10-701
Machine Learning

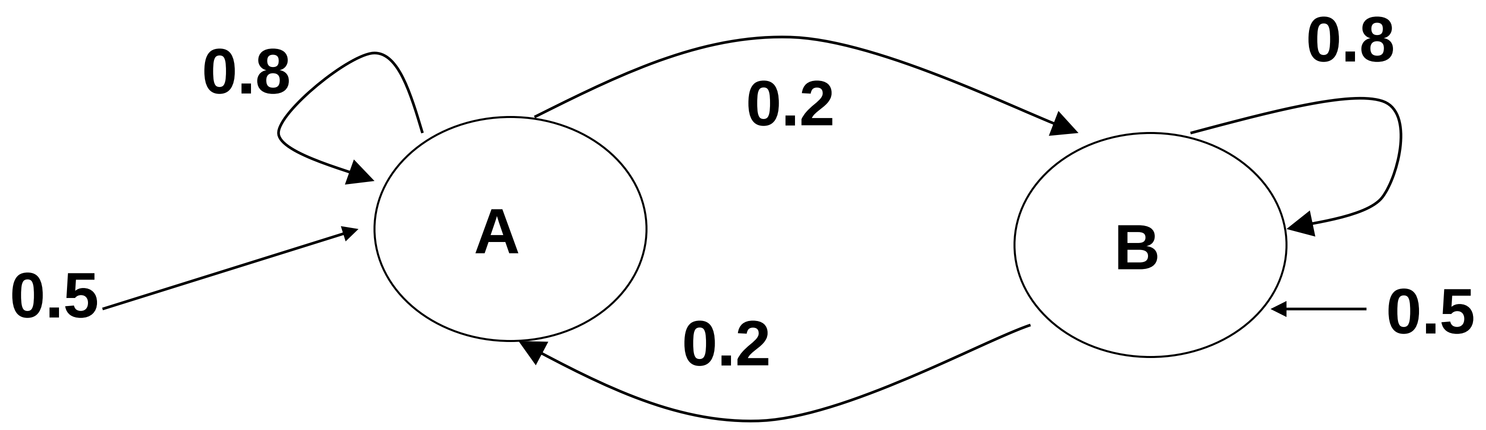
Learning HMMs

A Hidden Markov model

- A set of states $\{s_1 \dots s_n\}$

  - In each time point we are in exactly one of these states denoted by q_t
- Π_i, the probability that we *start* at state s_i
- A transition probability model, $P(q_t = s_i \mid q_{t-1} = s_j)$
- A set of possible outputs Σ

  - At time *t* we emit a symbol $\sigma \in \Sigma$
- An emission probability model, $p(o_t = \sigma \mid s_i)$



Inference in HMMs

- Computing P(Q) and $P(q_t = s_i)$ √
- Computing P(Q | O) and $P(q_t = s_i | O)$ √
- Computing $\mathrm{argmax}_Q P(Q)$ √

Learning HMMs

- Until now we assumed that the emission and transition probabilities are known
- This is usually not the case

  - How is “AI” pronounced by different individuals?

  - What is the probability of hearing “class” after “AI”?

While we will discuss learning the transition and emission models, we will not discuss how to learn the set of states.

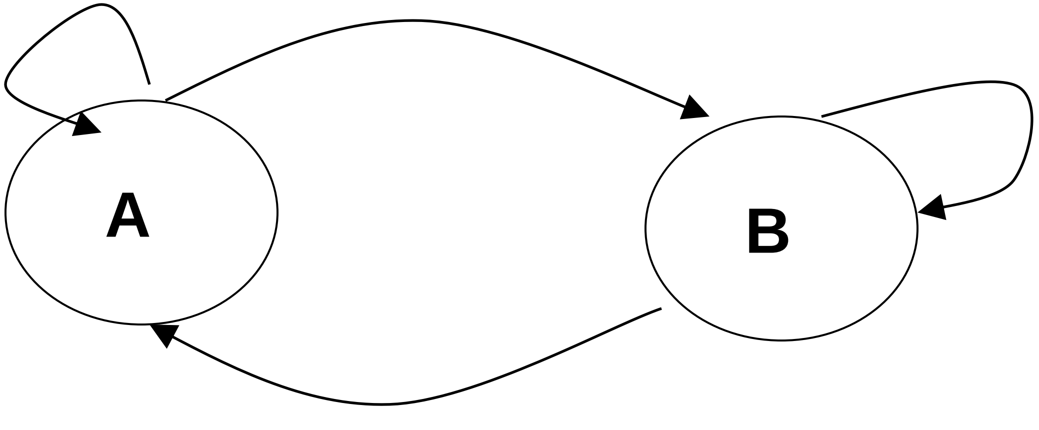
This is usually a function of domain knowledge.

Example

- Assume the model below
- We also observe the following sequence:

1,2,2,5,6,5,1,2,3,3,5,3,3,2 …..

- How can we determine the initial, transition and emission probabilities?



MLE when states are observed

- We will initially assume that we can observe the states themselves
- Obviously, this is not the case. We will relax this assumption to both, infer the states and learn the parameters.

Initial probabilities

Q: assume we can observe the following sets of states:

AAABBAA
AABBBBB
BAABBAB

how can we learn the initial probabilities?

A: Maximum likelihood estimation

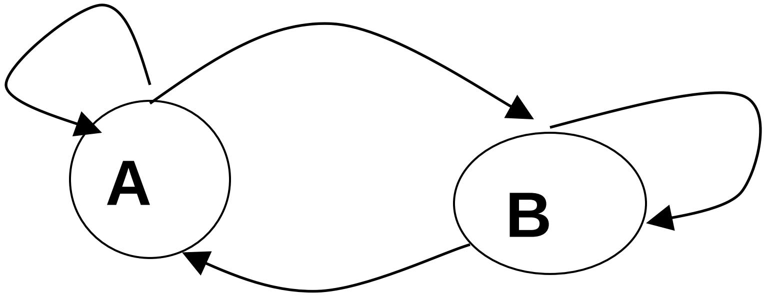
Find the initial probabilities π such that

k is the number of sequences avialable for training

$$\pi^* = \arg\max_{\pi} \prod_{k} \pi(q_1) \prod_{t=2}^{T} p(q_t \mid q_{t-1}) \Rightarrow$$

$$\pi^* = \arg\max_{\pi} \prod_{k} \pi(q_1)$$

$$\pi_A = \#A / (\#A + \#B)$$



Transition probabilities

Q: assume we can observe the set of states:

AAABBAAAABBBBBAAAABBBB

how can we learn the transition probabilities?

