10: Drawing things together - some perspectives

Kevin Gurney

Dept. Human Sciences, Brunel University
Uxbridge, Middx.

It is the intention of this final part to look back on the technical material and try to perceive some overall structure. This is done principally by developing a neural net taxonomy or classification scheme but also by providing historical and disciplinary perspectives.

1 Nets vs. Symbols (revisited)

This was the point of departure at the beginning of the course but we review the two paradigms again, this time in more depth and with some historical background.

From the early days of computing in the late 1940s and early '50s, there have existed two approaches to the problem of developing machines which exhibit 'intelligent' behaviour. One of these tries to capture knowledge in some domain as a set of atomic semantic objects or *symbols*, and to manipulate these according to a set of formal algorithmic rules. This symbolic- algorithmic paradigm has, over the last twenty years, represented the mainstream of research in Artificial Intelligence, and indeed the very term 'AI' is usually taken to refer to this school of thought.

Concurrent with this however, has been another line of research which uses machines whose architecture is loosely based on that of the animal brain, and which learn from a training environment, rather than being preprogrammed in some high level computer language. Work with these so-called *neural networks* was very active in the 1960s, suffered a loss of popularity, during the '70s and early '80s, but is now enjoying a revival of interest.

1.1 The symbolic paradigm

Although the first electronic digital computers were designed to perform numerical calculations, it was apparent to the early workers that the machines they had built were also capable of manipulating symbols, since the machines themselves knew nothing of the semantics of the bit-strings stored in their memories. Thus Alan Turing speaking in 1947 about the design for the proposed Automatic Computing Engine (ACE), saw the potential to deal with complex game playing situations 'Given a position in chess the machine could be made to list all the "winning combinations" to a depth of about three moves....' quoted from [18].

The machines on which the modern AI fraternity now run their algorithms have not changed in any fundamental conceptual way from the Pilot ACE which was eventually built; all of these being examples of the classic Von Neumann architecture. Granted, there has been a speed increase of several orders of magnitude, and hardware parallelism is sometimes available, but contemporary 'AI engines' are still vehicles for the instantiation of the theoretic stance which claims that cognition can be described completely as a process of formal, algorithmic symbol manipulation.

Mainstream AI has proved successful in many areas and, indeed, with the advent of expert systems has become big business. For a brief history of its more noteworthy achievements see [29].

However AI has not fulfilled much of the early promise that was conjectured by the pioneers in the field. This is brought home by Dreyfus in his book 'What Computers Can't Do' [8] where he criticizes the early extravagant claims and outlines the assumptions made by AI's practitioners. Principal among these are the belief that all knowledge or information can be formalised, and that the mind can be viewed as a device that operates on information according to formal rules. It is precisely in those domains of experience where it has proved extremely difficult to formalise the environment, that the 'brittle' rule-based procedures of AI have failed.

The differentiation of knowledge into that which can be treated formally and that which cannot, is made explicit by Smolensky (1988) [34] where he makes the distinction between *cultural*, or *public knowledge* and *private*, or *intuitive knowledge*. The stereotypical examples of the former are found in science and mathematics, whereas the latter describes, for instance, the skills of a native speaker or the intuitive knowledge of an expert in some field. In the *connectionist* view, intuitive knowledge cannot be captured in a set of linguistically formalized rules and a completely different strategy must be adopted.

1.2 The connectionist paradigm

The central idea is that in order to recreate some of processing capabilities of the brain it is necessary to recreate some of its architectural features. Thus a connectionist machine, or *neural net*, will consist of a highly interconnected network of comparatively simple processors (the *nodes*, *units* or *artificial neurons*) each of which has a large fan-in and fan-out. In biological neurons the distinctive processing ability of each neuron is supposed to reside in the electro-chemical characteristics of the inter-neuron connections, or synapses. In many connectionist systems this is modeled by assigning a connection strength or *weight* to each input. However, there are other ways of associating a set of parameters to a node which capture its functionality as in, for example, the cubic nodes. In all cases the net ensemble of these is the embodiment of the knowledge the system possesses.

Moving to dynamics, biological neurons communicate by the transmission of electrical impulses, all of which are essentially identical, so that information is contained in the spatio-temporal relationships between them. Neurons are continually summing or *integrating* the effects of all its incoming pulses and, depending on whether the result is excitatory or inhibitory, an output pulse may or may not be generated. In artificial nets, each node continually updates its state by generating an internal *activation value* which is a function of its inputs and internal parameters. This is then used to generate an output via some activation-output function which is typically a squashing function like the sigmoid.

At the system level, it is possible to draw up a list of features displayed by many artificial neural nets but, since the connectionist banner has been attached to such a wide diversity of systems, any such list will inevitably not apply to all of them; however the following is a typical set of characteristics:

- The node parameters are trained to their final values by continually presenting members from a set of patterns or *training vectors* to the net, allowing the net to respond to each presentation, and altering the weights accordingly; that is, they are adaptive rather than preprogrammed systems.

- Their action under presentation of input is often best thought of as the time evolution of a dynamical physical system and there may even be an explicit description of this in terms of a set of differential equations. This was the nature, for example, of the continuous valued Hopfield net and the competitive nets.

