Uncertain Reasoning and Data Engineering: Overview

Wednesday, March 1, 2000

William H. Hsu
Department of Computing and Information Sciences, KSU

Readings:
Chapter 15, Russell and Norvig

“Bayesian Networks Without Tears”, Charniak

Graphical Models of Probability Distributions

• Idea
  - Want: model that can be used to perform inference
  - Desired properties
    - Ability to represent functional, logical, stochastic relationships
    - Express uncertainty
    - Observe the laws of probability
    - Tractable inference when possible
    - Can be learned from data

• Additional Desiderata
  - Ability to incorporate knowledge
    - Knowledge acquisition and elicitation: in format familiar to domain experts
    - Language of subjective probabilities and relative probabilities
    - Support decision making
    - Represent utilities (cost or value of information, state)
    - Probability theory + utility theory + decision theory
    - Ability to reason over time (temporal models)

Bayesian Belief Networks (BBNS): Definition

• Conditional Independence
  - X is conditionally independent (CI) from Y given Z (sometimes written X ⊥ Y | Z) if
    \[ P(X | Y, Z) = P(X | Z) \]
  - For all values of X, Y, and Z

• Bayesian Network
  - Directed graph model of conditional dependence assertions (or CI assumptions)
  - Vertices (nodes): denote events (each a random variable)
  - Edges (arcs, links): denote conditional dependencies

• General Product (Chain) Rule for BBNS

• Example (“Sprinkler” BBN)

Unsupervised Learning and Conditional Independence

• Given: (n + 1)-Tuples
  - No notion of instance variable or label
  - After seeing some examples, want to know something about the domain
  - Correlations among variables
  - Probability of certain events
  - Other properties

• Want to Learn: Most Likely Model that Generates Observed Data
  - In general, a very hard problem
  - Under certain assumptions, have shown that we can do it

• Assumption: Causal Markovity
  - Conditional independence among “effects”, given “cause”
  - When is the assumption appropriate?

• Structure Learning
  - Can we learn more general probability distributions?
  - Example: automatic speech recognition (ASR), natural language, etc.

Bayesian Belief Networks: Properties

• Conditional Independence
  - Variable (node): conditionally independent of non-descendants given parents
  - Example

  \[
  P(X_1, X_2, \ldots, X_n) = \prod \left( P(X_i | \text{parents}(X_i)) \right)
  \]

• Bayesian Network: Probabilistic Semantics
  - Node: variable
  - Edge: one axis
  - Descendant/Non Descendant
  - Result: chain rule for probabilistic inference
  - Edge: one axis of a conditional probability table (CPT)

Lecture Outline

• Readings: 6.11, Mitchell: Chapter 15, Russell and Norvig: Charniak Tutorial

• Suggested Reference: Lectures 9-13, CIS 786 (Fall, 1999)

• This Week’s Review: “A Theory of Inferred Causation”, Pearl and Verma

• Graphical Models of Probability
  - Bayesian networks: introduction
    - Definition and basic principles
    - Conditional independence and causal Markovity
  - Inference and learning using Bayesian networks
    - Acquiring and applying distributions (conditional probability tables)
    - Learning tree dependent distributions and polytrees

• Learning Distributions for Networks with Specified Structure
  - Gradient learning
  - Maximum weight spanning tree (MWST) algorithm for tree-structured networks

• Reasoning under Uncertainty: Applications and Augmented Models

• Next Lecture: (More on) Learning Bayesian Network Structure

Overview

• Knowledge acquisition
  - Can be learned from data

• Tractable inference when possible

• Express uncertainty

• Additional Desiderata
  - Ability to reason under time (temporal models)

Graphical Models of Probability Distributions

• Idea
  - Want: model that can be used to perform inference

• Desired properties
  - Ability to represent functional, logical, stochastic relationships

• Express uncertainty
  - Observe the laws of probability

• Tractable inference when possible

• Can be learned from data
Bayesian Belief Networks: Inference

- **Problem Definition**
  - Given:
    - Bayesian network with specified CPTs
    - Observed values for some nodes in network
  - Return: inferred (probabilities of) values for query node(s)

- **Implementation**
  - Bayesian network contains all information needed for this inference
  - If only one variable with unknown value, easy to infer it
  - In general case, problem is intractable (NP-hard: reduction to 3-CNF-SAT?)
  - In practice, can succeed in many cases using different methods
  - Exact inference: work well for some network structures
  - Monte Carlo: “simulate” network to randomly calculate approximate solutions
  - Key machine learning issues
    - Feasible to select this information or learn it from data?
    - How to learn structure that makes inference more tractable?

