Recommended reading

Here’s an HMM

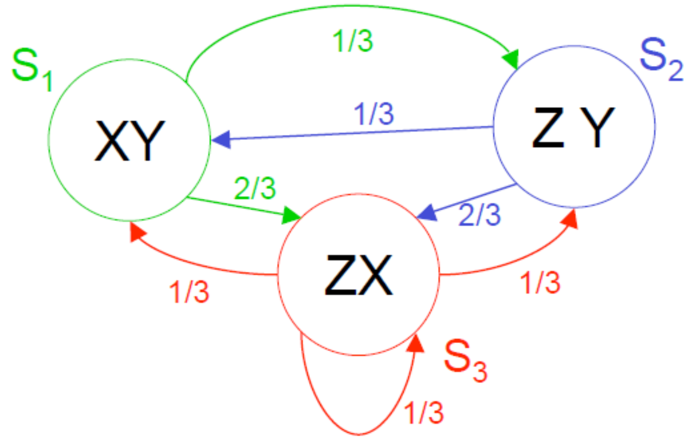
- Start randomly in state s_1 or s_2
- Choose one out of two output symbols $v_1=X$, $v_2=Y$, $v_3=Z$ in each state with equal probability



Here's an HMM

USI/SUPSI
Istituto Dalle Molle di studi sull'intelligenza artificiale
IDSIA

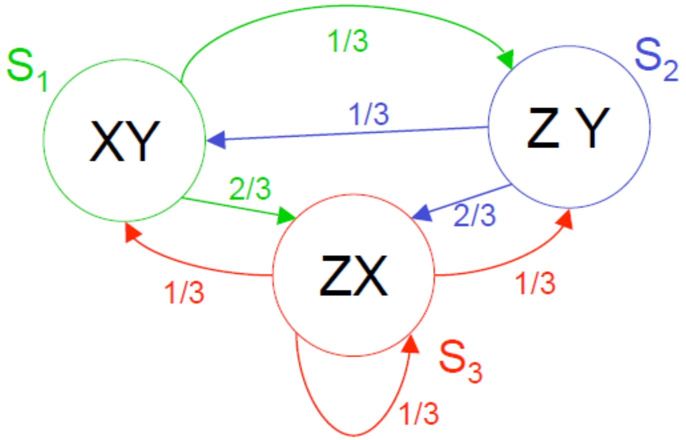
- Start randomly in state s_1 or s_2
- Choose one out of two output symbols v_1=X, v_2=Y, v_3=Z in each state with equal probability



$$\pi_1 = \frac{1}{2}, \pi_2 = \frac{1}{2}, \pi_3 = 0, \qquad A = \begin{pmatrix} 0 & 1/3 & 2/3 \\ 1/3 & 0 & 2/3 \\ 1/3 & 1/3 & 1/3 \end{pmatrix}, \qquad B = \begin{pmatrix} 1/2 & 1/2 & 0 \\ 0 & 1/2 & 1/2 \\ 1/2 & 0 & 1/2 \end{pmatrix}$$

