Designing a Learning System

DESIGNING A LEARNING SYSTEM

1. **Choosing the Training Experience**

- The type of training experience available can have a significant impact on success or failure of the learner.
- One key attribute is whether the training experience provides direct or indirect feedback regarding the choices made by the performance system.
  - For example, in learning to play checkers, the system might learn from ***direct*** training examples consisting of individual checkers board states and the correct move for each. Alternatively, it might have available only ***indirect*** information consisting of the move sequences and final outcomes of various games played.
  - Here the learner faces an additional problem of ***credit assignment,*** or determining the degree to which each move in the sequence deserves credit or blame for the final outcome.
- A second important attribute of the training experience is the degree to which the learner controls the sequence of training examples.
  - For example, the learner might rely on the teacher to select informative board states and to provide the correct move for each. Alternatively, the learner might itself propose board states that it finds particularly confusing and ask the teacher for the correct move.
- **A** third important attribute of the training experience is how well it represents the distribution of examples over which the final system performance P must be measured.

DESIGNING A LEARNING SYSTEM

- We now have a fully specified learning task.
- **A checkers learning problem:**
  - Task T: playing checkers
  - Performance measure ***P:*** Percent of games won in the world tournament
  - Training experience E: games played against itself
- In order to complete the design of the learning system, we must now choose

  1. the exact type of knowledge to be learned

  2. a representation for this target knowledge

  3. a learning mechanism

DESIGNING A LEARNING SYSTEM

2. Choosing the Target Function

- To determine exactly what type of knowledge will be learned and how this will be used by the performance program.
- Let us begin with a checkers-playing program that can generate the ***legal*** moves from any board state. The program needs only to learn how to choose the ***best*** move from among these legal moves.
  - Given this setting where we must learn to choose among the legal moves, the most obvious choice for the type of information to be learned is a program, or function, that chooses the best move for any given board state.
  - Let us call this function ***ChooseMove*** and use the notation ***ChooseMove*** : $B \rightarrow M$ to indicate that this function accepts as input any board from the set of legal board states *B* and produces as output some move from the set of legal moves ***M.***
- An alternative target function and one that will turn out to be easier to learn in this setting-is an evaluation function that assigns a numerical score to any given board state.
  - Let us call this target function ***V*** and again use the notation ***V*** : $B \rightarrow R$ to denote that ***V*** maps any legal board state from the set B to some real value. We intend for this target function ***V*** to assign higher scores to better board states.
  - Let us therefore define the target value ***V(b)*** for an arbitrary board state ***b*** in *B,* ***as*** follows:

1. if ***b*** is a final board state that is won, then ***V(b)*** = 100

2. if b is a final board state that is lost, then ***V(b)*** = -100

3. if b is a final board state that is drawn, then ***V(b)*** = ***0***

4. 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 (assuming the opponent plays optimally, as well).

DESIGNING A LEARNING SYSTEM

3. **Choosing a Representation for the Target Function**

- we have specified the ideal target function V, we must choose a representation that the learning program will use to describe the function c that it will learn.
- let us choose a simple representation: for any given board state, the function c will be calculated as a linear combination of the following board features:
  - ***x1:*** the number of black pieces on the board
  - ***x2:*** the number of red pieces on the board
  - ***x3:*** the number of black kings on the board
  - **x4:** the number of red kings on the board
  - ***x5:*** the number of black pieces threatened by red (i.e., which can be captured on red's next turn)
  - ***X6:*** the number of red pieces threatened by black

Thus, our learning program will represent c(b) as a linear function of the form

$$\hat{V}(b) = w_0 + w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + w_5x_5 + w_6x_6$$

where ***wo*** through ***W6*** are numerical coefficients, or weights, to be chosen by the learning algorithm.

DESIGNING A LEARNING SYSTEM

- Partial **design of a checkers learning program:**
  - Task T: playing checkers
  - Performance measure ***P:*** percent of games won in the world tournament
  - Training experience E: games played against itself
  - Target function: V:Board →R
  - Target function representation

$$\hat{V}(b) = w_0 + w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + w_5x_5 + w_6x_6$$

DESIGNING A LEARNING SYSTEM

4. **Choosing a Function Approximation Algorithm**

- In order to learn the target function $\hat{V}$ we require a set of training examples, each describing a specific board state b and the training value $V_{train}(b)$ for b.
- In other words, each training example is an ordered pair of the form (b, $V_{train}(b)$).
- For instance, the following training example describes a board state b in which black has won the game (note ***x2*** = 0 indicates that **red** has no remaining pieces) and for which the target function value Vtrain(b)s therefore **+100.**

$$\langle\langle x_1 = 3, x_2 = 0, x_3 = 1, x_4 = 0, x_5 = 0, x_6 = 0\rangle, +100\rangle$$

