What is the Learning Problem?

Learning = Improving with experience at some task

- Improve over task T,
- with respect to performance measure P,
- based on experience E.

E.g., Learn to play checkers

- T: Play checkers
- P: % of games won in world tournament
- E: opportunity to play against self

Learning to Play Checkers

- T: Play checkers
- 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?

Type of Training Experience

- Direct or indirect?
- Teacher or not?

A problem: is training experience representative of performance goal?

Choose the Target Function

- $ChooseMove : Board \rightarrow Move$??
- $V : Board \rightarrow \Re$??
- ...

Possible Definition for Target Function V

- 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 a not a final 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.

This gives correct values, but is not operational

Choose Representation for Target Function

- collection of rules?
- neural network ?
- polynomial function of board features?
- ...

A Representation for Learned Function

$\hat{V}(b) = w_0+w_1\cdot bp(b)+w_2\cdot rp(b)+w_3\cdot bk(b)+w_4\cdot rk(b)+w_5\cdot bt(b)$-

- $bp(b)$: the number of black pieces on board b
- $rp(b)$: the number of red pieces on board b
- $bk(b)$: the number of black kings on board b
- $rk(b)$: the number of red kings on board b
- $bt(b)$: the number of red pieces threatened by black (i.e., which can be taken on black's next turn)
- $rt(b)$: the number of black pieces threatened by red

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

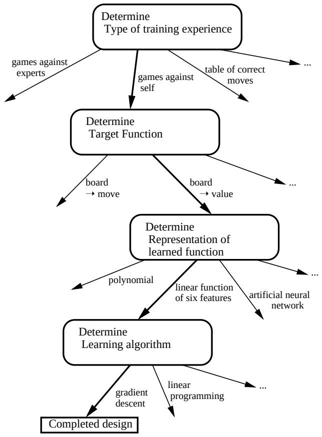
LMS Weight update rule:

Do repeatedly:

- Select a training example b at random
  1. Compute the $error(b)$ for this training example:
     $$error(b) = V_{train}(b) - \hat{V}(b)$$
  2. For each board feature f_i, update weight w_i as follows:
     $$w_i \leftarrow w_i + c \cdot f_i \cdot error(b)$$

c is some small constant, say 0.5, to moderate the rate of learning

Design Choices



Determine Type of training experience

games against experts

games against self

table of correct moves

...

Determine Target Function

board → move

board → value

...

Determine Representation of learned function

polynomial

linear function of six features

artificial neural network

...

Determine Learning algorithm

gradient descent

linear programming

...

Completed design

Some Issues in Machine Learning

- What algorithms can approximate functions well (and when)?
- How does number of training examples influence accuracy?
- How does complexity of hypothesis representation impact it?
- How does noisy data influence accuracy?
- 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 representations?