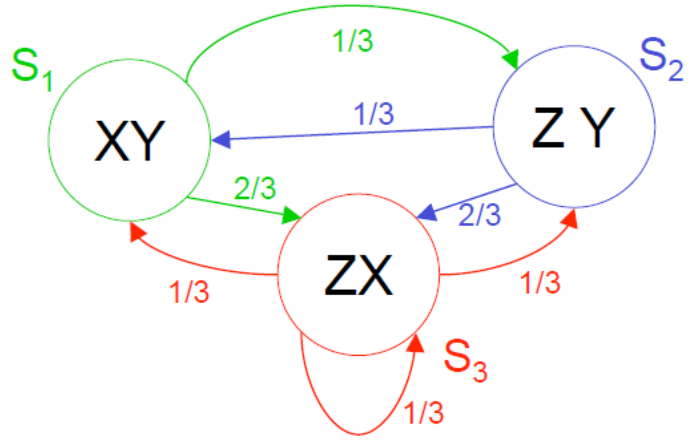
Let’s generate an observation sequence

Istituto Dalle Molle di studi sull’intelligenza artificiale
USI/SUPSI IDSIA



q_0=		O_0=	
q_1=		O_1=	
q_2=		O_2=	

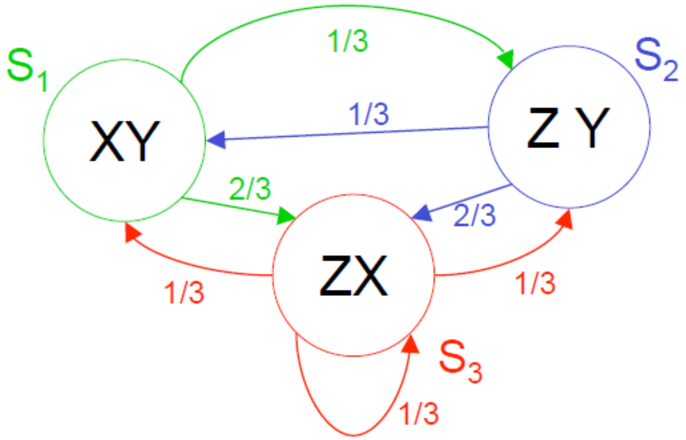
Let’s generate an observation sequence



Start: 50-50 choice between S_1 and S_2

q_0=		O_0=	
q_1=		O_1=	
q_2=		O_2=	

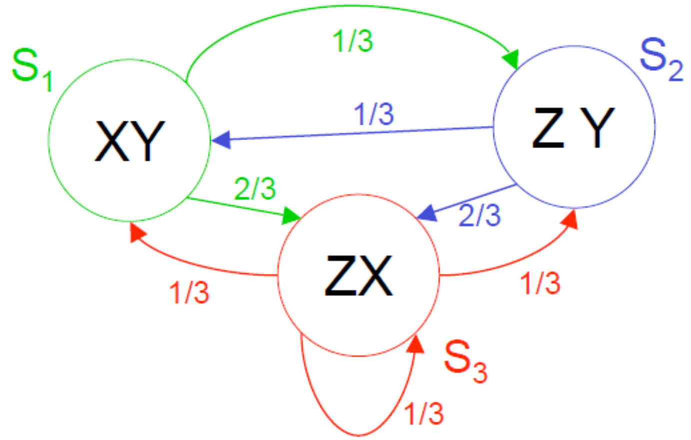
Let’s generate an observation sequence



50-50 choice
between X and Y

q_0=	S_1	O_0=	
q_1=		O_1=	
q_2=		O_2=	

Let's generate an observation sequence

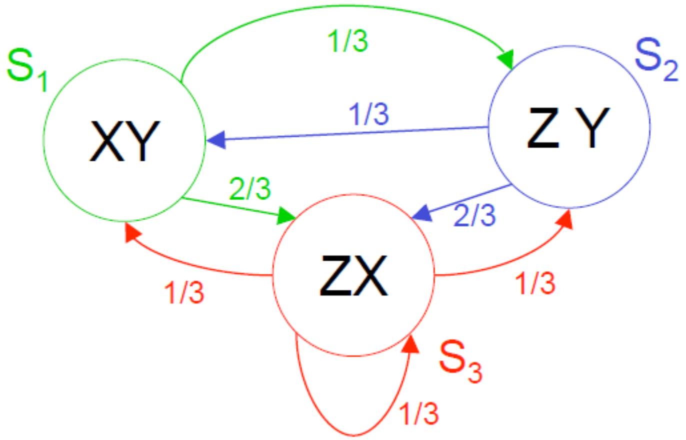


Go to S_3 with prob. 2/3, or S_2 with prob. 1/3

q_0=	S_1	O_0=	X
q_1=		O_1=	
q_2=		O_2=	

Let's generate an observation sequence

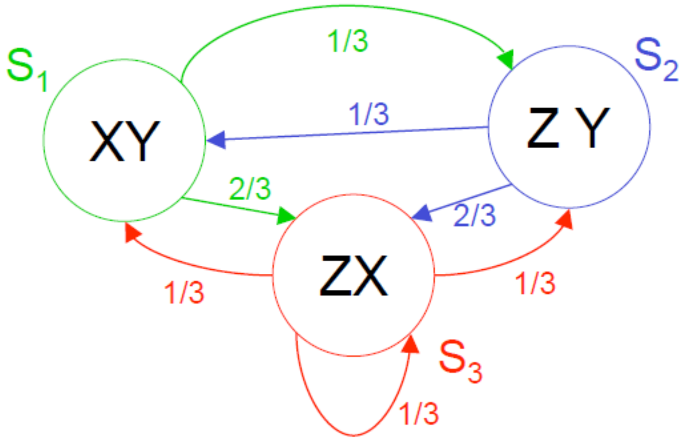
USI/SUPSI
Istituto
Dalle Molle
di studi
sull'intelligenza
artificiale
IDSIA

