1

The Machine That Could Learn Anything

A highlight of the eighteenth SPWBA conference was "The Machine That Can Learn Anything." Devised by the succinctly named "Professor A.," this exhibit drew attention from a broad cross section of delegates. Its success appears to have been partly due to its striking visual appearance. While other exhibits sported the chromium hi-tech look, the "Machine That Can Learn Anything" offered something more primitive. A small, black tent adorned with astrological motifs, relieved by the color screen of a laptop just visible through a velvet-edged opening. On the outside of the tent, a handwritten sheet welcomed visitors and encouraged them to submit trial learning tasks to the machine. Instructions at the end of the sheet specified the terms of engagement. "Tasks must be presented in the form of example association pairs," it asserted. "The machine can then be tested by presentation of test items. A task will be considered to have been successfully learned if the machine is able to produce correct associations for test items in the majority of cases."

Visitors wishing to test the machine's ability had thus to express learning tasks in the form of association pairs. Let's say the visitor wished to test machine's ability to learn addition. He or she had first to write down a list of suitably representative associations. For example:

```
1 1 -> 2
1 2 -> 3
2 5 -> 7
4 2 -> 6
8 1 -> 9
```

This list had then to be presented to the machine via its "sensory organ": a hole in the tent's rear panel. Having passed in their list of associations, testers had then to move around to the screen at the front of the tent and wait until such time as the machine printed out an instruction inviting the submission of a query. Testers could then type in items and evaluate the machine's responses (suggested associations) as correct or incorrect. At each stage the machine would present performance statistics, that is, show the proportion of test items upon which correct associations had been generated. And without exception, testers found that the machine performed in an exemplary fashion. It was, indeed, able to generate correct associations in the majority of cases.

Many visitors were visibly impressed by the behavior of the machine. Some were inclined to reserve judgment; a few showed complete disinterest. But from the practical point of view, the exhibit was a runaway success. In addition to hosting an estimated 2000 visitors in less than a week, it was the recipient of a quite unexpected level of media attention. The day after opening, a national newspaper printed a 1000-word report on Professor A.'s work under the banner headline "Learn-Anything Machine Is a Labour of Love." And on the final two days of the exhibition, no fewer than three TV stations ran news items covering the exhibit itself. The media take on A.'s machine was unanimously upbeat, and commentators were uncharacteristically supportive. In just a few years' time, one suggested, members of the public could expect to be "teaching" their computers to do things rather than laboriously "commanding" them using the mouse and keyboard.

When asked how the machine worked, Professor A. noted there was no magic involved; the machine simply applied well-known techniques from the field of *machine learning*, a subfield of computer science concerned with intelligent computers. But he admitted that the role played by the machine's "sensory organ" was significant. "The key to the machine's success," he noted, "is that users can only present learning tasks in a particular way, namely as association or prediction tasks. This is the format assumed by many methods in machine learning. By forcing users to present their problems this way, we open up the whole repertoire of machine learning methods. We make it possible to employ any method we like on any problem we're presented with."

With his bold rhetoric, Professor A. quickly swayed the media to his side. But the academic rank and file were less easily persuaded. As the conference neared a conclusion, grassroots opinion turned against Professor A., and at a plenary session held on the final day, his machine became the focus of a rising tide of hostile attention. One critic suggested that rather than being able to "learn anything," A.'s machine was actually limited to the solving of formally specified prediction problems. Another argued that since the machine had no way of actively interacting with the world, there was no basis upon which it could carry out *any* sort of cognitive activity, let alone learning.

Professor A. remained unmoved. He accepted the proposition that the machine's abilities involved prediction. But, somewhat to the surprise of his detractors, he rejected the idea that there was any real difference between this task and that of learning. He then went on to challenge the audience to come up with a learning problem that could *not* be interpreted as a type of prediction problem.

The assembly was temporarily silenced. Then a shout rang out. "What about concept learning?" demanded a man standing at the very rear of the hall. Professor A. contemplated the question for a moment and then moved cautiously toward the overhead projector. "OK. But let us pretend that I have never heard of concept learning," he said, taking a green felt-tip from his pocket. "Now you tell me how you would like to specify a concept learning problem."

