Probabilistic temporal models

1.Value of X at a given time is called the state (usually discrete, finite)

2.The transition model P(Xt | Xt-1) specifies how the state evolves over time

3.Stationarity assumption: same trans

Joint distribution P(X0,…, XT)=P(X0)∏t P(Xt|Xt-1)

4. Are Markov models a special case of Bayes nets? – Yes (Directed acyclic graph, joint) and No (Infinitely many variables, Repetition of transition model not part of standard Bayes net syntax)

5.Markov assumption: Xt+1, … are independent of X0,…,Xt-1 given Xt, A kth-order model allows dependencies on k earlier steps

6. $P(X_t)=\sum_{xt-1} P(X_t,X_{t-1}=x_{t-1})=\sum_{xt-1} P(X_{t-1}=x_{t-1}) P(X_t|X_{t-1}=x_{t-1})$

7.Application of Stationary Distributions: Web Link Analysis, Gibbs Sampling

8.HMM: (1) Initial distribution: P(X0) (2) Transition model: P(Xt|Xt-1) (3) Emission model: P(Et|Xt)

9.HMM: $P(X0,X1,E1,\ldots,XT,ET)=P(X0)\prod_{t=1:T} P(X_t|X_{t-1})P(E_t|X_t)$

10.Independence in HMM: (1) Future states are independent of the past given the present. (给定 Xt 则 Xt+1 与前面都无关) (2) Current evidence is independent of everything else given the current state(给定 Xt 则 Et 与其他任何都无关)

11. (1) Filtering: $P(X_t|e_{1:t})$ belief state (posterior distribution over the most recent state given all evidence.) (2) Prediction: $P(X_{t+k}|e_{1:t})$ for k>0.posterior distribution over a future state given all evidence

(3) Smoothing: $P(X_k|e_{1:t})$ for 0≤k<t (posterior distribution over a past state given all evidence) (4) Most likely explanation: $\text{argmax}_{x1:t} P(x_{1:t}|e_{1:t})$ Ex: speech recognition, decoding with a noisy channel

12.$P(X_{t+1}|e_{1:t+1})=P(X_{t+1}|e_{1:t},e_{t+1})=\alpha P(e_{t+1}|X_{t+1},e_{1:t})P(X_{t+1}|e_{1:t})=\alpha P(e_{t+1}|X_{t+1})P(X_{t+1}|e_{1:t})= \alpha P(e_{t+1}|X_{t+1}) \sum_{xt} P(x_t|e_{1:t})P(X_{t+1}|x_t,e_{1:t}) =\alpha P(e_{t+1}|X_{t+1})\sum_{xt} P(x_t|e_{1:t})P(X_{t+1}|x_t)$,where $\alpha=1/P(e_{t+1}|e_{1:t})$

13. (Forward Algorithm) $P(X_{t+1}|e_{1:t+1}) = \alpha P(e_{t+1}|X_{t+1}) \sum_{x_t} P(x_t|e_{1:t})P(X_{t+1}|x_t)$, $O(|X|^2)$where |X| is the number of states

14.State trellis: graph of states and transitions over time(交叉线那个图)

15.Viterbi algorithm (MLE): computes best paths $\text{argmax}_{x1:t} P(x_{1:t}|e_{1:t})$，每次迭代找指向它路径里面概率最大的那个

16.forward 和 Viterbi 的时间是 $O(|X|^2T)$，空间 $O(|X|T)$，T 是每个 state 可以取的状态数

17. Every discrete DBN can be represented by an HMM, Advantages: Sparse dependencies, exponentially fewer parameters

18. Particle Filtering:|X|太大，reasonable samples 太少。先按照概率放置等权重 sample，然后让 sample 按照 $x_t|x_{t-1}$ 迁移，新的权重是 $P(e_t|x_t)$，然后 resample(权重再调成相等，归一) Propagate forward → Weight → Resample

Markov Decision Process

1.(1)A set of states s ∈ S (2)A set of actions a ∈ A (3)A transition function T(s, a, s'), Probability that a from s leads to s', i.e. P(s'|s,a), Also called the model or the dynamics (4) A reward function R(s, a, s') Sometimes just R(s) or R(s') (5)A start state (6)Maybe a terminal state