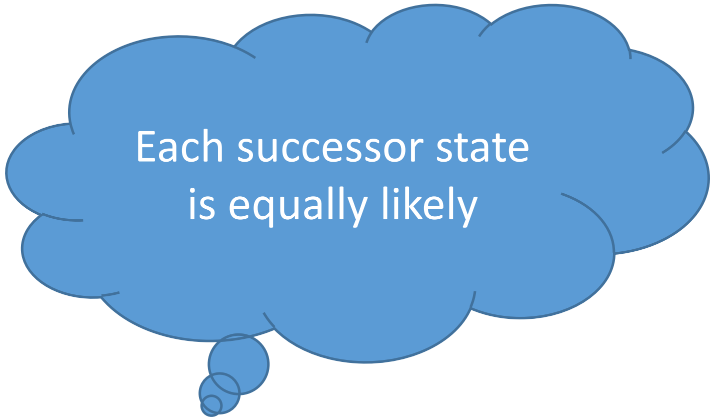


50-50 choice
between X and Z

q_0=	S_1	O_0=	X
q_1=	S_3	O_1=	
q_2=		O_2=	

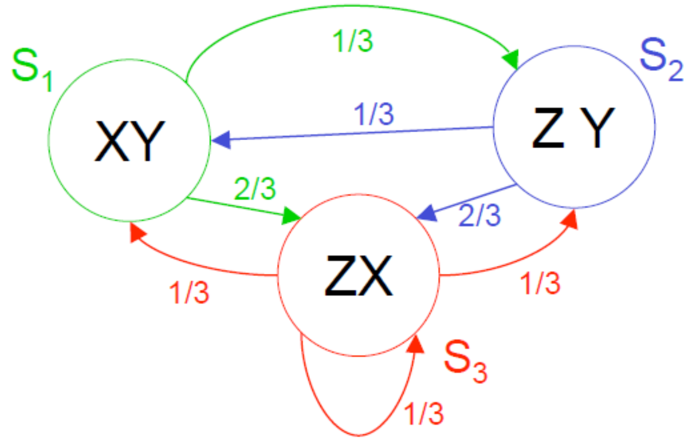
Let’s generate an observation sequence

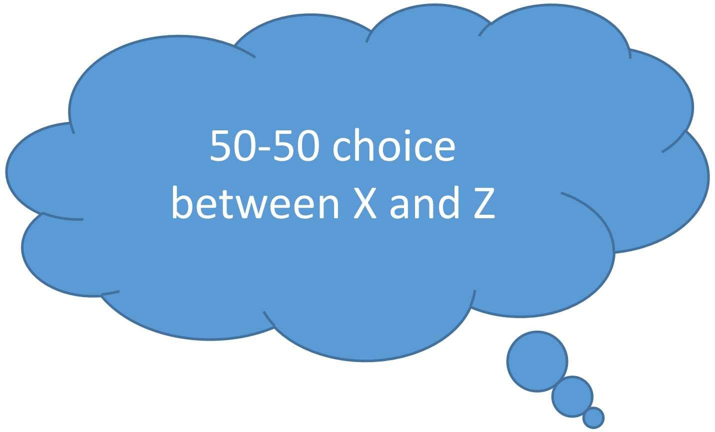




q_0=	S_1	O_0=	X
q_1=	S_3	O_1=	X
q_2=		O_2=	

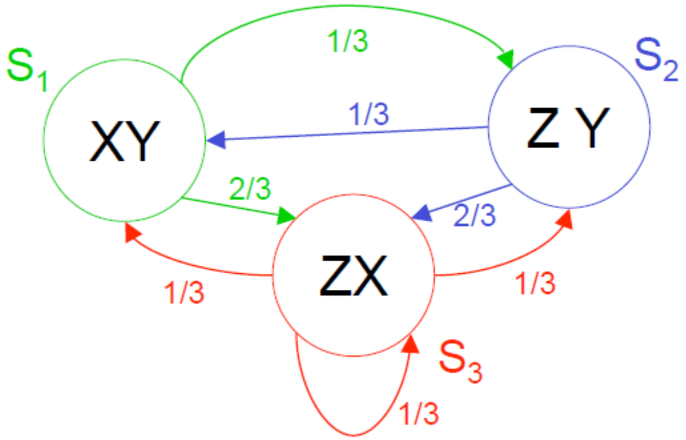
Let's generate an observation sequence





q_0=	S_1	O_0=	X
q_1=	S_3	O_1=	X
q_2=	S_3	O_2=	

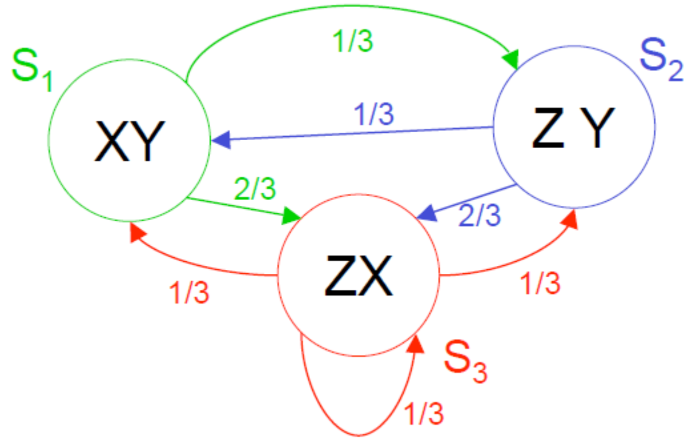
Let’s generate an observation sequence



q_0=	S_1	O_0=	X
q_1=	S_3	O_1=	X
q_2=	S_3	O_2=	Z

Let's generate an observation sequence

Istituto Dalle Molle di studi sull'intelligenza artificiale
USI/SUPSI IDSIA



But the states are hidden!

This is what we work with:

q_0=	?	O_0=	X
q_1=	?	O_1=	X
q_2=	?	O_2=	Z

HMM Standard Algorithms

- First question: What is $P(O) = P(O_0, O_1, \ldots, O_T)$ for a given observation?

HMM Standard Algorithms

- First question: What is $P(O) = P(O_0, O_1, \ldots, O_T)$ for a given observation?
- Clearly,

$$P(O) = \sum_{Q \in \text{paths of length T}} P(O \wedge Q) = \sum_{Q \in \text{paths of length T}} P(O|Q) \cdot P(Q)$$

- How do we compute $P(Q)$ and $P(O|Q)$, for given observation and path?

HMM Standard Algorithms

USI/SUPSI
Istituto Dalle Molle di studi sull'intelligenza artificiale
IDSIA

- First question: What is *P(O) = P(O_0, O_1, …, O_T)* for a given observation?
- Clearly,

$$P(O) = \sum_{Q \in \text{paths of length T}} P(O \wedge Q) = \sum_{Q \in \text{paths of length T}} P(O|Q) \cdot P(Q)$$

- How do we compute *P(Q)* and *P(O|Q),* for given observation and path?
- Example with three states/observations (as before):
  - $P(Q) = P(q_0, q_1, q_2) = P(q_0) \cdot P(q_1|q_0) \cdot P(q_2|q_1) = \pi_0 \cdot a_{q_0 q_1} \cdot a_{q_0 q_1}$ (Markov)
