

Neural Networks versus Artificial Intelligence: The Phoney War

May 21, 2003

Abstract

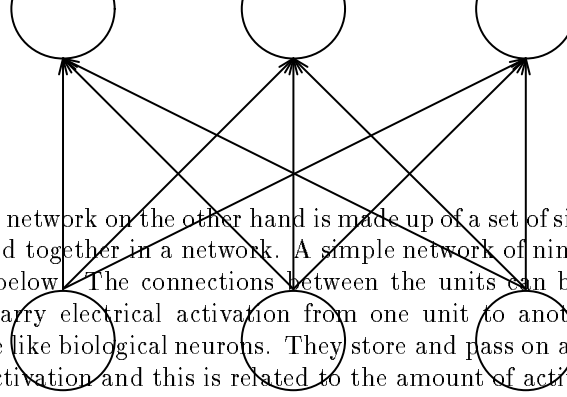
The paper evaluates the significance of recent developments in the field of neural networks with respect to the field of artificial intelligence. It argues that neural network research is not in competition with classical artificial intelligence research but rather represents the latest development in an alternative paradigm *within* artificial intelligence. The paper also introduces a basic neural learning mechanism (back-propagation) and shows how it can be used to derive a diagnostic mechanism from a training set of examples.

1 What is a neural network?

It is no surprise to find that many people are still quite unclear as to what neural networks (NNs) actually are and what they can do. There are many sources of confusion. For example, when AI researchers talk about a “neural network” they are typically referring to a particular type of computational architecture even though the term is used by others to denote a network of *biological* neurons. Resolving this terminological confusion only leads us into another more fundamental one; namely, if a neural network is a type of computational architecture, what is its relationship to the conventional, von Neumann architecture with which we are familiar?

To resolve these confusions we need to clearly characterise the relationship between the neural network and the conventional computational mechanism. The neural network does not work in the same way as the conventional computer and it has very different computational properties. The conventional computer has two main components: a memory and some kind of processing device (eg. a CPU). Information is represented in terms of structures of symbols and the way in which this information is processed all depends on the way in which the processing device manipulates the symbol structures stored in the memory.¹

¹If the computer is to produce useful behaviour, the processing device must manipulate symbols in a way that is compatible with the relationships between the items of information represented.



The neural network on the other hand is made up of a set of simple processing units connected together in a network. A simple network of nine units is shown in the figure below. The connections between the units can be thought of as wires which carry electrical activation from one unit to another. The units themselves are like biological neurons. They store and pass on a certain amount of electrical activation and this is related to the amount of activation that they receive via their input connections. Information is represented in terms of levels of activation and the way in which it is **processed** all depends on the way in which activation propagates through the network; ie. it all depends on the connections between the units.²

Of course, this way of characterising the two types of computer glosses over many important details. It also tends to give the impression that the difference between the conventional computer and the “neural” computer is all to do with the physical components from which they are made. But this is not the case. The fact that we can simulate any mechanism whatsoever in a conventional computer means that we can build a neural network (or, for that matter, a conventional computer) out of simulated rather than physical components.³

²Hence the name *connectionism*.

³The resulting device is called a *virtual machine* since it exists only in the sense that it can be used *as if* it was a machine.

Surprisingly enough, in a great many cases, neural networks *are* built this way — as virtual machines running in conventional computers. The details of the simulation vary from case to case. But very often the activation storing units are represented as record fields, connections are represented as pointers from one field to another and activation levels are represented as real numbers between 0 and 1.

2 The rise and fall and rise of neural networks

Artificial intelligence research has, naturally enough, tended to focus on one type of architecture or the other. Over time this split has become institutionalised and we now have what are, in effect, two paradigms of research. Sometimes the two paradigms are differentiated by labelling the von Neumann-oriented work *symbolic* and the neural network-oriented work *subsymbolic* (the reasons for this should become clear below).

In the earliest years of AI, the symbolic paradigm was dominant and attention tended to focus on algorithmic models such as Newell and Simon's *Logic Theorist*. But in the late 50s and early 60s the subsymbolic paradigm bubbled up with Rosenblatt's work on the Perceptron architecture. This system appeared — initially at least — to exhibit quite startling learning behaviours. But following Minsky and Papert's exhaustive evaluation of the theoretical properties of the perceptron [1] the subsymbolic paradigm sank back while the symbolic paradigm came to the fore with systems such as Evans' analogy solving program [2] and later Winston's work on learning and vision [3, 4].

