Announcements

- W4 out, due next week Monday
- P4 out, due next week Friday
- Mid-semester survey

Announcements II

- Course contest
- Regular tournaments. Instructions have been posted!
- First week extra credit for top 20, next week top 10, then top 5, then top 3.
- First nightly tournament: tentatively Monday night

Today

- Dynamic Bayes Nets (DBNs)
  - [sometimes called temporal Bayes nets]
- Demos:
  - Localization
  - Simultaneous Localization And Mapping (SLAM)
- Start machine learning

P4: Ghostbusters 2.0

- Plot: Pacman’s grandfather, Grandpac, learned to hunt ghosts for sport.
- He was blinded by his power, but could hear the ghosts’ banging and clanging.
- Transition Model: All ghosts move randomly, but are sometimes biased
- Emission Model: Pacman knows a “noisy” distance to each ghost

Dynamic Bayes Nets (DBNs)

- We want to track multiple variables over time, using multiple sources of evidence
- Idea: Repeat a fixed Bayes net structure at each time
- Variables from time t can condition on those from t-1

Discrete valued dynamic Bayes nets are also HMMs
Exact Inference in DBNs
- Variable elimination applies to dynamic Bayes nets
- Procedure: "unroll" the network for T time steps, then eliminate variables until $P(X_T|e_1:T)$ is computed

Online belief updates: Eliminate all variables from the previous time step; store factors for current time only

DBN Particle Filters
- A particle is a complete sample for a time step
- Initialize: Generate prior samples for the $t=1$ Bayes net
  - Example particle: $G_1^a = (3,3)$ $G_1^b = (5,3)$
- Elapse time: Sample a successor for each particle
  - Example successor: $G_2^a = (2,3)$ $G_2^b = (6,3)$
- Observe: Weight each entire sample by the likelihood of the evidence conditioned on the sample
  - Likelihood: $P(E_1^a|G_1^a) * P(E_1^b|G_1^b)$
- Resample: Select prior samples (tuples of values) in proportion to their likelihood

DBN Particle Filters
- A particle is a complete sample for a time step
- Initialize: Generate prior samples for the $t=1$ Bayes net
  - Example particle: $G_1^a = (3,3)$ $G_1^b = (5,3)$
- Elapse time: Sample a successor for each particle
  - Example successor: $G_2^a = (2,3)$ $G_2^b = (6,3)$
- Observe: Weight each entire sample by the likelihood of the evidence conditioned on the sample
  - Likelihood: $P(E_1^a|G_1^a) * P(E_1^b|G_1^b)$
- Resample: Select prior samples (tuples of values) in proportion to their likelihood

Trick I to Improve Particle Filtering Performance: Low Variance Resampling
- Advantages:
  - More systematic coverage of space of samples
  - If all samples have same importance weight, no samples are lost
  - Lower computational complexity

Trick II to Improve Particle Filtering Performance: Regularization
- If no or little noise in transitions model, all particles will start to coincide
  - Regularization: introduce additional (artificial) noise into the transition model

SLAM
- SLAM = Simultaneous Localization And Mapping
  - We do not know the map or our location
  - Our belief state is over maps and positions!
  - Main techniques: Kalman filtering (Gaussian HMMs) and particle methods
  - [DEMOS]
Robot Localization

- In robot localization:
  - We know the map, but not the robot’s position
  - Observations may be vectors of range finder readings
  - State space and readings are typically continuous (works basically like a very fine grid) and so we cannot store $B(X)$
  - Particle filtering is a main technique

[Demos]
Global-floor

SLAM

- SLAM = Simultaneous Localization And Mapping
  - We do not know the map or our location
  - State consists of position AND map!
  - Main techniques: Kalman filtering (Gaussian HMMs) and particle methods

Particle Filter Example

- DEMOS
  - fastslam.avi, visionSlam_heliOffice.wmv

Further readings

- We are done with Part II Probabilistic Reasoning
- To learn more (beyond scope of 188):
  - Koller and Friedman, Probabilistic Graphical Models (CS281A)
  - Thrun, Burgard and Fox, Probabilistic Robotics (CS287)

Part III: Machine Learning

- Up until now: how to reason in a model and how to make optimal decisions
- Machine learning: how to acquire a model on the basis of data / experience
  - Learning parameters (e.g. probabilities)
  - Learning structure (e.g. BN graphs)
  - Learning hidden concepts (e.g. clustering)
### Machine Learning Today