  - $P(O|Q) = P(O_0|q_0) \cdot P(O_1|q_1) \cdot P(O_2|q_2) = b_{q_0}(O_0) \cdot b_{q_1}(O_1) \cdot b_{q_2}(O_2)$
- but performing such computations for *many* paths is infeasible!

The Forward Algorithm

- Let us do the computation in DP-style, just like last time
- Define $\alpha_t(i) = P(O_0, \dots, O_t \wedge q_t = s_i)$ for t = 0, …, T
  - this is the probability to have seen the first *t* observations and then being in state s_i

The Forward Algorithm

- Let us do the computation in DP-style, just like last time
- Define $\alpha_t(i) = P(O_0, \ldots, O_t \wedge q_t = s_i)$ for t = 0, …, T
  - this is the probability to have seen the first *t* observations and then being in state s_i
- Just like last time, we can define the α_t recurrently:
  - $\alpha_{t+1}(j) = \sum_i \alpha_t(i) \cdot a_{ij} \cdot b_j(O_{t+1})$ with $\alpha_0(i) = \pi_i \cdot b_i(O_0)$
  - in other words, we get the next α_t by summing over
    - the previous α_t
    - multiplied by the relevant state transition probability and the observation probability

The Forward Algorithm

USI/SUPSI IDSIA Istituto Dalle Molle di studi sull'intelligenza artificiale

- Let us do the computation in DP-style, just like last time
- Define $\alpha_t(i) = P(O_0, \ldots, O_t \wedge q_t = s_i)$ for t = 0, …, T
  - this is the probability to have seen the first *t* observations and then being in state s_i
- Just like last time, we can define the α_t recurrently:
  - $\alpha_{t+1}(j) = \sum_i \alpha_t(i) \cdot a_{ij} \cdot b_j(O_{t+1})$ with $\alpha_0(i) = \pi_i \cdot b_i(O_0)$
  - in other words, we get the next α_t by summing over
    - the previous α_t
    - multiplied by the relevant state transition probability and the observation probability