- They are robust under the presence of noise on the inter-unit signal paths and exhibit graceful degradation under hardware failure. (Recall the examples on the 1st problem sheet)

- A characteristic feature of their operation is that they work by extracting statistical regularities or *features* from the training set. This allows the net to respond to novel inputs in a useful way by classifying them with one of the previously seen patterns or by assigning them to new classes. (Recall the simulation exercise with the 11-input TLU)

- Typical modes of operation are as associative memories, retrieving complete patterns from partial data (Recall the Hopfield net demo), and as pattern classifiers (The A/B classification with delta rule simulation)

- There is no simple correspondence between nodes and high level semantic objects. Rather, the representation of a 'concept' or 'idea' within the net is via the complete vector of unit activities, being distributed over the net as a whole, so that any given node may partake in many semantic representations. Think about the 'codes' in the 424 encoders etc. which form distributed (internal) representations of the vectors which are then explicitly represented in a one-to-one way at the outputs.

Concerning this last point there is a divergence of opinion in the connectionist camp. Many workers have exhibited nets that contain at least some nodes which denote high level semantic constructs [32, 37]. In [34] Smolensky argues for the 'Proper treatment of Connectionism', in which nets can only operate at a *sub-symbolic* level and where there are no local high- level semantic representations. He notes that, otherwise, connectionism runs the risk of degenerating into a parallel implementation of the symbolic paradigm. Indeed, high level semantic nets have been studied outside the connectionist milieu as 'pulse networks' on weighted digraphs [7] where no 'neural' analogy is implied.

2 A brief history of neural nets

2.1 The early years

The ideas concerning machines that incorporate neural features have always been contemporaneous with work on the general purpose computers in common use today. In fact the analogy between computing and the operation of the brain was to the fore in much of the early work in this area. Thus von Neumann, in the first draft of a report on the EDVAC [35], makes several correspondences between the proposed circuit elements and animal neurons.

In 1942 Norbert Weiner [15] and his colleagues were formulating the ideas that were later christened 'Cybernetics', and which he defined as 'control and communication in the animal and the machine'. Central to this programme, as the description suggests, is the idea that biological mechanisms can be treated from an engineering and mathematical perspective. Of central importance here is the idea of *feedback*

[demo xpt on feedback - 'fingers together']

With the rise of AI and cognitive science, the term 'Cybernetics' has become unfashionable in recent years [1] it might be argued that, because of its interdisciplinary nature, the new- wave of connectionism should properly be called a branch of Cybernetics: certainly many of the early neural-net scientists would have described their activities in this way.

In the same year that Weiner was formulating Cybernetics, McCulloch and Pitts [26] published the first formal treatment of artificial neural nets. The main result in this historic (but opaque) paper, as summarized in [36] is that any well defined input-output relation can be implemented in a formal neural network.

One of the key attributes of nets is that they can learn from their experience in a training environment. In 1949, Donald Hebb [13] indicated a mechanism whereby this may come about in real animal brains. Essentially, synaptic strengths change so as to reinforce any simultaneous correspondence of activity levels between the pre-synaptic and post-synaptic neurons. Translated into the language of artificial neural nets, the weight on an input should be augmented to reflect the correlation between the input and the unit's output. Learning schemes based on this 'Hebb rule' have always played a prominent role.

The next milestone is probably the invention of the *Perceptron* by Rosenblatt in 1957; much of the work with these is described in his book 'Principles of Neurodynamics' [30]. One of the most significant results presented there was the proof that a simple training procedure (the perceptron training rule - lecture 2) would converge if a solution to the problem existed.

Rumelhart and Zipser (1985) [31] give some interesting anecdotes and reminiscences from this era.

In 1969 enthusiasm for neural nets was dampened somewhat by the publication of Minsky and Papert's book 'Perceptrons' [27]. Here, the authors show that there is an interesting class of problems (those that are not linearly separable) that single layer perceptron nets cannot solve, and they held out little hope for the training of multi-layer systems that might deal successfully with some of these. Minsky had clearly had a change of heart since 1951, when he and Dean Edmonds had built one of the first network-based learning machines. The fundamental obstacle to be overcome is the so-called 'credit assignment problem': in a multilayer system, how much does each unit (especially one not in the output layer) contribute to the error the net has made in processing the current training vector? (This is the problem to which BP is addressed).

In spite of 'Perceptrons', much work continued in what was now an unfashionable area, living in the shadow of symbolic AI: Grossberg was laying the foundations for his Adaptive Resonance Theory (ART) [6, 11]. Fukushima was developing the cognitron [9]; Kohonen was investigating nets that used topological feature maps [22] (lecture 7), and Aleksander [2] was building hardware implementations of the nets based on the n-tuple technique of Bledsoe and Browning [5].

There have been several developments over the last few years that have led to a resurgence of interest in nets. Some of these factors are technical and show that Minsky and Papert had not said the last word on the subject, while others are more general; in the next section we look at both these strands of thought.

2.2 The neural net renaissance

In 1982 John Hopfield [19], then a physicist at Caltech, showed that a highly interconnected network of threshold logic units could be analyzed by considering it to be a physical dynamic system possessing an 'energy'. The process of associative recall, where the net is started in some initial random state and goes on to some stable final state, parallels the action of the system falling into a state of minimal energy.

This novel approach to the treatment of nets with feedback loops in their connection paths (recurrent nets), has proved very fruitful and has led to the involvement of the physics community, as the mathematics of these systems is very similar to that used in the Ising-spin model of magnetic phenomena in materials [3]. In fact something very close to the 'Hopfield model' had been introduced previously by Little , but here the emphasis was on this analogy with spin systems, rather than the energy-based description. Little [24] also made extensive use of a quantum mechanical formalism, and this may have contributed to the paper's lack of impact.