- An ML Example: Parameter Estimation
  - Maximum likelihood
  - Smoothing
  - Applications
  - Main concepts
  - Naïve Bayes

### Estimation: Parameter Estimation

- Estimating the distribution of a random variable
- Elicitation: ask a human (why is this hard?)
- Empirically: use training data (learning!)
  - E.g.: for each outcome $x$, look at the empirical rate of that value:
  $$P_{ML}(x) = \frac{\text{count}(x)}{\text{total samples}}$$
  - This is the estimate that maximizes the likelihood of the data
  - $P_0(x) = \begin{cases} \frac{1}{6} & \text{if } x \in \{H, T\} \\ 0 & \text{otherwise} \end{cases}$
- Issue: overfitting. E.g., what if only observed 1 jelly bean?

### Estimation: Smoothing

- Relative frequencies are the maximum likelihood estimates
  $$\hat{g}_{ML} = \arg\max_\theta P(X|\theta)$$
  $$\hat{P}_{ML}(x) = \frac{\text{count}(x)}{\text{total samples}}$$
- In Bayesian statistics, we think of the parameters as just another random variable, with its own distribution
  $$\hat{g}_{MAP} = \arg\max_\theta P(\theta|X)$$
  $$\hat{P}(\theta|X) = \frac{P(X|\theta)P(\theta)}{\int P(X|\theta)P(\theta)\,d\theta}$$

### Estimation: Laplace Smoothing

- Laplace’s estimate:
  - Pretend you saw every outcome once more than you actually did
  $$P_{LAP}(x) = \frac{c(x) + 1}{\sum_y [c(y) + 1]}$$
  - Can derive this as a MAP estimate with Dirichlet priors (see cs281a)
  $$P_{MAP}(X) = \frac{\alpha + |X|}{\beta + |Y|}$$
- Example: Spam Filter
  - Input: email
  - Output: spam/ham
  - Setup:
    - Get a large collection of example emails, each labeled “spam” or “ham”
    - Note: someone has to hand label all this data
    - Want to learn to predict labels of new, future emails
  - Features: The attributes used to make the ham/spam decision
    - Words: FREE!
    - Text Patterns: $d$, CAPS
    - Non-text: SenderInContacts
  - Example email:
    - Dear Sir. First, I must solicit your confidence in this transaction, this is by virtue of its nature as being utterly confidential and top secret.
    - To be removed from future mailings, simply reply to this message and put "REMOVE" in the subject.
    - 99 million email addresses for only $99
  - Ok, I know this is blatantly OT but I'm beginning to go insane. Had an old Dell Dimension XPS sitting in the corner and decided to plug it in, it's all the power nothing happened.
Example: Digit Recognition

- **Input:** images / pixel grids
- **Output:** a digit 0-9
- **Setup:**
  - Get a large collection of example images, each labeled with a digit
  - Note: someone has to hand label all this data!
  - Want to learn to predict labels of new, future digit images
- **Features:** The attributes used to make the digit decision
  - Pixels: (6,8)=ON
  - Shape Patterns: NumComponents, AspectRatio, NumLoops
  - ...

Other Classification Tasks

- **In classification, we predict labels y (classes) for inputs x**
- **Examples:**
  - Spam detection (input: document, classes: spam / ham)
  - OCR (input: images, classes: characters)
  - Medical diagnosis (input: symptoms, classes: diseases)
  - Automatic essay grader (input: document, classes: grades)
  - Fraud detection (input: account activity, classes: fraud / no fraud)
  - Customer service email routing
  - … many more
- **Classification is an important commercial technology!**

Important Concepts

- **Data:** labeled instances, e.g. emails marked spam/ham
  - Training set
  - Held-out set
  - Test set
- **Features:** attribute-value pairs which characterize each x
- **Experimentation cycle**
  - Learn parameters (e.g. model probabilities) on training set
  - (Tune hyperparameters on held-out set)
  - Compile accuracy of test set
  - Very important: never “peek” at the test set!
- **Evaluation**
  - Accuracy: fraction of instances predicted correctly
- **Overfitting and generalization**
  - Want a classifier which does well on test data
  - Overfitting: fitting the training data very closely, but not generalizing well
  - We’ll investigate overfitting and generalization formally in a few lectures

Training Data

Held-Out Data

Test Data