A: Maximum likelihood estimation

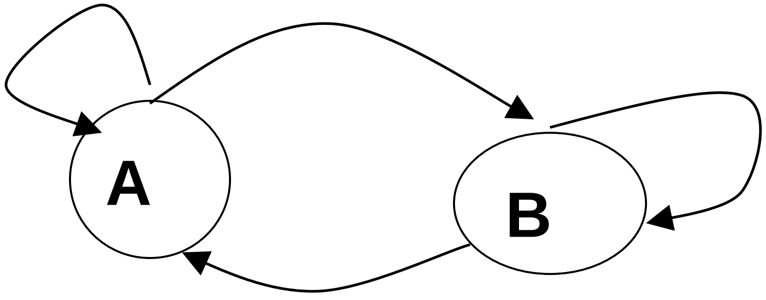
Find a transition matrix *a* such that

remember that we defined $a_{i,j}=p(q_t=s_i|q_{t-1}=s_j)$

$$a^* = \arg\max_a \prod_k \pi(q_1) \prod_{t=2}^{T} p(q_t \mid q_{t-1}) \Rightarrow$$

$$a^* = \arg\max_a \prod_{t=2}^{T} p(q_t \mid q_{t-1})$$

$$a_{A,B} = \#AB / (\#AB + \#AA)$$



Emission probabilities

Q: assume we can observe the set of states:

A A A B B A A A A B B B B B A A

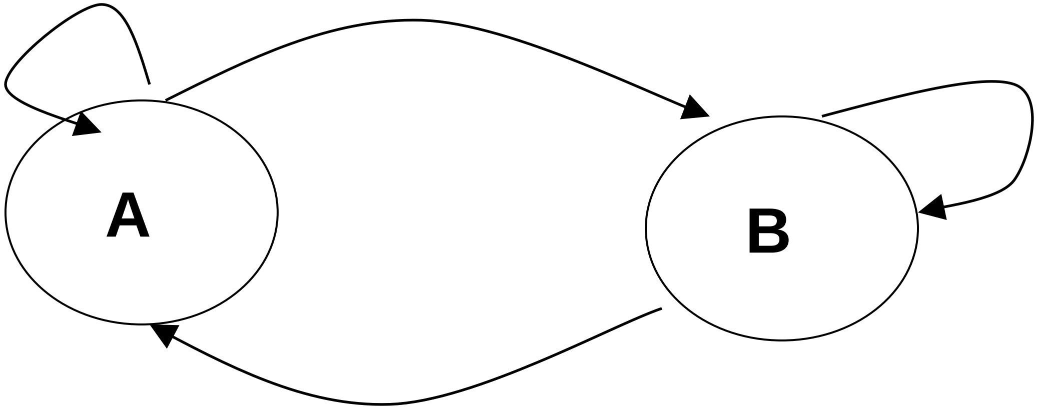
and the set of dice values

1 2 3 5 6 3 2 1 1 3 4 5 6 5 2 3

how can we learn the emission probabilities?

A: Maximum likelihood estimation

$b_A(5) = \#A5 / (\#A1 + \#A2 + \ldots + \#A6)$



Learning HMMs

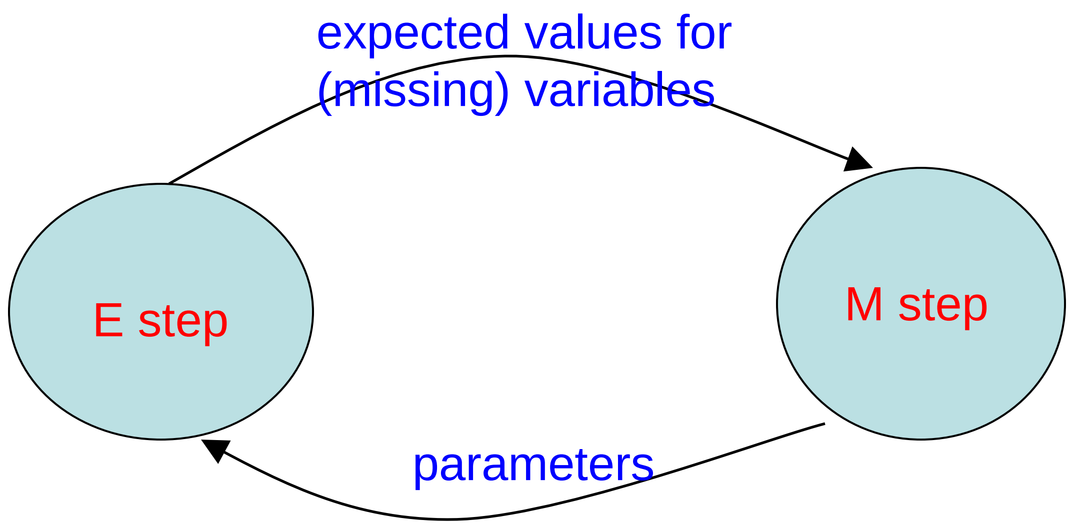
- In most case we do not know what states generated each of the outputs (fully unsupervised)
- … but had we known, it would be very easy to determine an emission and transition model!
- On the other hand, if we had such a model we could determine the set of states using the inference methods we discussed

Expectation Maximization (EM)

- Appropriate for problems with ‘missing values’ for the variables.
- For example, in HMMs we usually do not observe the states

Expectation Maximization (EM): Quick reminder

- Two steps
- E step: Fill in the expected values for the missing variables
- M step: Regular maximum likelihood estimation (MLE) using the values computed in the E step and the values of the other variables
- Guaranteed to converge (though only to a local minima).