A similar breakthrough occurred in connection with non-recurrent (feedforward) nets, when it was shown that the credit assignment problem had an exact solution. The resulting algorithm, 'Back error propagation' or simply *Backpropagation* also has claim to multiple authorship, as noted by Grossberg in [12]. Thus it was discovered by Werbos [38] rediscovered by Parker [28], and discovered again and made popular by Rumelhart, Hinton and Williams [32].

Aside from these technical advances in analysis, there is also a sense in which neural networks are just one example of a wider class of systems that the physical sciences have started to investigate which include cellular automata [33], fractals [25], and chaotic phenomena [16]. Traditionally physics has sought to explain by supposing that there is some simple underlying global model, usually consisting of a differential equation, to which the real situation is supposed to approximate as a perturbation. Now some investigators are meeting the complexity of dynamical systems 'head on' as it were, and are supposing that any macroscopic ordering is an emergent property, arising from the action of a large number of simple stochastic elements acting in concert. Statistical physics represents the vanguard of this movement that now includes the areas noted above. Neural nets are also complex dynamical systems with emergent collective properties, as noted by Hopfield in the title of his original paper [19] and it is not surprising, in hindsight, that the solid-state statistical physicists have managed to apply their theories to the subject.

Now in order to investigate these new models, it is often necessary to resort to computer simulation, and the power of this method has obviously been greatly enhanced by advances in technology. Thus certain experiments would simply have been unthinkable fifteen years ago by the average worker, given the accessibility and power of the computing resources available.

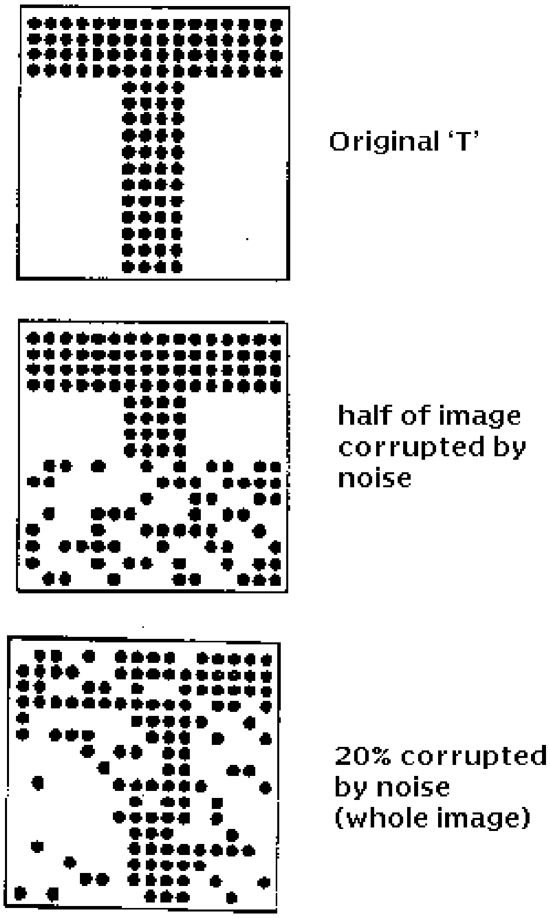
In summary therefore, we may say that there has, in recent years, been a marked increase in neural net research because of new mathematical insights, a conducive scientific *zeitgeist*, and enormous improvement in the ability to perform simulation.

3 Classifying neural net structures

On a first exposure to the neural net literature, it may seem that the whole area is simply a large collection of different architectures, node, types etc., which are somewhat *ad hoc* and which don't appear related in any way. It is the intention of this section to try and give a framework for classifying any network which enables it to be viewed as a special instance of, or as a composite of, only a handful of structures. All of these have been presented in previous lectures, we simply draw them together here for a comparative review. We start, however by reviewing the types of task that neural nets can perform.

3.1 Neural net tasks

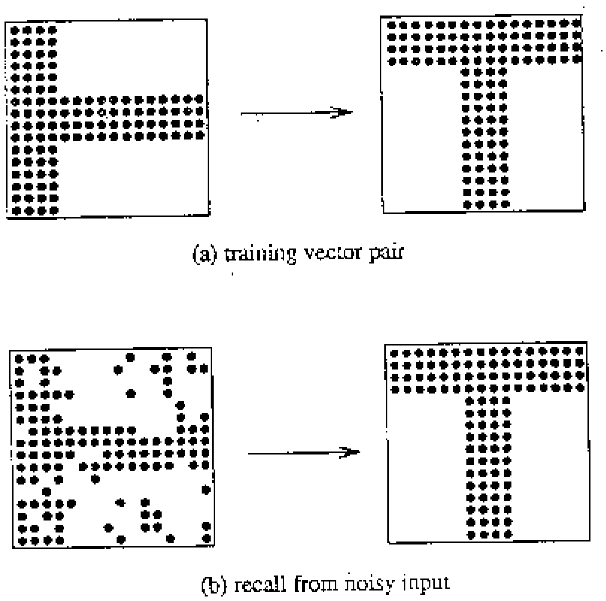
Recall the example of pattern recall with the 'T's and the examples in the Hopfield net demo (figure reproduced here for clarity).