The man thought for a few moments before responding. "Something that can do concept learning obviously has to be able to acquire the ability to distinguish between things which are part of the concept and things which are not part of the concept."

"Fine," said A. "And how should we specify this result?"

The man sensed that the question was intended to snare him. But he was unable to prevent himself from falling into the professor's trap. In attempting to provide a formal specification for a concept learning problem, the man found himself beginning to talk in terms of a *mapping* between certain situations and certain responses.

"But this mapping you are describing can also be viewed as specifying a prediction problem, can it not?" replied the professor when the man finally come to a stop. No answer was forthcoming. The professor

continued to his punch line. "And this is exactly the format which is required by my machine, yes? So we find that in formally specifying a learning problem we inevitably produce something which can be interpreted as a prediction problem. One has to conclude there is no *formal* difference between the two types of tasks."

It was a well-rehearsed performance. But still, many members of the audience remained unconvinced. Some went so far as to offer further responses to the "challenge." The professor was thus given the opportunity to demonstrate by similar reasoning that several other forms of learning—including skill acquisition, function learning, language development, classification learning, and behavior learning—were all equivalent, under formal specification, to the task of prediction. When it became obvious that there was no mileage to be gained on this territory, the flow of criticism began to dry up. One hardy individual, however, refused to give in.

"It amazes me," he commented bluntly, "that anyone could think that prediction and learning were the same thing. Surely it is obvious that many natural organisms do the latter but not the former."

"Well, that may be," agreed the professor. "But so what? I never claimed that prediction and learning are the same thing. The processes may be—probably are—quite different. What I showed was that specifications of learning tasks are always equivalent to specifications for prediction tasks. So the tasks have to be the same. Even if the solutions are different."

"But aren't you just making a theoretical distinction?" responded the truculent delegate. "Most interesting learning tasks can't be given formal specifications in advance. So the real issue is how a learning agent can develop behavior that doesn't have a neat, formal specification."

The professor nodded, considering (or pretending to consider) the point at length. "Well, I'm not sure that it makes sense to say you have a learning task if you cannot formally specify what that task is. That seems to me to be a contradiction. And changing the topic to behavior learning makes no difference. Either there is a behavior or there is not a behavior. And if there is, it must be possible, at least in principle, to say *what* that behavior is, that is, to give it a formal specification. I cannot see how we

can escape this. I really can't. So it seems to me unavoidable that I have been right all along."

1.1 Back to Reality

So much for the story of Professor A. What are *we* to make of the Machine That Can Learn Anything? How should we interpret the professor's immodest defense at the plenary session? Is the machine itself some sort of fake? Are the professor's arguments about the formal equivalence of learning tasks and prediction tasks mere sophistry? The reader will probably have formed an opinion. But the line pursued in this book will be essentially *pro*. That is to say, it will tend to go along with the argument that learning can be treated as an attempt to solve a prediction task. The idea sounds implausible—even absurd—at first hearing. But it becomes more digestible with familiarity.

Any dictionary definition will confirm that learning involves the acquisition of knowledge or behavior. But since knowledge acquisition can always be viewed as the learning of new "conceptual behavior," we can justifiably treat *all* learning as some form of behavior learning. This simplifies things considerably. But we can go a stage further.

A complete specification of a behavior must show how it involves the production of certain actions in certain situations. So whenever we attempt to fully specify a learning task, we must identify the relevant associations between situations and actions. But as soon as we do this, we are caught on the professor's hook. Our problem specification defines the task in terms of a *mapping*. We can always read this mapping in two ways: as saying what actions should be produced in a given situation or as *predicting* which actions should be produced in a given situation. The two readings are essentially equivalent. It does not make any difference if we treat the mapping as specifying a prediction task or a learning task.

Professor A. is thus right in claiming that learning tasks and prediction tasks are equivalent. But what of his claim that his machine can learn *anything*? The professor's argument rests on the fact that he can get his machine to produce above-chance performance on any prediction problem. But does this prove anything? Can it really support the claim that

the machine can perform universal learning? To get a better handle on these questions, we need to take a closer look at the process of prediction. We need to see what it involves and what sort of performance is generally achievable.