Forward-Backward

- We already defined a *forward* looking variable

$$\alpha_t(i) = P(O_1 \ldots O_t \wedge q_t = s_i)$$

- We also need to define a *backward* looking variable

$$\beta_t(i) = P(O_{t+1}, \cdots, O_T \mid s_t = i)$$

Forward-Backward

- We already defined a *forward* looking variable

$$\alpha_t(i) = P(O_1 \ldots O_t \wedge q_t = s_i)$$

- We also need to define a *backward* looking variable

$$\beta_t(i) = P(O_{t+1}, \cdots, O_T \mid q_t = s_i) =$$
$$\sum_j a_{i,j} b_j(O_{t+1}) \beta_{t+1}(j)$$

Forward-Backward

- We already defined a *forward* looking variable

$$\alpha_t(i) = P(O_1 \ldots O_t \wedge q_t = s_i)$$

- We also need to define a *backward* looking variable

$$\beta_t(i) = P(O_{t+1}, \cdots, O_T \mid q_t = s_i)$$

- Using these two definitions we can show

P(A|B)=P(A,B)/P(B)

$$P(q_t = s_i \mid O_1, \cdots, O_T) = \frac{\alpha_t(i)\beta_t(i)}{\sum_j \alpha_t(j)\beta_t(j)} \stackrel{def}{=} S_t(i)$$

State and transition probabilities

- Probability of a state

$$P(q_t = s_i \mid O_1, \cdots, O_T) = \frac{\alpha_t(i)\beta_t(i)}{\sum_j \alpha_t(j)\beta_t(j)} \overset{def}{=} S_t(i)$$

- We can also derive a transition probability

$$P(q_t = s_i, q_{t+1} = s_j \mid o_1, \cdots, o_T) = S_t(i, j)$$

$$P(q_t = s_i, q_{t+1} = s_j | o_1, \cdots, o_T) = \\ = \frac{\alpha_t(i) P(q_{t+1} = s_j | q_t = s_i) P(o_{t+1} | q_{t+1} = s_j) \beta_{t+}}{\sum_k \alpha_t(k)\, \beta_t(k)} \\ \overset{\text{def}}{=} S_t(i, j)$$

E step

- Compute $S_t(i)$ and $S_t(i,j)$ for all *t, i,* and *j* ($1 \le t \le n$, $1 \le i \le k$, $2 \le j \le k$)

$$P(q_t = s_i \mid O_1, \cdots, O_T) = S_t(i)$$

$$P(q_t = s_i, q_{t+1} = s_j \mid o_1, \cdots, o_T) = S_t(i, j)$$

M step (1)

Compute emission probabilities (here we assume a multinomial distribution):

define:

$$B_k(j) = \sum_{t|o_t=j} S_t(k)$$

then

$$b_k(j) = \frac{B_k(j)}{\sum_i B_k(i)}$$

M step (2)

Compute transition probabilities:

$$a_{i,j} = \frac{\hat{n}(i,j)}{\sum_k \hat{n}(i,k)}$$

where

$$\hat{n}(i,j) = \sum_t S_t(i,j)$$

Complete EM algorithm for learning the parameters of HMMs (Baum-Welch)

- Inputs: 1 .Observations $O_1 \ldots O_T$

  2. Number of states, model

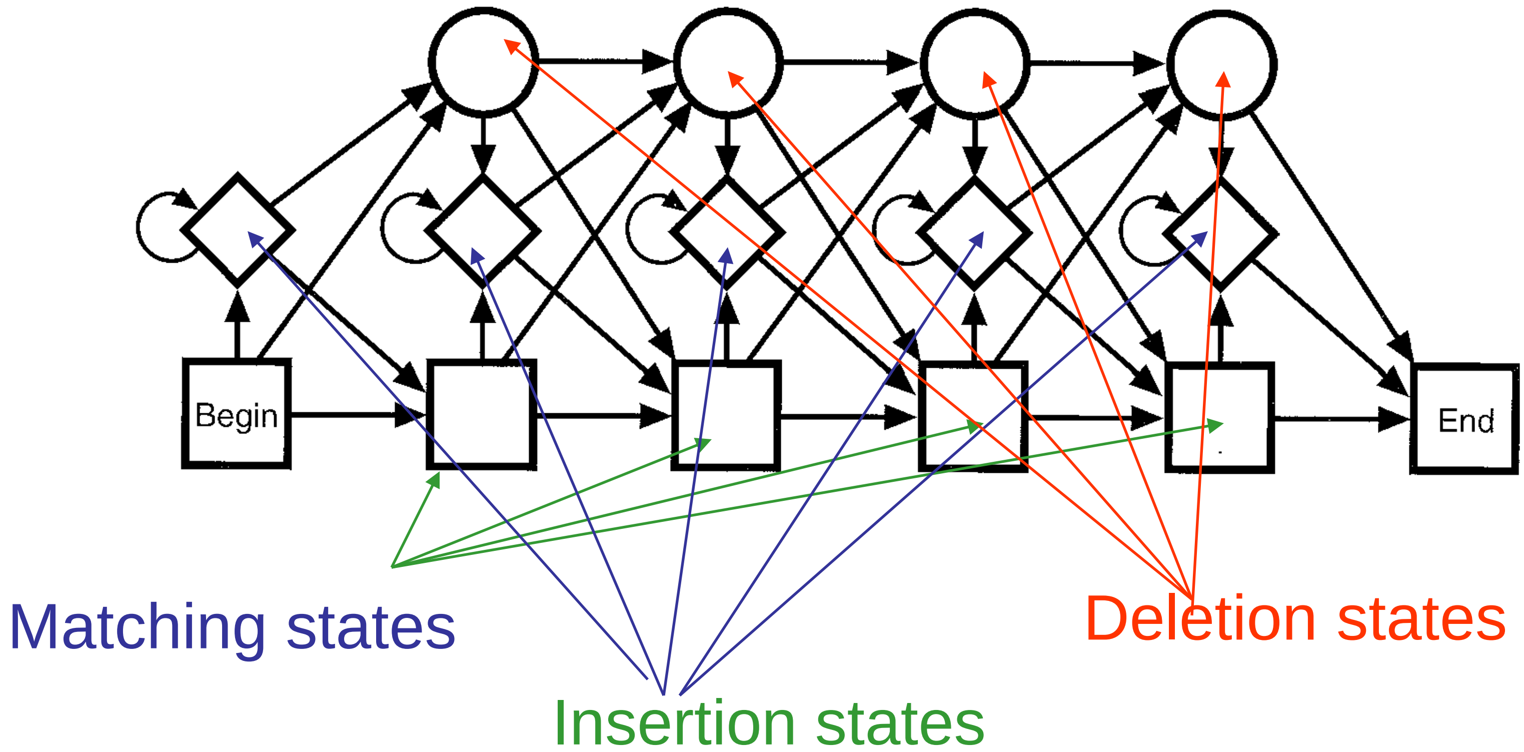
1. Guess initial transition and emission parameters
2. Compute E step: $S_t(i)$ and $S_t(i,j)$
3. Compute M step
4. Convergence? No
5. Output complete model