2. MDPs are non-deterministic search problems, an explicit policy defines a reflex agent

$$V^*(s) = \max_a Q^*(s,a)$$

$$Q^*(s,a) = \sum_{s'} T(s,a,s')\left[R(s,a,s') + \gamma V^*(s')\right]$$

$$V^*(s) = \max_a \sum_{s'} T(s,a,s')\left[R(s,a,s') + \gamma V^*(s')\right]$$

The Bellman

3.Value Iteration: Complexity of each iteration: $O(S^2A)$

$$V_{k+1}(s) \leftarrow \max_a \sum_{s'} T(s,a,s')\left[R(s,a,s') + \gamma V_k(s')\right]$$

稳态 max 改 argmax 就是 policy extraction

4.Actions are easier to select from q-values than values!

5. Policy iteration: (1) Policy evaluation: calculate utilities for some fixed (not optimal) policy (2) Policy improvement: update policy using one-step look-ahead with resulting converged (not optimal!) utilities as future values. Can converge (much) faster under some conditions (可以迭代($O(S^2)$一步)也可以解线性方程组,因为策略固定. 先随便固定一个策略,计算 value,然后找最好的 policy.

Reinforcement learning

1.(1) Exploration(探索): you have to try unknown actions to get information (2) Exploitation(开发): eventually, you have to use what you know (3) Regret(后悔): even if you learn intelligently, you make mistakes (4) Sampling: because of chance, you have to try things repeatedly (5) Difficulty: learning can be much harder than solving a known MDP (6*) 概率 T 和回报 R 都是不知道的,要学 optimal policy and values.

2.Model Based: 先估计概率,再用这个概率算期望. Model free: 不算概率，用均值之类的做期望.

3.Direct Evaluation: 最后会收敛到对的值,但是 It wastes information about state connections, Each state must be learned separately, It takes a long time to learn.

4. TD Learning: learn immediately from every experience.

Sample of V(s): $sample = R(s,\pi(s),s') + \gamma V^\pi(s')$

Update to V(s): $V^\pi(s) \leftarrow (1-\alpha)V^\pi(s) + (\alpha)sample$

Makes recent samples more important, decreasing learning rate (alpha) can give converging averages

$$\bar{x}_n = \frac{x_n + (1-\alpha)\cdot x_{n-1} + (1-\alpha)^2 \cdot x_{n-2} + \ldots}{1 + (1-\alpha) + (1-\alpha)^2 + \ldots}$$

5.Q learning

Q-value iteration

$$Q_{k+1}(s,a) \leftarrow \sum_{s'} T(s,a,s')\left[R(s,a,s') + \gamma \max_{a'} Q_k(s',a')\right]$$

$$sample = R(s,a,s') + \gamma \max_{a'} Q(s',a')$$

$$Q(s,a) \leftarrow (1-\alpha)Q(s,a) + (\alpha)\,[sample]$$

6. Q-learning converges to optimal policy even if you're acting suboptimally (off-policy learning), learning rate 要不断减小，但不能减小太快

Known MDP: Offline Solution

Goal	Technique
Compute V*, Q*, π*	Value / policy iteration
Evaluate a fixed policy π	Policy evaluation

Unknown MDP: Model-Free

Goal	Technique
Compute V*, Q*, π*	Q-learning
Evaluate a fixed policy π	TD Value Learning

Unknown MDP: Model-Based

Goal	Technique
Compute V*, Q*, π*	VI/PI on approx. MDP
Evaluate a fixed policy π	PE on approx. MDP

7. Exploration vs. Exploitation: ε-greedy

With (small) probability ε, act randomly, With (large) probability 1-ε, act on current policy

Exploration function: takes a Q-value estimate u and a visit count n, and returns an optimistic utility, e.g.f(u,n)=u+k/n.

