The Probabilistic Revolution in Artificial Intelligence

Michael Kearns
Division Manager, AI Principles Research
AT&T Laboratories – Research

Road Map

- Some History and Sociology of AI Research
- Facing and Embracing Uncertainty
- Case Study I: Graphical Models for Probabilistic Inference
- Case Study II: Markov Decision Processes for Planning
- Case Study III: Some Recent Advances in Machine Learning

Goals of Artificial Intelligence

- Build systems that approach human performance on "difficult" tasks (even in limited domains)
- Develop models of perception and behavior that explain or shed light on true mechanisms (biology)
- Develop rich representations of knowledge that are amenable to algorithmic manipulation (computer science)

The “California or Bust” Approach

- Attack a problem of sufficient generality that many aspects of “intelligence” must be solved simultaneously
- Examples: Carnegie-Mellon SOAR project (real-time constraints); MIT Cog project (physical movement); SHRDLU project
- Very little work on definitions and mathematical foundations
- Beyond criticism if successful
- Decreasing in popularity

The Balkanization of AI

- Machine Learning/Neural Networks
- Reasoning and Knowledge Representation
- Planning
- Speech Recognition
- Natural Language Processing
- Computer Vision

An explosion of formalisms, specialized methods and tools

Pitfalls of Balkanization: “Type Mismatches”

- Logic-based reasoning systems:
  - furniture(x) $\rightarrow$ object(x)
  - vehicle(x) $\rightarrow$ object(x)
  - father(x, y) **and** father(y, z) $\rightarrow$ grandfather(x, z)
  - Inferences are **exact**
  - Assume predicates **learned** from experience
- Machine learning systems:
  - Learn **approximate** predicates over an atomic feature set from **statistical** observations
  - four-legs(x) **and** can-sit(x) $\rightarrow$ chair(x)

Problem: learning system run separately on chairs and furniture may result in representations violating chair(x) $\rightarrow$ furniture(x)!

Recent Trends: The Probabilistic Revolution

- Logic-based inference $\rightarrow$ (approximate) inference in graphical models; Bayesian networks
- Operator-based planning $\rightarrow$ (approximately) optimal policy computation in Markov decision processes; reinforcement learning
- Learning logical representations $\rightarrow$ learning "distributed" representations; neural networks

Features of the Revolution

- A giant step towards reality
- **Explicit** modeling of both “genuine” uncertainty (probabilistic behavior) and “ignorance” uncertainty
- Accompanied by an acceptance of and emphasis on **approximate** solutions and computations
- Probability theory provides a **unifying** language; reduces type mismatches
- Starting to see significant unification of **algorithms**

Causes for guarded optimism on the grand goals.

Representing Distributions by Bayesian Networks

- Have a joint distribution P over random variables
- Start with conditional decomposition of joint:

$$\begin{aligned} P(X_1, \ldots, X_N) &= P(X_1)P(X_2|X_1)P(X_3|X_1, X_2)\cdots P(X_N|X_1, \ldots, X_{N-1}) \\ &= \Pi_{i=1}^{N} P(X_i|X_1, \ldots, X_{i-1}) \end{aligned}$$

- Idea: hope that conditional **independences** lead to simplifications like $P(X_5|X_1, X_2, X_3, X_4) = P(X_5|X_1, X_3)$

Example: Burglar Alarm Model

- Variables A(larm), B(urglar), E(arthquake), J(ohn), M(ary)
- Full joint distribution $P(A, B, E, J, M)$
- Try exploiting intuitions about **causality**: B and E are independent of everything; A depends only on B and E; J and M depend only on A
- Decomposition:

$$P(A, B, E, J, M) = P(B)P(E)P(A|B, E)P(J|A)P(M|A)$$

- Instead of $2^5 - 1 = 31$ parameters, need only $1+1+4+2+2 = 10$ parameters
- Natural representation as a **directed graph**

Some Properties of Bayesian Networks

- Edges encode conditional dependences; **absence** of edges encode conditional **independences**
- Potential for dimensionality reduction: $O(2^N) \to O(2^k N)$ parameters, $k =$ maximum in-degree
- Caveat: order of decomposition can be crucial!
- Ease of extracting causal mechanisms, difficulty of assigning probabilities
- Hidden variables can simplify representation
- Permit development of graph-theoretic algorithms for natural problems

Subtleties of Conditional Independence: Examples

- Crucial distinction: some variables may be **observed** (**evidence**), others are unobserved

- B and E are independent with **no** evidence, but become **dependent** if A is observed

- If alarm went off, but **then** we learn there was an earthquake, we are more likely to dismiss burglary ("explaining away")

- Another example: if we already know that the car won't start, and **then** discover our battery is dead, we are less likely to believe we're out of gas

- **d-separation**: graph-theoretic characterization of when variables X and Y are independent given variable set Z