	$\alpha_t(0)$	$\alpha_t(1)$	…	$\alpha_t(M)$
t=0				
t=1				
…				
t=T				

fill out like this

The Forward Algorithm

- Let us do the computation in DP-style, just like last time
- Define $\alpha_t(i) = P(O_0, \ldots, O_t \wedge q_t = s_i)$ for t = 0, …, T
  - this is the probability to have seen the first *t* observations and then being in state s_i
- Just like last time, we can define the α_t recurrently:
  - $\alpha_{t+1}(j) = \sum_i \alpha_t(i) \cdot a_{ij} \cdot b_j(O_{t+1})$ with $\alpha_0(i) = \pi_i \cdot b_i(O_0)$
  - in other words, we get the next α_t by summing over
    - the previous α_t
    - multiplied by the relevant state transition probability and the observation probability
- Finally, $P(O) = \sum_i \alpha_T(i)$.

	$\alpha_t(0)$	$\alpha_t(1)$	…	$\alpha_t(M)$
t=0				
t=1				
…				
t=T				

fill out like this

State estimation

- How do we compute $P(q_T = s_i \mid O_0, \ldots, O_T)$?
  - That's simple now!

$$P(q_T = s_i | O) = \frac{P(q_T = s_i \wedge O)}{P(O)} = \frac{\alpha_T(i)}{\sum_j \alpha_T(j)}$$

- We have done the first step towards solving a fundamental HMM "problem": Out of observations, we can now compute the state we are in!
  - Does this remind you of classification problems?

Most probable path

- Second HMM questions:
- What is the most probable path given an observation? Determine

$$\mathrm{argmax}_{q_0,q_1,\dots,q_T} P(q_0, q_1, \dots, q_T | O_0, O_1, \dots, O_T)$$

- In principle,

$$\mathrm{argmax}_Q P(Q|O) = \frac{\mathrm{argmax}_Q P(O|Q) \cdot P(Q)}{P(O)} = \mathrm{argmax}_Q P(O|Q) \cdot P(Q)$$

Most probable path

- We can also compute that with DP!
- Define

$$\delta_t(i) = \max_{q_0,\ldots,q_{t-1}} P(q_0,\ldots,q_{t-1},q_t = s_i, O_0,\ldots,O_t)$$

- that is the probability of the path of length *t* with the *maximum* chance of:
  - occuring
  - ending up in state s_i
  - producing the output $O_0, \ldots, O_t$

Most probable path

- Just as before, find a recursive definition of the $\delta_t(i)$:
  - $\delta_0(i) = \pi_i \cdot b_i(O_0)$
  - $\delta_{t+1}(j) = \max_i(\delta_t(i) \cdot a_{ij} \cdot b_j(O_{t+1}))$, that is, we maximize over a set of possible transitions in the last step
- How to determine the best *path?*
  - So far we have the *probability* of the best path
  - During the recursive computation, required to save a “backpointer” to the predecessor state which yielded the maximum probability
  - When the entire observation has been processed (say, at time step *T*), just maximize over all $\delta_T(i)$ and step back through time to derive the most probable path

HMM Training

- We have done a lot of computations of the form $P(O_0, \ldots, O_t \mid \lambda)$, where λ stands for the parameters of the HMM (so far, the a_{ij} and b_i).
- Now assume we have some observations, and want to expect the HMM from them
  - we assume the number of states is fixed
- As usual, use maximum likelihood criterion:

$$\hat{\lambda} = \mathrm{argmax}_{\lambda} P(O_0, \ldots, O_T | \lambda)$$

- no closed-form solution available, but we do as for the Gaussian mixtures:
  - assume that the path $Q = (q_0, \ldots, q_t)$ is a hidden variable
  - alternatingly reestimate maximum-likelihood paths and HMM state parameters
  - *Expectation Maximization*

HMM Training

- We define
  - $\gamma_t(i) = P(q_t = s_i | O_0, \dots, O_T, \lambda)$ (probability that state i is reached)
  - $\xi_t(i,j) = P(q_t = s_i \wedge q_{t+1} = s_j | O_0, \dots, O_T, \lambda)$ (probability of transitions i -> j)
  - (note the difference between variable *t* and observation length *T* – in these definitions we always consider the *entire* observation sequence)
- Given fixed HMM parameters, these values can be computed by a DP algorithm (for details see Rabiner's paper)

