Graphical Models and Hidden Markov Models

STAT5241 Section 2

Statistical Machine Learning

Xiaofei Shi

Generative models are powerful

- Conditional generative model P(zebra images| horse images)

- Style Transfer

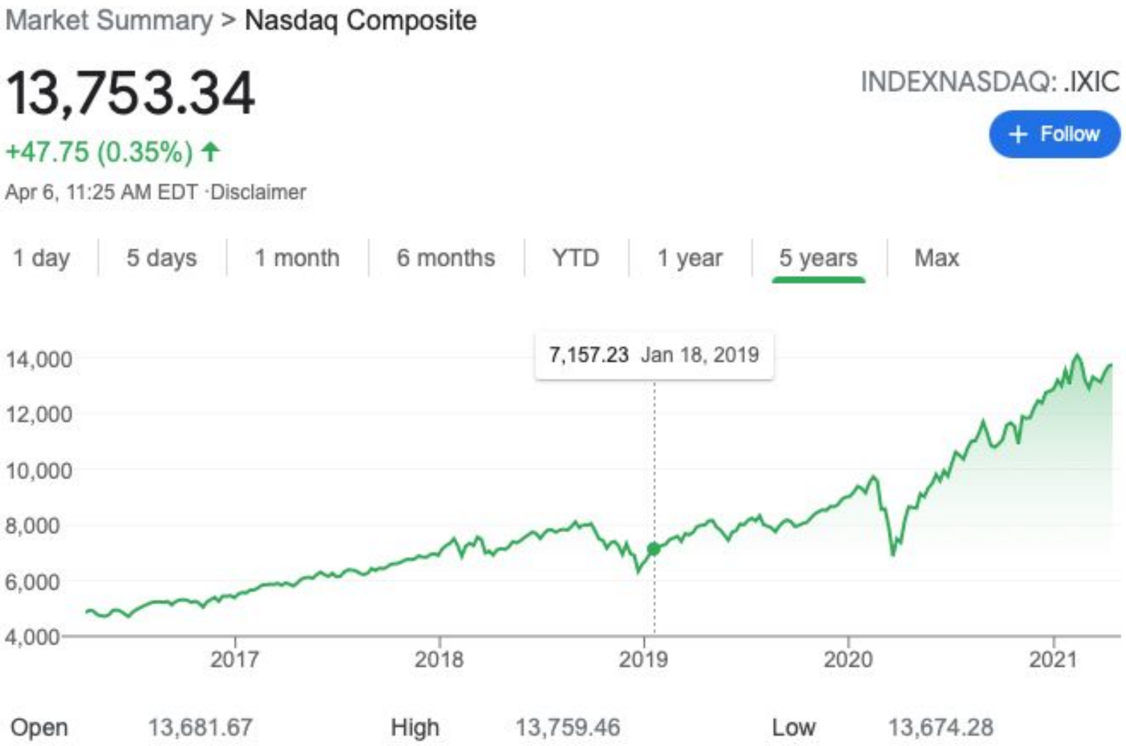
Input Image Monet Van Gogh

Zhou el al., Cycle GAN 2017

Generative models are powerful

Example of the Progression in the Capabilities of GANs From 2014 to 2017. Taken from The Malicious Use of Artificial Intelligence: Forecasting, Prevention, and Mitigation, 2018.

Generative models are powerful



COLUMBIA UNIVERSITY

Generative Models

- An important “unsupervised learning” problem is learning distributions over input variables, also known as “generative models”

  - given such distributions, can do “probabilistic reasoning”
  - does not require “labels”; such unlabeled data is plentiful; big open problem in ML to get “good” generative models!

- Applications of generative models and probabilistic reasoning, especially for time series analysis:

  - Computer vision and graphics
  - Natural language processing
  - Information retrieval
  - Robotic control
  - Computational biology
  - Medical diagnosis
  - Finance and economics
  - …

However, generative models are...

- **very specific** (e.g. multivariate Gaussian distribution, or mixture of multivariate Gaussians) so that they might not be applicable to your given dataset ("data does not look Gaussian!")

- **have a very large number of parameters** (e.g. kernel density estimation)

Graphical Models

- Graphical models thread this needle: flexible, compact, and also interpretable (uses graphs, that are intuitive even to lay users)

- A marriage of probability theory (distributions among random variables) and graph theory (uses graphs to represent distributions)

Graphical Models

- Directed Graphical Models

- Undirected Graphical Models

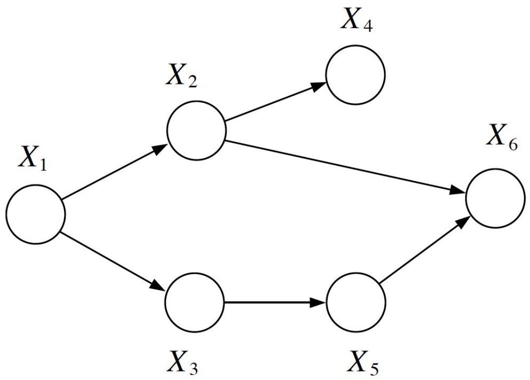
Directed Graphical Models

- Directed Graph is a pair $G = (V, E)$, where V is a set of nodes, E is a set of directed (also called oriented) edges. We will assume G is acyclic.

- Each node $i \in V$ is associated with a random variable X_i.

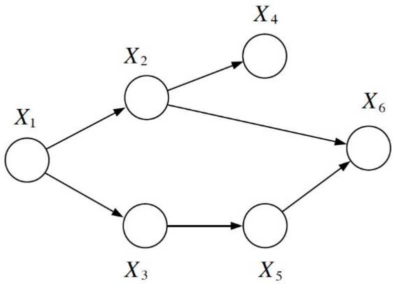
- Letting $V = \{1, 2, \ldots, n\}$, the set of random variables is $\{X_1, X_2, \ldots, X_n\}$.

- We will use node i and the associated random variable X_i interchangeably (though graph nodes and random variables are different formal objects!)



COLUMBIA UNIVERSITY

Directed Graphical Models



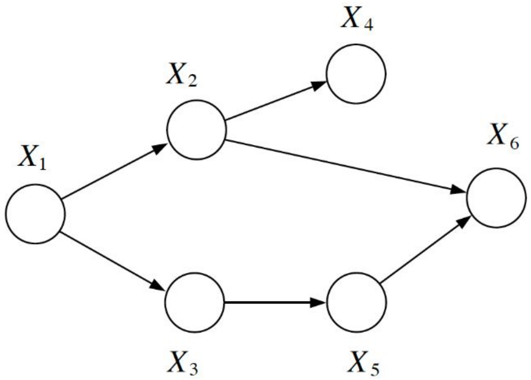
- Each node $i \in V$ has a set of parents π_i
  - $\pi_6 = \{X_2, X_5\}$.
- Let $V = \{1, 2, \ldots, n\}$. Given a set of functions $\{f_i(x_i, x_{\pi_i}) : i \in V\}$, we define a joint probability distribution:

$$p(x_1, \ldots, x_n) := \prod_{i=1}^{n} f_i(x_i, x_{\pi_i}).$$

Only assumptions:
* Non-negative
* Sum to one as a function of x_i

- Is it a proper distribution?
  - Is it non-negative?
  - Does it sum to one? (Yes, provided each function $f_i(x_i, x_{\pi_i})$ sums to one as a function of x_i.

