What Counts as Learning?

Here are some ways that a program can update its own database. Which of these, if any, counts as “learning”?

- From `P` and `P -> Q` learn that `Q`
- From `Q` and `P -> Q` learn that `P`
- From `P(x1)` and `P(x2)` and `P(x3)` … learn that `for all x, P(x)`

Turning the Question Around: What Does Learning "Count As"?

In different contexts, learning can take various forms:

- information gathering
  - abduction, induction, deduction?
- hypothesis formation
  - abduction, deduction?
- generalization
  - induction, abduction?
- speedup
  - deduction!, abduction, induction

Just defining the learning problem is difficult. But doing so can constrain the implementation task quite a bit!

An Exercise

I have a graduate student who wants to build a program that learns to play checkers better.

Help him out by suggesting answers to the following questions:

- What kind of knowledge should his program try to learn?
- What should it do to learn this knowledge?
- How will we know if the program has learned anything?

Your answers need not deal with details of the program, but rather with the game of checkers and the task of software engineering.

As a thinking aid, you might ask yourself the same questions about how **you** learn to play a game (well)!

What is the Learning Problem?

improving with experience at some task

- Improve at task T...
- ... based on some experience E ...
- ... with respect to performance standard P.

For the task of learning checkers, my graduate student might fill in the variables with:

- T = playing checkers
- E = playing human opponents when they are available; playing itself all other times
- P = its performance rating in match play and tournament play against players who strength is known

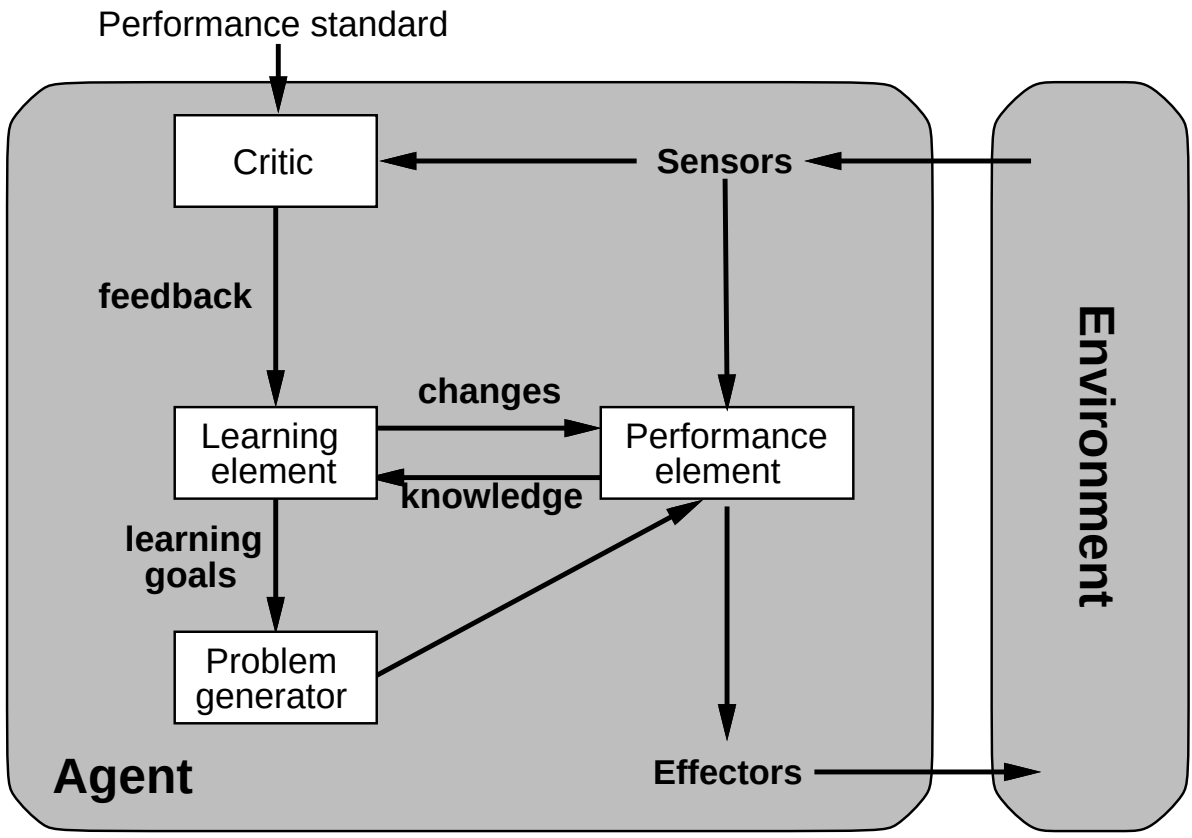
Learning in Specific Contexts

An agent is given a sequence of problems from a particular class of problems. As a result of solving a problem **and getting feedback**, the agent should

