

Jin Gu

Department of Automation, Tsinghua University

Email: jgu@tsinghua.edu.cn

Phone: (010) 62794294-866

Chapter 5 Dynamic Models

2021 Fall

Jin Gu (古瑾)

Outlines

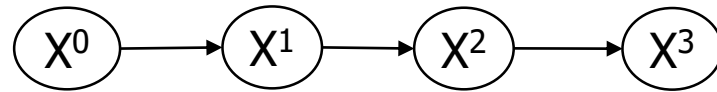
- Stochastic Process and Markov property
- Dynamic Bayesian Networks
- Hidden Markov Models (HMMs)
 - State-Observation Models
 - A Few Examples of HMM Representations
 - Calculations for HMMs
 - Generate Simulated Data
- Extended Models

Textbook References

- Textbook 2
 - Chapter 17 (Methods in Details)
- Textbook 1
 - Chapter 6.1 ~ 6.3

Markov / Memoryless Property

- Add the time dimension to variables $X \rightarrow X^{(t)}$
- Assumptions



- Time can be discretized into interesting points

t_1, t_2, \dots, t_n

$$P(X) = \prod_{t=1}^n P(X(t) | X(1), \dots, X(t-1))$$

- Markov assumption holds

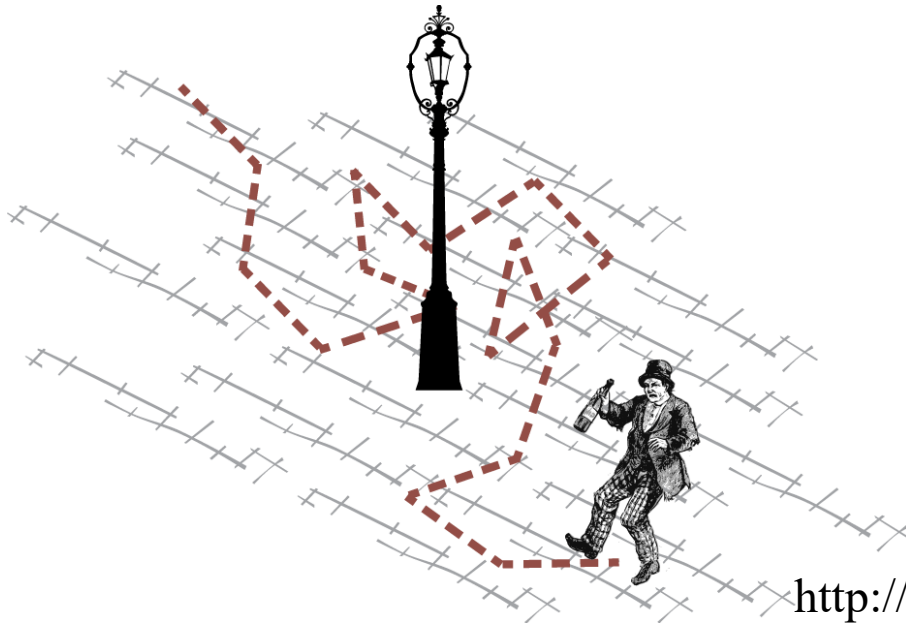
$$P(X) = \prod_{t=1}^n P(X(t) | X(t-1))$$

- Distribution is stationary (time invariant or homogeneous)

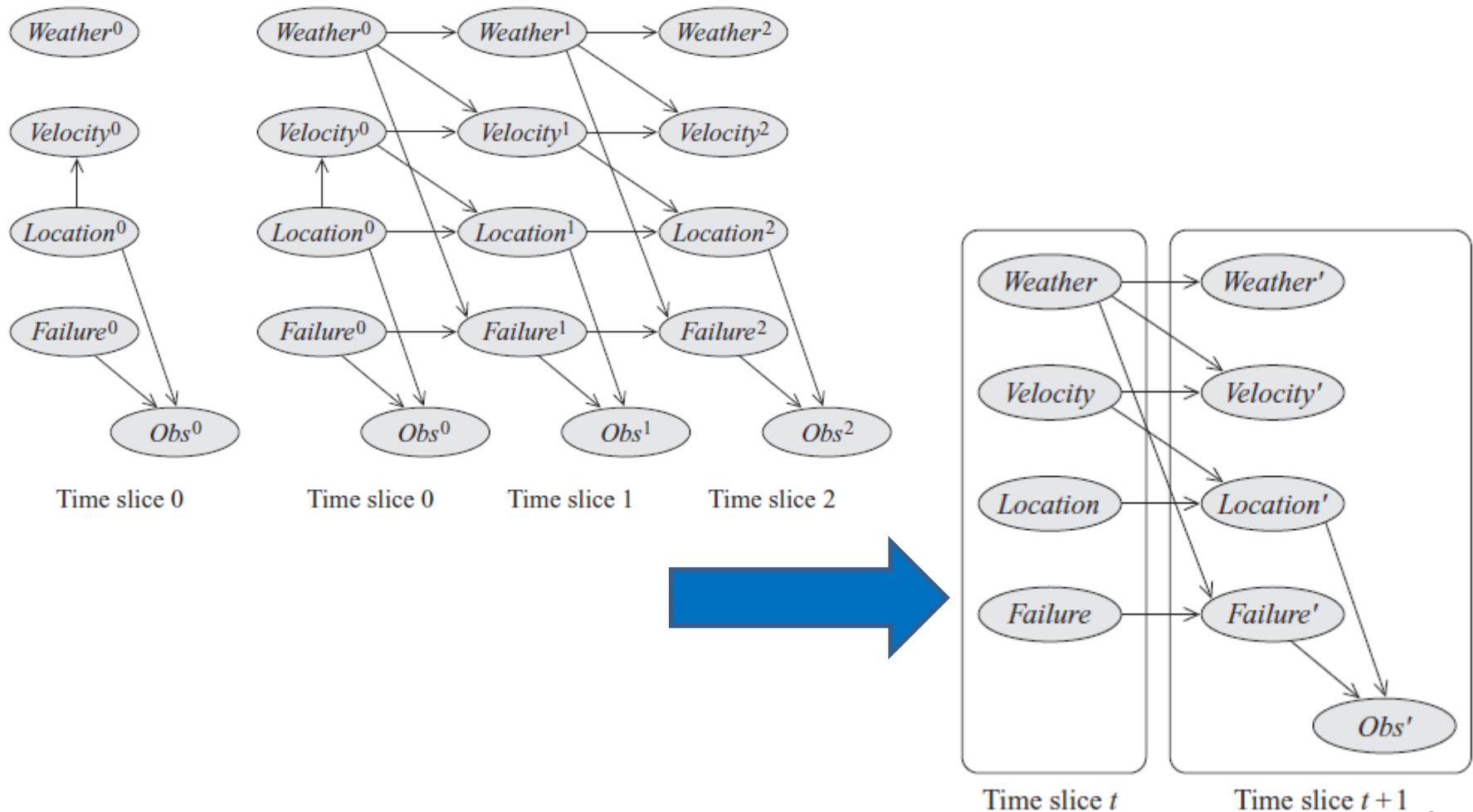
$$\forall i, j : P(X(i) | X(i-1)) = P(X(j) | X(j-1))$$

Example: Random Walk

- Define $X^{(t)}$ as the coordinate of particle at time point t . The particle can randomly move to neighbor variables (nodes) with fixed probability. Then, $P(X^{(t+1)})$ is only determined by $X^{(t)}$.