Directed Graphical Models



$$p(x_1, x_2, x_3, x_4, x_5, x_6) = p(x_1)p(x_2 \mid x_1)p(x_3 \mid x_1)p(x_4 \mid x_2)p(x_5 \mid x_3)p(x_6 \mid x_2, x_5)$$

Columbia University

Chain Rule

$$p(x_1, x_2, x_3, x_4, x_5, x_6)$$
$$= p(x_1)p(x_2 \,|\, x_1)p(x_3 \,|\, x_1, x_2)p(x_4 \,|\, x_1, x_2, x_3)p(x_5 \,|\, x_1, x_2, x_3, x_4)p(x_6 \,|\, x_1, x_2, x_3, x_4, x_5)$$

In General:

$$p(x_1, x_2, \ldots, x_n) = \prod_{i=1}^{n} p(x_i \,|\, x_1, \ldots, x_{i-1}).$$

Holds for all distributions

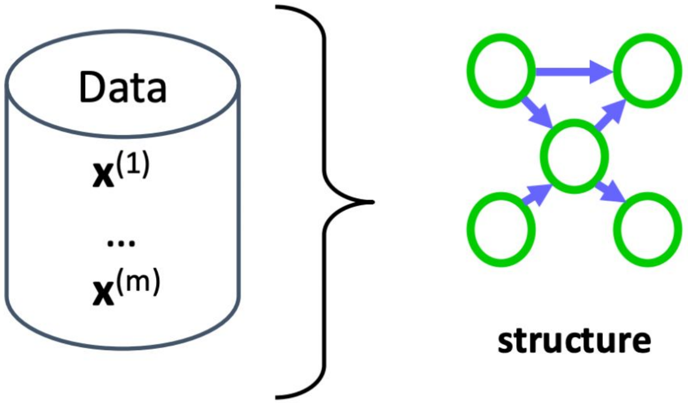
Compare to:

$$p(x_1, x_2, \ldots, x_n) = \prod_{i=1}^{n} p(x_i \,|\, x_{\pi_i}).$$

Holds for directed graphical model distributions

Conditional probability tables (CPT)

Learning Directed Graphical Models



structure

+

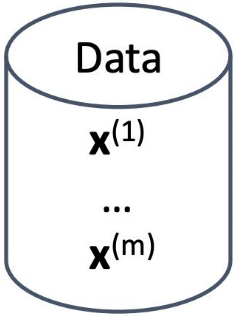
CPTs – $P(X_i \mid \mathbf{Pa}_{X_i})$

parameters

Given set of m independent samples (assignments of random variables),

find the best (most likely?) Bayes Net (graph Structure + CPTs)

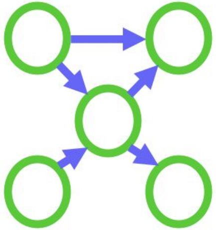
Learning CPTs with Given Structure



For each discrete variable X_k

Compute MLE or MAP estimates for

$$p(x_k|\mathrm{pa}_k)$$

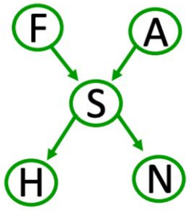


Recall

MLE: $P(X_i = x_i \mid X_j = x_j) = \dfrac{\text{Count}(X_i = x_i, X_j = x_j)}{\text{Count}(X_j = x_j)}$

MAP: Add psuedocounts

Learning CPTs with Given Structure



- Given structure, log likelihood of data

$$
\begin{aligned}
&\log P(\mathcal{D} \mid \theta_{\mathcal{G}}, \mathcal{G}) \\
&= \log \prod_{j=1}^{m} P(f^{(j)}) P(a^{(j)}) P(s^{(j)} | f^{(j)}, a^{(j)}) P(h^{(j)} | s^{(j)}) P(n^{(j)} | s^{(j)}) \\
&= \sum_{j=1}^{m} [\log P(f^{(j)}) + \log P(a^{(j)}) + \log P(s^{(j)} | f^{(j)}, a^{(j)}) + \log P(h^{(j)} | s^{(j)}) + \log P(n^{(j)} | s^{(j)})] \\
&= \underbrace{\sum_{j=1}^{m} \log P(f^{(j)})}_{\theta_F} + \underbrace{\sum_{j=1}^{m} \log P(a^{(j)})}_{\theta_A} + \underbrace{\sum_{j=1}^{m} \log P(s^{(j)} | f^{(j)}, a^{(j)})}_{\theta_{F,A}} + \\
&\quad \underbrace{\sum_{j=1}^{m} \log P(h^{(j)} | s^{(j)})}_{\theta_{H|S}} + \underbrace{\sum_{j=1}^{m} \log P(n^{(j)} | s^{(j)})}_{\theta_{N|S}}]
\end{aligned}
$$

Depends only on

Can compute MLEs of each parameter independently!

Information Theoretic Interpretation

$$\log P(\mathcal{D} \mid \theta_{\mathcal{G}}, \mathcal{G}) = \sum_{j=1}^{m} \sum_{i=1}^{n} \log P\left(X_i = x_i^{(j)} \mid \mathbf{Pa}_{X_i} = \mathbf{x}_{\mathbf{Pa}_{X_i}}^{(j)}\right)$$

$$= \sum_{i=1}^{n} \sum_{x_i} \sum_{\mathbf{x}_{\mathbf{Pa}_{X_i}}} \text{count}(X_i = x_i, \mathbf{Pa}_{X_i} = \mathbf{x}_{\mathbf{Pa}_{X_i}}) \log P\left(X_i = x_i \mid \mathbf{Pa}_{X_i} = \mathbf{x}_{\mathbf{Pa}_{X_i}}\right)$$

Plugging in MLE estimates: ML score

$$\log \widehat{P}(\mathcal{D} \mid \widehat{\theta}_{\mathcal{G}}, \mathcal{G}) = \sum_{j=1}^{m} \sum_{i=1}^{n} \log \widehat{P}\left(x_i^{(j)} \mid \mathbf{x}_{\mathbf{Pa}_{X_i}}^{(j)}\right)$$

$$= m \sum_{i=1}^{n} \sum_{x_i} \sum_{\mathbf{x}_{\mathbf{Pa}_{X_i}}} \widehat{P}(x_i, \mathbf{x}_{\mathbf{Pa}_{X_i}}) \log \widehat{P}\left(x_i \mid \mathbf{x}_{\mathbf{Pa}_{X_i}}\right)$$

Reminds of entropy

Columbia University

Information Theoretic Interpretation

$$\log \hat{P}(\mathcal{D} \mid \hat{\theta}_{\mathcal{G}}, \mathcal{G}) = m \sum_{i=1}^{n} \sum_{x_i} \sum_{\mathbf{x}_{\mathbf{Pa}_{X_i}}} \hat{P}(x_i, \mathbf{x}_{\mathbf{Pa}_{X_i}}) \log \hat{P}\left(x_i \mid \mathbf{x}_{\mathbf{Pa}_{X_i}}\right)$$

$$= -m \sum_{i=1}^{n} \hat{H}(X_i \mid \mathbf{Pa}_{X_i})$$

$$= m \sum_{i=1}^{n} [\hat{I}(X_i, \mathbf{Pa}_{X_i}) - \underbrace{\hat{H}(X_i)}_{\text{Doesn't depend on graph structure } \mathcal{G}}]$$

