VE492 Final Recitation Class

Yunpeng Jiang, Zhengjie Ji

UMJI

{*jyp9961, jizhengjie*}*@sjtu.edu.cn*

July 29, 2021

Table of Contents

Probability - Outline

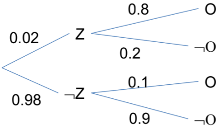
- Random Variables
- Joint and Marginal Distributions
- Conditional Distribution
- Product Rule, Chain Rule, Bayes' Rule
- Inference
- (Conditional) Independence

Probability - Background knowledge

- X, Y independent if and only if $\forall x, y : P(x, y) = P(x)P(y)$
- X, Y are conditionally independent given Z if and only if: $X \perp\!\!\!\perp Y|Z, \quad \forall x, y, z : P(x, y|z) = P(x|z)P(y|z)$
- Conditional Probability: $P(x|y) = P(x, y)/P(y)$
- Product rule: $P(x, y) = P(x|y)P(y)$
- Chain rule: $P(X_1, X_2, ..., X_n) = \prod_{i=1}^{n} P(X_i|X_1, ..., X_{n-1})$
- Sum rule (marginalization): $p(X) = \sum_y P(X, y)$
- Variant of sum rule: $p(X) = \sum_y P(X|y)P(y)$
- Bayes rule: $P(y|x) = p(x|y)p(y)/P(x)$

Quick Example: Bayes Rule

- $P(Z) = 0.02$ (zebra in 2% of images)
- $P(O|Z) = 0.8$ (true positive)
- $P(O|Z) = 0.1$ (false positive)
- We want to calculate: $p(Z|O)$
- Apply Bayes rule!



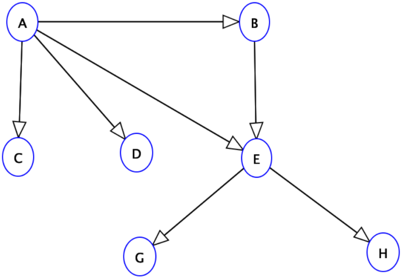
Bayes Nets - Outline

- Representation
- Conditional Independence
  - D-separation
  - Active / Inactive Paths
- Probabilistic Inference
  - Enumeration
  - Variable elimination
  - Probabilistic inference
  - Sampling

Bayes Nets - Sample Questions

- How to get formula of joint distribution from BN graphs?
- How to count the degree of freedom of BN graphs?
- How to run variable elimination? What is the best ordering for VE? What is the largest generated factor? What is the cutset?
- What are the difference of the four sampling methods? What is the time complexity?

Example: Small Bayes Net



- Provide the formula of the joint distribution over all the variables given by the Bayes net.
- Provide the number of degrees of freedom of the BN.
- Run VE to compute to compute $P(A|H = h)$. Provide the list of the sizes (i.e., number of variables) of the factors obtained at the end of each iteration in VE.

Hidden Markov Models - Outline

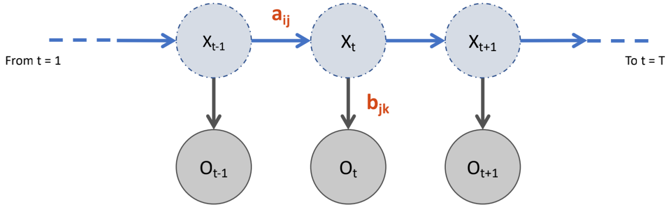
- Markov Models and Hidden Markov Models
- Forward algorithm
- Viterbi algorithm
- Particle filtering
- Dynamic Bayesian Network

Hidden Markov Models - Sample Questions

- How to compute stationary distribution?
- How to do filtering / prediction / smoothing / explanation?
- What is particle filtering?
- How to distinguish Dynamic Bayes Nets and Bayes Nets?

HMM Terminology

Time instants	t *in* {1,2 . . . T}
Hidden States / States / Emitters	X_t
Outputs / Emissions / Observations / Visible States	O_t
All possible states / states set	X_t *in* {1,2 . . . N}
All possible emissions / emissions set	O_t *in* {1,2 . . . K}
Initial state distribution / Initial state probabilities	p_i in q *or* π_i in π
Transition probabilities / State transition probabilities	a_{ij} *in* row-stochastic matrix A
Emission probabilities / Observation probabilities	b_{jk} *in* row-stochastic matrix B

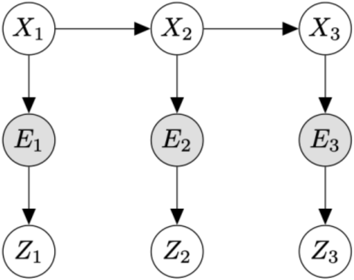


Example - Adapt the Forward Algorithm

Filtering Algorithm

$$P(X_{t+1}|e_{1:t+1}) = \alpha P(e_{t+1}|X_{t+1}) \sum_{x_t} P(x_t|e_{1:t}) P(X_{t+1}|x_t)$$

- Adapt the forward algorithm to this variant of HMM.
- Step 1. Predict step
- Step 2. Update