DESIGNING A LEARNING SYSTEM

4.1 **ESTIMATING TRAINING VALUES**

- $V(b)$: the true target function
- $\hat{V}(b)$: the learned function
- $V_{train}(b)$: the training value

One rule for estimating training values:

- $V_{train}(b) \leftarrow \hat{V}(Successor(b))$

DESIGNING A LEARNING SYSTEM

4.2 **ADJUSTING THE WEIGHTS**

- One common approach is to define the best hypothesis, or set of weights, as that which minimizes the square error E between the training values and the values predicted by the hypothesis $\hat{V}$.

$$E \equiv \sum_{\langle b, V_{train}(b) \rangle \in \text{ training examples}} (V_{train}(b) - \hat{V}(b))^2$$

LMS weight update rule.

For each training example $\langle b, V_{train}(b) \rangle$

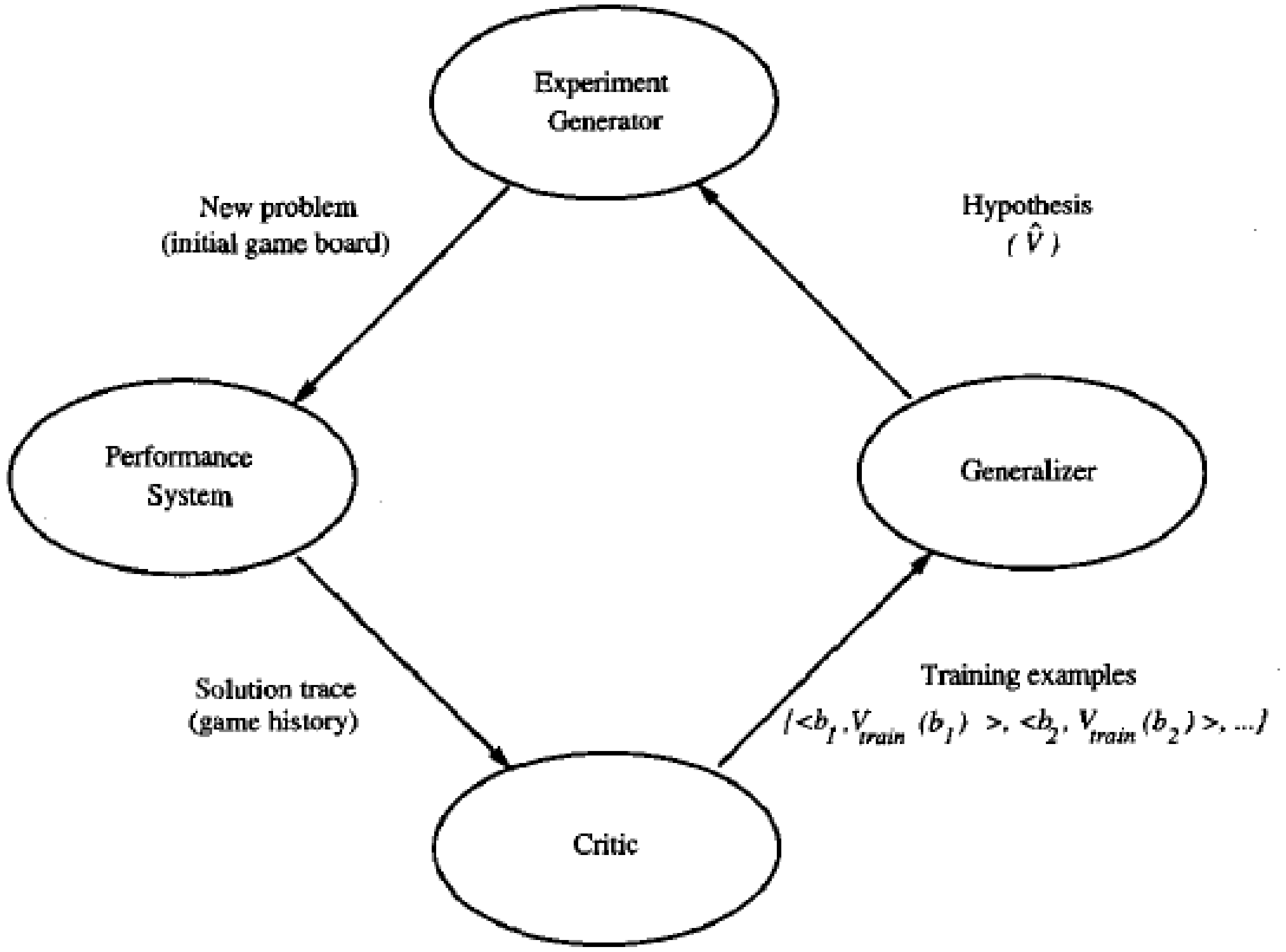
- Use the current weights to calculate $\hat{V}(b)$
- For each weight w_i, update it as

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

when the error $(V_{train}(b) - \hat{V}(b))$ is zero, no weights are changed.