ML score for graph structure $\mathcal{G}$

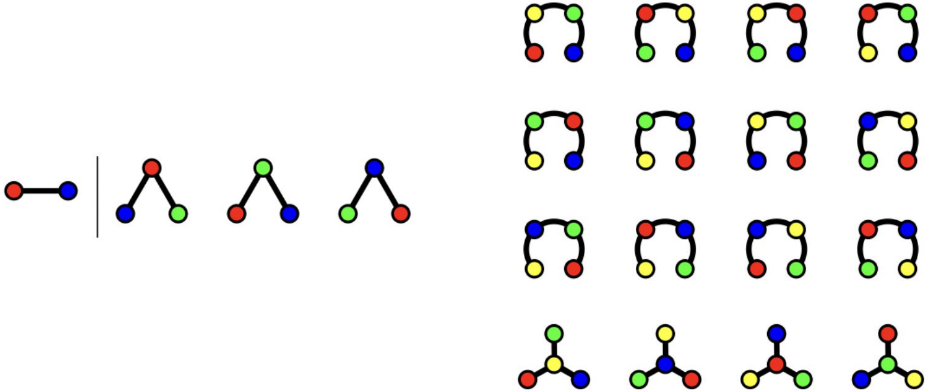
$$\arg\max_{\mathcal{G}} \log \hat{P}(\mathcal{D} \mid \hat{\theta}_{\mathcal{G}}, \mathcal{G}) = \arg\max_{\mathcal{G}} \sum_{i=1}^{n} \hat{I}(X_i, \mathbf{Pa}_{X_i})$$

COLUMBIA UNIVERSITY

What if graphical structure is not given?

How many trees are there?

- Trees – every node has at most one parent
- n^{n-2} possible trees (Cayley's Theorem)



Nonetheless – Efficient optimal algorithm finds best tree!

COLUMBIA UNIVERSITY

Focusing on Tree Structure

$$\arg\max_{\mathcal{G}} \log \widehat{P}(\mathcal{D} \mid \widehat{\theta}_{\mathcal{G}}, \mathcal{G}) = \arg\max_{\mathcal{G}} \sum_{i=1}^{n} \widehat{I}(X_i, \mathbf{Pa}_{X_i})$$

Equivalent Trees (same score): I(A,B) + I(B,C)

$A \rightarrow B \rightarrow C$ $\qquad$ $A \leftarrow B \leftarrow C$ $\qquad$ $A \leftarrow B \rightarrow C$

Score provides indication of structure:

$A \rightarrow B \rightarrow C$

I(A,B) + I(B,C)

$B \leftarrow A \rightarrow C$

I(A,B) + I(A,C)

Chow-Liu Algorithm

- For each pair of variables X_i, X_j
  - Compute empirical distribution: $\widehat{P}(x_i, x_j) = \frac{\text{Count}(x_i, x_j)}{m}$
  - Compute mutual information:

$$\widehat{I}(X_i, X_j) = \sum_{x_i, x_j} \widehat{P}(x_i, x_j) \log \frac{\widehat{P}(x_i, x_j)}{\widehat{P}(x_i)\widehat{P}(x_j)}$$

- Define a graph
  - Nodes $X_1, \ldots, X_n$
  - Edge (i,j) gets weight $\widehat{I}(X_i, X_j)$

- Optimal tree BN
  - Compute maximum weight spanning tree (e.g. Prim's, Kruskal's algorithm O(nlog n))
  - Directions in BN: pick any node as root, breadth-first-search defines directions

Columbia University

Learning Bayesian Networks for General Graphs

Theorem: The problem of learning a BN structure with at most *d* parents is NP-hard for any (fixed) *d*>1 (Note: tree d=1)

- Mostly heuristic (exploit score decomposition).
- Chow-Liu: provides best tree approximation to any distribution.
- Graph that maximizes ML score: complete graph!

  - Adding a parent always increase ML score!

  - The more edges, the fewer independence assumptions, the higher the likelihood of the data, but will overfit…
- Start with Chow-Liu tree.

  - Add, delete, invert edges.

  - Evaluate ML score + penalty

Learning Graphical Models

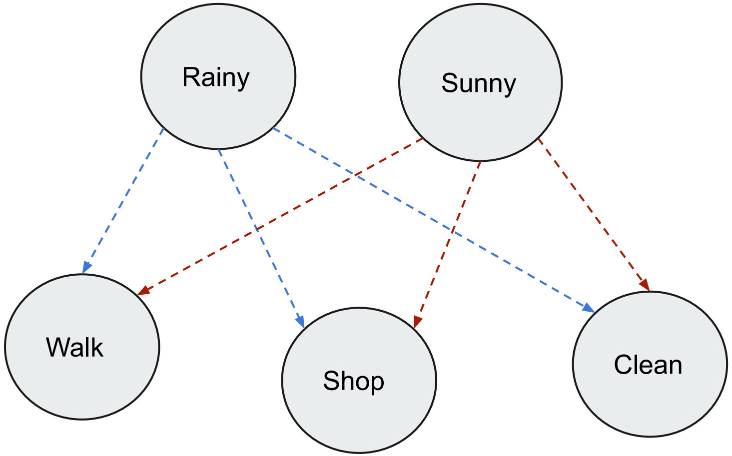
- Two Step Procedures:
  - 1. **Model Selection**; estimate graph structure
  - 2. **Parameter Inference** given graph structure
- **Score Based Approaches**: **search** over space of graphs, with a score for graph based on parameter inference
- **Constraint-based Approaches**: estimate individual edges by **hypothesis tests** for conditional independences
- Caveats: (a) difficult to provide guarantees for estimators; (b) estimators are NP-Hard

COLUMBIA UNIVERSITY

Recall Bayesian networks

Consists of 2 parts:

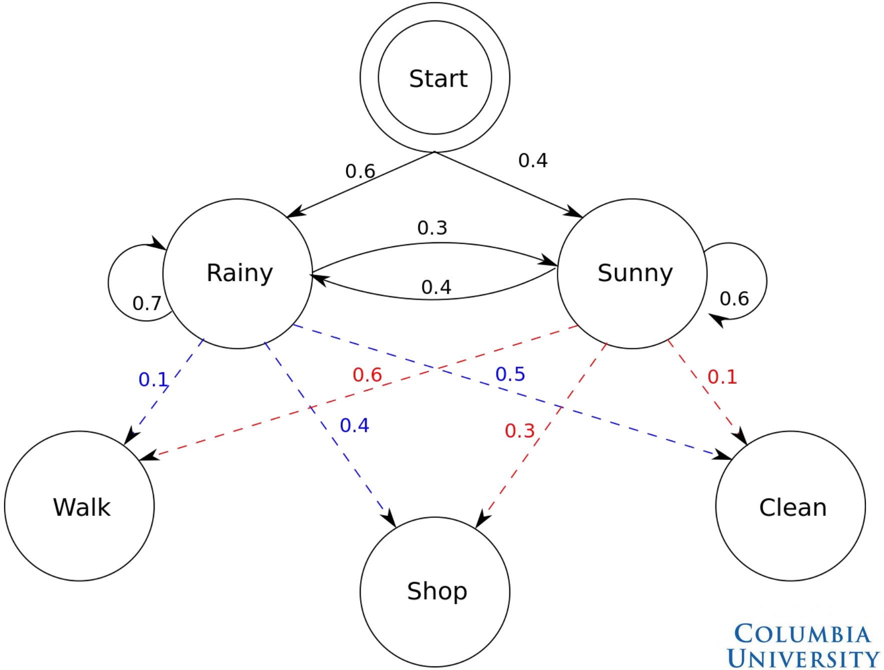
- Probability belongs to each class
- Class conditional probability



There exists a direction!

