Lecture 0

A Brief Survey of Machine Learning

Tuesday, August 24, 1999

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

Readings:
Chapter 1, Mitchell

Lecture Outline

- Course Information: Format, Exams, Homework, Programming, Grading
- Class Resources: Course Notes, Web/News/Mail, Reserve Texts
- Overview
  - Topics covered
  - Applications
  - Why machine learning?
- 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

Course Information and Administrivia

- Instructor: William H. Hsu
  - E-mail: bhsu@cis.ksu.edu
  - Phone: (785) 532-6350 (office), (785) 539-7180 (home)
  - Office hours: after class; 1-3pm Monday, 9-11am Wednesday; by appointment
- Grading
  - Assignments (5): 50%, paper summaries (13): 15%, midterm: 15%, final: 20%
  - Lowest homework score and lowest 3 paper summary scores dropped
- Homework
  - Five assignments: written (3) and programming (2)
  - Late policy: due on Thursdays; free extension to following Tuesday (*if needed by due date*); -10% credit per day thereafter
  - Cheating: don't do it; see class web page for policy
- Project Option
  - 1-hour project option for graduate students
  - Term paper or semester research project
  - Sign up by September 23, 1999 if interested (see class web page)

Class Resources

- Web Page (Required)
  - http://ringil.cis.ksu.edu/Courses/Fall-1999/CIS798
  - Lecture notes (MS PowerPoint 97, PostScript)
  - Homeworks (MS Word 97, PostScript)
  - Exam and homework solutions (MS Word 97, PostScript)
  - Class announcements (students responsibility to follow) and grade postings
- Course Notes at Copy Center (Required)
- Newsgroup (Recommended)
  - ksu.class.cis798whh
  - Mirror of announcements from class web page
  - Discussions (instructor and other students)
  - Dated research announcements (seminars, conferences, calls for papers)
- Mailing List (Optional)
  - CIS798WHH-L@cis.ksu.edu
  - Sign-up sheet (if interested)
  - Reminders, related research announcements

Course Overview

- Learning Algorithms and Models
  - Models: decision trees, linear threshold units (winnow, weighted majority), neural networks, Bayesian networks (polytrees, belief networks, influence diagrams, HMMs), genetic algorithms, instance-based (nearest-neighbor)
  - Algorithms (e.g., for decision trees): ID3, C4.5, CART, OC1
  - Methodologies: supervised, unsupervised, reinforcement; knowledge-guided
- Theory of Learning
  - Computational learning theory (COLT): complexity, limitations of learning
  - Probably Approximately Correct (PAC) learning
  - Probabilistic, statistical, information theoretic results
- Multistrategy Learning: Combining Techniques, Knowledge Sources
- Data: Time Series, Very Large Databases (VLDB), Text Corpora
- Applications
  - Performance element: classification, decision support, planning, control
  - Database mining and knowledge discovery in databases (KDD)
  - Computer inference: learning to reason

Why Machine Learning?

- New Computational Capability
  - Database mining: converting (technical) records into knowledge
  - Self-customizing programs: learning news filters, adaptive monitors
  - Learning to act: robot planning, control optimization, decision support
  - Applications that are hard to program: automated driving, speech recognition
- Better Understanding of Human Learning and Teaching
  - Cognitive science: theories of knowledge acquisition (e.g., through practice)
  - Performance elements: reasoning (inference) and *recommender* systems
- Time is Right
  - Recent progress in algorithms and theory
  - Rapidly growing volume of online data from various sources
  - Available computational power
  - Growth and interest of learning-based industries (e.g., data mining/KDD)

Rule and Decision Tree Learning

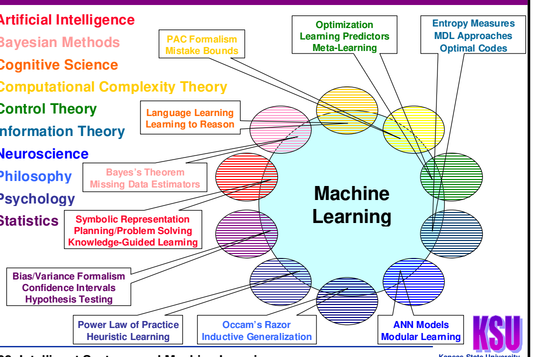
- Example: Rule Acquisition from Historical Data
- Data
  - Patient 103 (time = 1): Age 23, First-Pregnancy: no, Anemia: no, Diabetes: no, Previous-Premature-Birth: no, Ultrasound: *unknown*, Elective C-Section: *unknown*, Emergency-C-Section: *unknown*
  - Patient 103 (time = 2): Age 23, First-Pregnancy: no, Anemia: no, Diabetes: yes, Previous-Premature-Birth: no, Ultrasound: abnormal, Elective C-Section: no, Emergency-C-Section: *unknown*
  - Patient 103 (time = n): Age 23, First-Pregnancy: no, Anemia: no, Diabetes: no, Previous-Premature-Birth: no, Ultrasound: *unknown*, Elective C-Section: no, Emergency-C-Section: YES
- Learned Rule
  - IF *no previous vaginal delivery*, AND *abnormal 2nd trimester ultrasound*, AND *malpresentation at admission*, AND *no elective C-Section*
    THEN *probability of emergency C-Section is 0.6*
  - Training set: 26/41 = 0.634
  - Test set: 12/20 = 0.600