All through the 70s the symbolic paradigm was in a leading position but towards the end of the decade the subsymbolic paradigm began to show a marked increase in activity and in the early years of the next decade (the 80s) we saw the development of, first, the Boltzmann machine learning algorithm [5] and then the back-propagation algorithm [6]. The emergence of these two learning systems marked an important step forward in the subsymbolic field since they provided a solution to the long-standing problem of learning in *complex* networks, ie. networks with internal units.

In recent years the two paradigms have begun to coexist in a more mutually supportive fashion. There is certainly a huge amount of work going on in both fields as evidenced by the sizes of the two leading conferences (IJCNN and IJCAI). And relatively recently, researchers from both sides of the fence have begun to carry out comparative work to try to establish the precise strengths and weaknesses of symbolic and subsymbolic mechanisms (eg. [7, 8, 9]). Initial results seem to suggest that in the field of learning, symbolic and subsymbolic algorithms are of comparable performance overall. Of course the existence of such comparative studies is not enough to prevent extremists in one paradigm or the other making exaggerated claims about the strengths of their own paradigm or the invalidity of the opposing paradigm. And, certainly, in recent years there has been a surfeit of claims to the effect that neural networks overcome *all* the problems associated with classical (ie. algorithmic) AI, which is assumed to

therefore be defunct. But such claims are totally at odds with the evidence and should not be taken too seriously.

3 Principal Methods

A principal distinguishing characteristic of subsymbolic systems is the fact that they are produced by *training* rather than explicit *programming*. Thus work in the subsymbolic paradigm has tended to concentrate on the development of *learning procedures*. One of the main families of algorithms is based on the perceptron learning rule developed by Frank Rosenblatt [10]. This learning procedure is closely related to standard statistical procedures for deriving discriminant functions using least-mean-squares (LMS) regression.

How does the perceptron learning rule (PLR) work? In the simplest case we have a neural network that is made up of some *input units* connected to one or more *output units*. We want to derive a set of weights for the connections between input and output units so that when we feed activation representing a particular input into the input units we obtain a desired output at the output units. Output units in the perceptron are linear-threshold units. These units compute a linear sum of their input activations and then threshold the result.

Applying the perceptron learning rule involves presenting examples of desired input/output associations and then changing weights so as to reduce the difference between the actual outputs generated and the desired outputs. Let us see what this means in a particular case.

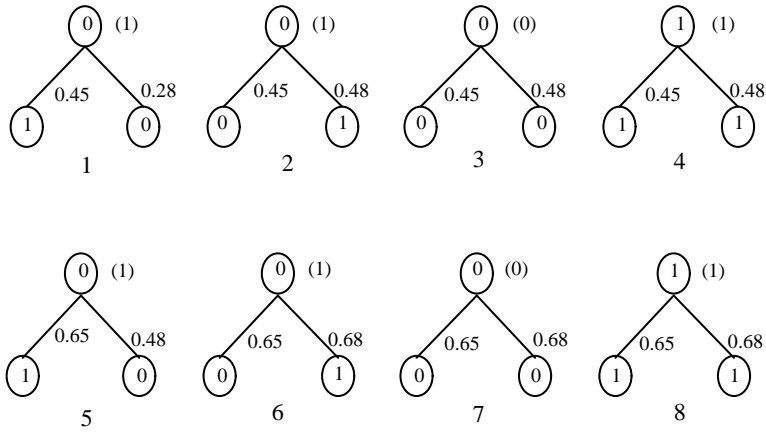
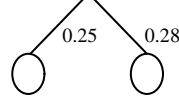
3.1 Learning to compute OR

Imagine that the following are a set of desired input/output associations.

```
<[1 0] [1]>  
<[0 1] [1]>  
<[0 0] [0]>  
<[1 1] [1]>
```

We have written out the associations with with one association per line. On each line, the first sequence in square brackets is the input and the second sequence in square brackets is the desired output. In this case the inputs and outputs are bit vectors. Each input has two bits and each output has one bit. This means that we need a network with two input units and one output unit, see the figure below. (Note that this particular set of input/output associations correspond to the logic function OR.)

$\langle [0 \ 1] \ [1] \rangle$
 $\langle [0 \ 0] \ [0] \rangle$
 $\langle [1 \ 1] \ [1] \rangle$