Limitation of Bayesian networks

- Doing full Bayesian networks are computationally expensive;
- Performance is suboptimal on high-dimensional data;
- Bayesian networks are directed acyclic graphical models, where the directed edges are used to capture causal relationship between random variables, but we need undirected graphical model with potential loops in between nodes to model correlations.
- Bayesian networks cannot be used to model temporal/sequence data.



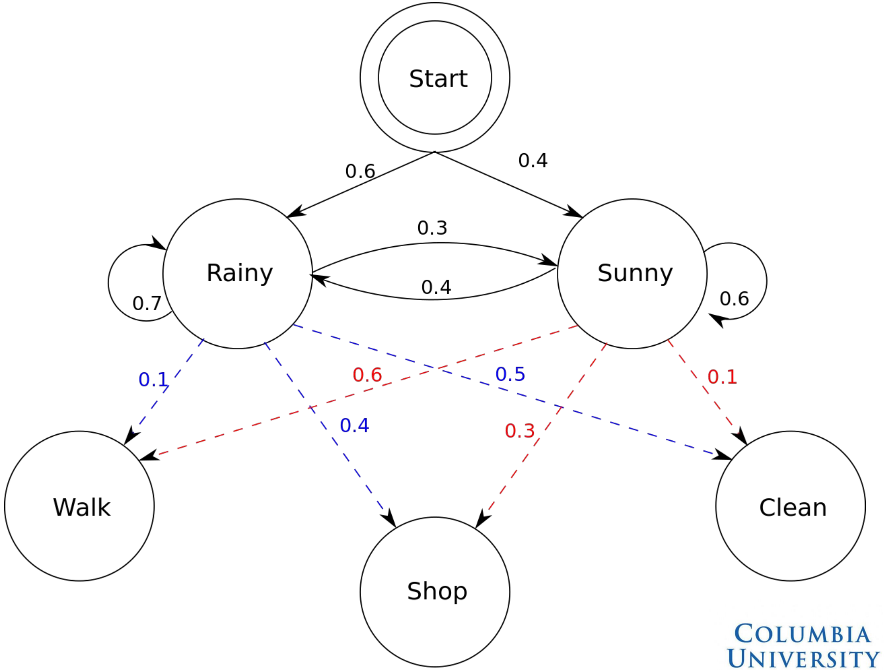
Hidden Markov models

Model a set of observations using a set of hidden states:

(observations, hidden states) such as:

- (pixel inputs, features)
- (range/visual sensor, location)
- (sound/visual signal, language/situation/words)
- (facial expression, results from the driving test)

hidden state generates observation;
hidden state transitions to other hidden states

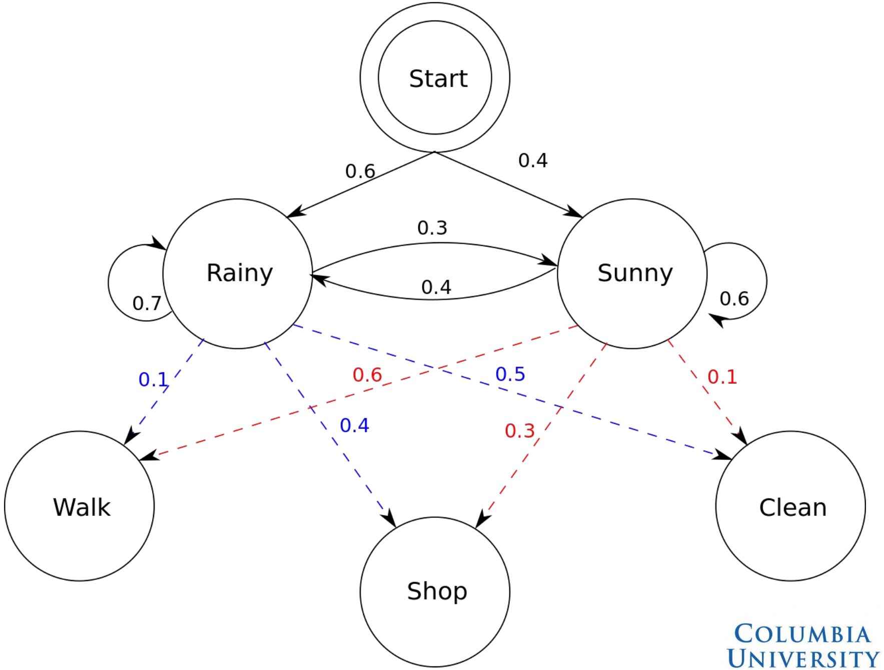


COLUMBIA UNIVERSITY

Example: Daisy's diary

Daisy kept a diary on what she was doing but she forgot to keep track of the weather. Let's treat the weather conditions as hidden states:

- With 60%/40%, today is rainy/sunny
- If today is rainy, it would either remain rainy (70%), or could be sunny tomorrow (30%).



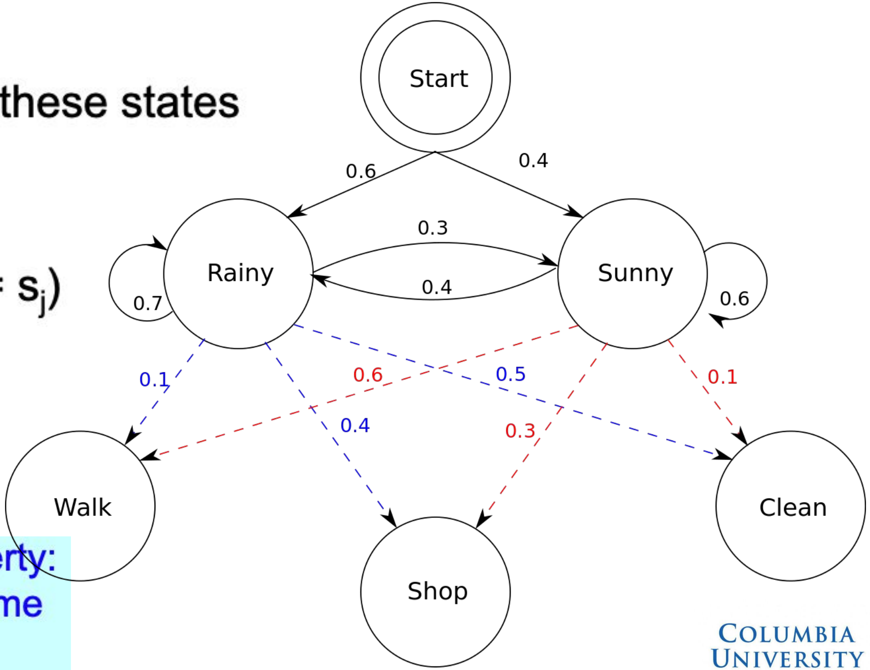
COLUMBIA UNIVERSITY

Definition of 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, $p(o_t = \sigma \mid s_i)$