HMM Training

- With
  - $\gamma_t(i) = P(q_t = s_i | O_0, \dots, O_T, \lambda)$
  - $\xi_t(i,j) = P\left(q_t = s_i \curlywedge q_{t+1} = s_j \middle| O_0, \dots, O_T, \lambda\right)$
- the HMM parameters are reestimated as follows:

$$\hat{\pi}_i = \gamma_0(i)$$
$$\hat{a}_{ij} = \frac{\sum_{t=0}^{T} \xi_t(i,j)}{\sum_{t=0}^{T-1} \gamma_t(i,j)}$$
$$\hat{b}_j(k) = \frac{\sum_{t:O_t=v_k} \gamma_t(j)}{\sum_t \gamma_t(j)}$$

- The two steps of path reestimation (E-step) and parameter reestimation are (M-step) repeated until a convergence criterion is satisfied.
- For HMMs, this is known as the *Baum-Welch Algorithm.*

Continuous-observation HMMs

- So far, we have considered HMMs whose outputs are *discrete*
- It is easy to extend the formulation to HMMs with continuous outputs, which are modeled by GMMs

Continuous-observation HMMs

USI/SUPSI Istituto Dalle Molle di studi sull'intelligenza artificiale IDSIA

- For applying the HMM, we can directly use the formulas as described before. As an example, consider the observation probability computation

$$P(O) = \sum_{Q \in \text{paths of length T}} P(O|Q) \cdot P(Q)$$

for which we used the forward algorithm, defining:

$$\alpha_{t+1}(j) = \sum_i \alpha_t(i) \cdot a_{ij} \cdot b_j(O_{t+1}) \text{ with } \alpha_0(i) = \pi_i \cdot b_i(O_0)$$

In the continuous case, simply define $P(O_t|q_t = s_i) \coloneqq p_i(O_t)$ with $p_i(O) = \sum_k w_k \mathcal{N}(O|\mu_k, \sigma_k)$, then:

$$\alpha_{t+1}(j) = \sum_i \alpha_t(i) \cdot a_{ij} \cdot p_j(O_{t+1}) \text{ with } \alpha_0(i) = \pi_i \cdot p_i(O_0)$$

(w_k are the mixture weights – I have renamed them because the π is already used for the HMM initial probabilities)

Continuous-observation HMMs

- Reestimating the HMM parameters now includes reestimating the means, covariance matrices, and weights of the Gaussians attached to each HMM state.
- Example: The mean of the *k*-th Gaussian of state s_j can be reestimated by the formula

$$\hat{\mu}_{jk} = \frac{\sum_t \gamma_t(j,k) \cdot o_t}{\sum_t \gamma_t(j,k)}$$

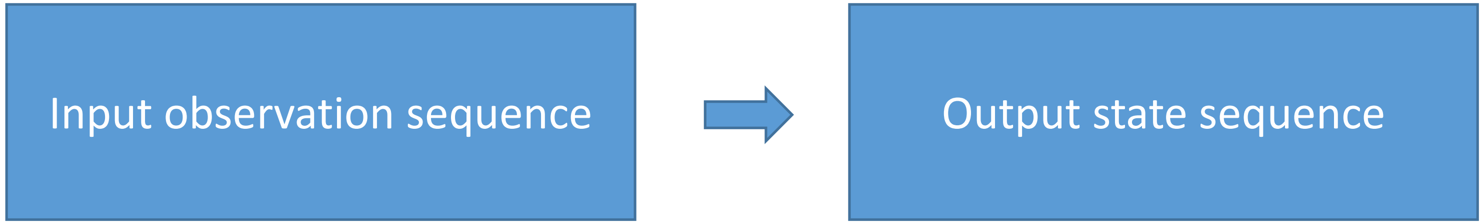
  where $\gamma_t(j,k)$ is the probability of being in state *j* at time *t*, with Gaussian *k* accounting for the observation.
- For details, see Rabiner's paper

HMMs as Classifiers

- How do we use HMMs for classification?