- solve the same problem **better**
- solve **other** problems from the class

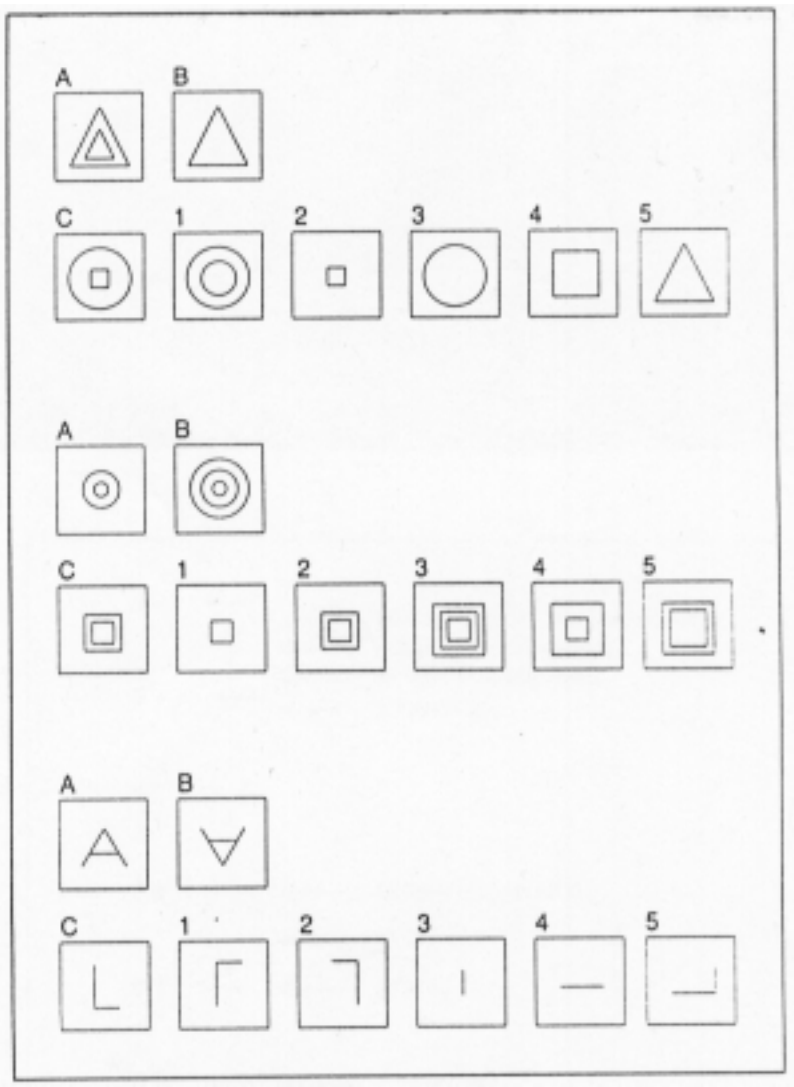
An agent is given a sequence of attribute vectors to classify into n different categories. Each vector is accompanied by its correct category. The agent should have more success categorizing a vector as the number of examples it sees increases.

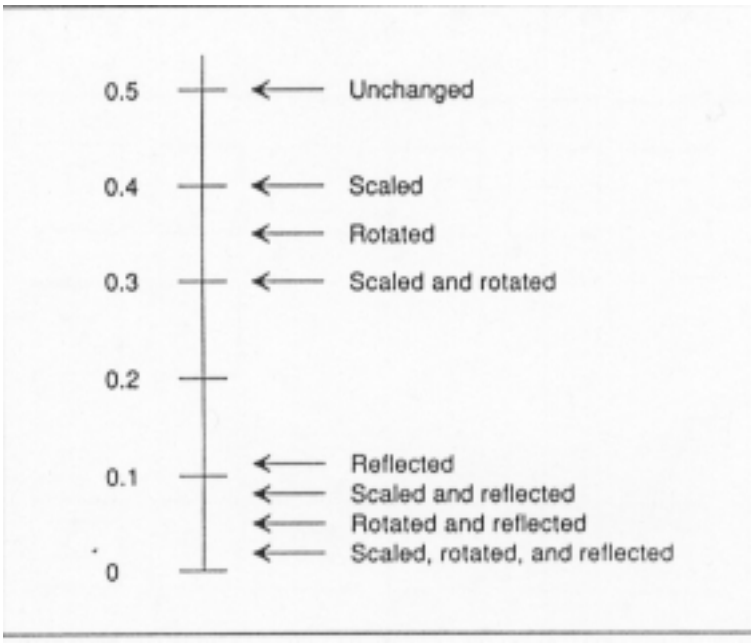
An agent is given a sequence of percepts (in the form of attribute vectors), and for each it must suggest an action. The more vectors it sees, the better the action it suggests.


Performance standard
Critic
Sensors
feedback
changes
Learning element
Performance element
knowledge
learning goals
Problem generator
Agent
Effectors
Environment

Flashback to the ACT

For each of these three analogy problems, select the correct answer:


A
B
C
1
2
3
4
5
A
B
C
1
2
3
4
5
A
B
C
1
2
3
4
5


0.5
Unchanged
0.4
Scaled
Rotated
0.3
Scaled and rotated
0.2
0.1
Reflected
Scaled and reflected
Rotated and reflected
Scaled, rotated, and reflected
0

Learning as Finding a Function

A learning agent encounters a number of data when working on a problem (say, playing checkers). It faces a tough problem: