Lecture 29

Conclusions and Final Review

Thursday, December 06, 2001

William H. Hsu
Department of Computing and Information Sciences, KSU
http://www.cis.ksu.edu/~bhsu

Readings:
Chapters 1-10, 13, Mitchell
Chapters 14-21, Russell and Norvig

Lecture 0:
A Brief Overview of Machine Learning

- Overview: Topics, Applications, Motivation
- Learning = Improving with Experience at Some Task
  - Improve over task *T*,
  - with respect to performance measure *P*,
  - based on experience *E*.
- Brief Tour of Machine Learning
  - A case study
  - A taxonomy of learning
  - Intelligent systems engineering: *specification of learning problems*
- Issues in Machine Learning
  - Design choices
  - The performance element: intelligent systems
- Some Applications of Learning
  - Database mining, reasoning (inference/decision support), acting
  - Industrial usage of intelligent systems

Lecture 1:
Concept Learning and Version Spaces

- Concept Learning as Search through *H*
  - Hypothesis space *H* as a state space
  - Learning: finding the correct hypothesis
- General-to-Specific Ordering over *H*
  - Partially-ordered set: Less-Specific-Than (More-General-Than) relation
  - Upper and lower bounds in *H*
- Version Space Candidate Elimination Algorithm
  - *S* and *G* boundaries characterize learner's uncertainty
  - Version space can be used to make predictions over unseen cases
- Learner Can Generate Useful Queries
- Next Lecture: When and Why Are Inductive Leaps Possible?

Lectures 2-3: Introduction to COLT

- The Need for Inductive Bias
  - Modeling inductive learners with equivalent deductive systems
  - Kinds of biases: preference (search) and restriction (language) biases
- Introduction to Computational Learning Theory (COLT)
  - Things COLT attempts to measure
  - Probably-Approximately-Correct (PAC) learning framework
- COLT: Framework Analyzing Learning Environments
  - Sample complexity of *C*, computational complexity of *L*, required expressive power of *H*
  - Error and confidence bounds (PAC: $0 < ? < 1/2$, $0 < ? < 1/2$)
- What PAC *Prescribes*
  - Whether to try to learn *C* with a known *H*
  - Whether to try to reformulate *H* (apply change of representation)
- Vapnik-Chervonenkis (VC) Dimension: Measures Expressive Power of *H*
- Mistake Bounds

Lectures 4-5: Decision Tree Induction

- Model ("Architecture"): Decision Trees (DTs)
- Algorithm *Build-DT*: Top Down Induction
- Entropy and Information Gain: *ID3*
  - Goal: to measure *uncertainty removed* by splitting on a candidate attribute *A*
  - *ID3* ? *Build-DT* using *Gain*(•)
- ID3 as Hypothesis Space Search (in State Space of Decision Trees)
- Data Mining using *MLC++* (Machine Learning Library in C++)
- Occam's Razor and Decision Trees
  - Preference biases versus language biases
  - Minimum Description Length (MDL) – justification for Occam's Razor biases
- Overfitting
  - Problem: fitting training data too closely
  - Overfitting prevention, avoidance, and recovery techniques
- Other Ways to Make Decision Tree Induction More Robust

Lectures 6-7: ANNs (Perceptrons, MLPs, Backprop)

- Neural Networks: Parallel, Distributed Processing Systems
  - Biological and artificial (ANN) types
  - Perceptron (linear threshold unit/gate, aka LTU/LTG): model neuron
- Update Rules for Single-Layer Networks
  - Multiplicative (Hebbian, Winnow), additive (gradient: Perceptron, Delta Rule)
  - Batch versus incremental mode
- Advantages and Disadvantages of LTG
  - "Disadvantage" (tradeoff): simple and restrictive
  - "Advantage": perform well on many realistic problems (e.g., some text learning)
- Multi-Layer ANNs and Backprop
  - Backpropagation of error: distributes penalty (loss) function throughout network
  - Gradient learning: takes derivative of error surface with respect to weights
  - Error is based on difference between desired output (t) and actual output (o)
- Overfitting in ANNs
- Recurrent ANNs: Can Express Temporal Depth (Non-Markovity)

Kansas State University
Department of Computing and Information Sciences

Lecture 8: Statistical Evaluation of Hypotheses

- **Statistical Evaluation Methods for Learning: Three Questions**
  - **Generalization quality**
    - ***How well does observed accuracy estimate generalization accuracy?***
    - **Estimation bias and variance**
    - **Confidence intervals**
  - **Comparing generalization quality**
    - ***How certain are we that h_1 is better than h_2?***
    - **Confidence intervals for paired tests**
  - **Learning and statistical evaluation**
    - ***What is the best way to make the most of limited data?***
    - ***k*-fold CV**
- **Tradeoffs: Bias versus Variance**
- **Next: Sections 6.1-6.5, Mitchell (Bayes's Theorem; ML; MAP)**