Basic Operations on Bayesian Networks

- **Inference**:
  - Given some evidence (observations) Z, what is the conditional distribution on **query** variable X?
  - Example: What is $P(E|J)$?
  - Example: What is the probability we're out of gas given that the car won't start and the battery is dead?
- **Learning**:
  - Parameter estimation: given the graphical structure, learn the conditional probability tables from observed data
  - Structure learning: learn the graphical structure

Inference: State of the Art

- Intractable in the general case
- Important special case: **polytree** graphs permit efficient algorithm based on dynamic programming
- Efficient mean-field methods for **approximating** $P(X|Z)$ in "dense" networks
- Recently: connection between inference in Bayesian networks and decoding of "turbocodes"

Some Applications of Bayesian Networks

- PATHFINDER system (Stanford): large Bayesian network for medical diagnosis built in consultation with physicians
- Microsoft troubleshooting scripts

A Probabilistic Approach to Planning

- Bayesian networks a **static** and **descriptive** representation of knowledge
- Do not explicitly model **utility** of actions, or **dynamics** of environment
- How do we model our possible **actions** in an uncertain world, and their consequences?

Markov Decision Processes

- **State space**: $1, \ldots, N$ (N small for now)
- **Actions**: $a_1, \ldots, a_k$
- **Transition probabilities**: $P_a(ij)$ is probability of going from state i to state j under action a
- **Rewards**: $R_a(i)$ is (expected) reward for executing action a in state i
- **Policy**: a mapping π from states to actions

Want to find a policy with high reward...

Two Measures of Policy Performance

If following a policy yields actual rewards $r_1, r_2, \ldots, r_T$:

- Discounted return: $r_1 + \gamma r_2 + \gamma^2 r_3 + \cdots + \gamma^{T-1} r_T$
- Undiscounted return: $(1/T)(r_1 + r_2 + \cdots + r_T)$

In either case, want a policy that maximizes expected return.

Computing the Optimal Policy Given the Model

- **Value** of state i:

$$
\begin{aligned}
V^*(i) &= \max_a \{R_a(i) + \gamma \sum_j P_a(ij) V^*(j)\} \\
&= \max_a \{Q(i,a)\}
\end{aligned}
$$

- **Value Iteration Algorithm**: Make **guesses** for the $V^*(i)$, iterate on guesses

- Converges to $V^*(i)$ rapidly; greedy policy on values gives optimal policy

Undiscounted case: PSPACE-hard, but polynomial time for finite horizon.

Learning a Good Policy

- Q**-learning**: If a from state i gave reward r and state j

$$Q(i,a) \leftarrow (1-\alpha)Q(i,a) + \alpha(r + \gamma \max_{a'} Q(j,a'))$$

- **If** all states are visited infinitely often, yields true Q-values and optimal policy
- Problem: **Exploitation-Exploration Trade-Off**

Recent work: algorithm for general MDP's that learns a policy within ϵ of optimal in time polynomial in N and $1/\epsilon$.

When N is Large

- State aggregation
- Function approximation
- Assumptions on the MDP: e.g., factorization

An Application: Tesauro's TD-Gammon

- Only get reward at end of game
- Function approximation by neural network
- Trained by self-play for 300K games
- Competitive with top three human players

Another application: job-shop scheduling, actions are local modifications to schedule

Recent Topics in Machine Learning

- Adopted probabilistic and statistical views long ago
- Quantifying generalization ability: methods from statistical mechanics
- A new framework for learning: boosting

The Usual Framework

- Input space X, distribution P
- Unknown **target** function $f : X \to Y$
- **Training data** $S = (x_1, y_1), \ldots, (x_n, y_n)$, $y_i = f(x_i)$ (+noise), x_i i.i.d. from P
- Use S to choose **hypothesis** $h \in H$, **training** error $\hat{\epsilon}(h)$
- **Generalization** error $\epsilon(h) = \text{Pr}_P[h(x) \neq f(x)]$

Learning Curves and Statistical Mechanics

- **Learning curve** of an algorithm: $\epsilon(h)$ as a function of n
- "Classical" analyses: **bound** learning curve by $\epsilon_0 + c_0/\sqrt{m}$
- Stat mech analysis: each $h \in H$ is a system state of "energy" $\widehat{\epsilon}(h)$ and probability $e^{-(1/T)\widehat{\epsilon}(h)}$
- Discovery of **phase transitions** and **symmetry breaking** in neural networks

An Alternate Framework: Boosting Mediocre Algorithms

- Give sample S to algorithm A, get hypothesis h_1
- Modify distribution on S, give to A, get $h_2 \dots$
- **Combine** $h_1, h_2, \dots$ (e.g. weighted voting)
- **Weak learner** A may be complex algorithm, or simple search for best single variable

Benefits of the Probabilistic Revolution

- More realistic and tractable models
- Common mathematical language for AI
- Reuse of algorithms: approximate optimization, Expectation-Maximization, gradient descent
- Connections with older disciplines and demystification