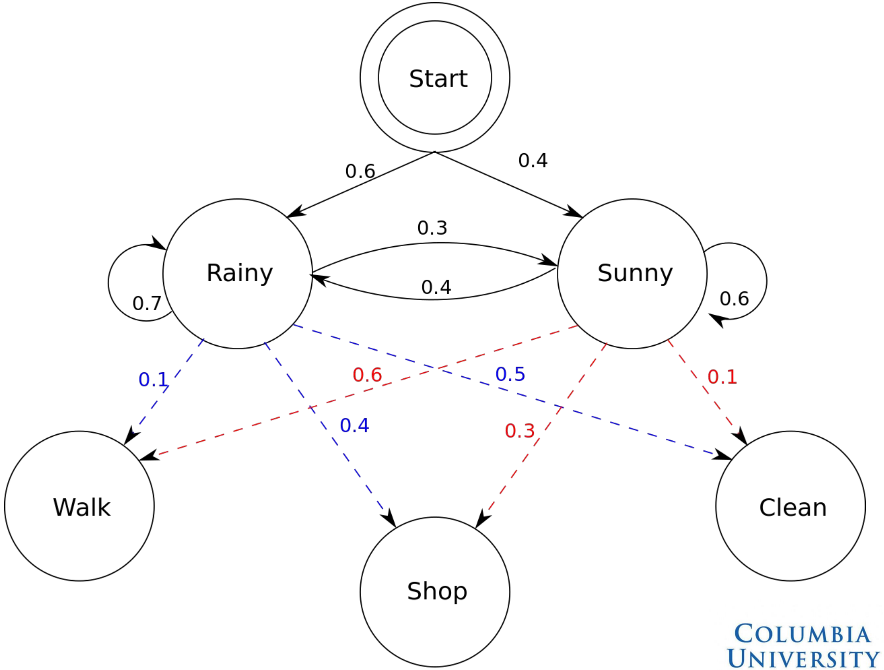
An important aspect of this definition is the Markov property: q_{t+1} is conditionally independent of q_{t-1} (and any earlier time points) given q_t

More formally $P(q_{t+1} = s_i \mid q_t = s_j) = P(q_{t+1} = s_i \mid q_t = s_j, q_{t-1} = s_j)$

Understanding HMM

Several questions people can ask:

- What is the current weather?
- What is the probability for shopping tomorrow?
- What is the probability for shopping next week?



Understanding HMM

When no observation:

- What is the current weather?

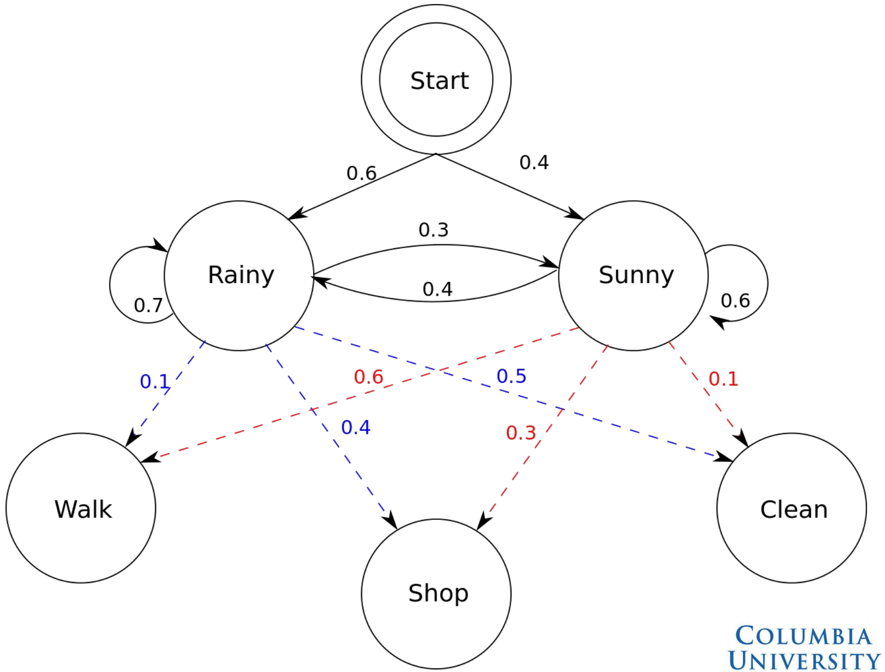
  P[rainy] = 0.6 P[sunny]=0.4

- What is the weather tomorrow?

  P[rainy] = 0.6 x 0.7 + 0.4 x 0.4

  P[sunny] = 0.6 x 0.3 + 0.4 x 0.6

- What is the weather on the t-th day?



COLUMBIA UNIVERSITY

Understanding HMM

When no observation:

- What is the current weather?

  P[rainy] = 0.6 P[sunny]=0.4

- What is the weather tomorrow?

  P[rainy] = 0.6 x 0.7 + 0.4 x 0.4

  P[sunny] = 0.6 x 0.3 + 0.4 x 0.6

- What is the weather on the t-th day?

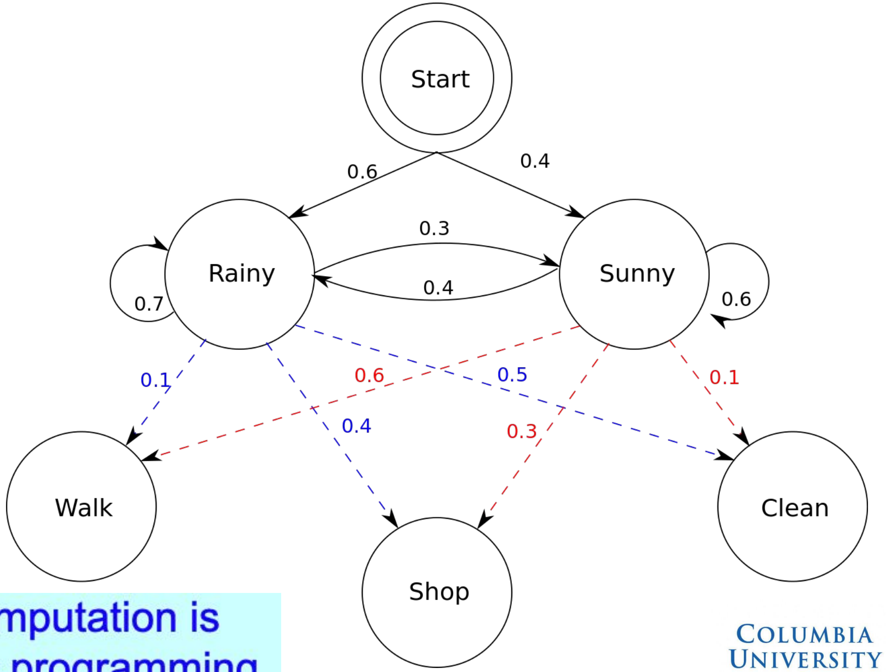
Lets define $p_t(i)$ = probability state i at time t = $p(q_t = s_i)$