Neural Network Learning

- Autonomous Learning Vehicle In a Neural Net (ALVINN): Pomerleau *et al*
  - http://www.cs.cmu.edu/afs/cs/project/alv/member/www/projects/ALVINN.html
  - Drives 70mph on highways

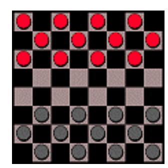
Relevant Disciplines

- Artificial Intelligence
- Bayesian Methods
- Cognitive Science
- Computational Complexity Theory
- Control Theory
- Information Theory
- Neuroscience
- Philosophy
- Psychology
- Statistics



Specifying A Learning Problem

- Learning = Improving with Experience at Some Task
  - Improve over task *T*,
  - with respect to performance measure *P*,
  - based on experience *E*.
- Example: Learning to Play Checkers
  - *T*: play games of checkers
  - *P*: percent of games won in world tournament
  - *E*: opportunity to play against self



- Refining the Problem Specification: Issues
  - What experience?
  - What *exactly* should be learned?
  - How shall it be *represented*?
  - What specific algorithm to learn it?
- Defining the Problem Milieu
  - Performance element: How shall the results of learning be applied?
  - How shall the performance element be evaluated? The learning system?

Example: Learning to Play Checkers

- Type of Training Experience
  - Direct or indirect?
  - Teacher or not?
  - Knowledge about the game (e.g., openings/endgames)?
- Problem: Is Training Experience *Representative* (of Performance Goal)?
- Software Design
  - Assumptions of the learning system: *legal* move generator exists
  - Software requirements: generator, evaluator(s), parametric target function
- Choosing a Target Function
  - *ChooseMove: Board → Move* (action selection function, or *policy*)
  - *V: Board → R* (board evaluation function)
  - Ideal target *V*; approximated target $\hat{}$
  - Goal of learning process: operational description (approximation) of *V*

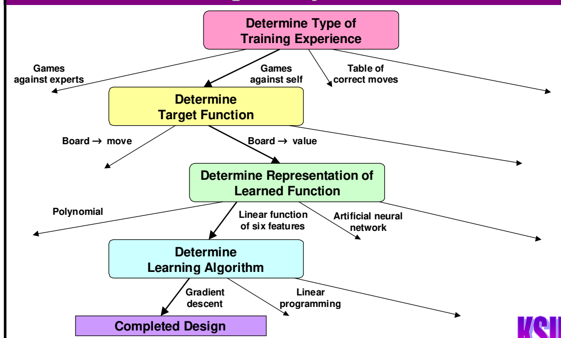
A Target Function for Learning to Play Checkers

- Possible Definition
  - If *b* is a final board state that is won, then *V(b)* = 100
  - If *b* is a final board state that is lost, then *V(b)* = -100
  - If *b* is a final board state that is drawn, then *V(b)* = 0
  - If *b* is not a final board state in the game, then *V(b)* = *V(b')* where *b'* is the best final board state that can be achieved starting from *b* and playing optimally until the end of the game
  - *Correct values, but not operational*
- Choosing a Representation for the Target Function
  - Collection of rules?
  - Neural network?
  - Polynomial function (e.g., linear, quadratic combination) of board features?
  - Other?
- A Representation for Learned Function
  - $\hat{V}(b) = w_0 + w_1 bp(b) + w_2 rp(b) + w_3 bk(b) + w_4 rk(b) + w_5 bt(b) + w_6 rt(b)$
  - bp/rp = number of black/red pieces; bk/rk = number of black/red kings; bt/rt = number of black/red pieces *threatened* (can be taken on next turn)

A Training Procedure for Learning to Play Checkers

- Obtaining Training Examples
  - $V(b)$ the target function
  - $\hat{V}(b)$ the learned function
  - $V_{train}(b)$ the training value
- One Rule For Estimating Training Values:
  - $V_{train}(b) \leftarrow \hat{V}(Successor(b))$
- Choose Weight Tuning Rule
  - Least Mean Square (LMS) weight update rule:
    REPEAT
    - Select a training example *b* at random
    - Compute the *error(b)* for this training example
      $error(b) = V_{train}(b) - \hat{V}(b)$
    - For each board feature f_i, update weight w_i as follows:
      $w_i \leftarrow w_i + c \cdot f_i \cdot error(b)$
      where *c* is a small, constant factor to adjust the learning rate

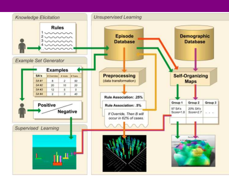
Design Choices for Learning to Play Checkers



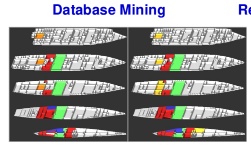
Some Issues in Machine Learning

- What Algorithms Can Approximate Functions Well? When?
- How Do Learning System Design Factors Influence Accuracy?
  - Number of training examples
  - Complexity of *hypothesis representation*
- How Do Learning Problem Characteristics Influence Accuracy?
  - Noisy data
  - Multiple data sources
- What Are The Theoretical Limits of Learnability?
- How Can Prior Knowledge of Learner Help?
- What Clues Can We Get From Biological Learning Systems?
- How Can Systems Alter Their Own Representation?

Interesting Applications


NCSA *D2K* - http://www.ncsa.uiuc.edu/STI/ALG
Database Mining

Cartia *ThemeScapes* - http://www.cartia.com
Reasoning (Inference, Decision Support)


DC-ARM - http://www-kbs.ai.uiuc.edu

Normal
Ignited
Engulfed
Destroyed
Extinguished
Fire Alarm
Flooding

Planning, Control