1.2 Prediction Games

A prediction task stripped to the bones is really just a type of guessing game. It is a contest in which an individual is given some information on a topic, and is then asked to guess information that has been held back. The game of "battleships" is a good example. In this game, two users provide information about their battleship formation on a turn-by-turn basis. The aim of the game is to sink the other person's ships. This involves guessing the locations of the opponent's ships from the information given.

Another common guessing game is that of *sequence prediction*. In this problem a string of numbers is given, and the task is to continue the sequence, that is, to make predictions about numbers that appear later on. For instance, if we are given the sequence

2, 4, 6, 8

and asked to predict the next number, we may well guess

10,

on the grounds that the numbers are increasing by values of 2. However, if we are asked to continue the sequence

2, 4, 6, 8, 10, 13, 16, 19,

we may guess that the next number is 22, or perhaps 23.

Of course, the data presented in prediction problems may be symbolic rather than numeric. They also may take the form of an unordered set rather than a sequence. For example, we might be presented with the data

orange, banana, pear

and asked to predict another item in the same set. A plausible response might be "apple" or "grape." Similarly, a likely guess regarding

Toyota, Ford, Mercedes, VW

might be Datsun.

A scenario that is particular interesting for present purposes occurs when the data are structured objects. For example, let us say we are given the following set of triples

⟨1, 4, 4⟩, ⟨8, 4, 1⟩, ⟨2, 6, 1⟩, ⟨3, 3, 3⟩, ⟨4, 2, 2⟩

and asked to guess another member of the same set. A plausible guess would be

⟨9, 1, 3⟩,

on the grounds that in all the examples, the largest value in the triple is perfectly divisible by both other values. Of course, there may be other plausible rules.

In a variation on the structured data theme, the aim is to predict missing values within *partially* complete data. For example, the task might involve using the examples

⟨1, 4, 4⟩, ⟨8, 0, 1⟩, ⟨2, 6, 1⟩, ⟨3, 3, 3⟩, ⟨4, 2, 2⟩

to fill in the missing values in

⟨6, ?, 1⟩, ⟨4, ?, ?⟩.

This actually brings us back to the data format required by Professor A. Examples had to be presented to his machine in the form of association pairs, that is, as objects consisting of a set of input values and an associated output. For example,

```
1 1 -> 2
1 2 -> 3
2 5 -> 7.
```

Such data are really just structured objects with an implicit partition between the input values and the output values. The examples above might have been written

```
<1 1 2>
<1 2 3>
<2 5 7>,
```

with the assumption being that the first two values in each object are the

given data and the final value is the answer. Correctly formatted test cases could then be written as partially complete triples, such as

```
<3 2 ?>
<4 1 ?>.
```

1.3 Supervised Learning

Prediction tasks presented using Professor A.'s association format are termed *supervised learning* tasks, on the grounds that the examples are like information given to the learner by a teacher or supervisor. When this terminology is used, the thing that is learned is generally termed the *target function*, and the inputs and associated outputs are treated as the arguments and values (respectively) of an unknown function. The learning is then conceptualized along computational lines. The given data (to the left of the arrow) are viewed as *input values*, and the to-be-predicted data (to the right) as *output values*. The learning process is then naturally viewed as the process of acquiring the ability to *compute* the target function.

Input values typically represent the attributes of an object or class. For example, in the association

```
red round smooth -> tasty
```

"red," "round," and "shiny" might be the color, shape, and texture attributes for a particular item (or class) of fruit. In such cases it is natural to view each set of input values as a description of an object written in terms of *variables*. A distinction may then be made between *input variables* and *output variables*, the former being placeholders for the input values and the latter being placeholders for the output values. Further, there is the obvious shortcut in which we refer to a complete set of input values simply as an *input* and a complete set of output values as an *output*. These conventions are illustrated in figure 1.1.