We can determine $p_t(i)$ by induction

1. $p_1(i) = \Pi_i$
2. $p_t(i) = \Sigma_j\, p(q_t = s_i \mid q_{t-1} = s_j)p_{t-1}(j)$

This type of computation is called dynamic programming

Complexity: $O(n^2*t)$



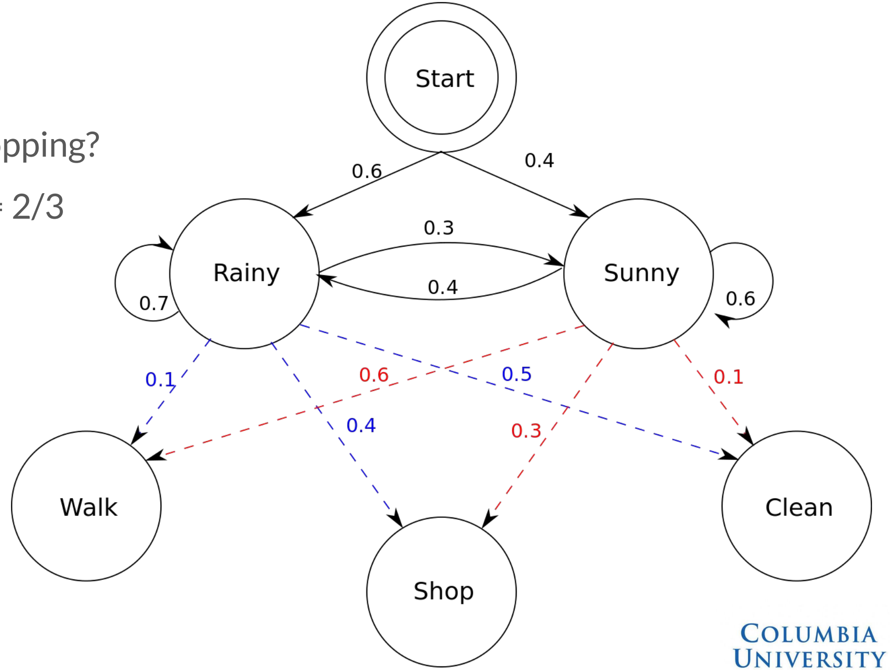
COLUMBIA UNIVERSITY

Understanding HMM

When with observations:

- What is the current weather given that Daisy is shopping?

  P[rainy|shopping] = 0.6 x 0.4/(0.6 x 0.4 +0.4 X 0.3) = 2/3



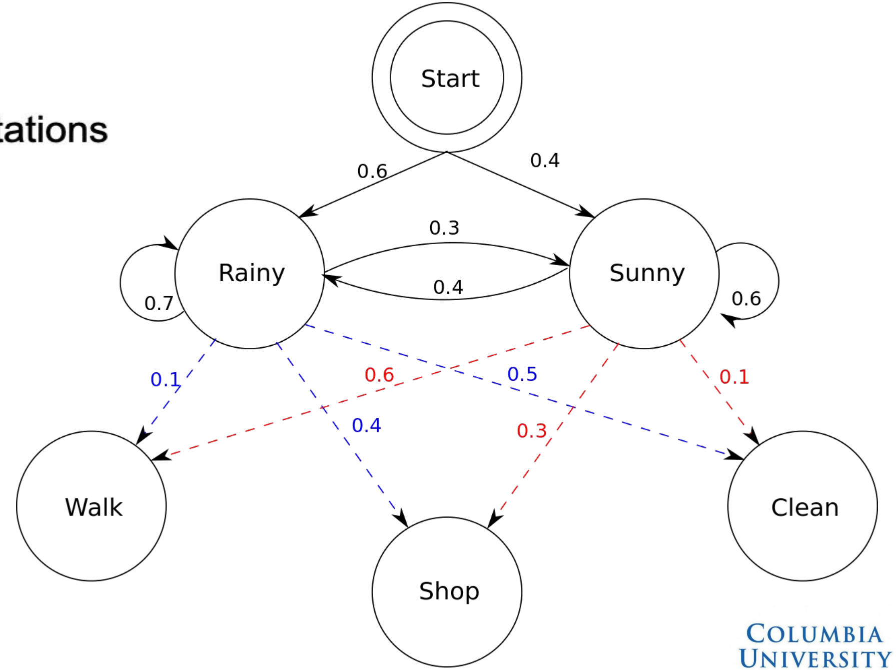
COLUMBIA UNIVERSITY

Inference in HMM

- We want to compute $P(q_t = A \mid O_1 \dots O_t)$
- For ease of writing we will use the following notations (commonly used in the literature)
- $a_{j,i} = P(q_t = s_i \mid q_{t-1} = s_j)$
- $b_i(o_t) = P(o_t \mid s_i)$

Transition probability

Emission probability



Columbia University

Inference in HMM

- We want to compute $P(q_t = A \mid O_1 \ldots O_t)$
- Lets start with a simpler question. Given a sequence of states Q, what is $P(Q \mid O_1 \ldots O_t) = P(Q \mid O)$?

  - It is pretty simple to move from $P(Q|O)$ to $P(q_t = A |O)$

  - In some cases $P(Q \mid O)$ is the more important question

    - Speech processing

    - NLP

Inference in HMM

$$P(Q|O) = \frac{P(O|Q)P(Q)}{P(O)}$$

Easy, $P(O \mid Q) = P(o_1 \mid q_1)\, P(o_2 \mid q_2) \ldots P(o_t \mid q_t)$

$P(Q) = P(q_1)\, P(q_2 \mid q_1) \ldots P(q_t \mid q_{t-1})$

But it is comparatively hard to compute P(O),

i.e. the probability of seeing a set of observations.

