



Lecture 0 of 42

Machine Learning / Advanced Topics in AI Course Organization and Survey

Friday, 18 January 2008

William H. Hsu

Department of Computing and Information Sciences, KSU

KSOL course pages: <http://snipurl.com/1y5gc> / <http://snipurl.com/1y5ih>
Course web site: <http://www.kddresearch.org/Courses/Spring-2008/CIS732>
Instructor home page: <http://www.cis.ksu.edu/~bhsu>

Reading for Next Class:
Syllabus and Course Intro
Handout: Chapters 1-2, Mitchell



Course Administration

- **Course Pages (KSOL):** <http://snipurl.com/1y5gc> / <http://snipurl.com/1y5ih>
- **Class Web Page:** www.kddresearch.org/Courses/Spring-2008/CIS732
- **Instructional E-Mail Addresses (for Topics in AI, substitute 830 for 732)**
 - * CIS732TA-L@listserv.ksu.edu (always use this to reach instructor and TA)
 - * CIS732-L@listserv.ksu.edu (this goes to everyone)
- **Instructor: William Hsu, Nichols 213**
 - * Office phone: +1 785 532 7905; home phone: +1 785 539 7180
 - * IM: AIM/MSN/YIM [hsuw@rizenabsith](#), ICQ 28651394/191317559, Google [banazir](#)
 - * Office hours: after class Mon/Wed/Fri; other times by appointment
- **Graduate Teaching Assistant: Jing Xia**
 - * Office location: Nichols 213a
 - * Office hours: to be announced on class web board
- **Grading Policy**
 - * Midterm: 15% (in-class, open-book); final (take-home): 20%
 - * Machine problems, problem sets (6 of 8): 30%; term project: 20%
 - * Paper reviews (10 of 12): 10%; class participation: 5% (HW, Q&A)





Class Resources

- **Web Page (Required)**
 - * <http://www.kddresearch.org/Courses/Spring-2008/CIS732>
 - * Lecture notes (MS PowerPoint 97-2003, PDF)
 - * Homeworks (MS PowerPoint 97-2003, PDF)
 - * Exam and homework solutions (MS PowerPoint 97-2003, PDF)
 - * Class announcements (students' responsibility) and grade postings
- **Course Notes at Copy Center (Required)**
- **Course Web Group**
 - * Mirror of announcements from class web page
 - * Discussions (instructor and other students)
 - * Dated research announcements (seminars, conferences, calls for papers)
- **Mailing List (Automatic)**
 - * CIS732-L@listserv.ksu.edu
 - * Sign-up sheet
 - * Reminders, related research announcements



Course Overview

- **Learning Algorithms and Models**
 - * **Models:** decision trees, winnow, artificial neural networks, naïve Bayes, genetic algorithms (GAs) and genetic programming (GP), instance-based learning (nearest-neighbor), inductive logic programming (ILP)
 - * **Algorithms:** for decision trees (ID3/C4.5/J48), ANNs (backprop), etc.
 - * **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 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, interest in 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: *unknown*, Elective C-Section: no, Emergency-C-Section: *unknown*
 - * Patient 103 (time = n): Age 23, First-Pregnancy: no, Anemia: no, Diabetes: yes, Previous-Premature-Birth: no, Ultrasound: abnormal, 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.***
 - ★ NAVLAB: <http://www.cs.cmu.edu/afs/cs/project/alv/www/index.html>
 - ★ Drives 70mph on highways