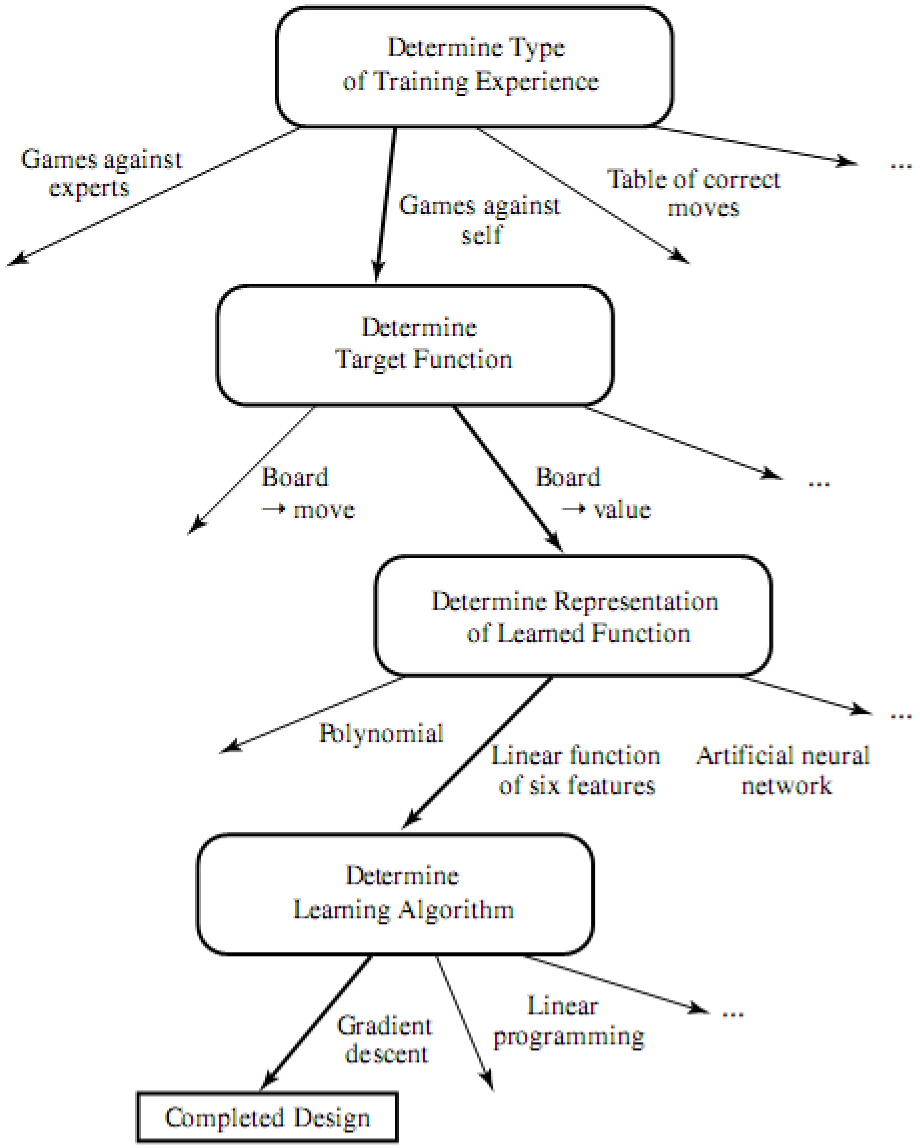
DESIGNING A LEARNING SYSTEM

The Final Design


Experiment Generator
New problem (initial game board)
Hypothesis ($\hat{V}$)
Performance System
Generalizer
Solution trace (game history)
Training examples
$\{<b_1, V_{train}(b_1)>, <b_2, V_{train}(b_2)>, ...\}$
Critic

DESIGNING A LEARNING SYSTEM

The Final Design



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

Exercise

- How many steps are involved in building the learning system?
-

Exercise

- Which of the following is not a step in building a learning system? *
  - Choosing the training experience
  - Choosing the function approximation algorithm
  - hyperparameter tuning
  - None of these

Exercise

- Weights are adjusted so that
  - Error can be minimized
  - Error can be maximized
  - Error does not have any relation

Exercise

- Distribution of training examples put impact on the performance of the model.
  - True
  - False

Exercise

- Explain the designing a learning system of sentiment analysis using neat and clean diagram.
- Explain the designing a learning system of Credit card fraud detection using neat and clean diagram.

Thank You !!!