# Machine Learning

Patricia J Riddle
Computer Science 367

# Textbook

- Tom M. Mitchell
  *Machine Learning*
  McGraw-Hill, New York, 1997

# Introduction

- Learn - improve automatically with experience
  - Database Mining - learning from medical records which treatments are most effective
  - Self Customizing programs -
    - houses learning to optimise energy costs based on particular usage patterns of their occupants
    - personal software assistants learning the evolving interests of their users in order to highlight relevant stories from online newspapers
  - Applications we can’t program by hand - autonomous driving - speech recognition
- Might lead to a better understanding of human learning abilities (and disabilities)

# Datamining versus Machine Learning

- Very large dataset
- Given to a person versus expert system
- Discovery versus Retrieval (only a matter of viewpoint) - retrieval plus change of representation - chess example

# Success Stories

- Learn to recognise spoken words
- Predict recovery rates of pneumonia patients
- Detect fraudulent use of credit cards
- Drive autonomous vehicles on public highways

# Success Stories II

- Play games such as backgammon at levels approaching the performance of human world champions
- Classification of astronomical structures

- References: Langley & Simon (1995) Applications of machine learning and rule induction Communications of the ACM, 38(11), 55-64
- Rumelhart, Widrow & Lehr (1994). The basic ideas in neural networks. Communications of the ACM 37(3), 87-92

# Definition of Learning

- A computer program is said to learn from experience **E**
  with respect to some class of tasks **T**
  and performance measure **P**,
  if its performance at tasks in **T**,
  as measured by **P**,
  improves with experience **E**.

# Example Learning Problems

- Handwriting Recognition:
  - T: recognising and classifying handwritten words with images
  - P: percent of words correctly classified
  - E: a database of handwritten words with given classifications

- Robot driving:
  - T: driving on public four lane highways using vision sensors
  - P: average distance traveled before an error (as judged by human overseer)
  - E: a sequence of images and steering commands recorded while observing a human driver

# Definition Continued

- Choice of P very important!! - Expert system or human comprehension? - Datamining!!

- Broad enough to include most tasks that we would call "learning tasks" but also programs that improve from experience in quite straightforward ways (rote learning or caching)!

- A database system that allows users to update data entries - it improves its performance at answering database queries, based on the experience gained from databases updates (same issue as what is intelligence)

# Designing a Learning System

- T: checkers (draughts)
- P: percent of games won in world tournament
- What experience?
- What exactly should be learned?
- How shall it be represented?
- What specific algorithm to learn it?

# Direct versus Indirect Learning

1. Individual checkers board states and correct move for each
2. Move sequences and final outcomes of various games played

- Credit assignment problem - the degree to which each move in the sequence deserves credit or blame for the final outcome - game can be lost even when early moves are optimal, if these are followed later by poor moves or vice versa

# Teacher or not?

- Degree to which learner controls the sequence of training examples

1. Teacher selects informative board states & provides the correct moves

2. For each proposed board state the learner finds particularly confusing it asks the teacher for correct move

3. Learner may have complete control as it does when it learns by playing itself with no teacher - learner may choose between experimenting with novel board states or honing its skill by playing minor variations of promising lines of play

# Choose Training Experience

- How well training experience represents the distribution of examples over which the final system performance P must be measured
- P is percent of games in the world tournament, obvious danger when E consists of only games played against itself (probably can't get world champion to teach computer!)
- Most current theories of machine learning assume that the distribution of training examples is identical to the distribution of test examples
- It is IMPORTANT to keep in mind that this assumption must often by violated in practice.
- E: play games against itself (advantage of getting a lot of data this way)

# Choose a Target Function

- ChooseMove: B -> M where B is any legal board state and M is a legal move (hopefully the "best" legal move)

- Alternatively, function V: B -> $\Re$ which maps from B to some real value where higher scores are assigned to better board states

- Now use the legal moves to generate every subsequent board state and use V to choose the best one and therefore the best legal move

# Choose a Target Function II

- $V(b) = 100$, if b is a final board state that is won
- $V(b) = -100$, if b is a final board state that is lost
- $V(b) = 0$, if b is a final board state that is a draw
- $V(b) = V(b')$, if b is not a final state where b´ is the best final board state starting from b assuming both players play optimally

- Not computable!! - non-operational definition (changes over time!!! - Deep Blue)
- Need Operational V - What are Realistic Time Bounds??
- May be difficult to learn an operational form of V perfectly - Function Approximation $V^{hat}$

# Choose Representation for Target Function

- Use a large table with an entry specifying a value for each distinct board state
- Collection of rules that match against features of the board state
- Quadratic polynomial function of predefined board features
- Artificial neural network
- NOTICE - choice of representation is closely tied to algorithm choice!!