We did not discuss initial probability estimation. These can be deduced from multiple sets of observation (for example, several recorded customers for speech processing)

HMM for DNA / Protein alignment

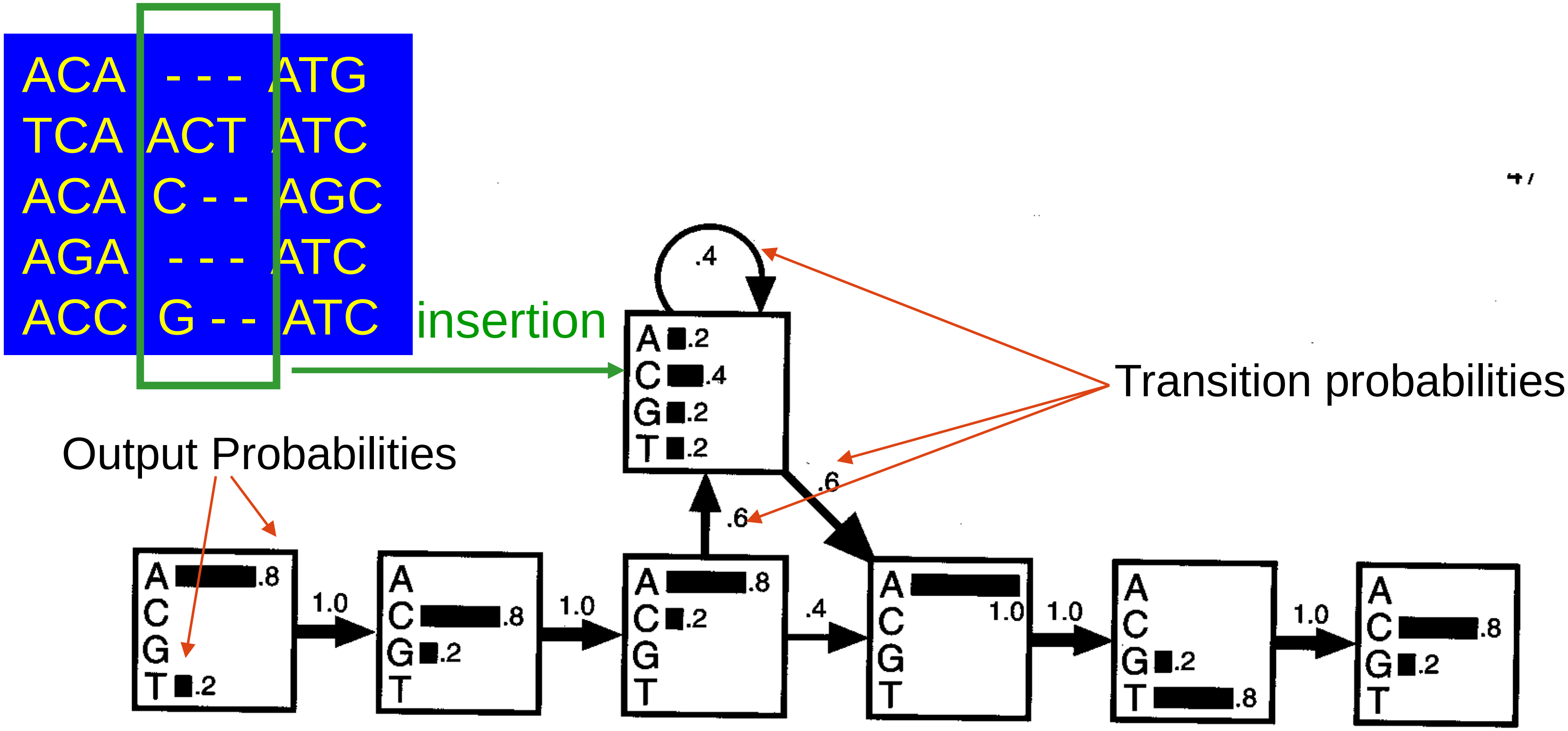
ACA	- - -	ATG
TCA	ACT	ATC
ACA	C - -	AGC
AGA	- - -	ATC
ACC	G - -	ATC

Building HMMs–*Topology*



No of matching states = average sequence length in the family
PFAM Database - of Protein families
(**http://pfam.wustl.edu**)

Building – *from an existing alignment*



A HMM model for a DNA motif alignments, The transitions are shown with arrows whose thickness indicate their probability. In each state, the histogram shows the probabilities of the four bases.

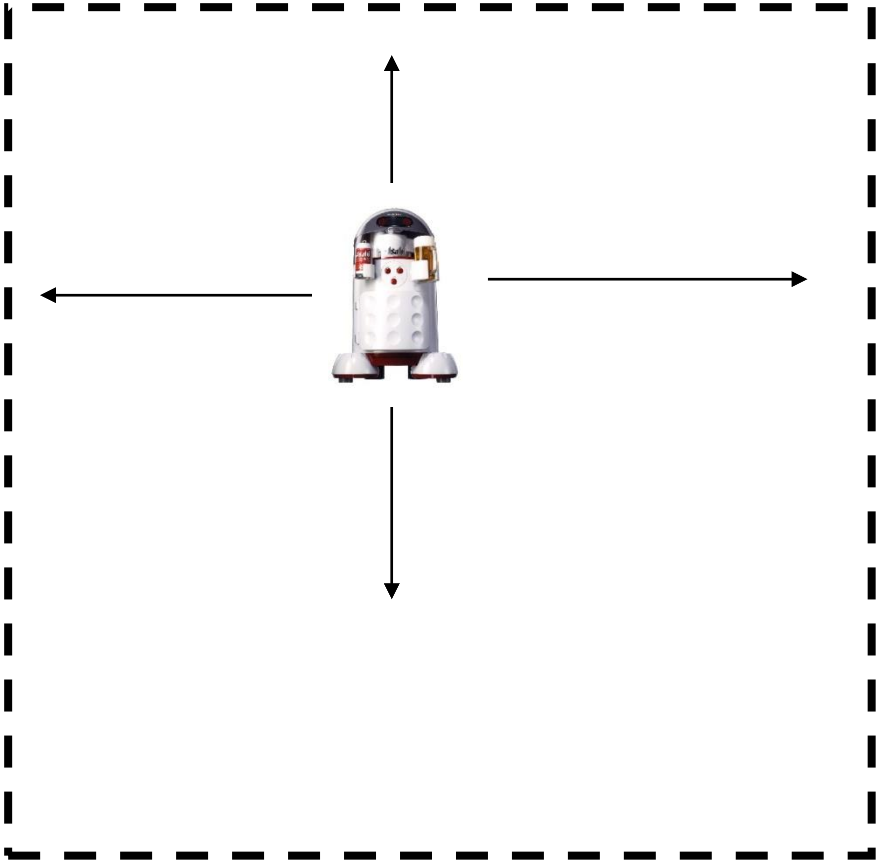
Advanced HMMs

- Factorial HMM's
- Input-output HMMs
- Dynamic Bayesian Networks (DBNs)