[slides of Ts]

The middle image is split into two halves: the top half or *key* is an uncorrupted portion of the original input vector; the rest of the input is random noise. If the net can retrieve the original training pattern from (b) then it is said to be acting as a *content addressable memory* or CAM. The lower image shows the original 'T' with the uniform addition of noise obtained by inverting each pattern element with probability 0.2. If the net can retrieve the original image then it is now behaving as a filter or noise extractor. Both of these are instances of *auto-associative recall* where a partial or noisy image is restored to some prototype.

The top image on the second slide shows a training vector pair, with input and output vectors on the left and right respectively.



[hetero-associative recall on ‘T’s]

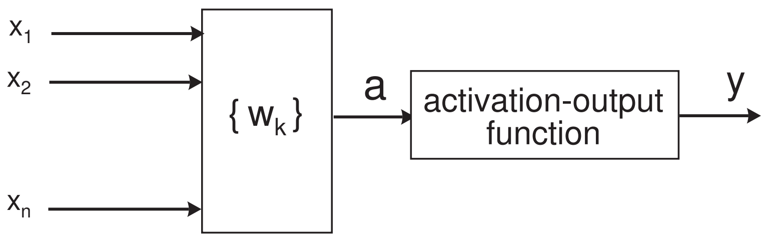
The lower image shows a noisy input recalling the original output vector. The split of the training pattern into two dissimilar halves, leads to the name *hetero-associative recall* for this process.

Suppose however, that in the last example we had unified the input and output patterns side by side into a single training vector. By keying on the left half of this and performing auto- association, we obtain a similar result (although not perhaps as noise resilient) to that obtained in the case of hetero- association. Thus conceptually both hetero- and auto-association can perform similar tasks; this point of view is useful later in comparing modes of learning.

As a final task, consider the hetero-associative case where the target or output pattern is much smaller than that on the input, and where it represents a highly compressed encoding of the input set (possibly with one node ‘on’, or several nodes ‘on’). In this case we talk of the net performing *pattern classification*, and introducing it in this way, classification can be seen as a special case of hetero-association.

3.2 A taxonomy of artificial neurons

It is useful to define the functional behaviour of all the unit types using the *activation-output* model introduced for the discussion of cubic nodes.



[activation-output model]

x_i is a set of n inputs, and ζ_k, a set of N internal parameters. The activation a is a function of the inputs and parameters, $a = a(\zeta_k, x_i)$. The output y is a function of the activation,

$y = y(a)$, defined by the activation-output relation. Splitting the node functionality up like this can assist the understanding of what 'ingredients' have been used in the node make-up.

We now describe some common node types and use these to draw up a list of criteria that may be used to classify them.

3.2.1 Nodes using a linear weighted sum of inputs

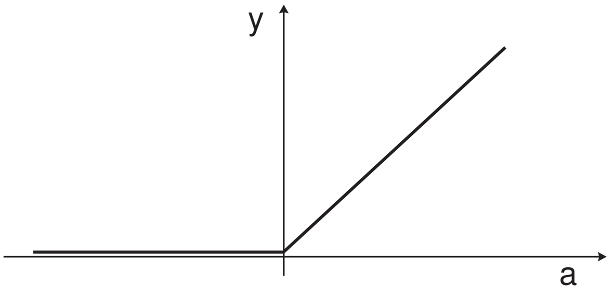
These are by far the most common. The parameters ζ_k now consist of a set of n *weights* w_i, one for each input, and a threshold parameter θ, that may be thought of as a weight associated with an input that is tied to '-1'. The activation a, is given by

$$a = \sum_{i=1}^{n} w_i x_i - \theta \tag{1}$$

The simplest possible activation-output a-o relation to use in conjunction with (1) is to put $y = a$. The resulting unit is found in the counterpropagation nets of Hecht-Nielson [14] Note that y takes on a continuous range of values and so networks of these units are computationally analogue.

Other possibilities for a-o include the threshold function used in TLUs which can therefore only use Boolean values. The other popular output law is based on the sigmoid function. Another function, used by Fukushima in his cognitron [18] is given by

$$y = \begin{cases} 0 & \text{if} \quad a < 0 \\ a & \text{if} \quad a \geq 0 \end{cases} \tag{2}$$



[ramp activation]

As well as using y directly as an analogue value, it is possible to interpret it as the probability that a '1' is emitted from the output in a stochastic, boolean valued node. This is used to good effect in the Reward Penalty algorithms [4] and the Boltzmann Machines [17].

The main feature of the hard-limiter and sigmoid functions is their nonlinearity, a feature which can be highly significant in the way that neural nets process information, and which Grossberg discusses at length in [12].

3.2.2 More complex nodes

It is possible to extend the simple activation relation (1) so that rather than just summing over terms linear in the inputs, we include higher order terms that are multilinear in these variables. This gives us the sigma-pi nodes. Any of the activation-output relations described above may be used with the sigma-pi activation.

Alternatively, we may define a node whose inputs and outputs are boolean, and which can perform *any* (perhaps stochastic) boolean function of its inputs (not necessarily linearly separable). The ζ_k are then the site-values at the corners of n-cube and we have the so-called cubic nodes.

3.2.3 Criteria for classification

In the light of the above examples, it is proposed that the following is a list of features that may act as dimensions in a 'node space' in the study of neural nets.