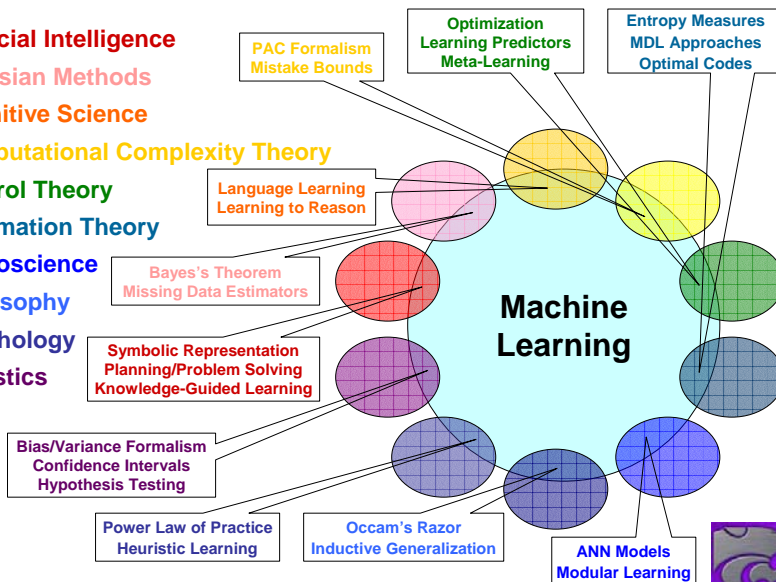


© 1999, 2001 Carnegie Mellon University



Relevant Disciplines

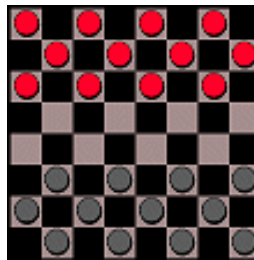
- **Artificial Intelligence**
- **Bayesian Methods**
- **Cognitive Science**
- **Computational Complexity Theory**
- **Control Theory**
- **Information Theory**
- **Neuroscience**
- **Philosophy**
- **Psychology**
- **Statistics**





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 tournament play
 - * 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 results of learning be applied?
 - * How shall performance element be evaluated? 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*: $Board \rightarrow Move$ – action selection function, or *policy*
 - * V : $Board \rightarrow R$ – evaluation function for game tree search (minimax / α - β)
 - * Ideal target V ; approximated target \hat{V}
 - * Goal of learning process: operational description (approximation) of V
- **Chinook: Checkers Solved by Game Tree Search (July 2007)**
- **Reference: <http://en.wikipedia.org/wiki/Checkers>**





A Target Function for Learning to Play Checkers

- **Possible Definition**
 - * If b is final board state that is won, then $V(b) = +100$ (or MAXINT)
 - * If b is final board state that is lost, then $V(b) = -100$ (or MAXINT)
 - * If b is final board state that is drawn, then $V(b) = 0$
 - * If b is not final board state in the game, then $V(b) = V(b')$ where b' is best final board state that can be achieved starting from b and playing optimally until end
 - * *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**
 - * $\hat{V}(b) = w_0 + w_1bp(b) + w_2rp(b) + w_3bk(b) + w_4rk(b) + w_5bt(b) + w_6rt(b)$
 - * 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)



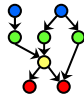
A Training Procedure for Learning to Play Checkers

- **Obtaining Training Examples**
 - * $V(b)$ target function
 - * $\hat{V}(b)$ learned function
 - * $V_{train}(b)$ training value ("signal")
 - **Rule For Estimating Training Values** $V_{train}(b) \leftarrow \hat{V}(Successor(b))$
 - **Rule for Training (Weight Tuning)**
 - * Least Mean Square (LMS) weight update rule
 - * REPEAT
 - Select training example b at random
 - Compute the *error*(b) for this training example

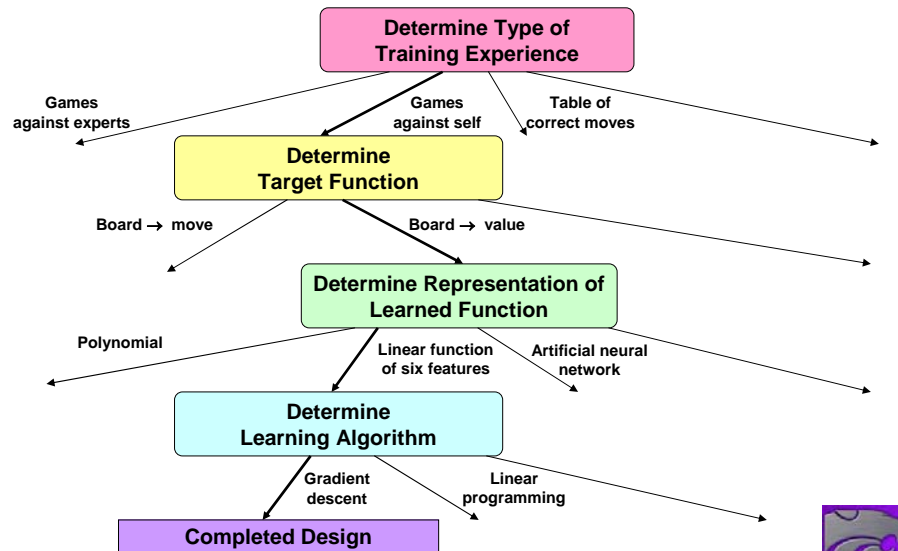
$$error(b) = V_{train}(b) - \hat{V}(b)$$
 - For each board feature f_i , update weight w_i as follows

$$w_i \leftarrow w_i + c \cdot f_i \cdot error(b)$$
- where c is small, constant factor to adjust learning rate