HMMs as Classifiers

- How do we use HMMs for classification?
- We assume that the *hidden sequence of states* is what we are interested in!
- We estimate it from the observation sequence, using the Viterbi algorithm
- Thus we have a sequence-to-sequence classifier:



HMMs as Classifiers

- How do we use HMMs for classification?
- We assume that the *hidden sequence of states* is what we are interested in!
- We estimate it from the observation sequence, using the Viterbi algorithm
- Thus we have a sequence-to-sequence classifier:

Input observation sequence → Output state sequence

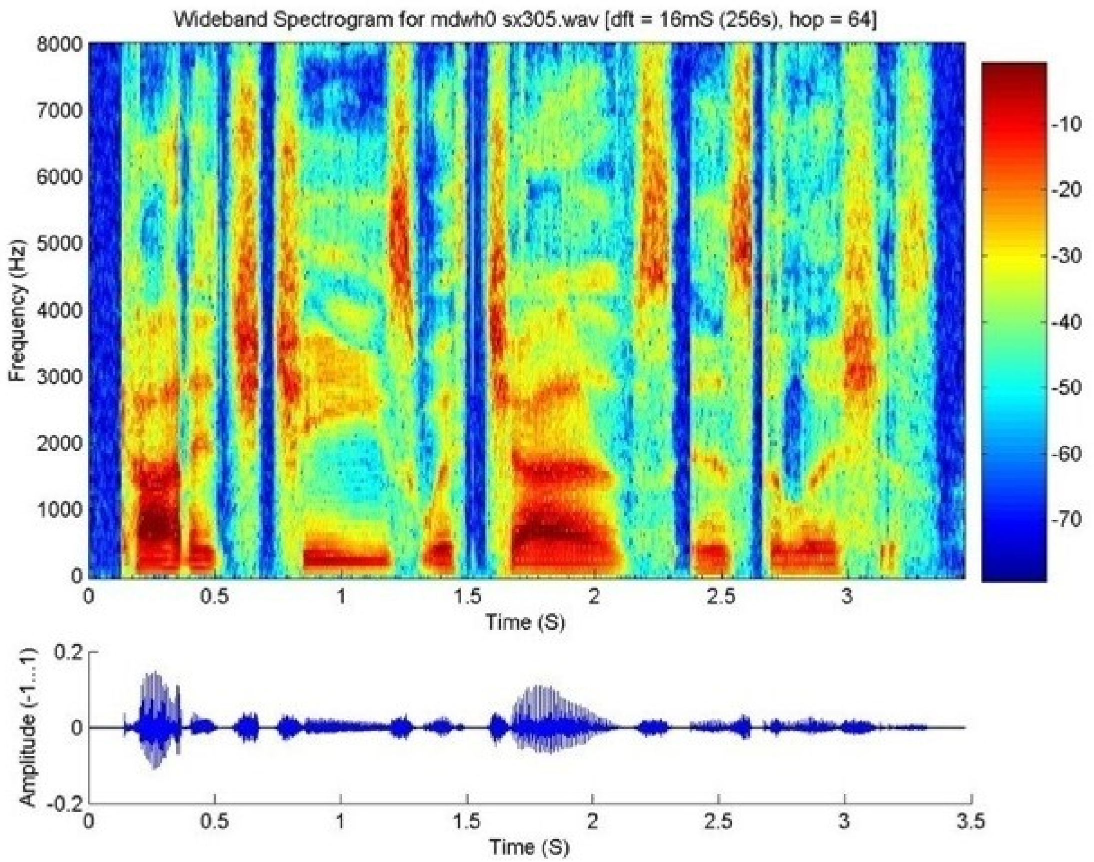
- Note that this gives a new meaning to the states as hidden variables.
- Also note that using a probabilistic *generative* model for *classification* comes with a set of issues which need to be carefully considered.

Speech Recognition with HMMs

in 5 minutes

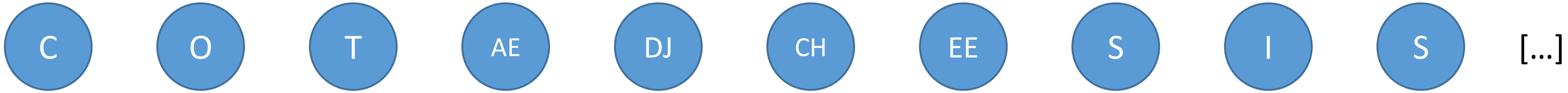
Speech Recognition

- HMM speech recognition setup:
  - The observations are usually spectral representations of speech
  - Example: *spectrogram* of the sentence “Cottage cheese with chives is delicious” (from the TIMIT corpus)
  - horizontal axis: time
  - vertical axis: frequency
  - colors: intensity
  - one *frame* every 10 ms