8.Regret:你的 reward 和 optimal reward 的差。随机和 explore function 最后都是最优的,但是 random 的 regret 更高

9. Linear Value Functions:

$$Q(s,a) = w_1 f_1(s,a) + w_2 f_2(s,a) + \ldots + w_n f_n(s,a)$$

$$\text{transition} = (s, a, r, s')$$

$$\text{difference} = \left[r + \gamma \max_{a'} Q(s', a')\right] - Q(s,a)$$

$$Q(s,a) \leftarrow Q(s,a) + \alpha\,[\text{difference}]$$

$$w_i \leftarrow w_i + \alpha\,[\text{difference}]\, f_i(s,a)$$

新系=原系数+a*(reward+r*max(Q(s',a'))-现在的 Q(s,a))*fi(s,a)

10. Observation: often the feature-based policies that work well (win games, maximize utilities) aren't the ones that approximate V / Q best.

Supervised learning

1.Training data includes desired outputs

2.(1)Classification(分类)=learning f with discrete output value (2)Regression(回归)=learning f with real-valued output value (3)Structured prediction(结构化预测)=learning f with structured output

3.Naive Bayes: Model based

$$P(Y, F_1 \ldots F_n) = P(Y) \prod_i P(F_i|Y)$$

Called "bag-of-words" because model is insensitive to word order or reordering

4. (1)Learn parameters (e.g. model probabilities) on training set (2)Tune hyperparameters on held-out set (3)Compute accuracy of test set (fraction of instances predicted correctly) (4)Very important: never "peek" at the test set!

5.MLE(最大似然估计)

- **MLE:** Choose θ to maximize probability of D

$$\hat{\theta} = \arg\max_\theta \ln P(\mathcal{D} \mid \theta) = \arg\max_\theta \ln \theta^{\alpha_H}(1-\theta)^{\alpha_T}$$

- **Set derivative to zero, and solve!**

$$\frac{d}{d\theta} \ln P(\mathcal{D} \mid \theta) = \frac{d}{d\theta}[\ln \theta^{\alpha_H}(1-\theta)^{\alpha_T}]$$

$$= \frac{d}{d\theta}[\alpha_H \ln\theta + \alpha_T \ln(1-\theta)]$$

$$= \alpha_H \frac{d}{d\theta}\ln\theta + \alpha_T \frac{d}{d\theta}\ln(1-\theta)$$

$$= \frac{\alpha_H}{\theta} - \frac{\alpha_T}{1-\theta} = 0 \qquad \hat{\theta}_{MLE} = \frac{\alpha_H}{\alpha_H + \alpha_T}$$

先把所有样本都按照给的分布乘起来=L，然后取对数(lnL)，然后 dlnL/da=0，求出参数。

6. (1)Overfitting: learn to fit the training data very closely, but fit the test data poorly.(2)Generalization: try to fit the test data as well (2.1)Training data is not representative of the true data distribution,Too few training samples,Training data is noisy (2.2)Too many attributes, some of them irrelevant to the classification task (2.3)The model is too expressive

7.Smoothing Laplace(X,Y 大的时候不好):

$$P_{LAP,k}(x) = \frac{c(x)+k}{N+k|X|} \quad P_{LAP,k}(x|y) = \frac{c(x,y)+k}{c(y)+k|X|}$$

Linear interpolation:

$$P_{LIN}(x|y) = \alpha \hat{P}(x|y) + (1.0 - \alpha)\hat{P}(x)$$

8.感知机: Perceptron,y=w*x,预测对了 w 不动，错了 w=w+y*f

y*是正确的 y，f 是特征(输入)

9.最小二乘法损失函数:找到 w 使其最小

$$L(\boldsymbol{w}) = \sum_i \left(y_i - h_w(\boldsymbol{x}_i)\right)^2 = \sum_i (y_i - \boldsymbol{w}^T \boldsymbol{x}_i)^2$$

找到的参数 $w^*=(X^TX)^{-1}X^Ty$，X 是把输入一行行写成矩阵，y 是每一行输入对应的 y 值向量。

10."Ockham's razor":prefer the simplest hypothesis consistent with the data

Supervised learning

1.Kmeans: Pick K random points as cluster centers (means) Loop {1. Assign data instances to closest mean 2. Assign each mean to the average of its assigned points} Until when no points' assignments change