Dynamic Bayesian Networks for Time *Invariant* Process/System

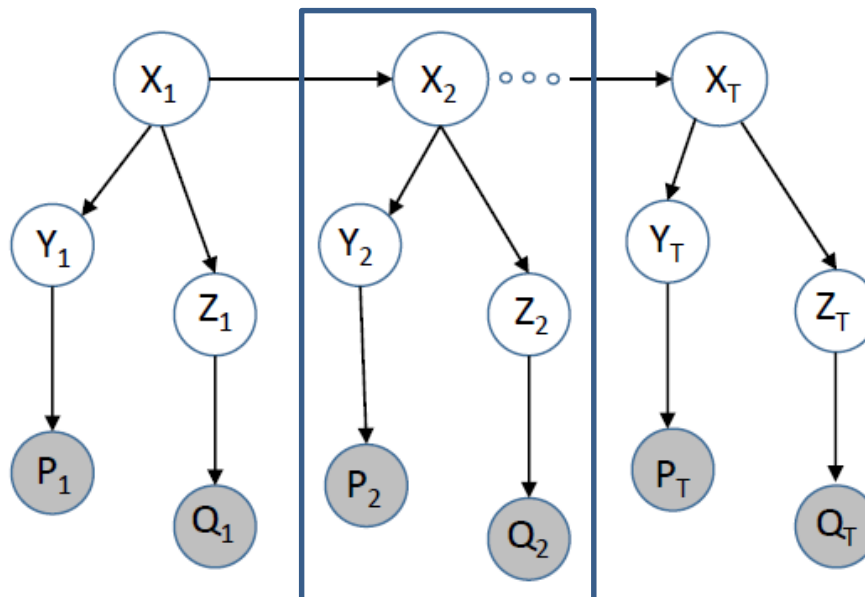


Recall Assignment #2(5)

Here is a HCI system which can recognize human's instructions based on his gesture (by a camera) and voice (by a microphone). The raw signals detected by the two sensors should be treated as the observations of the real gestures and voices. The computer samples the sensors 10 times per second. According to these descriptions:

- (1) Please draw a Bayesian network to model the human instruction recognition process at a given time.
- (2) Please extend your model to consider the continuous sampling process across a period of time.

(You can use the symbols you like to represent the essential variables you think, but please explain the meaning each variable symbol stands for.)



Multi-stream HMMs

What will you **LEARN** this week?

- How to model time-invariant systems by dynamic PGMs?
- How to introduce hidden variable(s) to model the dynamic property of a system?
- The basic calculations in HMMs

The Parameterization of Dynamic BNs

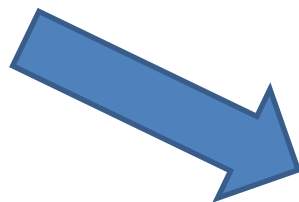
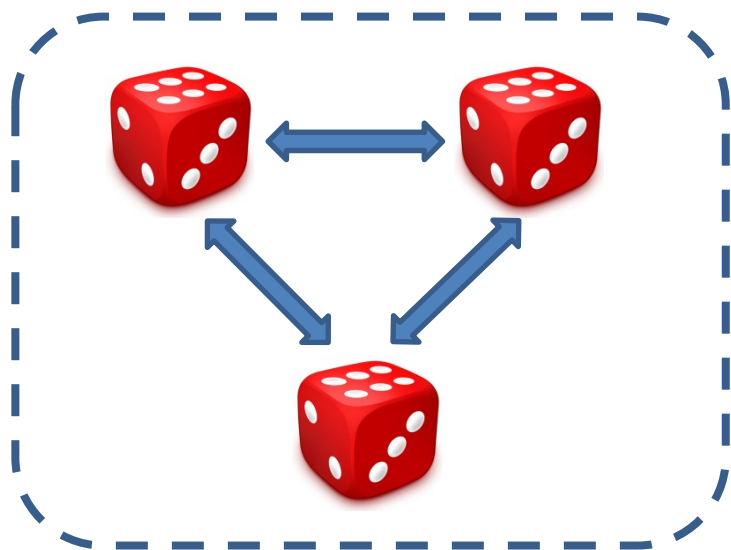
- Usually assume *time-invariant* systems (parameters are constant over time)
- The parameters **in** each time slice
 - The parameters between internal state variables
 - The emission or observation parameters between state variables and observation variables
- The transition parameters **between** two consecutive time slices

Dice problem

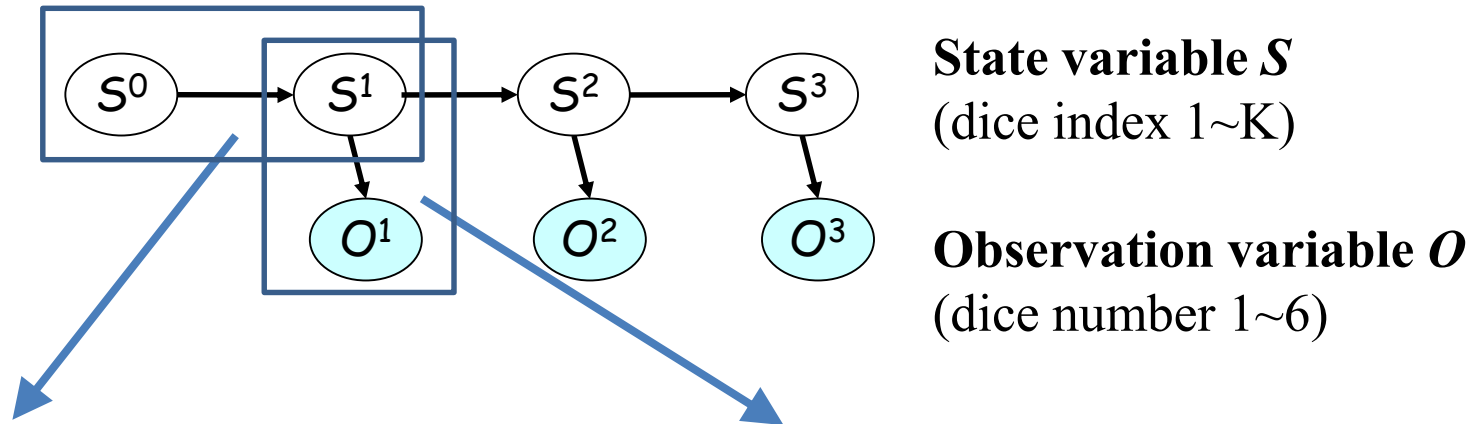
- You stay in a casino for a long time. You find that the probability of dice is equal for each of its six faces (1-6). But you doubt that there must be some trick on the dice. So you record the series of the dice number of each play, and want to find the problem....



Dice problem



Hidden Markov Models



Local structure #1

Hidden states (cannot be observed) are linked as Markov chain:

$$P(S^T | S^{t < T}) = P(S^T | S^{T-1})$$

Local structure #2

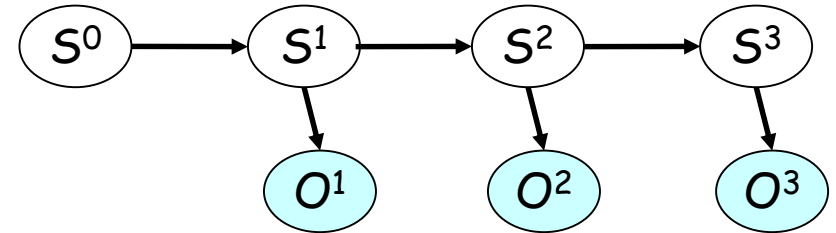
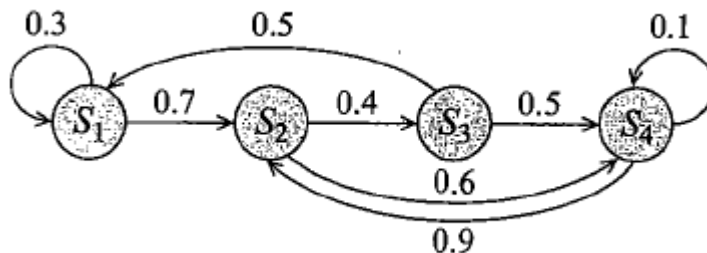
Observable variables are only determined by the hidden state:

$$P(O^T | S^{t \leq T}) = P(O^T | S^T)$$

State-Observation Models

- If the hidden variables can be *discretized* into several states, HMM can be represented as State-Observation Models.