Lectures 9-11: Bayesian Learning

- **Introduction to Bayesian Learning: Bayes's Theorem**
  - Definition of conditional (posterior) probability $P(x \mid y)$
  - Bayes's Theorem: $P(x \mid y) = P(y \mid x) \cdot P(x) / P(y)$
- **Maximum *A Posteriori* (MAP) and Maximum Likelihood (ML) Hypotheses**
  - Bayes's Rule and MAP
  - Uniform priors: allow use of MLE to generate MAP hypotheses
  - Relation to version spaces, candidate elimination
- **Minimum Description Length (MDL) Revisited**
  - Bayesian Information Criterion (BIC): justification for Occam's Razor
- **Bayes Optimal Classifier (BOC)**
  - Using BOC as a "gold standard"
- **Gibbs Classifier**
  - Ratio bound
- **Simple (Naïve) Bayes**
  - Rationale for assumption; pitfalls

$$v_{NB} = \arg\max_{v_j \in V} P(v_j) \prod_i P(x_i \mid v_j)$$

Lectures 12-13: Bayesian Belief Networks (BBNs)

- **Graphical Models of Probability**
  - Bayesian networks: introduction
    - Definition and basic principles
    - Conditional independence (causal Markovity) assumptions, tradeoffs
  - Inference and learning using Bayesian networks
    - Inference in polytrees (singly-connected BBNs)
    - Acquiring and applying CPTs: gradient algorithm *Train-BN*
- **Structure Learning in Trees: MWST Algorithm *Learn-Tree-Structure***
- **Reasoning under Uncertainty using BBNs**
  - Learning, eliciting, applying CPTs
  - In-class exercise: *Hugin* demo; CPT elicitation, application
  - Learning BBN structure: constraint-based versus score-based approaches
  - *K2*, other scores and search algorithms
- **Causal Modeling and Discovery: Learning Causality from Observations**
- **Incomplete Data: Learning and Inference (Expectation-Maximization)**

Lecture 15: EM, Unsupervised Learning, and Clustering

- Expectation-Maximization (EM) Algorithm
- Unsupervised Learning and Clustering
  - Types of unsupervised learning
    - Clustering, vector quantization
    - Feature extraction (typically, dimensionality reduction)
  - Constructive induction: unsupervised learning *in support of* supervised learning
    - Feature construction (*aka* feature extraction)
    - Cluster definition
  - Algorithms
    - EM: mixture parameter estimation (e.g., for *AutoClass*)
    - *AutoClass*: Bayesian clustering
    - Principal Components Analysis (PCA), factor analysis (FA)
    - Self-Organizing Maps (SOM): projection of data; competitive algorithm
  - Clustering problems: formation, segmentation, labeling
- Next Lecture: Time Series Learning and Characterization

Lecture 16: Introduction to Time Series Analysis

- **Introduction to Time Series**
  - 3 phases of analysis: forecasting (prediction), modeling, characterization
  - Probability and time series: stochastic processes
  - Linear models: ARMA models, approximation with temporal ANNs
  - Time series understanding and learning
    - Understanding: state-space reconstruction by delay-space embedding
    - Learning: parameter estimation (e.g., using temporal ANNs)
- **Further Reading**
  - Analysis: Box *et al*, 1994; Chatfield, 1996; Kantz and Schreiber, 1997
  - Learning: Gershenfeld and Weigend, 1994
  - Reinforcement learning: next…
- **Next Lecture: Policy Learning, Markov Decision Processes (MDPs)**
  - Read Chapter 17, Russell and Norvig, Sections 13.1-13.2, Mitchell
  - Exercise: 16.1(a), Russell and Norvig (bring answers to class; don't peek!)

Lecture 17: Policy Learning and MDPs

- **Making Decisions in Uncertain Environments**
  - Framework: Markov Decision Processes, Markov Decision Problems (MDPs)
  - Computing policies
    - Solving MDPs by dynamic programming given a *stepwise reward*
    - Methods: value iteration, policy iteration
  - Decision-theoretic agents
    - Decision cycle, Kalman filtering
    - Sensor fusion (*aka* data fusion)
  - Dynamic Bayesian networks (DBNs) and dynamic decision networks (DDNs)
- **Learning Problem**
  - Mapping from observed actions and rewards to decision models
  - Rewards/penalties: reinforcements
- **Next Lecture: Reinforcement Learning**
  - Basic model: *passive learning in a known environment*
  - *Q* learning: policy learning by adaptive dynamic programming (ADP)