The supervised learning paradigm provides a convenient way of packaging learning problems. But, appearances to the contrary, it does *not* not impose any restrictions or constraints. As the fictional Professor A. demonstrates in the story above, an association-mapping specification merely "fixes the goalposts." It is a specification of the *task* rather than

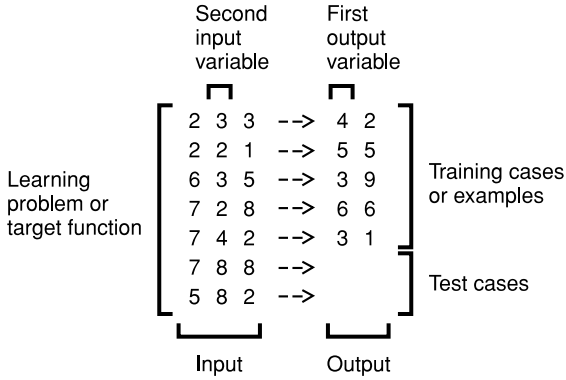

Figure 1.1
Supervised-learning terminology

the *solution*, and thus is completely neutral with respect to the way in which a particular problem may be solved.

1.4 Concept and Classification Learning

Although work in machine learning concerns itself primarily with supervised learning (prediction) tasks, researchers have focused on a number of variations. Attention has been given particularly to the so-called *concept learning problem*. This problem has the same form as the standard supervised learning problem except that target outputs are either "yes" or "no" (sometimes written + and −). Inputs that map onto "yes" are treated as positive examples of a particular concept. Inputs that map onto "no" are treated as negative examples (i.e., counterexamples). And the process of finding a solution to such a problem is then naturally viewed as the process of acquiring the relevant concept.

A sample concept learning problem appears in figure 1.2.[1] Here the inputs are lists of attributes for items of fruit, and the concept is that of edible fruit. Solving this problem can be viewed as the acquisition of the "edible-fruit" concept. Once the problem has been solved, it should be possible to classify test data, that is, novel items of fruit, as either edible or nonedible.

A variation on the theme of concept learning is the *classification problem*. This is just like the concept learning problem except for the fact

```
hairy  brown large hard --> no
smooth green small hard --> yes
hairy  red   small soft --> no
smooth red   large soft --> yes
smooth brown large hard -->
```

Figure 1.2
Edible-fruit concept learning problem

```
gasoline hatchback   FW-drive --> Ford
gasoline convertible FW-drive --> Ferrari
diesel   saloon      FW-drive --> Ford
gasoline hardtop     RW-drive --> Ferrari
diesel   hardtop     FW-drive -->
```

Figure 1.3
Car classification problem

that we now have a number of target outputs that are the labels of classes. The cases illustrate what sort of object belongs in which class. A sample problem involving the classification of cars appears in figure 1.3. (The variables here describe—working left to right—the fuel used, the body style, and the location of the drive wheels.)

In another version of the supervised learning problem, the inputs take the form of sets of truth-values, with "true" written as 1 and "false" as 0. The aim is to correctly learn the truth function exemplified by the examples. A sample problem appears in figure 1.4.

1.5 Behavior Learning

The supervised learning scenario also lends itself to the problem of *behavior learning*. For example, imagine that we have a simple two-wheeled mobile robot (a "mobot"), circular in shape and with two light sensors on its leading edge, as in figure 1.5(a). Imagine we would like the mobot to use a supervised learning method to learn how to steer away from sources of bright light, as illustrated in the plan diagram of figure 1.5(c) We might proceed by constructing a training set of examples in which each input is a combination of light-sensor values and each output

```
1 1 0 1 1 --> 1
1 0 0 0 0 --> 0
0 1 1 1 0 --> 1
1 1 0 0 1 --> 0
0 0 0 0 0 -->
```

Figure 1.4
Truth-function learning problem

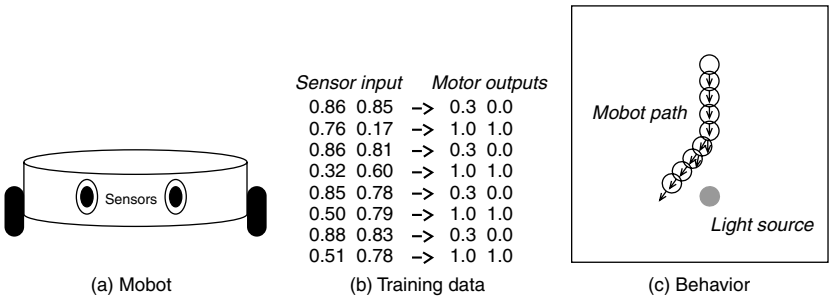

Figure 1.5
Learning light avoidance

is the combination of signals to be sent to the wheel motors. If the light sensors return higher values for stronger sources of light, and the motors produce an amount of wheel rotation proportional to the relevant signal, then a subset of the input/output pairs might be as shown in figure 1.5(b). The data exemplify the fact that rotational moves (achieved by turning the left wheel only) should be executed whenever either of the input signals exceeds a certain threshold.