	s_1	s_2	s_3	s_4
s_1	0.3	0.7	0	0
s_2	0	0	0.4	0.6
s_3	0.5	0	0	0.5
s_4	0	0.9	0	0.1



Hidden states (cannot be observed) are linked as Markov chain:

$$P(S^T | S^{t < T}) = P(S^T | S^{T-1})$$

Observable variables are only determined by the hidden state:

$$P(O^T | S^{t \leq T}) = P(O^T | S^T)$$

State-Observation Models

- State-State **Transmission**

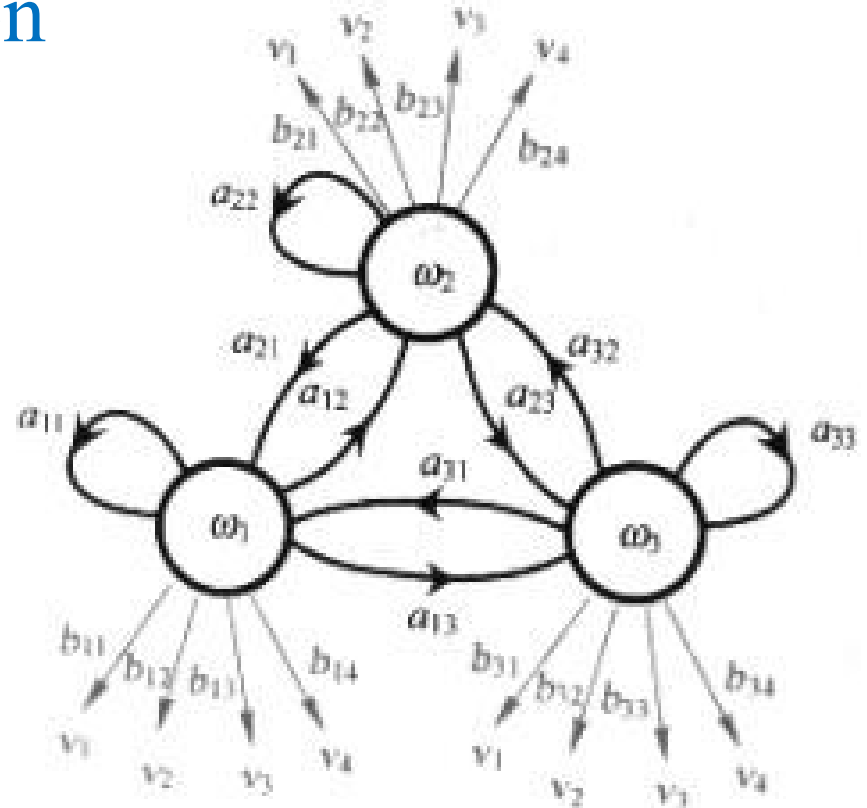
Matrix $T_{(n,n)}$

$$- S^t = TS^{t-1}$$

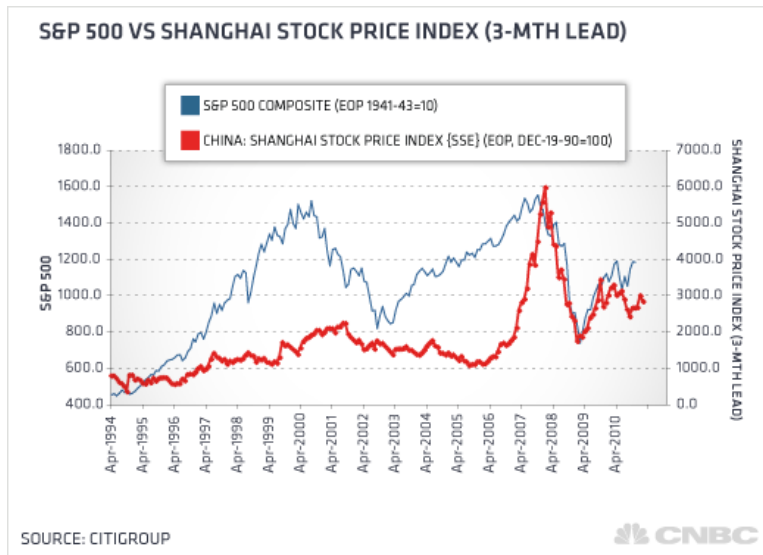
- State-Observation

Emission Matrix $E_{(m,n)}$

$$- O^t = ES^t$$



Example: Stock Market Index



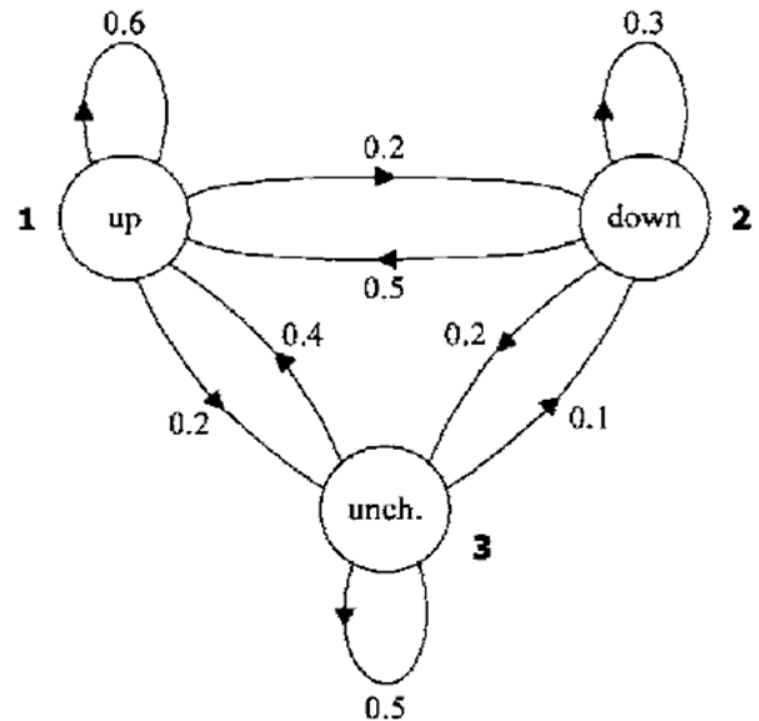
Initial state probability matrix

$$\pi = (\pi_i) = \begin{pmatrix} 0.5 \\ 0.2 \\ 0.3 \end{pmatrix}$$

State-transition probability matrix

$$\mathbf{A} = \{a_{ij}\} = \begin{bmatrix} 0.6 & 0.2 & 0.2 \\ 0.5 & 0.3 & 0.2 \\ 0.4 & 0.1 & 0.5 \end{bmatrix}$$

We can define **three** different states:
UP/DOWN/UNCHANGE



Could You Find Any Other Application?

App: Chromatin States

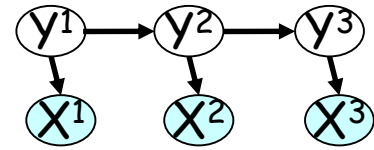
DNA methylation
DNA 甲基化

Histone modification
组蛋白修饰

Nucleosome positioning
核小体占位



Calculations in HMMs



- Problem 1: $P(\mathbf{X}|\boldsymbol{\theta})$, given the model and the observation sequence, infer the probability of getting that observation sequence from the model
- Problem 2: $\underset{Y}{\operatorname{argmax}} P(\mathbf{X}, \mathbf{Y}|\boldsymbol{\theta})$, given the model and the observation sequence, infer the hidden labels of the sequence
- Problem 3: $\underset{\boldsymbol{\theta}}{\operatorname{argmax}} P(\mathbf{X}|\boldsymbol{\theta})$, if parameters are unknown, learn them from the observation sequence