# Expressability Tradeoff

- Very expressive representations allow close approximations to the ideal target function V, but the more expressive the representation the more training data the program will require in order to choose among the alternative hypothesis

- Also depending on the purpose, a more expressive representation might make it more or less easy for people to understand!

# Choose SIMPLE Representation

- We choose: a linear combination of
  - $X_1$ the number of black pieces on the board
  - $X_2$ the number of red pieces on the board
  - $X_3$ the number of black kings on the board
  - $X_4$ the number of red kinds on the board
  - $X_5$ the number of black pieces threatened by red (which can be captured on red's next turn)
  - $X_6$ the number of red pieces threatened by black

- $V'(b) = w_0+w_1x_1+w_2x_2+w_3x_3+w_4x_4+w_5x_5+w_6x_6$
  - where $w_0$ through $w_6$ are numerical coefficients or weights to be chosen by the learning algorithm

# Design So Far

- T: Checkers
- P: percent of games won in world tournament
- E: games played against self
- V: Board -> $\Re$
- Target Function Representation:
  $V'(b) = w_0 + w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + w_5x_5 + w_6x_6$

# Choose Function Approximation Algorithm

- First need Set of training examples
  - $<b, V_{train}(b)>$
  - $<(x_1=3, x_2=0, x_3=1, x_4=0, x_5=0, x_6=0), +100>$ because $x_2=0$

- $V_{train}(b) <- V'(successor(b))$
  - Good if V´ tends to be more accurate for board positions closer to game's end

# Choose Learning Algorithm

- Learning Algorithm for choosing weights $w_i$ to best fit the set of training examples

  $\{<b, V_{train}(b)>\} \equiv \{<b, V'(Successor(b))>\}$

- Best fit could be defined as minimizes the squared error E

$$E \equiv \sum_{<b,V_{train}(b)>\in training-examples} (V_{train}(b) - V'(b))^2$$

$$E \equiv \sum_{<b,sucessor(b)>\in training-examples} (V'(successor(b)) - V'(b))^2$$

# Choose learning Algorithm II

- We seek the weights that minimise E for the observed training examples

- We need an algorithm that incrementally refines the weights as new training examples become available & is robust to errors in estimated training values

- One such algorithm is LMS (basis of Neural Network algorithms)

# Least Mean Squares

- LMS adjusts the weights a small amount in the direction that reduces the error on this training example

- Stochastic gradient-descent search through the space of possible hypothesis to minimize the squared error
  – Why stochastic ???

# LMS Algorithm

- LMS: For each <b,$V_{train}$(b)> use current weights to calculate V´(b). For each weight