Inference in HMM

- What is the probability of seeing a set of observations:
  - An important question in it own rights, for example classification using two HMMs
- Define $\alpha_t(i) = P(o_1, o_2 \ldots, o_t \wedge q_t = s_i)$
- $\alpha_t(i)$ is the probability that we:
  1. Observe $o_1, o_2 \ldots, o_t$
  2. End up at state i

Inference in HMM

- We want to compute P(Q | O)
- For this, we only need to compute P(O)
- We know how to compute $\alpha_t(i)$

From now its easy

$\alpha_t(i) = P(o_1, o_2 \ldots, o_t \wedge q_t = s_i)$

so

$P(O) = P(o_1, o_2 \ldots, o_t) = \Sigma_i P(o_1, o_2 \ldots, o_t \wedge q_t = s_i) = \Sigma_i \alpha_t(i)$

note that

$$p(q_t = s_i \mid o_1, o_2 \ldots, o_t) = \frac{\alpha_t(i)}{\sum_j \alpha_t(j)}$$

$P(A \mid B) = P(A \wedge B) / P(B)$

COLUMBIA UNIVERSITY

Inference in HMM

Q represents a set of state $Q = \{s_1, s_2 \ldots s_t\}$

O represents a set of emitted values $O = \{o_1, o_2 \ldots o_t\}$

- Computing $P(Q)$ and $P(q_t = s_i)$
  - If we cannot look at observations
- Computing $P(Q \mid O)$ and $P(q_t = s_i \mid O)$
  - When we have observation and care about the last state only
- Computing $\text{argmax}_Q P(Q \mid O)$
  - When we care about the entire path

COLUMBIA UNIVERSITY

Computational complexity

- How long does it take to compute P(Q | O)?
- P(Q): O(t)
- P(O|Q): O(t)
- P(O): $O(n^2t)$

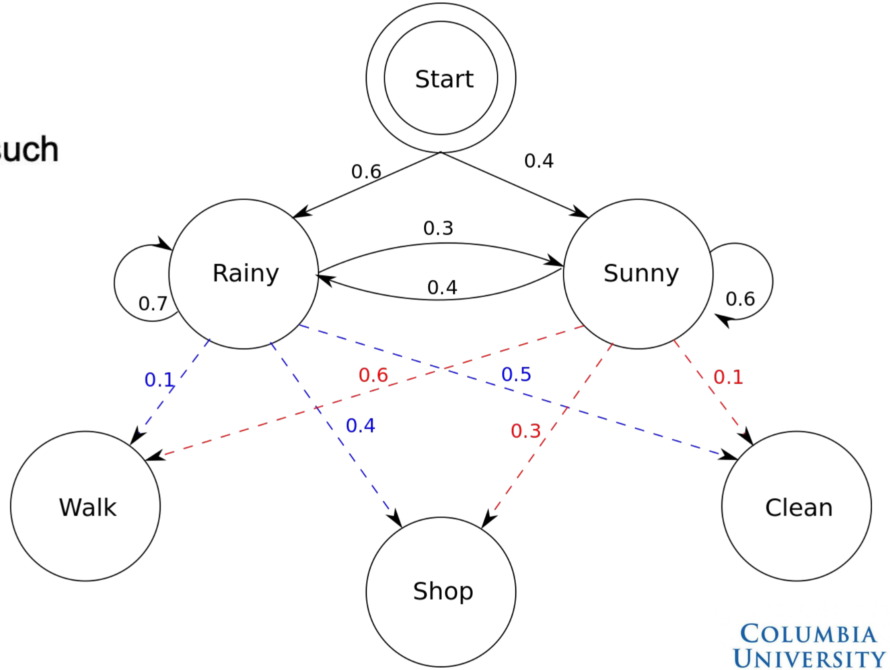
Inference in HMM

- We are almost done …
- One final question remains

  How do we find the most probable path, that is Q* such that

$$P(Q^* \mid O) = \text{argmax}_Q P(Q|O)?$$

- This is an important path
  - The words in speech processing
  - The set of genes in the genome
  - etc.

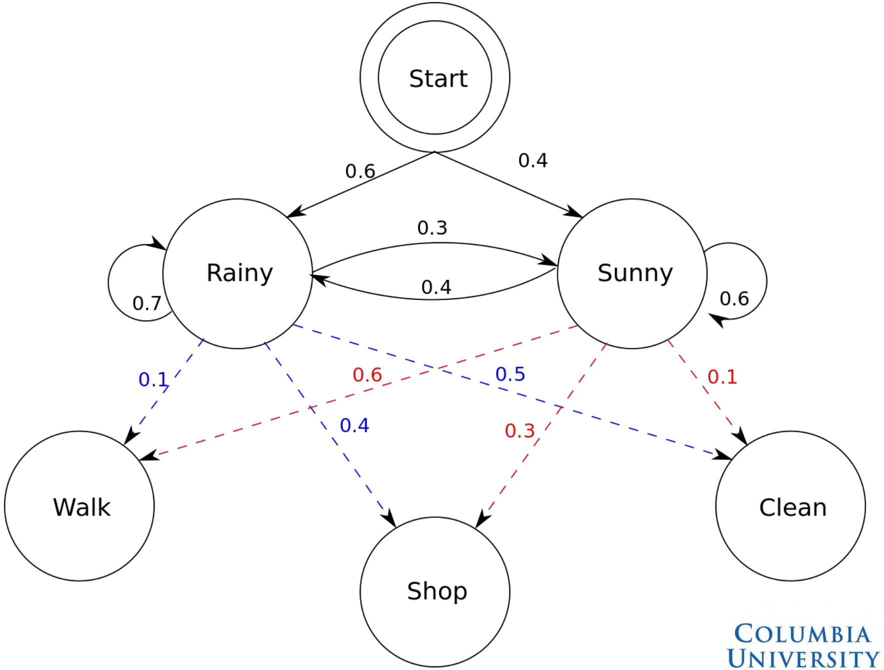


Inference in HMM

- What's the most likely sequence for you to see Daisy goes out for walking for 7 days in a week?

$$\arg\max_Q P(Q \mid O) = \arg\max_Q \frac{P(O \mid Q)P(Q)}{P(O)}$$

$$= \arg\max_Q P(O \mid Q)P(Q)$$



Columbia University

Inference in HMM

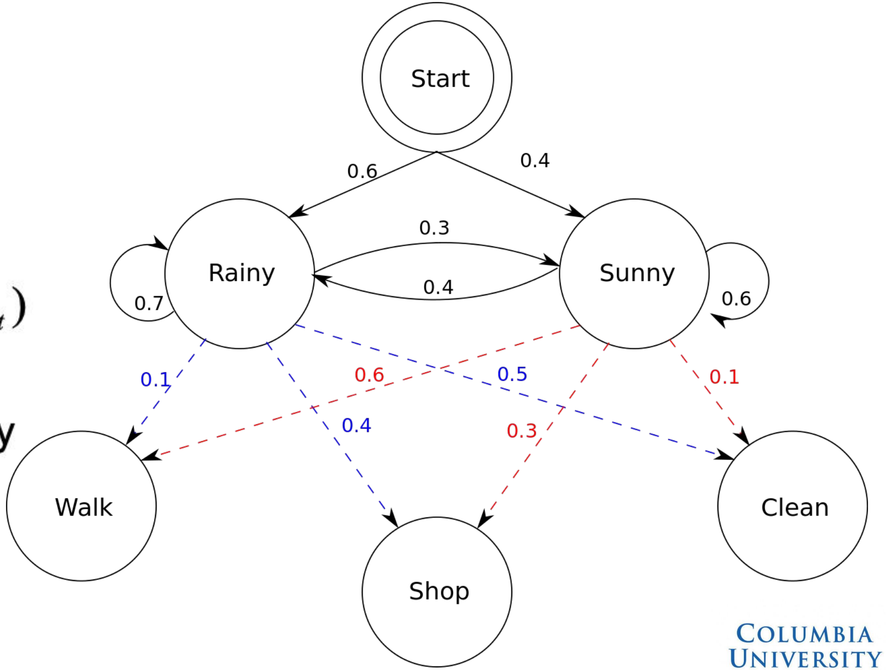
$$\arg\max_Q P(Q \mid O) = \arg\max_Q \frac{P(O \mid Q)P(Q)}{P(O)}$$