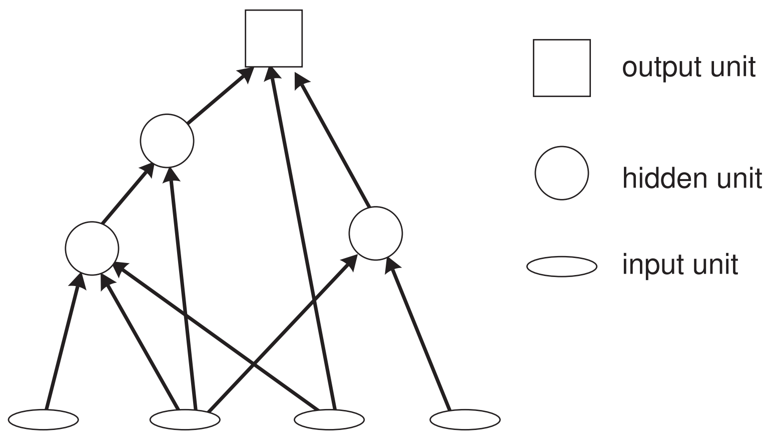
- The form of the function $a(\zeta_k)$ - e.g. linear, sigma-pi, cubic.
- The activation-output relation - linear, hard-limiter, or sigmoidal.
- The nature of the signals used to communicate between nodes - analogue or boolean.
- The dynamics of the node - deterministic or stochastic.

3.3 A taxonomy of net structures

The last section dealt with an analysis of the micro-structure of the net at node level; here we deal with the overall topology and dynamics. Quite generally, in connection with the latter, we usually recognize two phases of net operation: a *training phase*, when the node parameters ζ_k are being changed according to some learning rule, and a *testing phase* when these variables remain static and when we are particularly interested in the net's response to new, unseen patterns.

3.3.1 Feedforward nets

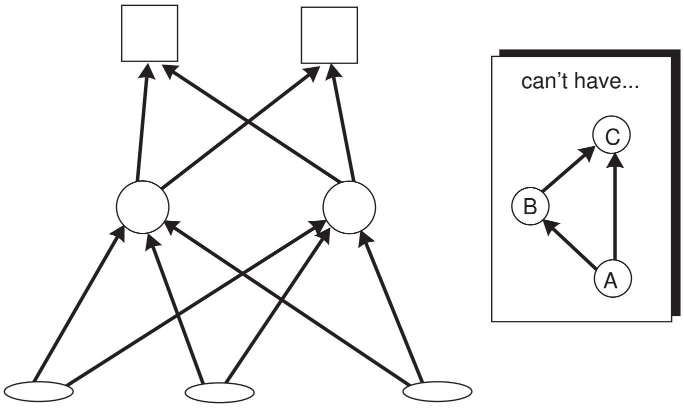
These are nets that contain no feedback loops in their inter-node connection paths. Thus, starting from any node in the net and tracing its fan-out, there will come a point after a finite number of steps when this process terminates at an *output node* which does not have a fan-out. Complementary to these are a set of *input nodes* or *input terminals* which have no fan-in; they have no functionality and are merely fan-out distribution points. The most general feedforward net is shown below



[general feedforward net]

The input and output nodes are accessible to the training environment and may therefore have signals directly imposed on them from outside. In between these two are a set of nodes referred to as *hidden* since they may not be accessed by the environment. It is the task of these units to develop an internal representation of the training set. In this context the hidden units are sometimes thought of as extracting *features*, or *encoding* the environment [10]. This is most apparent if there are fewer hidden than input nodes, and the net must perform some data compression. This is the basis for the operation of the encoder nets which were used in the simulation exercises.

One specialization of the general feedforward net is that where it is possible to distinguish a series of distinct layers. Formally this may be described in the following way: there does not exist a set of three units A, B and C, such that C receives input from A and B, and B receives input from A. The forbidden structure is shown below together with a typical 3-layer feedforward net.



[forbidden layered structure and a layered net]

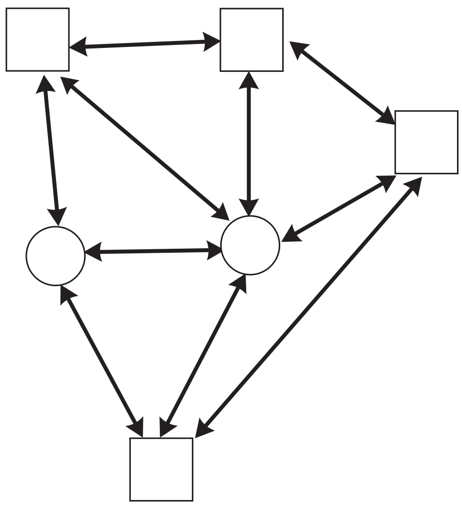
The dynamics of feedforward nets is straightforward. In the layered case, inputs are applied to the input nodes and subsequent layers evaluate their node outputs until the output layer is reached. In the more general case care has to be taken to ensure that all the inputs to a given node are valid before its evaluation. This whole process is referred to as a *forward pass.*

The perceptron rule and delta rule could be used to train feedforward nets with no hidden units. The backpropagation algorithm allowed training of hidden units in these nets.

3.3.2 Recurrent nets: topology

These are nets which have feedback loops in their inter-node connection paths. Thus there must exist at least one node such that, if the fan-out of that node is traced forward, then after a finite number of steps the node is revisited. In recurrent nets there is no distinction between input and output, rather there is a set of *visible units* which interface with the environment and, as in the feedforward case, a number of hidden units which do not. The way this interface operates is discussed below under 'dynamics'. The Hopfield net was an example of a recurrent net with no hidden units. The Boltzmann machines [17] (not dealt with on this course) allow hidden units

A typical recurrent net is shown below



[recurrent net]

The Hopfield net was a special case where there was total interconnectivity and all connections were symmetric.