2. Kmeans 循环的两步都不会让所有点到他们对应的 means 的距离之和变大，Also cannot increase total distance. The point y with minimum squared Euclidean distance to a set of points {x} is their mean

3. GMM: 用 gauss function(正态分布)的线性组合来拟合。如果知道 label, 则直接用每一组的均值和方差做 μ 和 σ,系数是这组占总数比例。

如果不知道，用最大似然算 No closed form solution，

4. EM (GMM): [E step] Compute probability of each instance having each possible label. [M step] Treating each instance as fractionally having both labels, compute the new parameter values 先随机几个分布，然后计算每个样本在不同分布下的概率(带进函数里),然后根据新的样本再计算分布函数(要乘那个比例)

θ (t) = {μ1(t)… μk(t), S1(t)… Sk(t), π1(t)…πk(t)}

$$\mathrm{P}\left(y_j = i \left| x_j, \theta^{(t)}\right.\right) \propto {\pi_i}^{(t)} N\left(x_j \left| {\mu_i}^{(t)}, {\Sigma_i}^{(t)}\right.\right)$$

$${\mu_i}^{(t+1)} = \frac{\sum_j \mathrm{P}\left(y_j = i \left| x_j, \theta^{(t)}\right.\right) x_j}{\sum_{j'} \mathrm{P}\left(y_{j'} = i \left| x_{j'}, \theta^{(t)}\right.\right)}$$

$${\pi_i}^{(t+1)} = \frac{\sum_j \mathrm{P}\left(y_j = i \left| x_j, \theta^{(t)}\right.\right)}{m}$$

m 是数据点的总数

5. 当所有的高斯函数都是球形,而且权重相同方差相同的时候,GMM 退化为 Kmeans(Kmeans 是 GMM 特例), GMM 是把每个点可以对应很多高斯函数,Kmeans 一个点只对应一个 means

6. EM for HMM: Baum–Welch algorithm

7.EM is coordinate ascent on F(θ, Q),E,M 都不会让 F 变小

Natural Language Parsing

1.Syntax: knowledge of the structural relationships between words

2. Grammar: the set of constituents and the rules that govern how they combine

3. Context-free grammars (CFGs), Also known as: Phrase structure grammars. One of the simplest and most basic grammar formalisms

4. A context-free grammar has four components. (1) A set Σ of terminals (words) (2) A set N of nonterminals (phrases) (3) A start symbol S∈N (4) A set R of production rules.

5. A sentence is ambiguous if it has more than one possible parse tree

6.CNF: 只有 A->B C 和 A->terminal 两种

Regular Grammer：Production rules are of the form A->aB or A->a 带概率能用 HMM

7. dependency grammar:看二元关系，CFG:看成分，优点：对于语序自由的语言好,更快,信息可能之后又用,不提取重复. Score 是每一个箭头的 score 之和，多个 score 选大的. Dependency grammars are a subclass of CFGs

8. 暴力枚举 Number of binary trees with n leaves is the Catalan number Cn-1(C(2n-2,n-1)/n)

9.CYK 算法

function CKY-PARSE(*words, grammar*) **returns** *table*

for $j \leftarrow$ **from** 1 **to** LENGTH(*words*) **do**
  $table[j-1, j] \leftarrow \{A \mid A \rightarrow words[j] \in grammar\}$
  for $i \leftarrow$ **from** $j-2$ **downto** 0 **do**
    for $k \leftarrow i+1$ **to** $j-1$ **do**
      $table[i,j] \leftarrow table[i,j] \cup$
        $\{A \mid A \rightarrow BC \in grammar,$
        $B \in table[i,k],$
        $C \in table[k,j]\}$

先填每个对角线，顺着向上找。每个(i, j)看看对应的行,列能不能拼成一个(有没有对应规则),如果能就填进去

10. 带概率的 CYK 如果有二义性，选概率大的那一个填进去

11. corpus 语料库,treebank：有监督的，所有 parsetree 已经写好,学习概率,MLE 效果不好,因为 tree 信息不够. Latent Variable Grammars(subtypes) Discriminative Parsing

12. Unsupervised Methods 没有标记 Structure search，Parameter learning，用 EM，EMM for HMM is special case