Weights are randomly initialised. Following each presentation, a test is made to see whether the output ~~unit's activation~~ ~~was too high or too low~~ (ie. 1 when it should have been 0, or 0 when it should have been 1). If the activation was too high then all incoming connections have their weights reduced by $r L_i$, where r is the learning rate and L_i is the activation of the unit at the other end of the connection. Over time this produces a set of weights that will give perfect performance (given that certain criteria are satisfied). The general form of the procedure is illustrated in the figure below.


```

29,F,f,f,f,f,f,f,f,f,f,f,f,t,2.5,t,2,t,91,t,1.42,t,64,f,?,other -> increased_binding_protein
43,M,t,f,f,f,f,f,f,f,f,f,f,f,t,0.05,f,?,t,160,t,1.03,t,156,f,?,other -> replacement_therapy
43,M,t,f,f,f,f,f,t,f,f,f,f,f,t,100,f,?,t,93,t,0.87,t,107,f,?,other -> underreplaced
85,F,f,f,f,f,f,f,t,f,f,f,f,f,t,44,t,2,t,38,t,1.03,t,37,f,?,SVI -> primary_hyperthyroid
30,F,t,f,f,f,f,f,f,f,f,f,f,f,t,0.1,t,3.1,t,202,t,1.74,t,116,f,?,STMW -> increased_binding_protein
84,F,t,f,f,f,f,f,f,f,f,f,f,f,t,0.3,f,?,t,154,t,0.97,t,159,f,?,other -> replacement_therapy
31,F,f,f,f,t,f,f,f,f,f,t,f,f,f,?,t,3,t,158,t,1.01,t,156,f,?,other -> hyperthyroid
29,M,f,f,f,f,f,f,f,f,f,f,f,t,2.6,t,1.6,t,135,t,0.86,t,157,f,?,other -> discordant_array_results
74,F,f,f,f,f,f,f,f,f,f,f,f,t,0.25,t,2.3,t,176,t,1.36,t,129,f,?,SVI -> increased_binding_protein
55,F,t,f,f,f,f,f,t,f,f,f,f,f,t,9.599999,t,2.4,t,136,t,1.48,t,92,f,?,other -> underreplaced
50,F,f,f,f,f,f,f,f,f,f,f,f,f,?,f,?,f,?,f,?,t,36,other -> elevated_TBG

```

Each line contains a single record. The fields of the records, which are separated by commas, represent the attributes of a particular patient. The attributes and their ranges are as shown in the table shown below. The final field of each record is separated by a preceding arrow. The value of the field is the diagnosis that was made for the individual in question.

	Attribute Name	Possible Values
1	age:	continuous.
2	sex:	M, F.
3	on thyroxine:	f, t.
4	query on thyroxine:	f, t.
5	on antithyroid medication:	f, t.
6	sick:	f, t.
7	pregnant:	f, t.
8	thyroid surgery:	f, t.
9	I131 treatment:	f, t.
10	query hypothyroid:	f, t.
11	query hyperthyroid:	f, t.
12	lithium:	f, t.
13	goitre:	f, t.
14	tumor:	f, t.
15	hypopituitary:	f, t.
16	psych:	f, t.
17	TSH measured:	f, t.
18	TSH:	continuous.
19	T3 measured:	f, t.
20	T3:	continuous.
21	TT4 measured:	f, t.
22	TT4:	continuous.
23	T4U measured:	f, t.
24	T4U:	continuous.
25	FTI measured:	f, t.
26	FTI:	continuous.
27	TBG measured:	f, t.
28	TBG:	continuous.

```

29 referral source:      WEST, STMW, SVHC, SVI, SVHD, other.
30 diagnosis           no_condition. hyperthyroid etc.

```

From these records we can derive a training set of examples. Each record gives us one example. The attributes provide us with the components of the input and the diagnosis provides us with the desired output. If we wish to use this training set of examples in training a neural network then we obviously have to convert the attribute values etc. into activation values. This can be done in a number of ways.

Below we see a training set that we have derived by (1) selecting a subset of particularly informative attributes⁵ and (2) converting the various attribute values into activation values (ie. numbers between 1 and 0). We have converted continuous attributes by normalising them into the range 0..1 and nominal attributes by mapping them onto orthogonal bit vectors.

```

{{0 0 0 0 0 0.10582 1 0 0 0} {1 0 0 0 0}}
{{0.055704 0.062147 0.042155 0.469388 0.023311 0 1 0 0 0}
 {0 1 0 0 0}}
{{0 0 0 0 0 0.470899 1 0 0 0} {1 0 0 0 0}}
{{0 0 0.531616 0.530612 0.256426 0 1 0 0 0} {0 0 1 0 0}}
{{0 0.412429 0.564403 0.455782 0.303048 0 0 1 0 0} {0 0 1 0 0}}
{{0.044075 0.254237 0.147541 0.551021 0.069934 0 0 0 1 0}
 {0 1 0 0 0}}
{{0.001512 0.265537 0.264637 0 0 0 1 0 0 0} {0 0 0 1 0}}
{{0 0 0 0 0 0.10582 1 0 0 0} {1 0 0 0 0}}
{{0 0.231638 0.210773 0.40136 0.12373 0 0 1 0 0} {0 0 0 1 0}}
{{0 0 0 0 0 0.132275 1 0 0 0} {1 0 0 0 0}}
{{0.006861 0.220339 0.206089 0.428572 0.116557 0 0 1 0 0}
 {0 0 0 1 0}}
{{0.130131 0.062147 0.046838 0.496599 0.023311 0 1 0 0 0}
 {0 1 0 0 0}}
{{0.00221 0.299435 0.269321 0 0 0 0 0 1 0} {0 0 0 1 0}}
{{0 0 0 0 0 0.206349 1 0 0 0} {1 0 0 0 0}}
{{0 0 0 0 0 0.132275 1 0 0 0} {1 0 0 0 0}}