3.3.3 Recurrent nets: dynamics

There are two fundamental modes of operation - *synchronous* or *parallel* update, and *asynchronous* update. In the former, all the nodes evaluate their new output together, and the net is 'clocked' globally. In the latter case, at each time step a node is selected at random to be updated. Both synchronous and asynchronous modes can be thought of as extreme instances of the more general case where there is a probability p that any node gets updated. For a comparison of net behaviour in these two modes see [10].

In the case of synchronous dynamics, and with deterministic nodes, the net is a *deterministic finite automaton* with a characteristic state space, where the state of the net is just the vector of unit outputs. These spaces may be partitioned into sets of states or *confluents* such that all states belong to only one confluent, and transitions are confined to pairs of states within each one. Examples of confluent structure were shown in the figure 'state diagrams for synchronous update' in lecture 5.

confluent structures

All confluents can be divided into two parts: a tree like fragment or *transient* and a *cycle*. In the left hand part of the figure, the cycle consists of a single state returning to itself - a single-state cycle or 1-cycle; in the right hand figure it includes several states, giving a multiple-cycle. Starting the net in a random state will usually mean beginning on a transient, and subsequent transitions will be made until a cycle is reached. Therefore this type of dynamics implies that we have to wait for the net to reach some kind of equilibrium.

Asynchronous operation implies the net is no longer deterministic, since nodes are selected for update at random. Therefore although we can define the state of the net at any time as before, the state structure is probabilistic and more complex than the one given above. (This leads to diagrams like the ones used with the Hopfield net). The concept of reaching equilibrium can be retained by the introduction of an 'energy function' associated with the net; this was one of the main insights given by Hopfield who showed that the process of coming to equilibrium may be thought of as one of energy minimisation. In this way memories could be thought of as states associated with stable energy minima, into which random starting states would eventually fall. The energy is chosen so that each neuron update implies a decrease in its value, and so we imagine an energy 'landscape' where, even though the next state of the system is not known *a priori*, the transition is guaranteed to be 'downhill' in the energy space.

We now turn to the question of how the network interacts with the environment. In training, a vector will be applied to these nodes in such a way that their outputs take the same values as the corresponding vector components. If there are any hidden nodes, the net is allowed to reach equilibrium by allowing these to update, if there are none we proceed directly to the use of the learning rule where the weights are then changed.

Under test, there are (at least) two ways of inputting information. One is to clamp a subset of the visible units and keep the clamp on for the entire test, while the net reaches equilibrium. The clamp can be freely reselected at each trial from the entire collection of visible units. This corresponds to the use of the net as a content addressable memory (see 'node tasks' section) and is the mode used with the Boltzmann Machines. Alternatively we may initialize the state of the net from an input vector, and then let the net operate completely freely, with *all* nodes partaking in the update process: this is the preferred mode when using the Hopfield nets and corresponds to recall from noisy data. Note, however, that the Hopfield net may also be used as a CAM.

3.4 Competitive learning nets

These are a kind of hybrid, where a feedforward structure contains at least one layer with intra-layer recurrence (see diagram in lecture 7). Each node is connected to a large neighbourhood of surrounding nodes by negative or *inhibitory* weighted inputs, and to itself (and possible a small local neighbourhood) via positive or *excitatory* inputs. These lateral connections are usually fixed in magnitude and do not partake in the learning process. It is also connected to the previous layer by a set of inputs which have trainable weights of either sign. The point of the lateral recurrent links is to enhance an initial pattern of activity over the layer, resulting in the node which was most active being turned fully 'on', and the others being turned 'off'; hence the label 'competitive' or 'winner-take-all' being applied to these nets.

The object here is to encode groups of patterns in a 1-out-of- n code, where each class is associated with an active node in the winner-take-all layer. This mechanism, or some minor variant, is a key component in Grossberg's ART [6, 11] Fukushima's cognitron [9] and Kohonen's nets that develop topological feature maps [21].

3.5 Criteria for classification

What was done for nodes in the last section is now done for net structures, so that the following is proposed as a set of attributes for net classification.

- Basic net topology - Recurrent , competitive, or feedforward.
- Hidden units - present or absent.
- Dynamics - Synchronous or asynchronous.

3.6 A taxonomy of training algorithms

These fall into three main categories. These are described individually and are followed by a discussion of the relation between the environment and the net, and the resources required for training.

3.6.1 Preprogramming

In this scheme the weights are not learnt as such by a gradual, iterative process of training vector presentation and update, rather they are fixed according to some prescription that makes use of all the vectors at once; in a sense the net is not trained but programmed. The best known example of this occurs in the Hopfield model.

3.6.2 Supervised learning

This is usually performed with feedforward nets where training patterns are composed of two parts, an input vector and an output vector, associated with the input and output nodes respectively. A training cycle consists of the following steps. An input vector is presented at the inputs together with a set of desired responses, one for each node, at the output layer. A forward pass is done and the errors or discrepancies, between the desired and actual response for each node in the output layer, are found. These are then used to determine weight changes in the net according to the prevailing learning rule.

The term 'supervised' originates from the fact that the desired signals on individual output nodes are provided by an external 'teacher'. The best known examples of this technique occur in the backpropagation algorithm, the delta rule and perceptron rule.