Coupling of hidden states

- A robot for tourists
- Location $\in$ {K by K grid}
- Language $\in$ {English, Spanish, French}
- Talk $\in$ {yes, no}

How do we design a HMM for this robot?

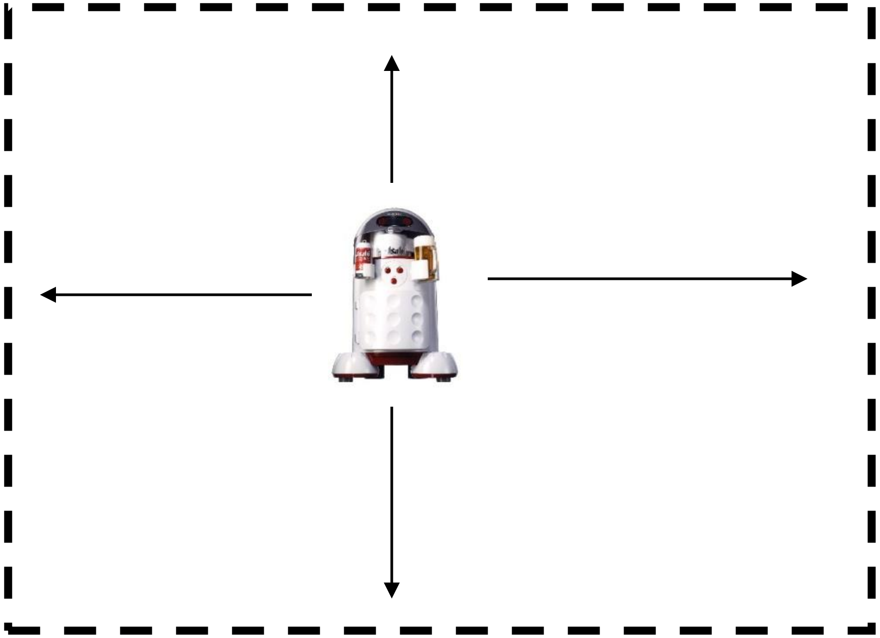


HMM for roboguide

• States: triplets {Loc, Lan, Tal}

• Emissions: based on location and presence / absence of a person

• Transitions: From one triplet to another



Example of transition:

P({Loc = (i,j), Lan = E, Tal = y} | ({Loc = (i,j-1), Lan = E, Tal = n})

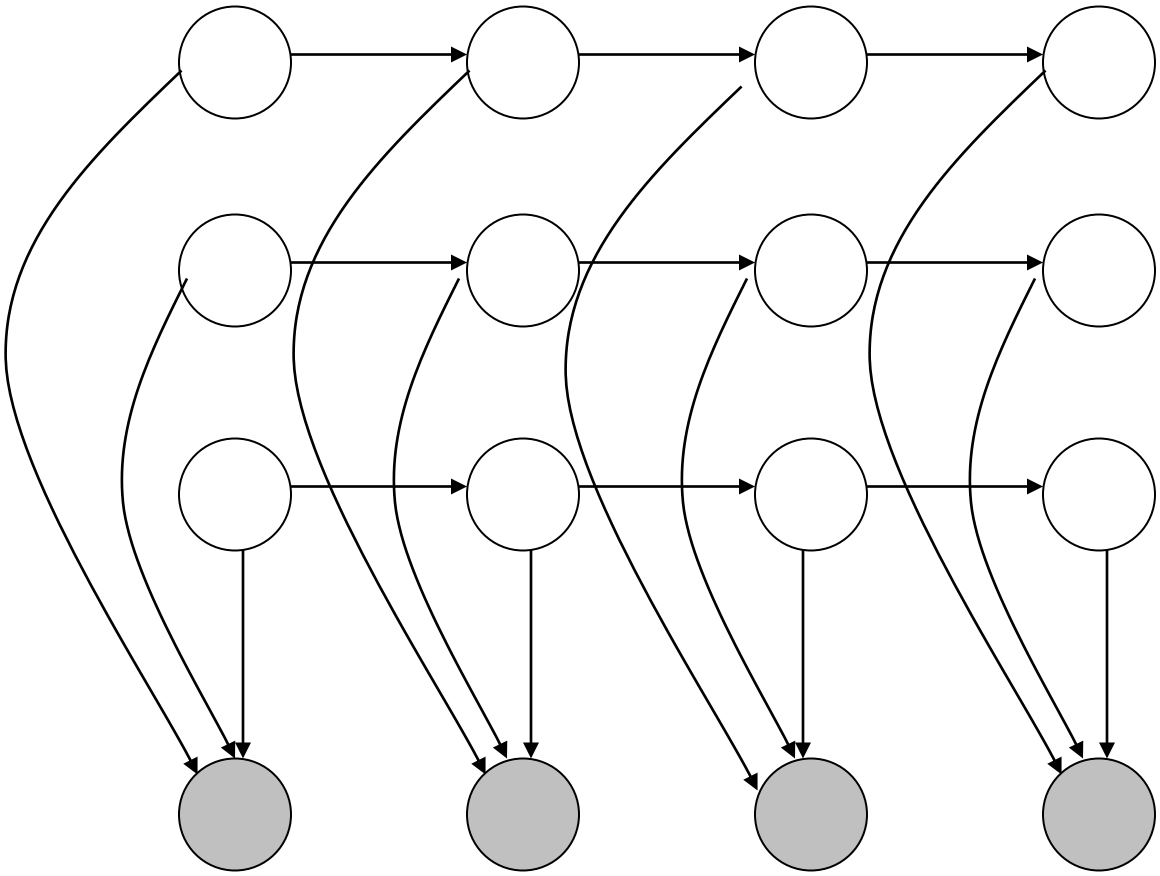
Problems?