Question #1: $P(\mathbf{X}|\boldsymbol{\theta})$

Inference

Formal Model Representation

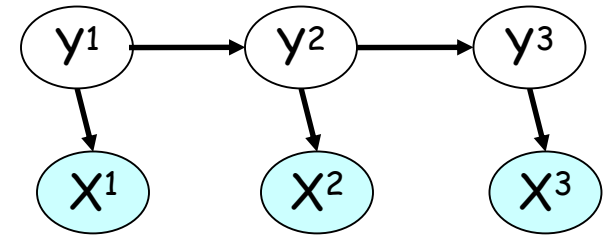
- HMMs (*discretized*) have the below parameters and the observation sequences X

$$\theta = \left\{ \begin{array}{l} T = \{t_{i,j}, \quad i, j = 1 \dots N\}, \\ E = \{e_{i,j}, \quad i = 1 \dots N, j = 1 \dots K\}, \\ \pi = \{\pi_i, \quad i = 1 \dots N\} \end{array} \right\}$$

$$X = \{x_t, \quad t = 1, \dots, T\}$$

Basic Idea

- $P(\mathbf{X}) = \sum_{\mathbf{Y}} P(\mathbf{XY})$



- By factorization theorem

- $P(\mathbf{XY}) = P(Y_1)P(X_1|Y_1)P(Y_2|Y_1)P(X_2|Y_2)\dots\dots$

- or

- $P(\mathbf{XY}) = P(X_T|Y_T)P(Y_T|Y_{T-1})P(X_{T-1}|Y_{T-1})P(Y_{T-1}|Y_{T-2})\dots\dots$

- So the model can be solved by eliminating Y_i iteratively in either order

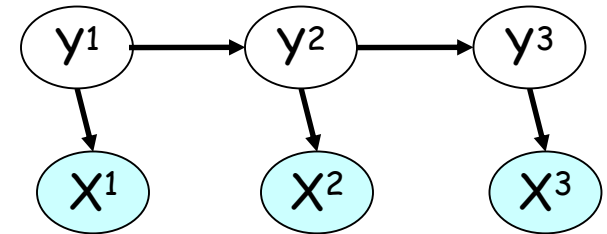
Compute $P(X|\theta)$

$$\alpha_t(i) = P(x_1, \dots, x_t, y_t = i \mid \theta)$$

Forward/Backward algorithm

- Initialization:

$$\alpha_1(i) = \pi_i e_{i,x_1}$$



- Induction:

$$\alpha_{t+1}(i) = \left[\sum_{j=1}^N \alpha_t(j) t_{j,i} \right] e_{i,x_{t+1}}$$

- Termination:

$$P(X \mid \theta) = \sum_{i=1}^N \alpha_T(i)$$

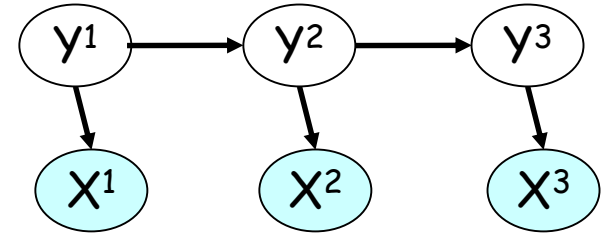
Compute $P(X|\theta)$

$$\beta_t(i) = P(x_{t+1}, \dots = i, \theta)$$

Forward/Backward algorithm

- Initialization:

$$\beta_{T+1}(i) = 1$$

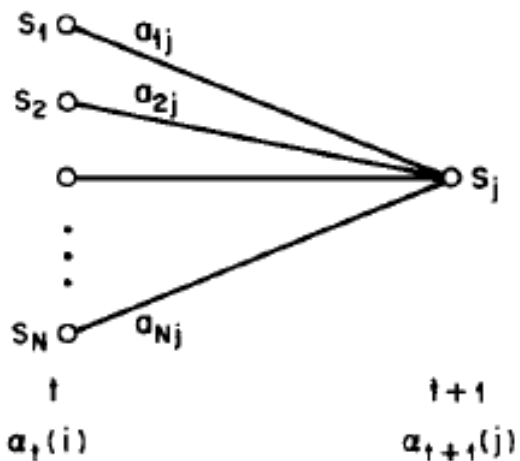


- Induction:

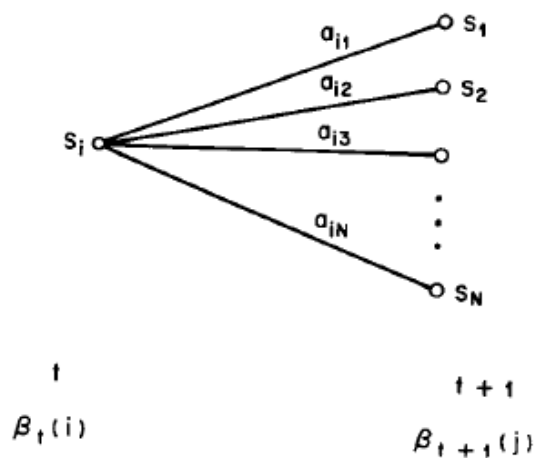
$$\beta_t(i) = \sum_{j=1}^N t_{i,j} e_{j,x_{t+1}} \beta_{t+1}(j)$$

- Termination:

$$P(X|\theta) = \beta_0(i) = \sum_{j=1}^N \pi_j e_{j,x_1} \beta_1(j)$$

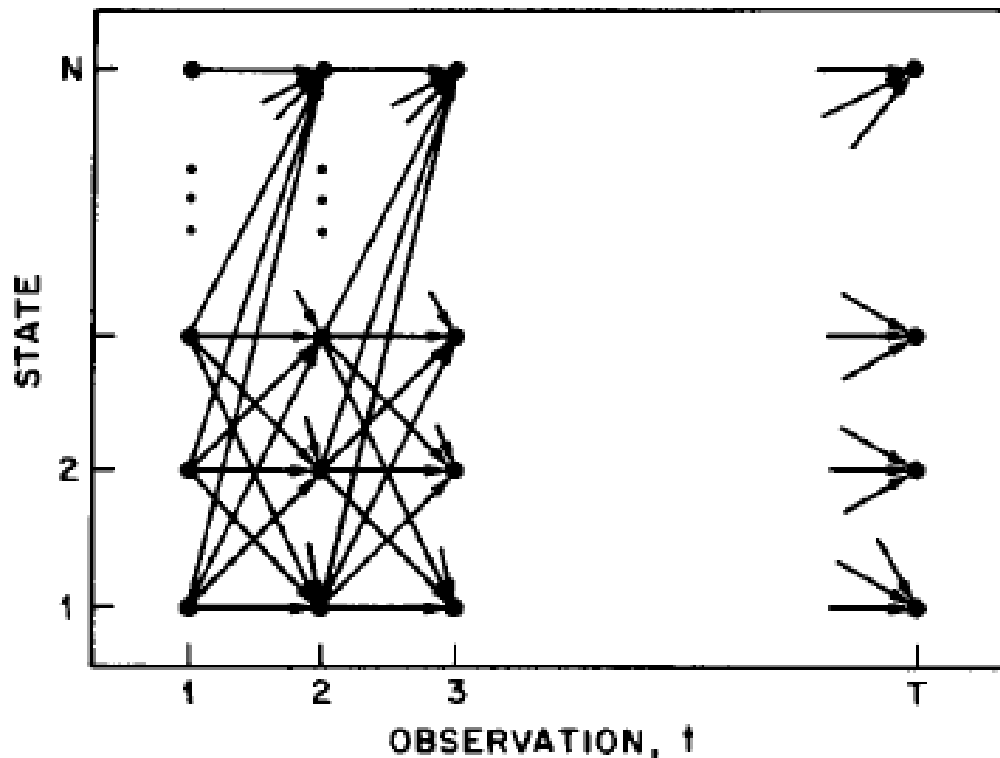


Local Transition (forward)



Local Transition (Backward)