A popular measure of the error E for a single training pattern, is the sum of square differences

$$E = \frac{1}{2} \sum_i (t_i - y_i)^2 \tag{3}$$

where t_i is the desired or target response on the ith unit, and $y_i$is that actually produced on the same unit. This is appealing because of its simplicity, but is somewhat *ad hoc* (why not use the fourth power of differences?) and Hopfield [20] has argued for the use of an alternative measure based on information theoretic ideas.

3.6.3 Unsupervised learning

This is usually found in the context of recurrent and competitive nets. There is no separation of the training set into input and output pairs. Typically a training cycle will consist of the following steps: a vector is applied to the visible nodes (or in the case of competitive learning, the input nodes); the net is allowed to reach equilibrium (or a 'winner' established); weight changes are made according to some prescription. It is the amalgamation of input-output pairs, and hence the disappearance of the external supervisor providing target outputs, that gives this scheme its name. This kind of learning is sometimes referred to as *self-organization*.

3.6.4 Computational resources and complexity

In [39] Williams makes the useful distinction between 'on-line' and 'off-line' learning. In the latter the net is trained not only with its own resources, but also with the aid of an auxiliary central computing facility. This orchestrates the learning process and may pass information to, and gather information from subsets of nodes in the network . This is clearly the case in the preprogramming style of learning used for Hopfield nets. It is also arguably the case in the backpropagation type algorithms, where errors and weight values are accumulated at a node from its fanout. In on-line training, the net only makes use of the facilities offered internally within each node, together with perhaps, a minimal number of control signals to initiate various stages in each learning cycle.

The on-line versus off-line distinction may therefore be reformulated as a local versus global one. Learning is local if processing is takes place principally in individual units, and if these make use of information immediately available both in space and time. This means that we are restricting operations to the use of a node's current inputs and output; storage of previously calculated values being prohibited. Learning is off- line if large amounts of data are continually being processed and redistributed around the net along pathways that do not form part of its natural connectivity.

Conceptually it may be possible to reformulate an off-line algorithm as an on-line one, if the boundary between the environment and the net is redrawn, so that resources previously within the 'net' are now thought of as within a more supervisory environment. In spite of the informality of these distinctions, it is believed useful to include them in the general net taxonomy.

3.7 Final remarks

Clearly not all nets will fall neatly into the categories described here - some nets will contain mixtures of features described here - but it is believed that some such taxonomy is useful in approaching a field which now has a large proliferation of architectures, structures and algorithms. One facet which has not been dealt with here is the relation between the operation of neural nets and conventional pattern classification and recognition techniques. Some of these links are established by Lippmann [23] who also gives a review of some specific neural net types.

4 Neural nets in different disciplines

Neural nets have attracted attention from workers in many, apparently disparate, fields. It is ironic that the interdisciplinary fusion that was cybernetics should have fallen into disrepute in the West (the word is still fashionable in the Eastern bloc). Much of the work done in cybernetics now goes under the title of 'General Systems Theory'. Neural nets would be subsumed under its general remit and the interdisciplinary links required for it to flourish would have been forged. There are other, related areas which would now also fall under the cybernetics umbrella, and from which the study of neural nets could benefit. These include the use Genetic Algorithms (GAs), the study of cellular automata, robotics, adaptive control and mainstream AI. Many scientists now recognise that there is a common thread in all this research and the term 'artificial life' has been coined to embrace these areas collectively. The key question appears to be "How can assemblies of simple system elements exhibit self-organising behaviour patterns and adapt to their environment in ways that appear intelligent?".

Returning to the confines of neural networks, their study can be approached from several different points of view leading to different ideas as to their utility and the goals of the research.

Neuroscientists and psychologists are interested in nets as computational models of the animal brain developed by abstracting, what are believed to be, those properties of real nervous tissue that are essential for information processing. Many biologists are sceptical about the ultimate power of some of the more impoverished models of neurons like TLUs and insist that more detail is necessary to explain the brain's function; only time will tell.

On the other hand, engineers and computer scientists see neural nets as one style of *parallel distributed computing* which may usefully be recruited for solving complex problems in pattern recognition and classification, associative memory, and function optimisation. The hope is that they may be more successful in dealing with real-world situations than the conventional algorithmic techniques that have predominated in machine intelligence. They do not concern themselves with biological realism and are perhaps culpable of sometimes ignoring relevant architectural cues from natural systems.

Physicists and mathematicians are drawn to the study of networks from an interest in non-linear dynamical systems, statistical mechanics and automata theory. They often treat recurrent nets by (like Hopfield nets) by taking the limit as the number of neurons tends to infinity and dealing with the resulting network as a thermodynamic system which avails itself of all the machinery of statistical mechanics. There is contact to be made here with some of the concepts in adaptive control theory and the use of an 'energy' function may be traced back to Lyapunov; hence they are often referred to as Lyapunov functions.

Some of the threads running through much of this work are the ability of neural nets to learn and self-organise, to generalise from training data, and to process information in other ways normally thought of as intelligent.

References

[1] I. Aleksander. Whatever happened to cybernetics. Technical Report N/S/103, Dept. Electrical Engineering, Brunel University, 1980.

[2] I. Aleksander and T.J. Stonham. Guide to pattern recognistion using random-access memories. *Computers and digital techniques*, 2:29 – 40, 1979.

[3] D.J. Amit and H. Gutfreund. Spin-glass models of neural networks. *Physical Review A*, 32:1007 – 1018, 1985.