Decoupling of states

- In many cases each state needs to be represented by a vector of attributes
- In these cases the transition between attributes may not depend on all the other attributes

  - For example, given a current location the next location is independent of the language being used
- In such cases it is better to use a different representation that is less complex and still captures the correct model

Factorial HMMs



Learning and inference in factorial HMMs

- M step: Same as in HMMs:

  - given expected state assignments compute initial, transition and emission probabilities (could couple states)

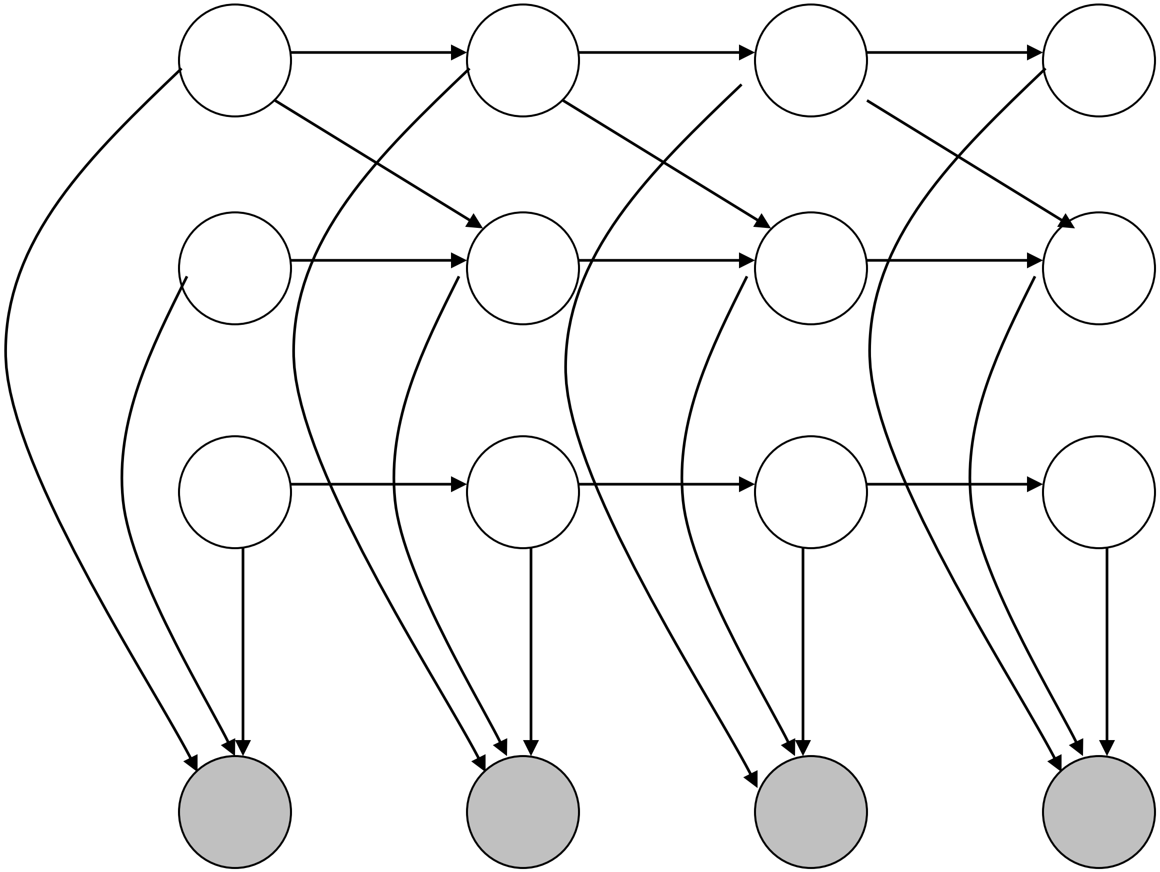
- E step: Harder, we cannot solve efficiently any more

  - The observations couple the states and the E step can be exponential in the number of different types of states

  - In practice people usually use a sampling (Monte Carlo) method for this task.

Factorial HMMs

Can also be used to represent relationships between different types of states. Again, inference is hard but if the states are known (estimated) we can compute transition and other probabilities (but we need more data).



Advanced HMMs

- Factorial HMM's √

- Input-output HMMs

- Dynamic Bayesian Networks (DBNs)

Static input

- Assume we are building a robot tracker which determines the location of a robot based on sensor information
- The model is to be used for 3 different types of robots, each has a slightly different speed and preference as to the next location

Handling the robotype model

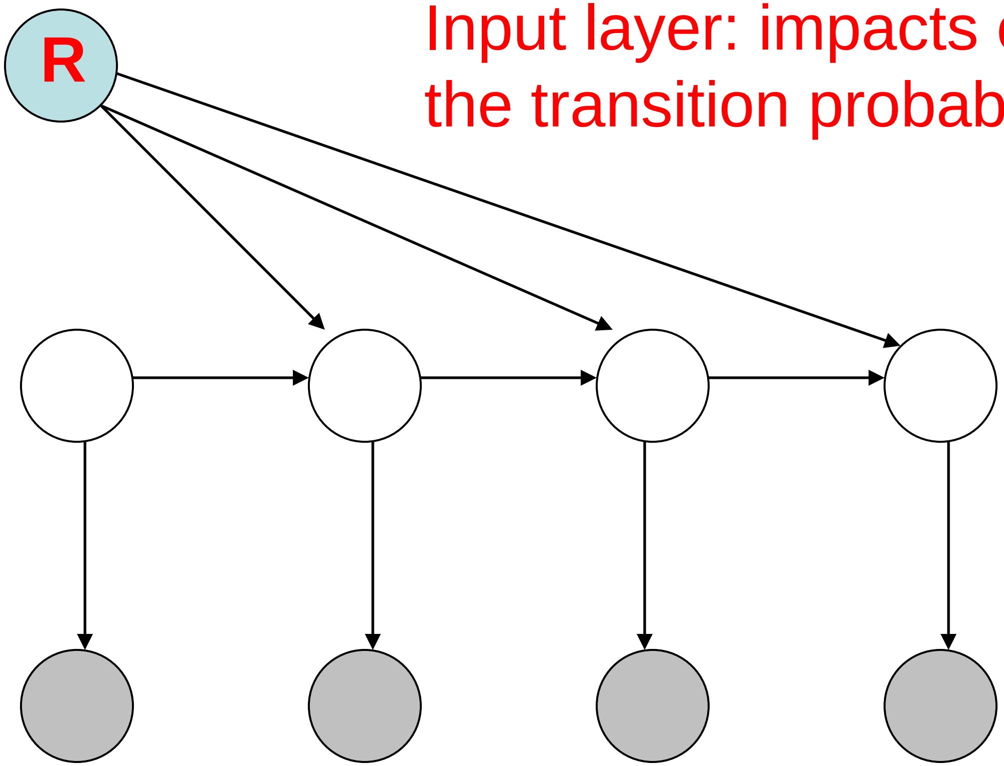
- We can always build separate models for the three different robots
- But this is a waste