```

The network used for the experiment is shown in the figure below. It has ten input units, three hidden units and five output units. All layers are fully interconnected.

⁵The following attributes were used TSH T3 TT4 T4U FTI TBG and referral_source.

If we look at the performance of the network before any training has taken place we see that the diagnoses produced (ie. the activations of the output vectors) do not correspond to any one of the desired output vectors. The outputs generated are shown in the following listing.⁶

```
Testing on training set
[pair 1 :
  [input {? ? ? ? ? 31 other}]
  [target {1 0 0 0 0}]
  [output [0.5814 0.5556 0.4712 0.5473 0.4669]]
  [learner_error 0.4919]]
[pair 2 :
  [input {24 0.6 21 0.97 22 ? other}]
  [target {0 1 0 0 0}]
  [output [0.5825 0.5555 0.4703 0.5495 0.4651]]
  [learner_error 0.5024]]
[pair 3 :
  [input {? ? ? ? ? 100 other}]
  [target {1 0 0 0 0}]
  [output [0.5799 0.5545 0.4705 0.5457 0.4684]]
```

⁶This was generated using the Poplog *learners* environment.

```

[learner_error 0.4918]]
[pair 4 :
[input {0.05 ? 230 1.06 217 ? other}]
[target {0 0 1 0 0}]
[output [0.585 0.5565 0.4714 0.5468 0.4668]]
[learner_error 0.5367]]
[pair 5 :
[input {0.05 3.7 244 0.95 256 ? SVI}]
[target {0 0 1 0 0}]
[output [0.5842 0.5534 0.4683 0.5447 0.469]]
[learner_error 0.5366]]
[pair 6 :
[input {19 2.3 66 1.09 61 ? SVHC}]
[target {0 1 0 0 0}]
[output [0.5725 0.5496 0.4656 0.5491 0.4674]]
[learner_error 0.501]]
[pair 7 :
[input {0.7 2.4 116 ? ? ? other}]
[target {0 0 0 1 0}]
[output [0.5834 0.5558 0.4712 0.5453 0.4681]]
[learner_error 0.5067]]

```

Following a certain amount of training the overall error of the network (ie. the overall degree to which the output values differ from desired output values) falls close to zero. At this point the performance is much improved and the network satisfactorily produces correct diagnosis in over 95% of all (seen) cases and over 92% of unseen cases.⁷ The input/output performance exhibited by the network for a small selection of examples is as follows.

```

** [pair 1 :
[input {? ? ? ? ? 31 other}]
[target {elevated_TBG}]
[output {elevated_TBG}]
[learner_error 0.0819]]
** [pair 2 :
[input {24 0.6 21 0.97 22 ? other}]
[target {primary_hypothyroid}]
[output {primary_hypothyroid}]
[learner_error 0.0]]
** [pair 6 :
[input {19 2.3 66 1.09 61 ? SVHC}]
[target {primary_hypothyroid}]
[output {primary_hypothyroid}]
[learner_error 0.0249]]
** [pair 7 :

```

⁷This performance is comparable to that reported in [9] for the same training set.

```

[input {0.7 2.4 116 ? ? ? other}]
[target {no_thyroid_condition}]
[output {no_thyroid_condition}]
[learner_error 0.0019]
** [pair 17 :
[input {140 ? 33 1.07 31 ? other}]
[target {primary_hypothyroid}]
[output {primary_hypothyroid}]
[learner_error 0.0001]]

```

6 How should we view neural nets?

Demonstrations such as this, in which a neural network is trained to perform expert diagnoses tend to give the impression that the technology of neural networks is an attractive alternative to classical artificial intelligence. It seems as if — with neural networks — we can avoid the long-winded process of knowledge engineering that is required with the AI approach. Using an easily-derived training set of examples and a fairly straightforward algorithm we can then derive a powerful, intelligent system. Furthermore, because neural units are technically quite simple, we have the opportunity to exploit the speed and efficiency advantages of hardware implementation.