[4] A.G. Barto and M.I. Jordan. Gradient following without backpropagation in layered networks. In *1st Int. Conference Neural Nets, San Diego*, volume 2. 1987.
(I have this).

[5] W.W. Bledsoe and I. Browning. Pattern recognition and reading by machines. In *Proceedings of the Eastern Joint Computer Conference*, pages 225 – 232. 1959.

[6] G. Carpenter and S. Grossberg. A massively parallel architecture for a self-organizing neural pattern recognition machine. *Computer Vision, Graphics, and Image Processing*, 37:54 – 115, 1987.

[7] J. Casti. *Connectivity, Complexity and Catastrophe in Largescale Systems*. Wiley, 1979.

[8] H.L. Dreyfus. *What Computers Can't Do - The Limits of Artificial Intelligence*. Harper and Row, 1979.

[9] K. Fukushima. Cognitron: a self-organizing multilayered neural network. *Biological Cybernetics*, 20:121 – 136, 1975.

[10] R.O. Grondin, W. Porod, C.M. Loeffler, and D.K. Ferry. Synchronous and asynchronous systems of threshold elements. *Biological Cybernetics*, 49:1 – 7, 1983.

[11] S. Grossberg. Competitive learning: from interactive activation to adaptive resonance. *Cognitive Science*, 11:23 – 63, 1987.

[12] S. Grossberg. Nonlinear neural networks: Principles, mechanisms, and architectures. *Neural Networks*, 1:17 – 61, 1988.

[13] D.O. Hebb. *The Organization of behaviour.* John Wiley, 1949.

[14] R. Hecht-Nielson. Counterpropagation networks. In *1st Int. Conference Neural Nets, San Diego*, volume 2. 1987.
(I have this).

[15] S.J. Heims. *John von Neumann and Norbert Weiner - From Mathematics to the Technologies of Life and Death.* Academic Press, 1982.

[16] A. Hilger. *Universality in Chaos.* ?, Bristol.

[17] G.E. Hinton, T.J. Sejnowski, and D. Ackley. Boltzmann machines: Constraint satisfaction networks that learn. Technical Report CMU-CS-84-119, Carnegie Mellon University, 1984.

[18] A. Hodges. *Alan Turing - The Enigma of Intelligence.* Counterpoint (Unwin), 1985.

[19] J.J. Hopfield. Neural networks and physical systems with emergent collective computational properties. *Proceedings of the National Academy of Sciences of the USA*, 79:2554 – 2588, 1982. Hopfield has another, related, model which uses continuous outputs. Beware when reading the literature which model is being discussed.

[20] J.J. Hopfield. Learning algorithms and probability distributions in feed-forward and feedback networks. *Proceedings of the National Academy of Sciences of the USA*, 84:8429 – 8433, 1987.

[21] T. Kohonen. Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, 43:59 – 69, 1982.

[22] T. Kohonen. *Self-organization and associative memory.* Springer Verlag, 1984.

[23] R.P. Lippmann. An introduction to computing with neural nets. *IEEE ASSP Magazine*, pages 4 – 22, 1987.

[24] W.A. Little. The existence of persistent states in the brain. *Mathematical Biosciences*, 19:101 – 120, 1974.

[25] B.B. Mandelbrot. *The Fractal Geometry of Nature.* Freeman, 1977.

[26] W. McCulloch and W. Pitts. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 7:115 – 133, 1943.

[27] M. Minsky and S. Papert. *Perceptrons.* MIT Press, 1969.

[28] D.B. Parker. Learning-logic. Technical Report 581-64, Office of Technology Licensing, Stanford University, 1982.

[29] R. Raj. Foundations and grand challenges of artificial intelligence. *AI Magazine*, 9:9 – 21, 1988.

[30] F. Rosenblatt. *Principles of Neurodynamics.* Spartan Books, 1962.

[31] D. Rumelhart and D. Zipser. Feature discovery by competitive learning. *Cognitive Science*, 9:75 – 112, 1985.

[32] D.E. Rumelhart, G.E. Hinton, and R.J. Williams. Learning representations by back-propagating errors. *Nature*, 323:533 – 536, 1986.

[33] Several authors. Cellular automata - proceedings of an interdisciplinary workshop, los alamos. In *Physica, vol. 10D, p. (special volume)*. van Nostrand ?, 1983.

[34] P Smolensky. On the proper treatment of connectionism. *Behavioural and Brain Sciences*, 11:1 – 74, 1988.

[35] John von Neumann. First draft of a report on the edvac. In W. Aspray and A. Burks, editors, *Papers of John von Neumann on Computing and Computer Theory, vol 12 in the Charles Babbage Institute Reprint Series for the History of Computing*. MIT Press, 1987.

[36] John von Neumann. The general and logical theory of automata. In W. Aspray and A. Burks, editors, *Papers of John von Neumann on Computing and Computer Theory, vol 12 in the Charles Babbage Institute Reprint Series for the History of Computing*. MIT Press, 1987.

[37] D.L. Waltz and J.B. Pollack. Massively parallel parsing: A strongly interactive model of natural language interpretation. *Cognitive Science*, 9:51 – 74, 1985.

[38] P. Werbos. *Beyond regression: New tools for prediction and analysis in the behavioural sciences*. PhD thesis, Harvard University, Cambridge, MA., 1974.

[39] R.J. Williams. Reinforcement-learning connectionist systems. Technical Report NU-CCS-87-3, Northeastern University Boston, 1987.