  - All robots share the same output probabilities

  - Perhaps they also share other properties (language, etc.).
- Instead, it would be better if we can design one model that will fit all of them

Input-output HMMs

Input layer: impacts only the transition probabilities

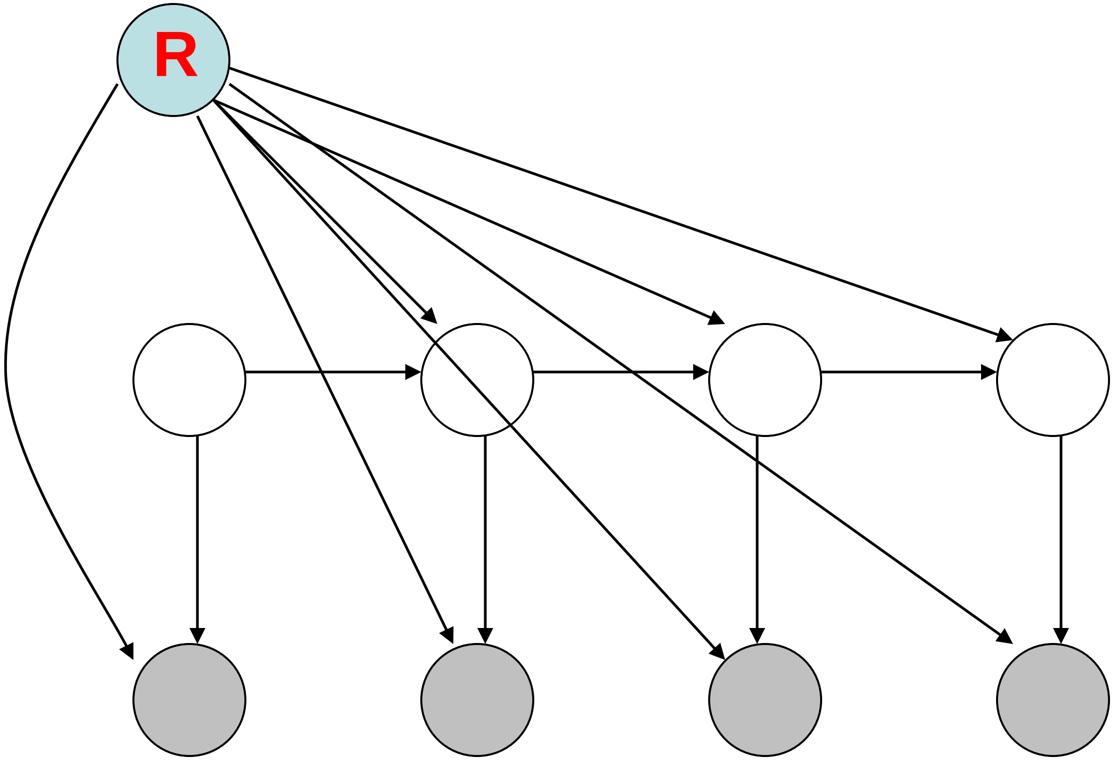


$$p(Q,O \mid M) = \pi(q_1)\,p(o_1 \mid q_1)\prod_t p(q_t \mid q_{t-1}, R)\,p(o_t \mid q_t)$$

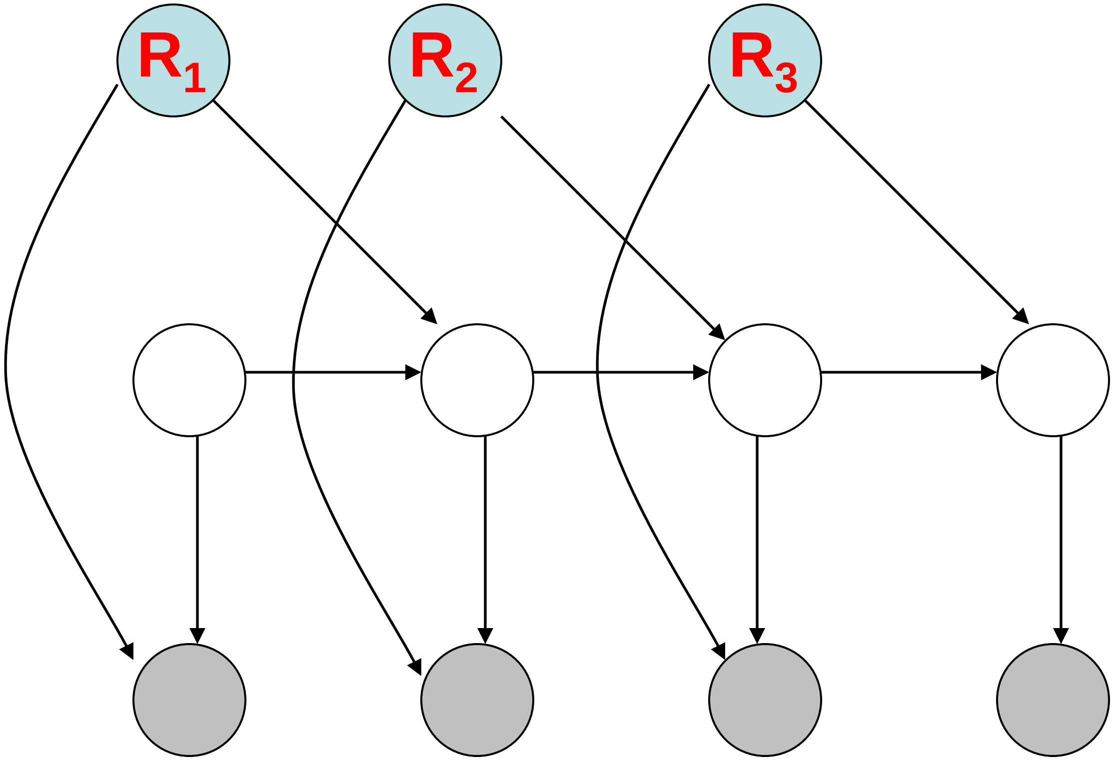
Learning and inference in input-output HMMs

- Depends on how the transition probability is modeled. Since R is given we can just learn a new transition table for each R value (note that the emission probabilities are still the same regardless of R).
- In some cases the transition probability may take on a different format (for example, logistic regression classifier). For such transitions learning is harder.