$$w_i \leftarrow w_i + \eta(V_{train}(b) - V'(b))x_i$$

- Where η is a small constant .01 that moderates the size of the weight update

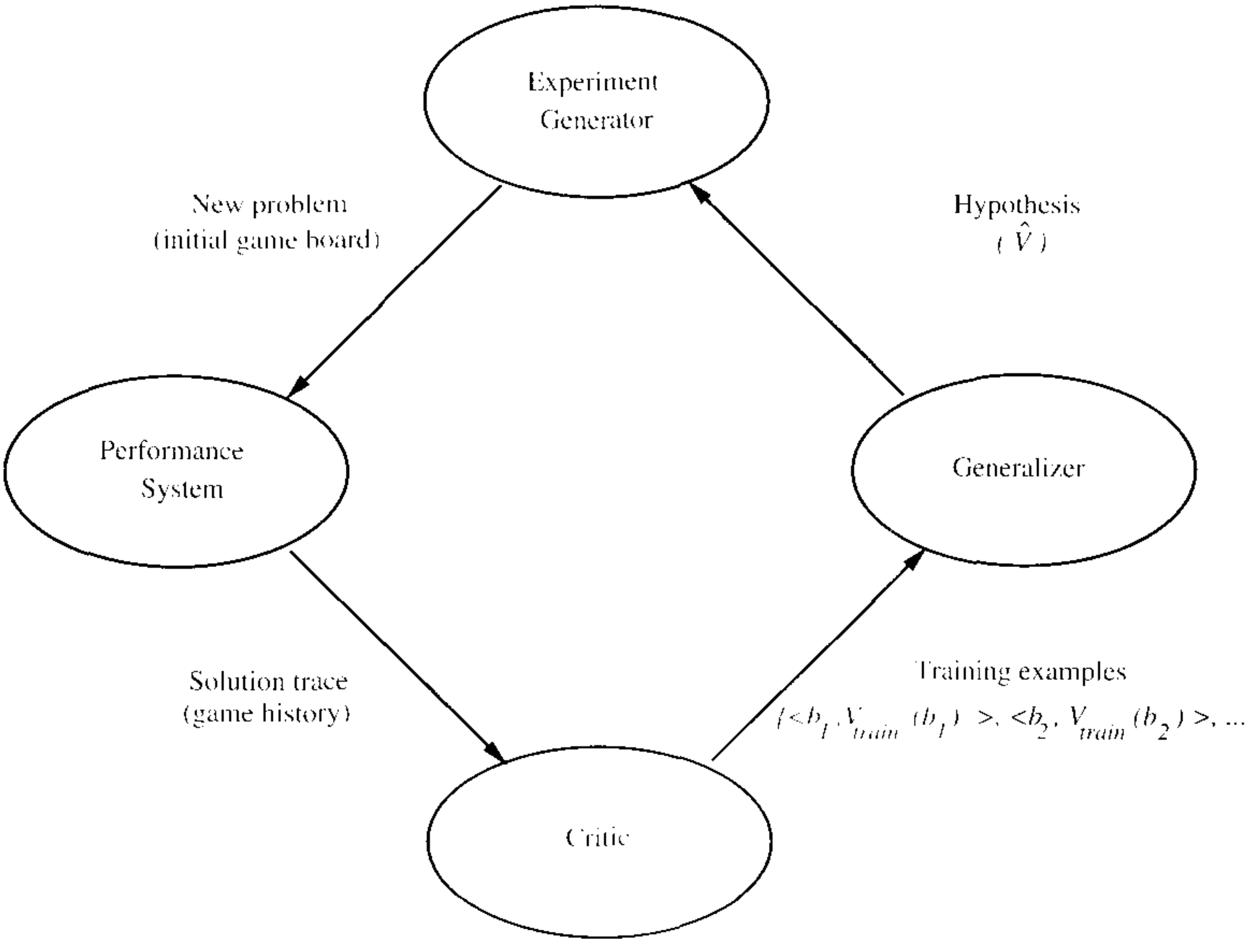
$$w_i \leftarrow w_i + \eta(V'(sucessor(b)) - V'(b))x_i$$