Full Chain



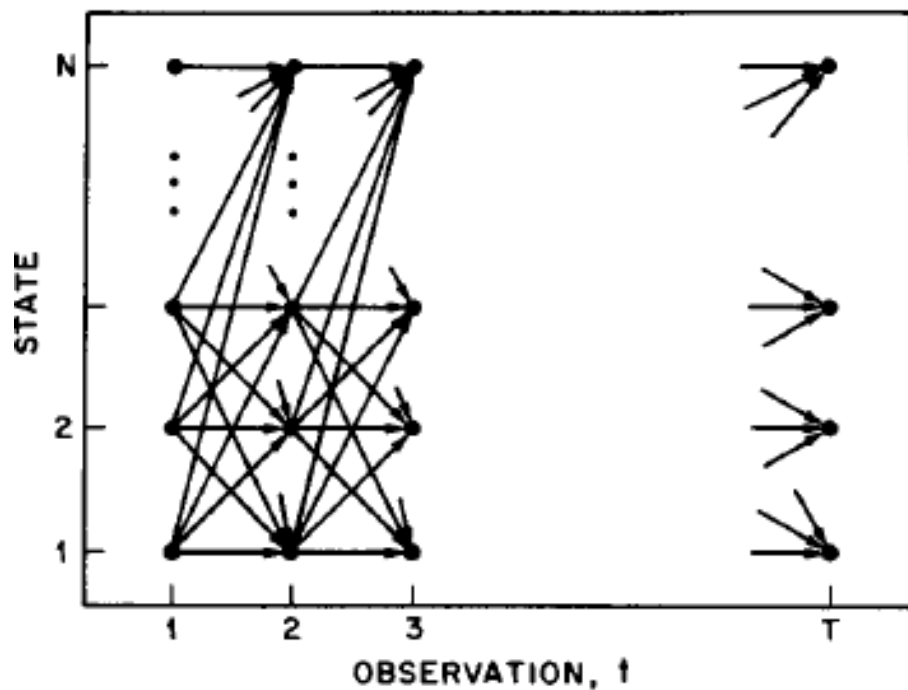
Figures from Rabiner LR.
Proceedings of the IEEE 1989,
 77(2):257-286.

Question #2: $\operatorname{argmax}_Y P(X, Y | \theta)$

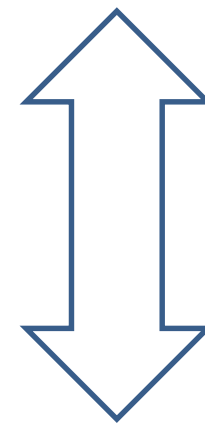
Inference

Full Chain

Find the optimal path



$$\operatorname{argmax}_Y P(X, Y | \theta)$$



Infer the most possible state

Figures from Rabiner LR.
Proceedings of the IEEE 1989,
77(2):257-286.

Compute $\underset{Y}{\operatorname{argmax}} P(X, Y|\theta)$

HMM inference (Viterbi algorithm)

- Known transition matrix and emission matrix
- Infer the maximum probability STATE series
- The probability at time 1 with observation x_1

$$\text{For } Y_1 = i: \quad \delta_{1,i} = \pi_i e_{i,x_1}$$

- The probability at time 2 with observation x_2

$$\delta_{2,i} = e_{i,x_2} \max_{y_1=1,\dots,N} \left(\pi_{y_1} e_{y_1,x_1} \times t_{y_1,i} \right) = e_{i,x_2} \max_{y_1=1,\dots,N} \left(\delta_{1,y_1} \times t_{y_1,i} \right)$$

$$\phi_2(i) = \arg \max_{y_1=1,\dots} \left(\delta_{1,y_1} \times t_{y_1,i} \right)$$

Important: Record **the optimal path** dynamically!!

HMM inference (Viterbi algorithm)

- The probability at time t with observation x_t

$$\delta_{t,i} = e_{i,x_t} \max_{y_{t-1}=1,\dots,N} \left(\delta_{t-1,y_{t-1}} \times t_{y_{t-1},i} \right)$$

$$\phi_t(i) = \arg \max_{y_{t-1}=1,\dots,N} \left(\delta_{t-1,y_{t-1}} \times t_{y_{t-1},i} \right)$$

- For the last observation

$$\delta_{T,i} = e_{i,x_T} \max_{y_{T-1}=1,\dots,N} \left(\delta_{T-1,y_{T-1}} \times t_{y_{T-1},i} \right)$$

$$\phi_T(i) = \arg \max_{y_{T-1}=1,\dots} \left(\delta_{T-1,y_{T-1}} \times t_{y_{T-1},i} \right)$$

$$y_T^* = \arg \max_{y_T=1,\dots} \left(\delta_{T,y_T} \right)$$

HMM inference (Viterbi algorithm)

- After y_t is inferred, trace back to get other y_i

$$y_T^* = \arg \max_{y_T=1, \dots, N} (\delta_{T, y_T})$$

$$\phi_t(i) = \arg \max_{y_{t-1}=1, \dots, N} (\delta_{t-1, y_{t-1}} \times t_{y_{t-1}, i})$$

- We can infer other y_i

$$y_{T-1}^* = \phi_T(i = y_T^*) = \arg \max_{y_{T-1}=1, \dots} (\delta_{T-1, y_{T-1}} \times t_{y_{T-1}, i})$$

$$y_{t-1}^* = \phi_t(i = y_t^*) = \arg \max_{y_{t-1}=1, \dots} (\delta_{t-1, y_{t-1}} \times t_{y_{t-1}, i})$$

Question #3: $\operatorname{argmax}_{\theta} P(X|\theta)$

Learning

Compute $\underset{\theta}{\operatorname{argmax}} P(X|\theta)$

Baum–Welch algorithm

- General strategy
 - 1) set an initial value of the parameters
 - 2) then do ***inferences of hidden values***
 - 3) then ***re-estimate or re-learn the parameters***
 - 4) Repeat the processes till **convergence**
- Possible problems
 - No guarantee of **convergence**
 - No guarantee of **global optimization**

Compute $\arg\max_{\theta} P(X|\theta)$

Baum–Welch algorithm

- Define an intermediate variables for inference
 - The probability of $y=i$ for time point t and $y=j$ for the following time point $t+1$

$$\xi_t(i, j) = P(y_t = i, y_{t+1} = j | X, \theta)$$

Compute $\underset{\theta}{\operatorname{argmax}} P(X|\theta)$

Baum–Welch algorithm


$$\theta^0 = \{T^0, E^0, \pi^0\}$$

- Use forward and backward algorithm to calculate probability

$$\alpha_t(i) = P(x_1, \dots, x_t, y_t = i | \theta^0)$$

$$\beta_t(i) = P(x_{t+1}, \dots, x_T | y_t = i, \theta^0)$$

$$\xi_t(i, j) = P(y_t = i, y_{t+1} = j | X, \theta)$$


$$\xi_t(i, j) = \frac{\alpha_t(i) t_{i,j} e_{j,x_{t+1}} \beta_{t+1}(j)}{\sum_{i=1}^Y \sum_{j=1}^Y \alpha_t(i) t_{i,j} \beta_{t+1}(j) e_{j,x_{t+1}}}$$

Compute $\underset{\theta}{\operatorname{argmax}} P(X|\theta)$

Baum–Welch algorithm

- The probability of $Y=i$ for time point t :

$$\gamma_t(i) = \sum_{j=1}^Y \xi_t(i, j)$$

- So expected times for state stayed in $Y=i$ and the expected times for state transition $i-j$:

$$\sum_{t=0}^{T-1} \gamma_t(i)$$

$$\sum_{t=0}^{T-1} \xi_t(i, j)$$

Compute $\underset{\theta}{\operatorname{argmax}} P(X|\theta)$

Baum–Welch algorithm

- Re-estimate all parameters (maximize)

$$t_{i,j} = \frac{\sum_{t=0}^{T-1} \xi_t(i,j)}{\sum_{t=0}^{T-1} \gamma_t(i)} \quad e_{i,x} = \frac{\sum_{t=0}^T \mathbf{I}(x_t = x) \gamma_t(i)}{\sum_{t=0}^T \gamma_t(i)}$$