If we equip the mobot with supervised learning capability and have it process the data in figure 1.5, then the result should be that the mobot acquires the ability to respond in the desired fashion with respect to sources of light.

1.6 Financial Prediction

One of the most intriguing supervised learning scenarios involves prediction of financial data, such as prices of stocks, bonds, and other

```
Price of x   Price of y       Future price of z

0.979248     0.058547   -->   0.057332
0.178428     0.784546   -->   0.139985
0.103902     0.024725   -->   0.002569
0.268517     0.639011   -->   0.171585
0.495132     0.159034   -->
```

Figure 1.6
Financial prediction problem

commodities. In this variant, the values of the variables are numeric. The input values are given financial data (i.e., current prices of given commodities) and the outputs are to-be-predicted financial data (e.g., the *future* price of some commodity). An illustrative set of data is shown in figure 1.6.

Finding a good solution to this sort of problem might enable us to make money, since it would provide the means of accurately predicting prices from current prices. Imagine that the output values in this mapping are wheat prices and that the application of a learning method to the data produces an input/output rule which, when applied to current prices, suggests that the price of wheat is about to increase. The implication is clear: if we buy wheat stocks now, we should realize a quick profit by selling after the price has risen.

1.7 Learning Problems to Solve by Hand

The best way to come to grips with what supervised learning is all about is to try to *do* it, that is, try to solve some nontrivial, supervised learning problems by hand. The three problems presented below may be used for this purpose. All follow the concept learning model, that is, they all take the form of an input/output mapping in which there are just two distinct outputs. And they all involve a manageably small amount of data, less than one page of input/output pairs. The first problem involves predicting the results of football games. The second involves distinguishing phrases with a certain linguistic property. The third involves deriving a plausible interpretation for a set of card game rules.

Problem 1: Predicting the Football Results

Careful investigation of the football results over a number of weeks shows that the performance of team X can be predicted on the basis of the results for teams A, B, C, D, and E. Thus, it is possible to predict whether X will win or lose just by looking at the results for A, B, C, D, and E. (It is assumed that X never plays A, B, C, D, or E.)

The data set below presents the results for teams A, B, C, D, E, and X recorded over 16 consecutive weeks. Each line shows the results for a particular week. The five input variables represent—working left to right—the results for teams A, B, C, D, and E, and the single output variable represents the result for team X. In all cases, 1 signifies "win" and 0 signifies "lose." The problem is to use the 16 examples to derive the rule that allows team X's result to be predicted from the other teams' results.

```
1 0 0 0 0 --> 0
0 0 0 0 1 --> 0
1 0 1 1 1 --> 1
1 0 1 0 1 --> 0
0 0 1 0 0 --> 0
1 0 0 1 0 --> 1
1 1 1 0 0 --> 1
0 0 0 0 0 --> 0
1 1 1 1 1 --> 1
1 0 1 1 0 --> 1
1 0 0 1 1 --> 1
1 0 0 0 1 --> 0
1 0 1 0 0 --> 0
1 1 1 1 0 --> 1
1 1 1 0 1 --> 1
0 1 1 0 0 --> 1
0 1 1 1 0 -->
1 1 0 1 1 -->
0 1 1 0 1 -->
0 0 1 1 0 -->
```