Lecture 18: Introduction to Reinforcement Learning

- Control Learning
  - Learning policies from <*state*, *reward*, *action*> observations
  - Objective: choose optimal actions given new percepts and incremental rewards
  - Issues
    - Delayed reward
    - Active learning opportunities
    - Partial observability
    - Reuse of sensors, effectors
- *Q* Learning
  - Action-value function *Q* : *state* ? *action* ? *value* (expected utility)
  - Training rule
  - Dynamic programming algorithm
  - *Q* learning for deterministic worlds
  - Convergence to true *Q*
  - Generalizing *Q* learning to nondeterministic worlds
- Next Week: More Reinforcement Learning (Temporal Differences)

Lecture 19: More Reinforcement Learning (TD)

- Reinforcement Learning (RL)
  - Definition: learning policies ? : *state* ? *action* from <<*state*, *action*>, *reward*>
    - Markov decision problems (MDPs): finding control policies to choose optimal actions
    - *Q*-learning: produces action-value function *Q* : *state* ? *action* ? *value* (expected utility)
  - Active learning: experimentation (exploration) strategies
    - Exploration function: *f*(*u*, *n*)
    - Tradeoff: greed (*u*) preference versus novelty (1 / *n*) preference, *aka* curiosity
- Temporal Diffference (TD) Learning
  - ? : constant for blending alternative training estimates from multi-step lookahead
  - *TD*(?): algorithm that uses recursive training rule with ?-estimates
- Generalization in RL
  - Explicit representation: tabular representation of *U*, *M*, *R*, *Q*
  - Implicit representation: compact (*aka* compressed) representation

Lecture 20: Neural Computation

- Review: Feedforward Artificial Neural Networks (ANNs)
- Advanced ANN Topics
  - Models
    - Modular ANNs
    - Associative memories
    - Boltzmann machines
  - Applications
    - Pattern recognition and scene analysis (image processing)
    - Signal processing
    - Neural reinforcement learning
- Relation to Bayesian Networks and Genetic Algorithms (GAs)
  - Bayesian networks as a species of connectionist model
  - Simulated annealing and GAs: MCMC methods
  - Numerical ("subsymbolic") and symbolic AI systems: principled integration
- Next Week: Combining Classifiers (WM, Bagging, Stacking, Boosting)

Lecture 21: Combiners (WM, Bagging, Stacking)

- Combining Classifiers
  - Problem definition and motivation: improving accuracy in concept learning
  - General framework: collection of weak classifiers to be improved (data fusion)
- Weighted Majority (WM)
  - Weighting system for collection of algorithms
    - Weights each algorithm *in proportion to its training set accuracy*
    - Use this weight in performance element (and on test set predictions)
  - Mistake bound for WM
- Bootstrap Aggregating (Bagging)
  - Voting system for collection of algorithms
  - Training set for each member: sampled with replacement
  - Works for unstable inducers
- Stacked Generalization (*aka* Stacking)
  - Hierarchical system for combining inducers (ANNs or other inducers)
  - Training sets for "leaves": sampled with replacement; combiner: validation set
- Next Lecture: Boosting the Margin, Hierarchical Mixtures of Experts

Lecture 22:
More Combiners (Boosting, Mixture Models)

- Committee Machines *aka* Combiners
- Static Structures (Single-Pass)
  - Ensemble averaging
    - For improving weak (especially unstable) classifiers
    - e.g., weighted majority, bagging, stacking
  - Boosting the margin
    - Improve performance of any inducer: weight examples to emphasize errors
    - Variants: filtering (*aka* consensus), resampling (*aka* subsampling), reweighting
- Dynamic Structures (Multi-Pass)
  - Mixture of experts: training in combiner inducer (*aka* gating network)
  - Hierarchical mixtures of experts: hierarchy of inducers, combiners
- Mixture Model (*aka* Mixture of Experts)
  - Estimation of mixture coefficients (i.e., weights)
  - Hierarchical Mixtures of Experts (HME): multiple combiner (gating) levels
- Next Week: Intro to GAs, GP (9.1-9.4, Mitchell; 1, 6.1-6.5, Goldberg)

Lecture 23: Introduction to Genetic Algorithms (GAs)

- Evolutionary Computation
  - Motivation: process of natural selection
    - Limited population; individuals compete for membership
    - Method for parallelizing and stochastic search
  - Framework for problem solving: *search, optimization, learning*
- Prototypical (Simple) Genetic Algorithm (GA)
  - Steps
    - Selection: reproduce individuals probabilistically, in proportion to fitness
    - Crossover: generate new individuals probabilistically, from pairs of "parents"
    - Mutation: modify structure of individual randomly
  - How to represent hypotheses as individuals in GAs
- An Example: GA-Based Inductive Learning (GABIL)
- Schema Theorem: Propagation of Building Blocks
- Next Lecture: Genetic Programming, The Movie

Lecture 24: Introduction to Genetic Programming (GP)