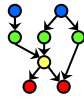


Design Choices for Learning to Play Checkers

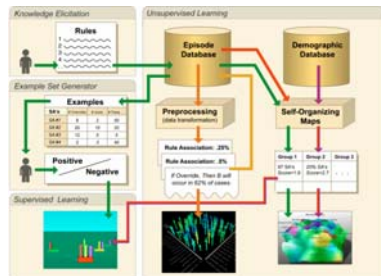


Some Issues in Machine Learning

- **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 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



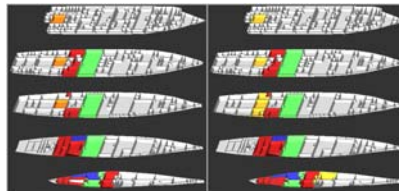
NCSA D2K - <http://alg.ncsa.uiuc.edu>

Database Mining



Clustering (Cartia ThemeScapes) - <http://snurl.com/1y511>

Reasoning (Inference, Decision Support)



Normal	Destroyed
Ignited	Extinguished
Engulfed	Fire Alarm
	Flooding

Planning, Control

DC-ARM - <http://www.stanford.edu/~dwilkins/members.htm>



What to Learn?

- **Classification Functions**
 - * Learning hidden functions: estimating ("fitting") parameters
 - * Concept learning (e.g., chair, face, game)
 - * Diagnosis, prognosis: risk assessment, medical monitoring, security, ERP
- **Models**
 - * Map (for navigation)
 - * Distribution (query answering, aka QA)
 - * Language model (e.g., automaton/grammar)
- **Skills**
 - * Playing games
 - * Planning
 - * Reasoning (acquiring representation to use in reasoning)
- **Cluster Definitions for Pattern Recognition**
 - * Shapes of objects
 - * Functional or taxonomic definition
- **Many Problems Can Be Reduced to Classification**



How to Learn It?

- **Supervised**

- * What is learned? **Classification function; other models**
- * Inputs and outputs? **Learning:** examples $\langle x, f(x) \rangle \rightarrow$ approximation $\hat{f}(x)$
- * How is it learned? **Presentation of examples to learner (by teacher)**

- **Unsupervised**

- * **Cluster definition, or *vector quantization* function (*codebook*)**
- * **Learning:** observations $x \times$ distance metric $d(x_1, x_2) \rightarrow$ discrete codebook $f(x)$
- * **Formation, segmentation, labeling of clusters based on observations, metric**

- **Reinforcement**

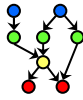
- * **Control policy (function from states of the world to actions)**
- * **Learning:** state/reward sequence $\langle \langle s_i, r_i \rangle : 1 \leq i \leq n \rangle \rightarrow$ policy $p: s \rightarrow a$
- * **(Delayed) feedback of reward values to agent based on actions**
- * **Model updated based on reward, (partially) observable state**



Supervised Inductive Learning: Classification and Regression

- **Given: Training Examples $\langle x, f(x) \rangle$ of Some Unknown Function f**
- **Find: A Good Approximation to f**
- **Examples (besides Concept Learning)**
 - * **Disease diagnosis**
 - x = properties of patient (medical history, symptoms, lab tests)
 - f = disease (or recommended therapy)
 - * **Risk assessment**
 - x = properties of consumer, policyholder (demographics, accident history)
 - f = risk level (expected cost)
 - * **Automatic steering**
 - x = bitmap picture of road surface in front of vehicle
 - f = degrees to turn the steering wheel
 - * **Part-of-speech tagging**
 - * **Computer security: fraud/intrusion detection, attack graphs**
 - * **Information extraction: clusters of documents**
 - * **Social networks and weblogs: predicting links, sentiment analysis**
 - * **Multisensor integration and prediction**





Learning and Types [1]: A Generic Supervised Learning Problem



Example	x_1	x_2	x_3	x_4	y
0	0	1	1	0	0
1	0	0	0	0	0
2	0	0	1	1	1
3	1	0	0	1	1
4	0	1	1	0	0
5	1	1	0	0	0
6	0	1	0	1	0

- **Input x_i : t_i , desired output y : t , "target" function f : $(t_1 \times t_2 \times t_3 \times t_4) \rightarrow t$**
- **Learning function: Vector $(t_1 \times t_2 \times t_3 \times t_4 \times t) \rightarrow (t_1 \times t_2 \times t_3 \times t_4) \rightarrow t$**