- Repeat above steps until convergence

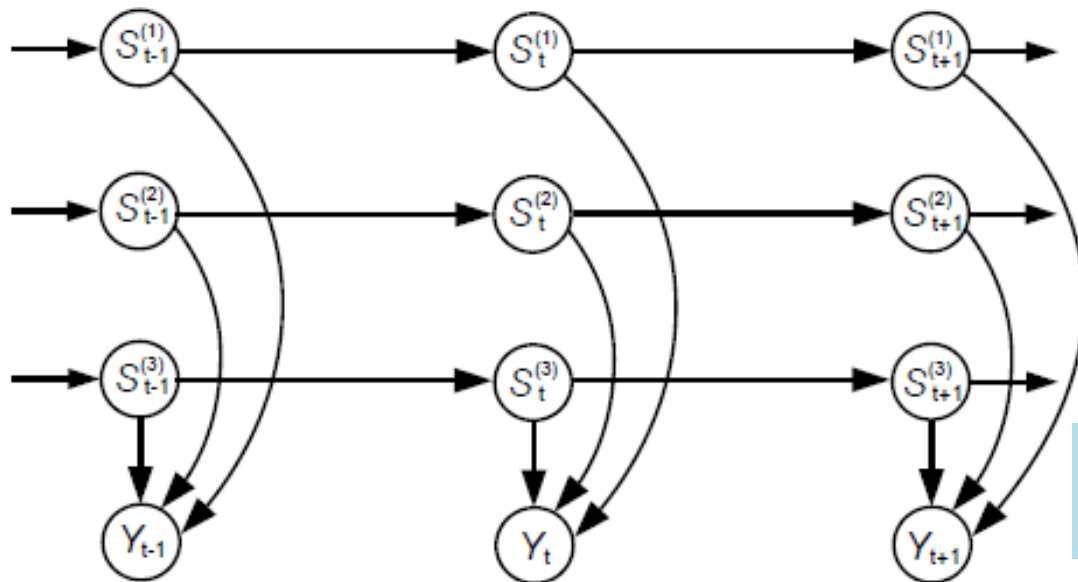
$$\left| \log(P(X|\theta)) - \log(P(X|\theta^0)) \right| < \varepsilon$$

Generate simulated data using HMMs

- Find the initial probability for hidden states
- Markov chain Monte Carlo (MCMC)
 - They are a class of algorithms for sampling from probability distributions based on **constructing a Markov chain that has the desired distribution as its equilibrium distribution**. The state of the chain after a large number of steps is then used as a sample of the desired distribution.
 - A good chain will have rapid mixing (**the stationary distribution is reached quickly starting from an arbitrary position**) described further under Markov chain mixing time.

Extended Models

- Factorial HMMs
- Hierarchical HMMs (GeneScan)
- Bayesian HMMs
- Two HMMs comparisons
- Max Entropy Markov Models
- Conditional Random Fields

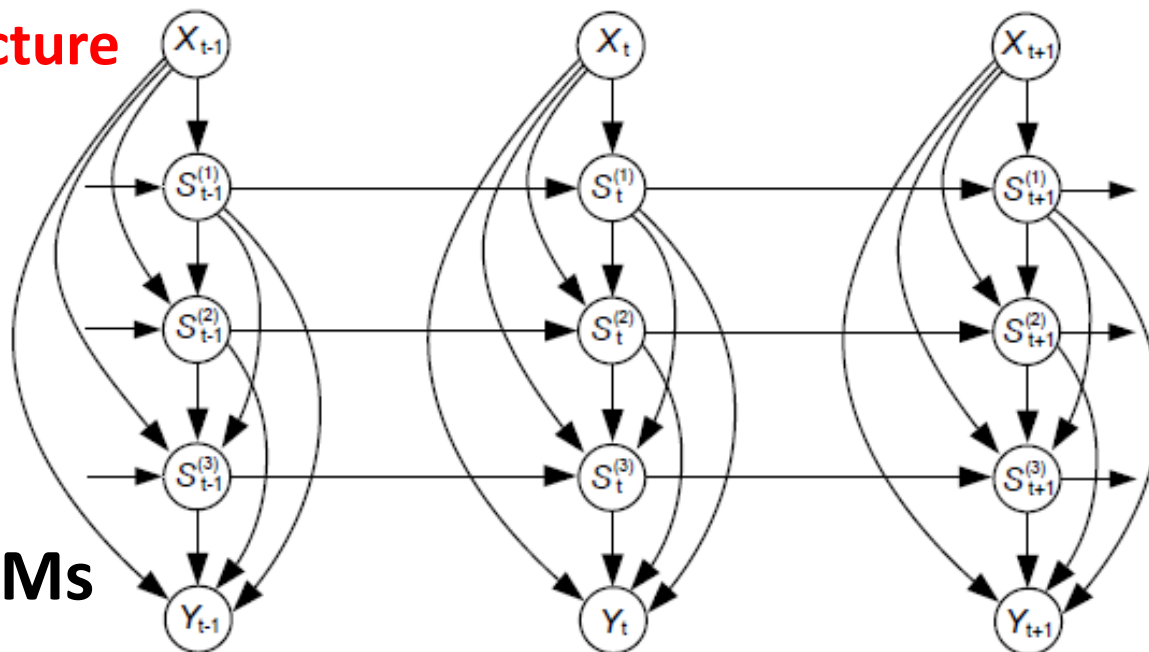


Factorial HMMs
Hierarchical HMMs

Q: Could you write down the probabilities of these models?

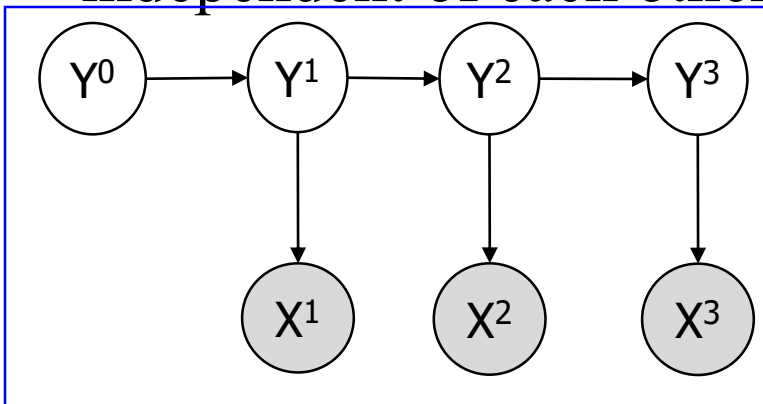
**What ever complex structure
 if you can solve it....**

Tree Structured HMMs

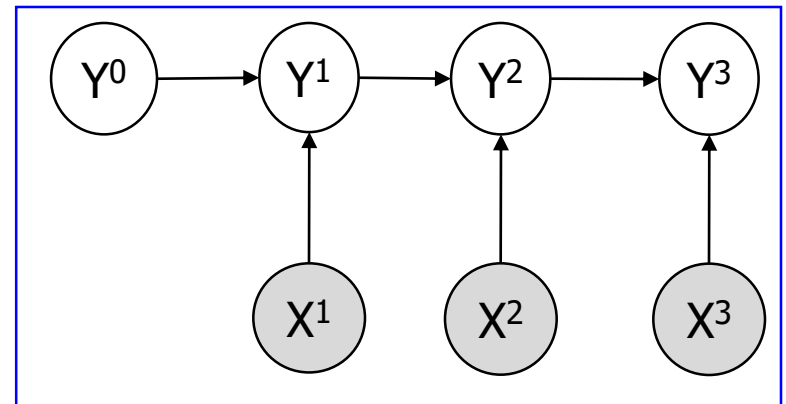


Max Entropy Markov Models

- In machine learning, a maximum-entropy Markov model (MEMM), or conditional Markov model (CMM), is a graphical model for sequence labeling that combines features of hidden Markov models (HMMs) and maximum entropy (MaxEnt) models. An **MEMM is a discriminative model** that extends a standard maximum entropy classifier by assuming that the unknown values to be learned are connected in a Markov chain rather than being conditionally independent of each other.