# LMS Intuition

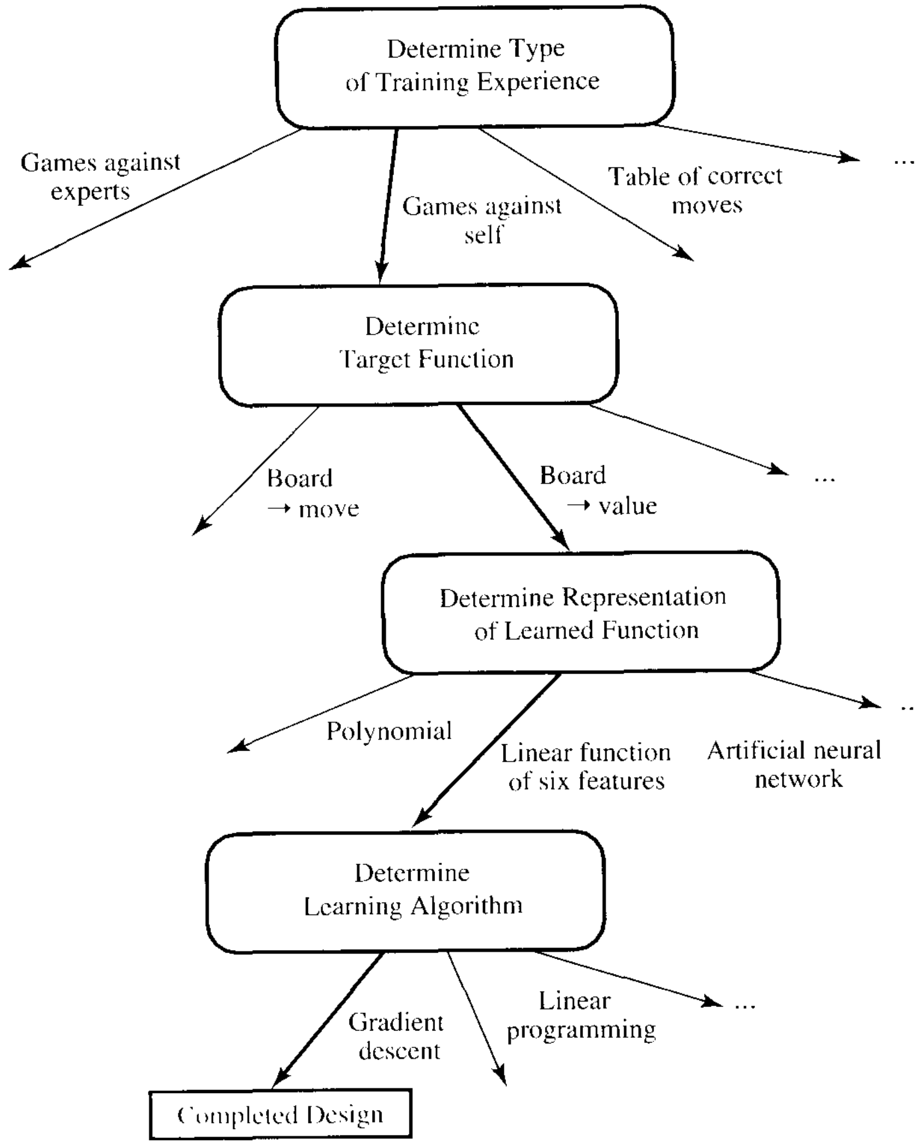
- To get an intuitive understanding notice that when the error is 0 no weights are changed, when it is positive then each weight is increased in proportion to the value of its corresponding feature

- Surprisingly, in certain settings this simple method can be proven to converge to the least squared approximation to $V_{train}$.
  - In how many training instances?
  - How understandable is the result? (Datamining)

# Design Methodology



# Design Choices



# Summary of Design Choices

- Constrained the learning task
- Single linear evaluation function
- Six specific board features

- If the true function can be represented this way we are golden, otherwise sunk

- Even if it can be represented, our learning algorithm might miss it!!!!
- Very few guarantees (some COLT) but pretty good empirically (like Quicksort)

- Our approach probably not good enough, but a similar approach worked for backgammon with a whole board representation and training on over 1 million games

# Other Approaches

- Store the training examples and pick closet - nearest neighbor
- Generate a large number of checker programs and have them play each other, keeping the most successful and elaborating and mutating them in a kind of simulated evolution - genetic algorithms
- Analyze or explain to themselves reasons for specific success or failures - explanation-based learning

# Learning as Search

- Search a very large space of possible hypothesis to find one that best fits the observed data

- For example, hypothesis space consists of all evaluation functions that can be represented by some choice of values for w0…w6
- The learner searches through this space to locate the hypothesis which is most consistent with the available training examples

- Choice of target function defines hypothesis space and therefore the algorithms which can be used.

- As soon as space is small enough just test them all chess -> tic-tac-toe

# Research Issues

- What algorithms perform best for which type of problems and representations?
- How much training data is sufficient?
- How can prior knowledge be used?
- How can you choose a useful next training experience?
- How does noisy data influence accuracy?
- How do you reduce a learning problem to a set of function approximations?
- How can the learner automatically alter its representation to improve its ability to represent and learn the target function?

# Summary

- Machine Learning is useful for
  - Datamining (credit worthiness)
  - Poorly understood domains (face recognition)
  - Programs that must dynamically adapt to changing conditions (Internet)
- Machine Learning draws on many diverse disciplines:
  - Artificial Intelligence
  - Probability and Statistics
  - Computational Complexity
  - Information Theory
  - Psychology and Neurobiology
  - Control Theory
  - Philosophy

# Summary II

- Learning problem needs well-specified task, performing metric, and source of training experience.

- Machine Learning approach involves a number of design choices:
  - type of training experience,
  - target function,
  - representation of target function,
  - an algorithm for learning the target function from the training data.

# Summary III

- Learning involves searching the space of possible hypothesis.
- Different learning methods search different hypothesis spaces (numerical functions, neural networks, decision trees, symbolic rules).

- There are some theoretical results which characterize conditions under which these search methods converge toward an optimal hypothesis.