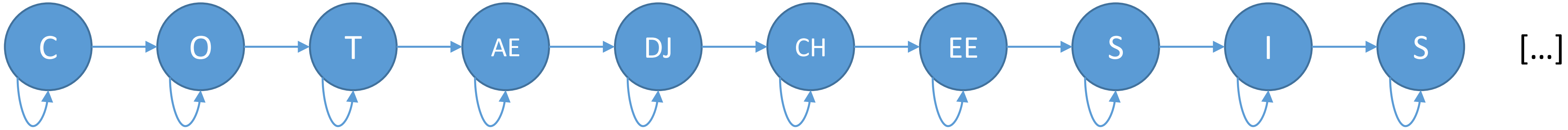
Speech Recognition

- HMM speech recognition setup:
  - The hidden states, in the most simple case, correspond to phones (speech sounds)



Speech Recognition

- HMM speech recognition setup:
  - The hidden states, in the most simple case, correspond to phones (speech sounds)

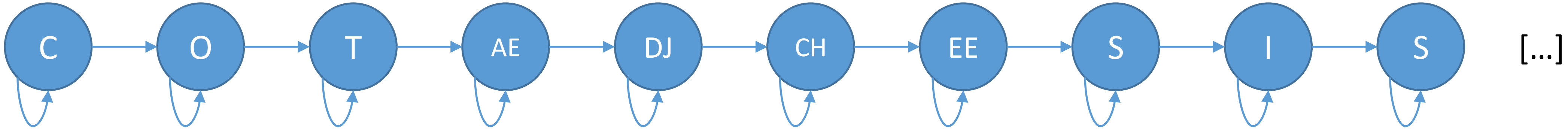


  - Left-to-right topology, with self loops, states can repeat, GMM observation models
  - One state every 10 ms

Speech Recognition

USI/SUPSI
Istituto Dalle Molle di studi sull'intelligenza artificiale
IDSIA

- HMM speech recognition setup:
  - The hidden states, in the most simple case, correspond to phones (speech sounds)



  - Left-to-right topology, with self loops, states can repeat, GMM observation models
  - One state every 10 ms
- Usually,
  - one splits each phone into three *subphones* (begin, middle, end) – better modeling
  - states depend on *context*
  - only the observation probabilities are trained, sequence structure injected by dictionary, language model, etc.

Speech Recognition

- Training the HMM model:
  - have a somewhat large speech corpus (hundreds of hours) with many observation sequences, requires frame-level annotations
  - perform EM for several epochs (maybe 10)
  - GMMs are often pretrained for speed
- Evaluating the trained model
  - search for the best state sequence given observations
  - the HMM grows to a tree/graph structure (for example, *prefix tree*)
    - this has to do with the fact that we have additional constraints (dictionary, language model)
  - time-synchronous *beam* search (prune low-prob. hypotheses)

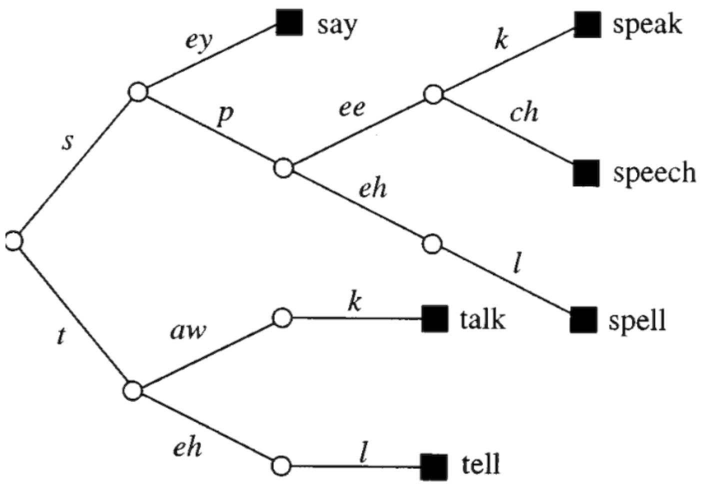


Image: Ortmanns/Ney, The time-conditioned approach in dynamic programming search for LVCSR

Conclusion / Outlook

- We have covered Hidden Markov models (HMMs)
  - the formalism
  - three fundamental questions
  - DP algorithms over and over again
  - EM training
  - usage for classification
- … and their application in speech recognition
  - requires several changes to the original formalism (in particular during search, where additional knowledge sources are integrated)
  - nonetheless highly successful algorithm, used for many years
  - more recently: integration of neural networks and HMMs, or complete substitution of HMMs by RNNs