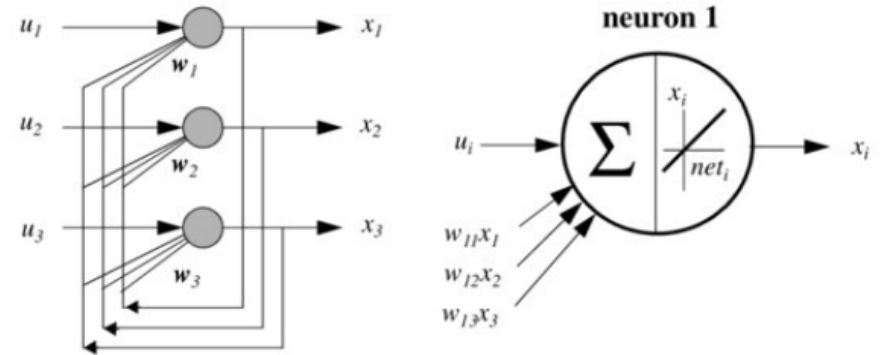
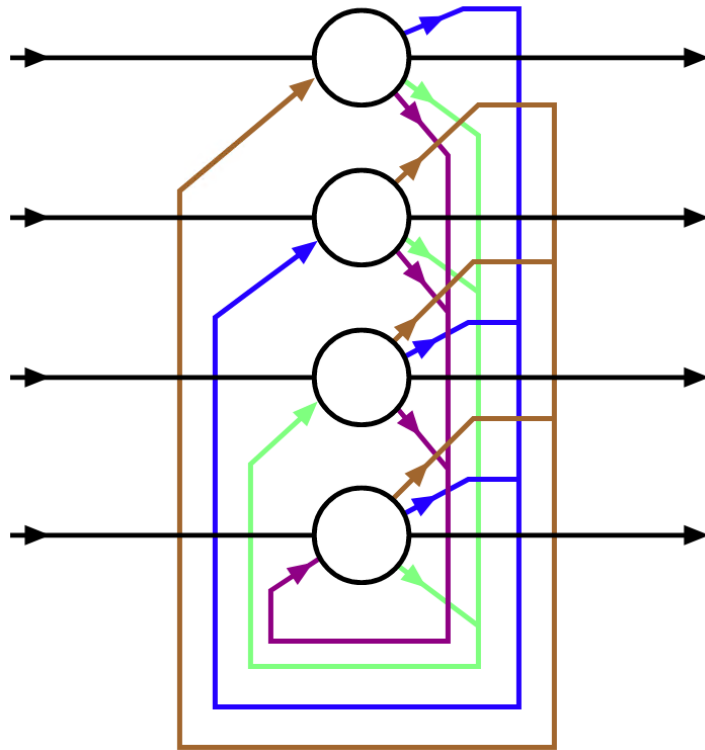


HMM



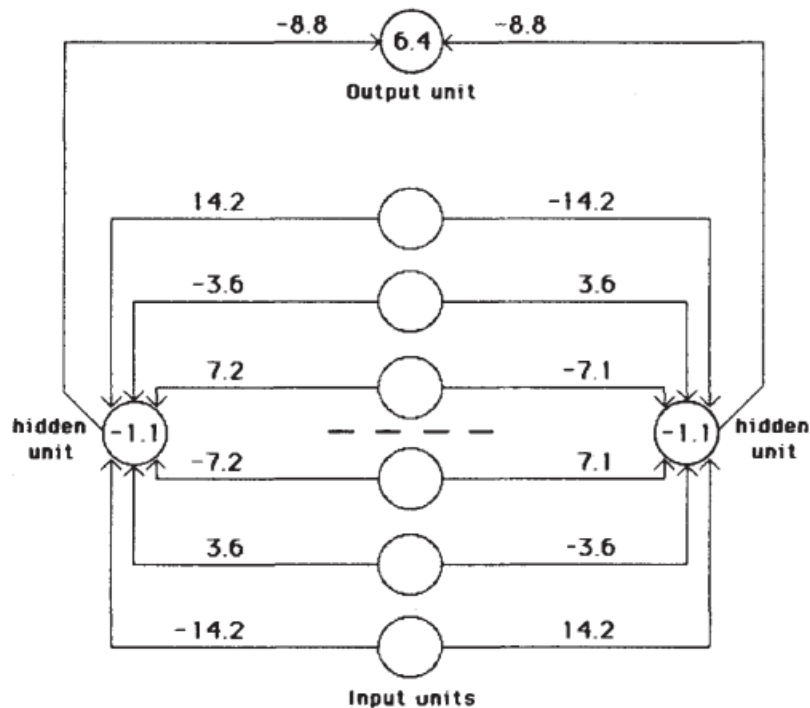
MEMM

Hopfield Network



From Wiki: A Hopfield network (or Ising model of a neural network or Ising–Lenz–Little model) is a form of recurrent artificial neural network popularized by John Hopfield in 1982, but described earlier by Little in 1974 based on Ernst Ising's work with Wilhelm Lenz.

Recurrent Neural Networks



Learning representations by back-propagating errors

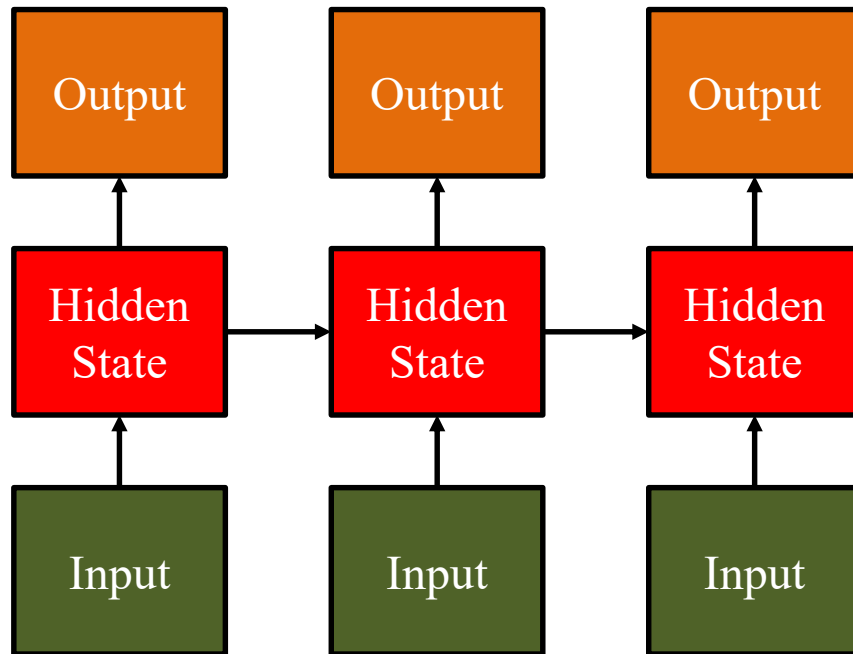
David E. Rumelhart*, Geoffrey E. Hinton†
& Ronald J. Williams*

* Institute for Cognitive Science, C-015, University of California, San Diego, La Jolla, California 92093, USA

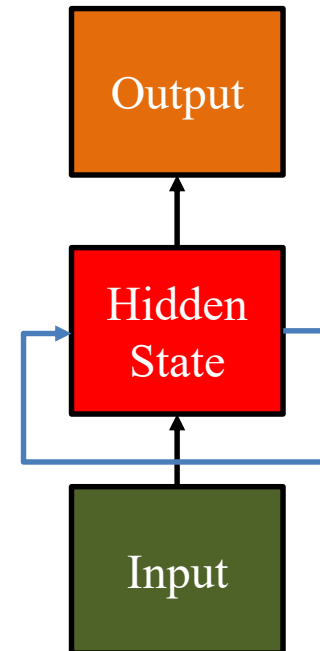
† Department of Computer Science, Carnegie-Mellon University, Pittsburgh, Philadelphia 15213, USA

Proposed by David Rumelhart et al.
in *Nature* 1986, 323, 533-536.