- **Genetic Programming (GP)**
  - Objective: program synthesis
  - Application of evolutionary computation (especially genetic algorithms)
    - Search algorithms
    - Based on mechanics of natural selection, natural genetics
  - *Design application*
- **Steps in GP Design**
  - Terminal set: program variables
  - Function set: operators and macros
  - Fitness cases: evaluation environment (compare: validation tests in software engineering)
  - Control parameters: "runtime" configuration variables for GA (population size and organization, number of generations, syntactic constraints)
  - Termination criterion and result designation: when to stop, what to return
- **Next Week: Instance-Based Learning (IBL)**

Lecture 25: Instance-Based Learning (IBL)

- Instance Based Learning (IBL)
  - *k*-Nearest Neighbor (*k*-NN) algorithms
    - When to consider: few continuous valued attributes (low dimensionality)
    - Variants: distance-weighted *k*-NN; *k*-NN with attribute subset selection
  - Locally-weighted regression: function approximation method, generalizes *k*-NN
  - Radial-Basis Function (RBF) networks
    - Different kind of artificial neural network (ANN)
    - Linear combination of local approximation ? global approximation to *f*(?)
- Case-Based Reasoning (CBR) Case Study: CADET
  - Relation to IBL
  - CBR online resource page: http://www.ai-cbr.org
- Lazy and Eager Learning
- Next Week
  - Rule learning and extraction
  - Inductive logic programming (ILP)

Lecture 26:
Rule Learning and Extraction

- Learning Rules from Data
- Sequential Covering Algorithms
  - Learning single rules by search
    - Beam search
    - Alternative covering methods
  - Learning rule sets
- First-Order Rules
  - Learning single first-order rules
    - Representation: first-order Horn clauses
    - Extending *Sequential-Covering* and Learn-One-Rule: variables in rule preconditions
  - FOIL: learning first-order rule sets
    - Idea: inducing logical rules from observed relations
    - Guiding search in FOIL
    - Learning recursive rule sets
- Next Time: Inductive Logic Programming (ILP)

Lecture 27: Inductive Logic Programming (ILP)

- Induction as Inverse of Deduction
  - Problem of induction revisited
    - Definition of induction
    - Inductive learning as specific case
    - Role of induction, deduction in automated reasoning
  - Operators for automated deductive inference
    - Resolution rule (*and operator*) for deduction
    - First-order predicate calculus (FOPC) and resolution theorem proving
  - Inverting resolution
    - Propositional case
    - First-order case (inverse entailment operator)
- Inductive Logic Programming (ILP)
  - Cigol: inverse entailment (very susceptible to combinatorial explosion)
  - Progol: sequential covering, general-to-specific search using inverse entailment
- Next Week: Knowledge Discovery in Databases (KDD), Final Review

Lecture 28:
KDD and Data Mining

- Knowledge Discovery in Databases (KDD) and Data Mining
  - Stages: selection (filtering), processing, transformation, learning, inference
  - Design and implementation issues
- Role of Machine Learning and Inference in Data Mining
  - Roles of unsupervised, supervised learning in KDD
  - Decision support (information retrieval, prediction, policy optimization)
- Case Studies
  - Risk analysis, transaction monitoring (filtering), prognostic monitoring
  - Applications: business decision support (pricing, fraud detection), automation
- Resources Online
  - Microsoft DMX Group (Fayyad): http://research.microsoft.com/research/DMX/
  - KSU KDD Lab (Hsu): http://ringil.cis.ksu.edu/KDD/
  - CMU KDD Lab (Mitchell): http://www.cs.cmu.edu/~cald
  - KD Nuggets (Piatetsky-Shapiro): http://www.kdnuggets.com
  - NCSA Automated Learning Group (Welge)
    - ALG home page: http://www.ncsa.uiuc.edu/STI/ALG
    - NCSA *D2K*: http://chili.ncsa.uiuc.edu

Meta-Summary

- **Machine Learning Formalisms**
  - Theory of computation: PAC, mistake bounds
  - Statistical, probabilistic: PAC, confidence intervals
- **Machine Learning Techniques**
  - Models: version space, decision tree, perceptron, winnow, ANN, BBN, SOM, *Q* functions, GA/GP building blocks (schemata), GP building blocks
  - Algorithms: candidate elimination, *ID3*, backprop, MLE, Simple (Naïve) Bayes, *K2*, EM, SOM convergence, LVQ, ADP, *Q*-learning, TD(?), simulated annealing, sGA
- **Final Exam Study Guide**
  - Know
    - Definitions (terminology)
    - How to solve problems from Homeworks 1 and 3 (problem sets)
    - How algorithms in Homeworks 2, 4, and 5 (machine problems) work
  - Practice
    - Sample exam problems (handout)
    - Example runs of algorithms in Mitchell, lecture notes
  - Don't panic!