We will use the following definition:

$$\delta_t(i) = \max_{q_1 \ldots q_{t-1}} p(q_1 \ldots q_{t-1} \wedge q_t = s_i \wedge O_1 \ldots O_t)$$

In other words we are interested in the most likely path from 1 to t that:

1. Ends in S_i
2. Produces outputs $O_1 \ldots O_t$



Inference in HMM

$$\arg\max_Q P(Q \mid O) = \arg\max_Q \frac{P(O \mid Q)P(Q)}{P(O)}$$

We will use the following definition:

$$\delta_t(i) = \max_{q_1 \ldots q_{t-1}} p(q_1 \ldots q_{t-1} \wedge q_t = s_i \wedge O_1 \ldots O_t)$$

In other words we are interested in the most likely path from 1 to t that:

1. Ends in S_i
2. Produces outputs $O_1 \ldots O_t$

Q: Given $\delta_t(i)$, how can we compute $\delta_{t+1}(i)$?

A: To get from $\delta_t(i)$ to $\delta_{t+1}(i)$ we need to

1. Add an emission for time t+1 (O_{t+1})
2. Transition to state s_i

$$\begin{aligned}\delta_{t+1}(i) &= \max_{q_1 \ldots q_t} p(q_1 \ldots q_t \wedge q_{t+1} = s_i \wedge O_1 \ldots O_{t+1}) \\ &= \max_j \delta_t(j) p(q_{t+1} = s_i \mid q_t = s_j) p(O_{t+1} \mid q_{t+1} = s_i) \\ &= \max_j \delta_t(j) a_{j,i} b_i(O_{t+1})\end{aligned}$$

Columbia University

Inference in HMM: the Viterbi algorithm

$$\delta_{t+1}(i) = \max_{q_1 \cdots q_t} p(q_1 \ldots q_t \wedge q_{t+1} = s_i \wedge O_1 \ldots O_{t+1})$$
$$= \max_j \delta_t(j) p(q_{t+1} = s_i \mid q_t = s_j) p(O_{t+1} \mid q_{t+1} = s_i)$$
$$= \max_j \delta_t(j) a_{j,i} b_i(O_{t+1})$$

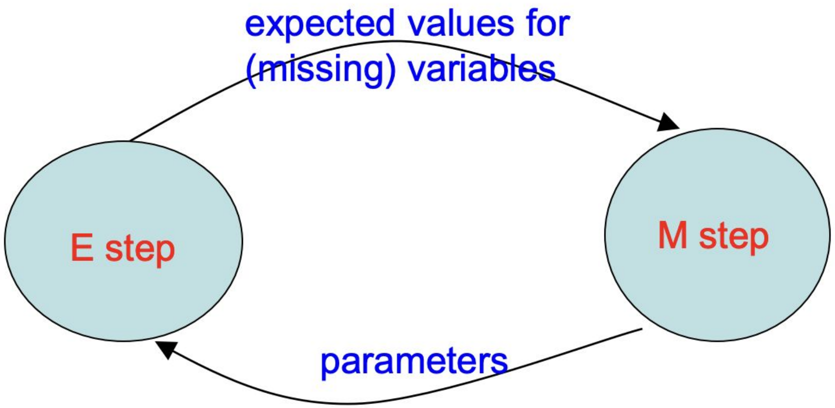
- Once again we use dynamic programming for solving $\delta_t(i)$
- Once we have $\delta_t(i)$, we can solve for our P(Q*|O)

By:

P(Q* | O) = argmax_QP(Q|O) = **Path defined by max_i** $\delta_t(i)$

Learning HMMs: EM algorithm revisit

- 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).



COLUMBIA UNIVERSITY

EM algorithm for HMMs (Baum-Welch)

E step:

- Compute $S_t(i)$ and $S_t(i,j)$ for all *t, i,* and *j*

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

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

M step:

Compute transition probabilities:

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

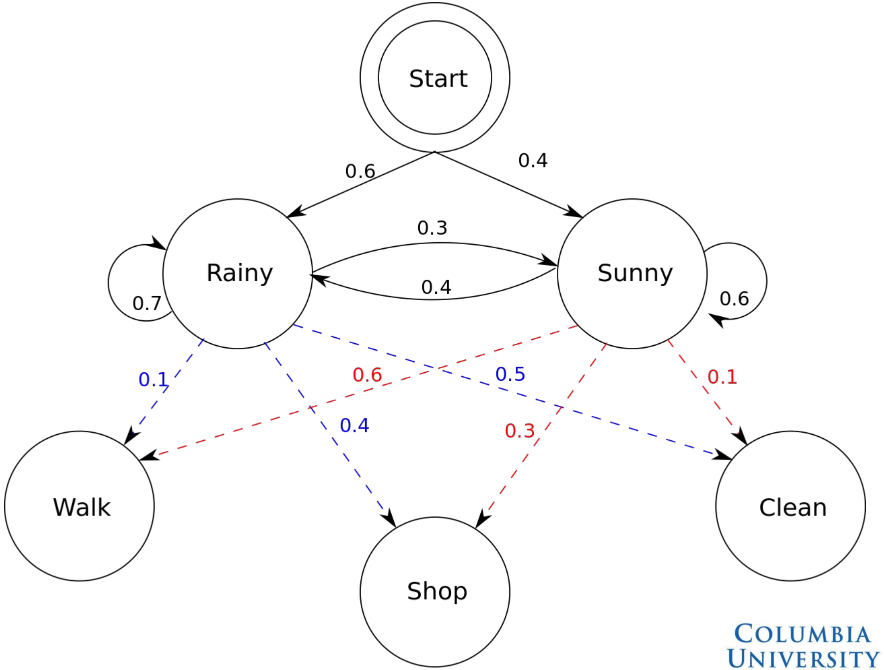
Compute emission probabilities (here we assume a multinomial distribution):

$$b_k(j) = \frac{B_k(j)}{\sum_i B_k(i)} \qquad B_k(j) = \sum_{t \mid o_t = j} S_t(k)$$

COLUMBIA UNIVERSITY

Takeaways

- Why HMMs? Which applications are suitable?
- Inference in HMMs
  - No observations
  - Probability of next state with observations
  - Maximum scoring path (Viterbi)
- Learning HMMs:
  - EM algorithm



References

- Christopher Bishop: Pattern Recognition and Machine Learning, Chapter 5
- Ziv Bar-Joseph, Tom Mitchell, Pradeep Ravikumar and Aarti Singh: CMU 10-701
- Ryan Tibshirani: CMU 10-725
- Ruslan Salakhutdinov: CMU 10-703
- https://machinelearningmastery.com/what-are-generative-adversarial-networks-gans/

Columbia University