Learning and Types [2]: Unrestricted Hypothesis Space

- **$|A \rightarrow B| = |B|^{|A|}$ – proof?**
- **$|H^4 \rightarrow H| = |\{0,1\} \times \{0,1\} \times \{0,1\} \times \{0,1\} \rightarrow \{0,1\}| = 2^{2^4} = 65536$ functions**
- **Complete Ignorance: Is Learning Possible?**
 - * **Need to see every possible input/output pair**
 - * **After 7 examples, still have $2^9 = 512$ possibilities (out of 65536) for f**

Example	x_1	x_2	x_3	x_4	y
0	0	0	0	0	?
1	0	0	0	1	?
2	0	0	1	0	0
3	0	0	1	1	1
4	0	1	0	0	0
5	0	1	0	1	0
6	0	1	1	0	0
7	0	1	1	1	?
8	1	0	0	0	?
9	1	0	0	1	1
10	1	0	1	0	?
11	1	0	1	1	?
12	1	1	0	0	0
13	1	1	0	1	?
14	1	1	1	0	?
15	1	1	1	1	?





Training Examples for Concept *EnjoySport*

- **Specification for Examples**
 - * Similar to a data type definition
 - * 6 attributes: Sky, Temp, Humidity, Wind, Water, Forecast
 - * Nominal-valued (symbolic) attributes - enumerative data type
- **Binary (Boolean-Valued or H -Valued) Concept**
- **Supervised Learning Problem: *Describe the General Concept***

Example	Sky	Air Temp	Humidity	Wind	Water	Forecast	Enjoy Sport
0	Sunny	Warm	Normal	Strong	Warm	Same	Yes
1	Sunny	Warm	High	Strong	Warm	Same	Yes
2	Rainy	Cold	High	Strong	Warm	Change	No
3	Sunny	Warm	High	Strong	Cool	Change	Yes



Representing Hypotheses

- **Many Possible Representations**
- **Hypothesis h : Conjunction of Constraints on Attributes**
- **Constraint Values**
 - * Specific value (e.g., *Water = Warm*)
 - * Don't care (e.g., "*Water = ?*")
 - * No value allowed (e.g., "*Water = \emptyset* ")
- **Example Hypothesis for *EnjoySport***
 - * Sky AirTemp Humidity Wind Water Forecast
 - <Sunny ? ? Strong ? Same>
 - * Is this consistent with the training examples?
 - * What are some hypotheses that are consistent with the examples?





Summary

- **Reading: Chapters 1-2, Mitchell**
- **Suggested Exercises: 2.2, 2.3, 2.4, 2.6**
- **Taxonomy of Learning Systems**
- **Week 1: Overview, Learning from Examples**
 - * (Supervised) concept learning framework
 - * Simple approach: assumes no noise; illustrates key concepts
- **Week 2: Hypothesis Learning and Inductive Bias**
 - * Sources: Mitchell (1997) – online notes and handout
 - * Wednesday: inductive learning, version space
 - * Friday: candidate elimination algorithm, active learning, inductive bias
 - * Background concepts: partially-ordered set (poset) formalism
- **Week 3: Decision Trees, Intro Computational Learning Theory (COLT)**
 - * First paper review due Wed 30 Jan 2008
 - * Sources: Kearns & Vazirani (1994), Han & Kamber 2nd edition (2006)



Terminology

- **Learning:** Improving at Task given Performance Measure, Experience
- **Performance Element:** Part of System that Applies Result of Learning
- **Types of Learning**
 - * **Supervised:** with “teacher” (often, classification from labeled examples)
 - * **Unsupervised:** from data, using similarity measure (unlabeled instances)
 - * **Reinforcement:** “by doing”, with reward/penalty signal
- **Supervised Learning: Target Functions**
 - * **Target function** – function c or f to be learned
 - * **Target** – desired value y to be predicted (sometimes “target function”)
 - * **Example / labeled instance** – tuples of the form $\langle x, f(x) \rangle$
 - * **Classification function, classifier** – nominal-valued f (enumerated return type)
- **Clustering: Application of Unsupervised Learning**
- **Concepts and Hypotheses**
 - * **Concept** – function c from observations to TRUE or FALSE (membership)
 - * **Class label** – output of classification function
 - * **Hypothesis** – proposed function h believed to be similar to c (or f)