Problem 2: The Incoherent Tutor

A student takes a course titled "The Production of Coherent Nonsense." Her tutor's method is to present his students with examples of good practice. In one session he takes a set of examples of three-word nonsense phrases and shows which of them should be classified as coherent and which as incoherent. The examples are shown below. The question is, What rule is the tutor using to classify the examples?

```
eypnv gdukk kaqpi --> coherent
psgdr gbaiz htyls --> incoherent
ihytw xbfkg yxcxw --> coherent
panct jlege kkirg --> incoherent
qpcrz vyqkr ygawe --> coherent
ahvlh xggcz nsgff --> incoherent
urmle zybyx gxslm --> incoherent
mbrfc plpkp rojva --> coherent
gdzxa vvjre ztdyj --> coherent
qpmuu begvu rmukx --> incoherent
riijf xdvxm xegum --> coherent
qpheq udrrw zguei --> coherent
qbiha zitck yegyx --> incoherent
sjvva ribyr qqeku --> incoherent
qcsgu qterv hulmf --> incoherent
duzsr rpjao zhmds --> coherent
iruih rxjaw xkgjn --> coherent
fppen mdasf wvfmj --> coherent
eatwk semqd cqewc --> incoherent
cnzbt ilzvl zzmkl --> coherent
hygtt xscza hiijl -->
xibkd uxgzl opcmf -->
eppps bbvtz zggil -->
lhwnu kltla kwzmg -->
```

Problem 3: The Card Player

One evening, a man wandered into town and bought himself a beer in the local saloon. While he drank his beer, he studied a group of settlers gathered around a large table. The settlers appeared to be engaged in

some sort of trading activity involving the exchange of small, rectangular cards. The man had no idea what the basis of this activity was. But, being bold, he strode across the room and sat down at the table. The settler immediately to his left nodded to him and began scribbling numbers on a sheet of paper. When he had filled the entire sheet, he pushed it across the table, saying, "If you want to play, you'd better learn the rules." The numbers on the sheet of paper were as shown below. How should they be interpreted? And what rule allows one to predict output values correctly?

```
13 1 9   4 12 1 9   1 9   4 --> 4
2  2 9   1 2  2 9   2 2   2 --> 5
13 3 4   1 11 4 11  4 11  1 --> 4
3  3 12  3 6  3 13  3 1   3 --> 6
2  1 2   1 2  4 12  1 2   2 --> 9
8  3 8   2 8  2 8   4 7   1 --> 9
4  4 4   2 4  4 5   2 4   1 --> 9
7  4 7   4 5  4 5   2 5   4 --> 5
6  4 12  1 6  1 6   4 9   1 --> 4
4  4 9   2 9  4 4   1 4   2 --> 5
6  4 10  4 13 4 9   4 8   4 --> 6
11 1 11  4 11 2 8   3 11  1 --> 9
4  1 7   1 5  1 1   1 13  1 --> 6
10 4 2   3 12 4 12  3 12  2 -->
5  1 7   1 1  1 13  1 4   1 -->
13 2 5   2 13 4 10  1 13  1 -->
```

1.8 A Reasonable Learning Criterion

To conclude the chapter, let us return once more to Professor A. and the Machine That Can Learn Anything. We obviously would like to know whether the machine really does what it is supposed to do, or whether there is some kind of trickery going on. The professor's arguments with respect to the formal equivalence of prediction tasks and learning tasks are sound. Thus we know that any attempt to discredit the machine on the basis of its input limitations fails. But does this mean we have to accept the professor's claims as stated?

The answer is No! Of course the ability to "learn anything" sounds impressive. But when we look carefully at what is really being offered, we find that what you see is not quite what you get. Recall that, according to the rules of engagement, the machine is to be deemed as having successfully learned the task presented, provided it gets the answers right in the *majority* of cases. This means it can get 49% of the associations wrong and still end up counting itself a winner! Some mistake, surely.

Undoubtedly, Professor A. is conning his audience. But the con has nothing to do with the restricted task presentation format. It has to do with the success criterion that is applied to the learning. When we read the phrase "the majority of cases," we tend to think in terms of figures like 80% or 90%. And, for the unwary, it may therefore sound perfectly reasonable that the machine should be deemed to have successfully learned something provided it produces appropriate answers in the majority of cases. But the criterion is too weak. We would never deem a person to have successfully learned something were he or she to produce inappropriate results in up to 49.99999% of cases. A.'s machine should be treated the same way. The Machine That Can Learn Anything thus has to be considered a fake, pending the introduction of a more restrictive success criterion.

1.9 Note

1. Note that this and the other "sample problems" in this chapter are merely illustrations. In practice, problems involve a much larger number of associations.