Tree Dependent Distributions

- **Polytrees**
  - Also singly-connected Bayesian networks
  - Definition: a Bayesian network with no undirected loops
  - Idea: restrict distributions (CPTs) to single nodes
  - Theorem: inference in singly-connected BBN requires linear time
    - Linear in network size, including CPT sizes
    - Much better than for unrestricted (multiply-connected) BBNs

- **Tree Dependent Distributions**
  - Further restriction of polytrees: every node has at one parent
  - Now only need to keep 1 prior, P(root), and n - 1 CPTs (1 per node)
  - All CPTs are 2-dimensional: P(y | parent)
  - Indepedence Assumptions
    - As for general BBN: x is independent of non-descendants given (single) parent z
    - Very strong assumption (applies in some domains but not most)

Inference in Trees

- **Inference in Tree-Structured BBNs (“Trees”)**
  - Generalization of Naive Bayes to model of tree dependent distribution
  - Given: tree T with all associated probabilities (CPTs)
  - Evaluate: probability of a specified event, P(x)

- **Inference Procedure for Polytrees**
  - Recursively traverse tree
    - Breadth-first, source(s) to sink(s)
    - Stop when query value P(x) is known
  - Perform inference at each node
    - \[ P(x) = \frac{\prod P(y_i | y_{parents}(y_i))}{\prod P(y_i | y_{parents}(y_i))} \]
  - NB: for trees, proceed root to leaves (e.g., breadth-first or depth-first)
  - Simple application of Bayes’s rule (more efficient algorithms exist)

Learning Distributions: Objectives

- **Learning The Target Distribution**
  - What is the target distribution?
    - Can’t use the “true” target distribution
    - Case in point: suppose target distribution was \( P_t \) (collected over 20 examples)
    - Using Naive Bayes would not produce an \( P(x) \) close to the MAP/ML estimate
  - Relating CI assumptions: expensive
    - MLE becomes intractable; BOC approximation, highly intractable
    - Instead, should make judicious CI assumptions
  - As before, goal is generalization
    - Given D (e.g., \( D_{101, 101, 1000} \))
    - Would like to know \( P(t|x) \) or \( P(t|x) = P_t \)
    - Several Variants
      - Known or unknown structure
      - Training examples may have missing values
    - Known structure and no missing values: as easy as training Naive Bayes

Learning Bayesian Networks: Partial Observability

- **Suppose Structure Known, Variables Partially Observable**
  - Example
    - Can observe ForestFire, Storm, BusTourGroup, Thunder
  - Similar to training artificial neural net with hidden units
  - Causes: Storm, BusTourGroup
    - Observable effects: ForestFire, Thunder
  - Intermediate variables: Lightning, Campfire
  - Learning Algorithm
    - Can use gradient learning (as for ANNs)
    - Converge to network \( h(x) \) that (locally) maximizes \( P(D | h) \)
  - Analogy: Medical Diagnosis
    - Causes: diseases or diagnostic findings
    - Intermediate: hidden causes or hypothetical inferences (e.g., heart rate)
    - Observables: measurements (e.g., from medical instrumentation)

Learning Distributions: Learning Distributions

- **Algorithm Train-BN (D)**
  - Let \( w_{ij} \) denote one entry in the CPT for variable \( Y_i \) in the network
    - \( w_{ij} = P(Y_i | y_j) \) parents(\( y_j \) = \{\text{the list of values}\})
    - e.g., if \( Y_i = \text{Campfire} \), then (for example) \( w_{ij} = \text{<Storm} = T, \text{BusTourGroup} = F> \)
  - WHILE termination condition not met DO
    - perform gradient ascent
    - Update all CPT entries \( w_{ij} \) using training data \( D \)
    - Renormalize \( w_{ij} \) to assure invariants:
      \[ \sum_j w_{ij} = 1, 0 \leq w_{ij} \leq 1 \]