More general Input-output HMMs



More general Input-output HMMs



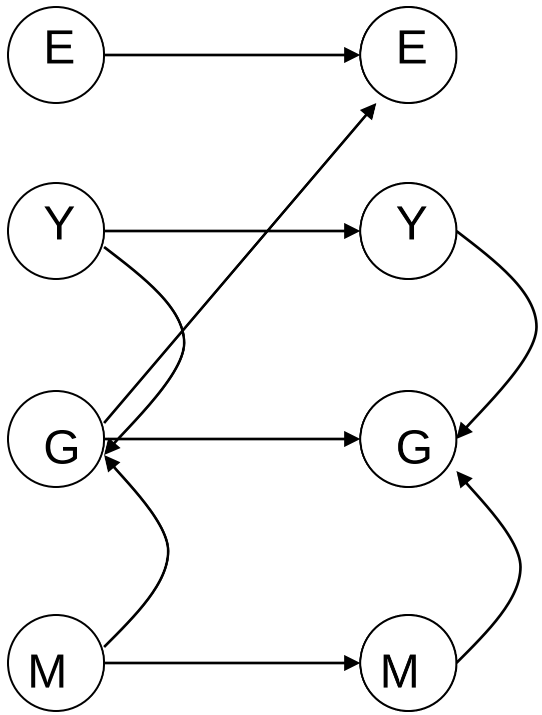
Advanced HMMs

- Factorial HMM’s √
- Input-output HMMs √
- Dynamic Bayesian Networks (DBNs)

Predicting stock prices

- If we knew the price for Microsoft, Yahoo and Ebay today and the price of Google yesterday, could we predict the new price for Google?
- In these and other cases there is no hidden states but there is a strong dependency over time

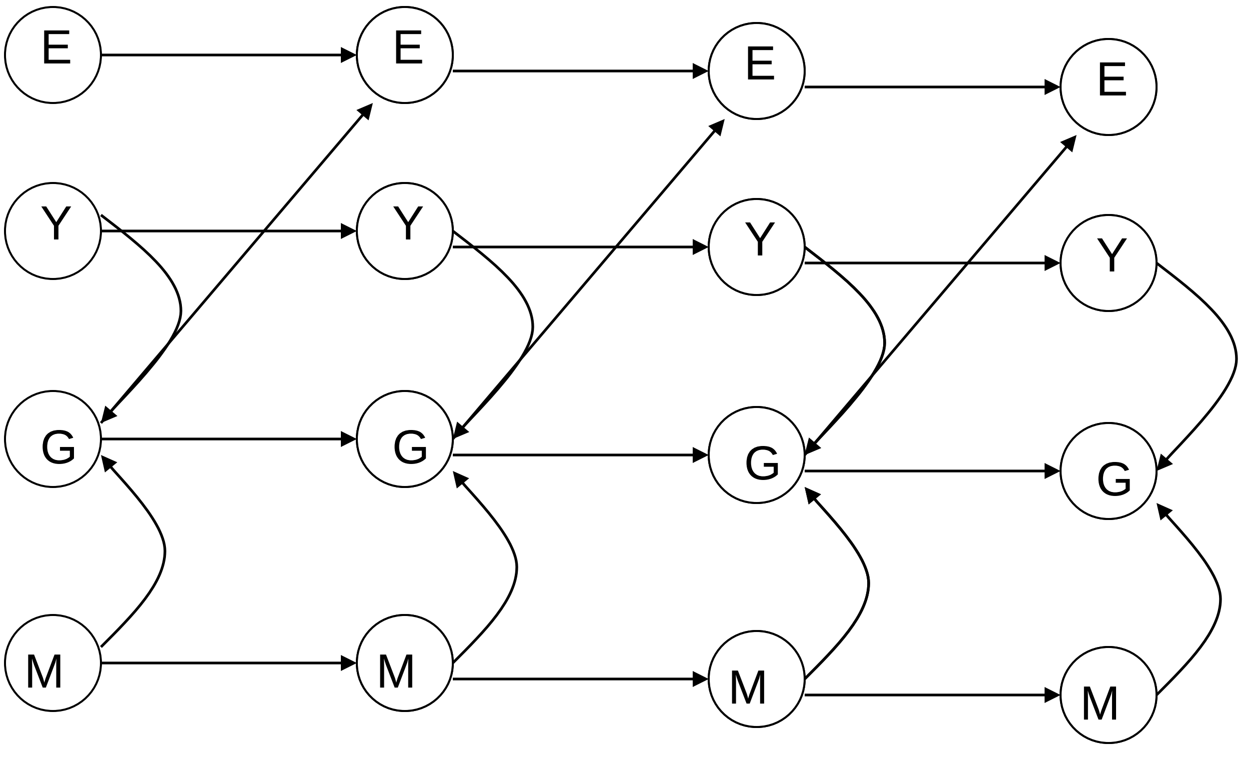
Dynamic Bayesian networks (DBNs)



- DBNs are an extension of Bayesian networks
- They follow the same semantics
- But they are repeated over time and so loops (between time units) are allowed.

$$P(X) = \prod_i p(x_i \mid Pa(x_i))$$

Dynamic Bayesian networks (DBNs)



$$P(X) = \prod_i p(x_i \mid Pa(x_i))$$

Learning and inference in DBNs

- This is really more similar to a Bayesian network (BN) than to a HMM
- Like all BNs, learning and inference is NP hard
- Which leads to several approximation methods:

  - Hill climbing

  - Annealing

  - Greedy algorithms

  - etc.

What you should know

- Why HMMs? Which applications are suitable?
- Learning HMMs: EM algorithm (Baum-Welch)
- Extensions of HMMs