Hidden Markov Models - Background knowledge

Forward Algorithm (rewrite)

$$\alpha_t(i) = [\sum_{j=1}^{N} \alpha_{t-1}(j) a_{ji}] b_i(O_t)$$

Viterbi Algorithm (rewrite)

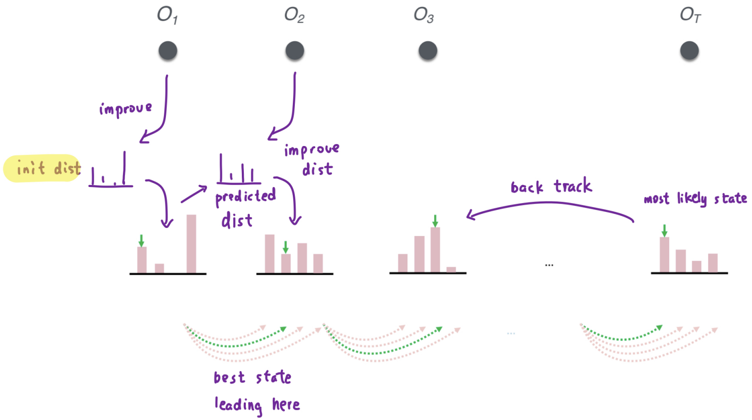
$$\delta_t(i) = max_{j \in \{1,\ldots,N\}} [\delta_{t-1}(j) a_{ji} b_i(O_t)]$$

- a_{ij}: state transition probabilities, b_{jk}: emission probabilities
- Forward Algorithm can be used to predict the current state given all of the current and past evidence.
- Viterbi algorithm can be used to calculate the most likely state sequence, namely $argmax_{X_{1:T}} P(X_{1:T} | O_{1:T}, \lambda)$, where $\lambda = \{A, B, \pi\}$.

Please refer to the lecture slides if this leads to any confusion!

Example: Viterbi Algorithm

- Viterbi: calculate most likely state sequence



Example: Viterbi Algorithm

A: $a_{ji} = P(X_{t+1} = j \mid X_t = i)$

$X_t \mid X_{t+1}$	A	B	H	S
A	0.6	0.1	0.1	0.2
B	0.0	03	0.2	0.5
H	0.8	0.1	0.1	0.1
S	0.2	0.0	0.1	0.7

B: $b_{ik} = P(O_t = k \mid X_t = i)$

$X_t \mid O_t$	p	e	b	l
A	0.6	0.2	0.1	0.1
B	0.1	0.4	0.1	0.4
H	0.0	0.0	0.7	0.3
S	0.0	0.0	0.1	0.9

$\pi = P(X_1 = i)$:

A	B	H	S
0.5	0.0	0.0	0.5

Observations :

$o_{1:4} = \{ b, p, l, e \}$

Find:

Most likely hidden state sequence: $X^*_{1:4}$

Naive Bayes model for classification

- Naive Bayes assumes all features are independent effects of the label.
- Naive Bayes for text: $P(Y, W_1, ..., W_n) = P(Y)\prod_i P(W_i|Y)$, where W_i is the word at position i, Y $\in$ {spam, ham}.

Maximum likelihood estimation

- Given the observed set D, find θ to maximize the probability of D
- $\theta = \underset{\theta}{argmax} P(D|\theta)$
- Set the derivative of $P(D|\theta)$ with respect to θ to zero, and solve for θ.

Laplace smoothing

- pretend that we have seen every outcome k extra times.
- $P_{Lap,k} = \frac{c(x)+k}{N+k|X|}$

Discriminative Learning

Linear classifiers

feature values(inputs), weights (learned), activation (sum, $\text{activation}_w(x) = \sum_i w_i \phi_i(x)$)

Binary perceptron learning process

- start with weights=0
- for each training instance (x,y*): classify with current weights, no change if correct else adjust the weight vector by adding or subtracting the feature vector (subtract if y*=-1).

Discriminative Learning

Multiclass perceptron learning process

- start with weights=0
- pick up training examples one by one
- predict with current weights $\hat{y} = argmax_y(w_y \cdot \phi(x))$, no change if correct. Otherwise, lower score of wrong answer and raise score of right answer $w_{\hat{y}} = w_{\hat{y}} - \phi(x)$, $w_{y*} = w_{y*} + \phi(x)$

Probabilistic Perceptron

- softmax function

Discriminative Learning

Learning by gradient descent

initialize w (e.g., randomly)

repeat for K iterations:

  for each example (x_i, y_i):

    compute gradient $\Delta_i = -\nabla_w \log P_w(y_i|x_i)$

  compute gradient $\nabla_w \mathcal{L} = \sum_i \Delta_i$

  $w \leftarrow w - \alpha \nabla_w \mathcal{L}$

$$\frac{d}{dw_y} \log P_w(y_i|x_i) = x_i(I(y = y_i) - P(y|x_i))$$

- α: learning rate —- hyperparameter that needs to be chosen carefully
- How? Try multiple choices
  - Crude rule of thumb: update should change w by about 0.1-1%

The End