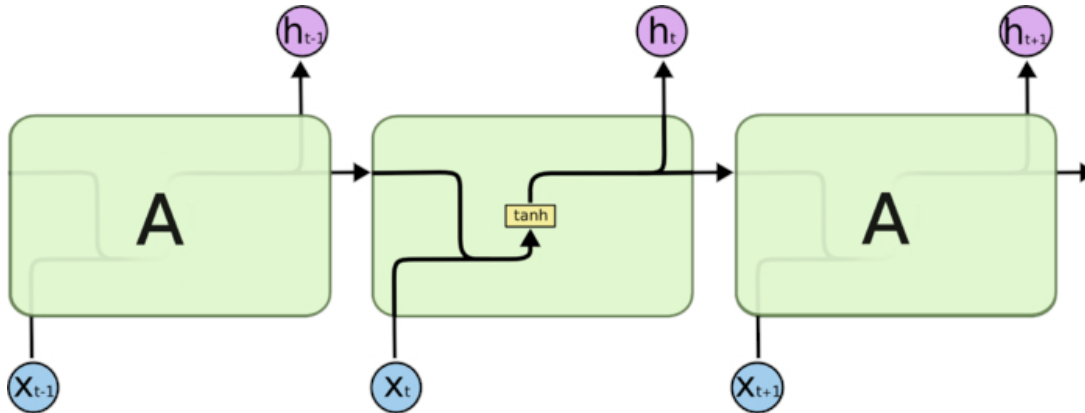
Recurrent Neural Networks



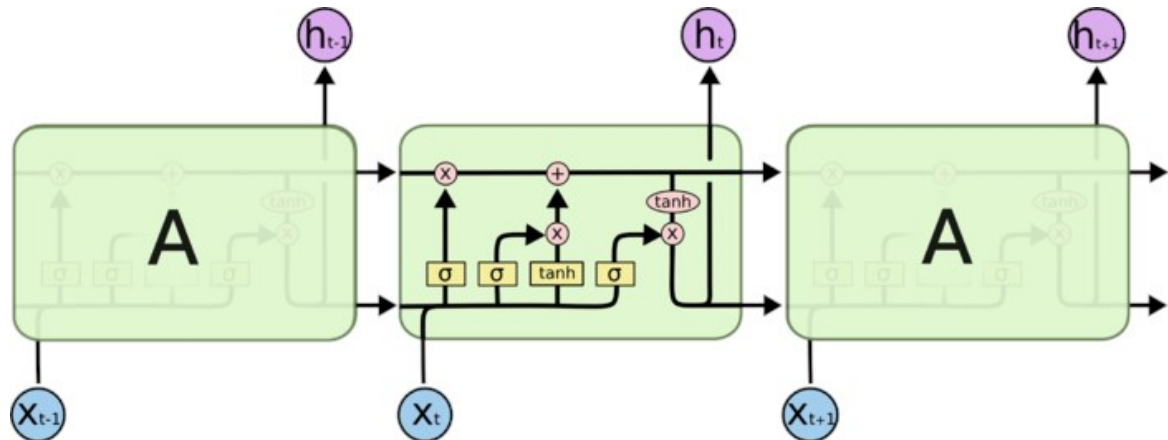
$$y^t = f(h^t)$$
$$h^t = g(x^t, h^{t-1})$$



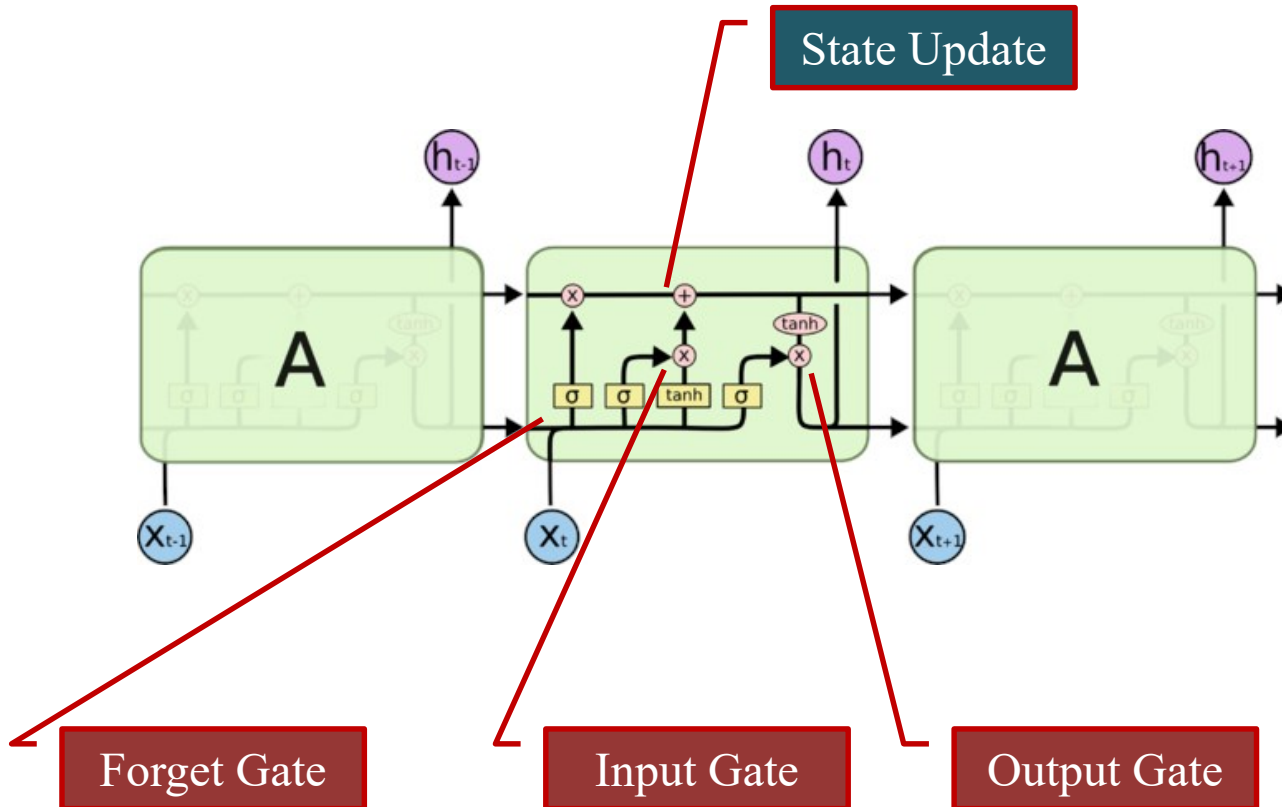
From RNN to LSTM



Disadvantages: strong Markov assumption



From RNN to LSTM



Supplementary Materials

- [1] Lawrence R Rabiner. **A tutorial on Hidden Markov Models and selected applications in speech recognition.** *Proceedings of the IEEE* 1989, 77(2):257-286.
- [2] Zoubin Ghahramani. **An introduction to Hidden Markov Models and Bayesian Networks.** *International Journal of Pattern Recognition and Artificial Intelligence* 2001, 15(1):9-42.
- [3] Lipton et al. **A critical review of recurrent neural networks for sequence learning.** arXiv:1506.00019
<https://arxiv.org/abs/1506.00019>

A Few Comments

- Dynamic or sequential models
 - Markov processes for time invariant systems
 - In-time models & between-time transitions
- Key for hidden models
 - **The design of hidden variables**
 - Significantly reduce the complexity of observed data (*introduce independent assumptions*)

The End of Chapter 5

HMM is a generative model

HMM is powerful for modeling
“hidden mechanisms”

近期重要时间节点提示

- 助教加强讨论、指导
- 第一次小测验：10月25日
- 课程论文自选题：11月7日
- “表示”部分集中答疑讨论
– 10月19日（周二）晚7点，地点待定