- **Applying Train-BN**
  - Learns CPT values
  - Useful in case of known structure
  - Next: learning structure from data
Tree Dependent Distributions: Learning The Structure

- Problem Definition: Find Most Likely \( T \) Given \( D \)
- Brute Force Algorithm
  - FOR each tree \( T \) DO
    - Compute the likelihood of \( T \):
      \[
      PT/D \approx \text{arg max } P(D|T) = \arg \max \prod_{x \in X} P(x, \text{parental}(x) | T)
      \]
    - RETURN the maximal \( T \)
- Is This Practical?
  - Typically no… (H analogous to that of ANN weight space)
  - What can we do about it?
- Solution Approaches
  - Use criterion / scoring function: Kullback-Leibler (K-L) distance
    \[
    D(P(T)||P(x)) = \sum_{x \in X} P(x) \log \left( \frac{P(x)}{P(x|T)} \right)
    \]
  - Measures how well a distribution \( P \) approximates a distribution \( P' \)
  - aka K-L divergence, aka cross-entropy, aka relative entropy

Tree Dependent Distributions: Maximum Weight Spanning Tree (MWST)

- Input: \( m \) Measurements (n-Tuples), i.i.d. – \( P \)
- Algorithm Learn-Tree-Structure (D)
  - FOR each variable \( X \) DO estimate \( P(x) \) // binary variables: \( n \) numbers
  - FOR each pair \( (X, Y) \) DO estimate \( P(x, y) \) // binary variables: \( m \) numbers
  - FOR each pair \( DO \) compute the mutual information (measuring the information \( X \) gives about \( Y \) with respect to this empirical distribution)
    \[
    I(X;Y) = \sum_{x,y} P(x,y) \log \left( \frac{P(x,y)}{P(x)P(y)} \right)
    \]
  - Build a complete undirected graph with all the variables as vertices
  - Let \( \lambda(x,y) \) be the weight of edge \( (X, Y) \)
  - Build a Maximum Weight Spanning Tree (MWST)
  - Transform the resulting undirected tree into a directed tree (choose a root, and set the direction of all edges away from it)
  - Place the corresponding CPTs on the edges (gradient learning)
  - RETURN: a tree-structured BBN with CPT values

Related Work In Bayesian Networks

- BBN Variants, Issues Not Covered Yet
  - Temporal models
    - Markov Decision Processes (MDPs)
    - Partially Observable Markov Decision Processes (POMDPs)
  - Useful in reinforcement learning
    - Decision theoretic model
    - Augments BBN with utility values and decision nodes
  - Unsupervised learning (EM, AutoClass)
    - Feature (subset) selection: finding relevant attributes
  - Current Research Topics Not Addressed In This Course
    - Hidden variables (introduction of new variables not observed in data)
    - Incremental BBN learning: modifying network structure online (“on the fly”)
  - Structure learning for stochastic processes
    - Noisy-OR Bayesian networks: another simplifying restriction

Summary Points

- Graphical Models of Probability
  - Bayesian networks; introduction
  - Definition and basic principles
  - Conditional independence (causal Markovity) assumptions, tradeoffs
- Inference and learning using Bayesian networks
  - Inference and learning using Bayesian networks
  - Acquiring and applying CPTs
  - Searching the space of trees: max likelihood
  - Examples: Sprinkler, Cancer, Forest-Fire, generic tree learning
- CPT Learning; Gradient Algorithm (Train-BN)
  - Structure Learning in Trees: MWST Algorithm Learn-Tree-Structure
  - Reasoning under Uncertainty: Applications and Augmented Models
- Some Material From: http://robotics.Stanford.EDU/~koller
  - Next Week: Read Heckerman Tutorial
  - Next Class: Presentation - “In Defense of Probability”, Cheesman