Of course appearances are deceptive. Although there is no doubting the fact that neural networks certainly can be used in this way — and in fact, the literature is full of examples in which neural networks have been trained to perform much more impressive feats of expertise than the one shown above — it has to be noted that the crucial element in the scenario is not the neural network itself but rather the learning algorithm that is applied to it. And as has now been shown in many cases, the learning performance that we can obtain with neural algorithms such as back-propagation can usually be matched — if not improved on — by conventional, symbolic learning algorithms and statistical procedures.

For example, Mooney et al. carried out an experimental comparison of symbolic and subsymbolic learning algorithms and, following some experiments with one symbolic algorithm (ID3) and two subsymbolic ones (back-propagation and perceptron) concluded that “the probability of correctly classifying new examples is about the same for the three systems, although there is indication that back-propagation classifies more accurately on noisy complex data sets.” [7, p. 779]. A table showing an approximation of their results is shown in the figure below.⁸ The labels along the top of the table correspond to well-known training-sets from the machine learning literature.⁹

Similar results are reported by Weiss and Kapouleas [8] following an empirical comparison against various statistical techniques. Fisher and McKusick also report comparable performance on a variety of tasks between ID3 — one of

⁸The percentages were derived from visual inspection of their histogram.

⁹Talk-A and Talk-full are two versions of the well-known NETtalk training set [12].

	Soybeans	Chess	Audiology	Talk-A	Talk-full
ID3	89%	96%	75%	55%	50%
Back-propagation	95%	96%	76%	63%	59%
Perceptron	92%	95%	72%	52%	48%

Figure 1:

the main symbolic learning algorithms and back-propagation, the leading neural network learning algorithm.

From the point of view of learning, then, there is no sense in which the subsymbolic paradigm with its neural network architectures has “defeated” the classical AI paradigm. It has unquestionably added a new dimension to it and contributed a wealth of new algorithms and techniques which need to be carefully explored and compared against existing methods. This extra dimension is particularly welcome now that the researchers are looking more eagerly for ways of exploiting parallel computing machinery. As Harrow Barrow has suggested, the artificial intelligence community should regard neural networks as neither threat nor menace but rather as an *opportunity* for further, productive interaction and exploration [13]. It is certainly the case that recent empirical work emphasizes the good sense of this philosophy.

References

- [1] Minsky, M. and Papert, S. (1969). *Perceptrons*. Cambridge, Mass.: MIT Press.
- [2] Evans, T. (1968). A program for the solution of a class of geometric-analogy intelligence-test questions. In M. Minsky (Ed.), *Semantic Information Processing*. Cambridge, Mass.: MIT Press.
- [3] Winston, P. (1970). Learning structural descriptions from examples. AI-TR-231, Ph.D thesis, Cambridge, Mass.: MIT AI Lab.
- [4] Winston, P. (1975). Learning structural descriptions from examples. In P. Winston (Ed.), *The Psychology of Computer Vision*. McGraw-Hill.
- [5] Ackley, D., Hinton, G. and Sejnowski, T. (1985). A learning algorithm for boltzmann machines. *Cognitive Science*, 9 (pp. 147-68).
- [6] Rumelhart, D., Hinton, G. and Williams, R. (1985). Learning internal representations by error propagation. ICIS report, Institute for Cognitive Science, UCSD.

- [7] Mooney, R., Shavlik, J., Towell, G. and Gove, A. (1989). An experimental comparison of symbolic and connectionist learning algorithms. *Proceedings of the Eleventh International Joint Conference On Artificial Intelligence* (pp. 775-780). Morgan Kaufmann.
- [8] Weiss, S. and Kapouleas, I. (1989). An empirical comparison of pattern recognition, neural nets and machine learning classification methods. *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence* (pp. 781-787). Morgan Kaufmann.
- [9] Fisher, D. and McKusick, K. (1989). An empirical comparison of ID3 and back-propagation. *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence* (pp. 788-793). Morgan Kaufmann.
- [10] Rosenblatt, F. (1962). *Principles of Neurodynamics*. New York: Spartan Books.
- [11] Minsky, M. and Papert, S. (1988). *Perceptrons: An Introduction to Computational Geometry* (expanded edn). Cambridge, Mass.: MIT Press.
- [12] Sejnowski, T. and Rosenberg, C. (1987). Parallel networks that learn to pronounce english text. *Complex Systems, 1* (pp. 145-68).
- [13] Barrow, H. (1989). AI, neural networks and early vision. *AISB Quarterly*, No. 69 (pp. 6-25).