Multiple Network Systems (Minos) Modules: Task Division and Module Discrimination[1]

F.J. Śmieja

German National Research Centre for Computer Science (GMD),
Schloß Birlinghoven, 5205 St. Augustin 1,
Germany.

ABSTRACT

It is widely considered an ultimate connectionist objective to incorporate neural networks into intelligent *systems.* These systems are intended to possess a varied repertoire of functions enabling adaptable interaction with a non-static environment. The first step in this direction is to develop various neural network algorithms and models, the second step is to combine such networks into a modular structure that might be incorporated into a workable system. In this paper we consider one aspect of the second point, namely: processing reliability and hiding of wetware details. Presented is an architecture for a type of neural expert module, named an *Authority.* An Authority consists of a number of *Minos* modules. Each of the Minos modules in an Authority has the same processing capabilities, but varies with respect to its particular *specialization* to aspects of the problem domain. The Authority employs the collection of Minoses like a panel of experts. The expert with the highest *confidence* is believed, and it is the answer and confidence quotient that are transmitted to other levels in a system hierarchy.

INTRODUCTION

The great majority of work on neural networks so far has concentrated on developing different models and observing their performance either on so-called "toy-problems" [13, 11, 19], in an attempt to improve them, or on highly preprocessed and isolated examples of aspects of real-world human processing [14, 20, 5, 12], in an attempt to demonstrate their potential. The ultimate objective is, with regard to Artificial Intelligence, to incorporate such networks into large hierarchical learning systems driven by centralized self-supervision and goal generators. We believe that only systems designed hierarchically mainly from modular neural network components will be able to solve the complex dynamic problems relevant to coping and evolving in a hostile environment. Ideally neural nets should enable the systems to adapt to new experiences in the environment, and the systems should respond in a controlled manner to new events and respond confidently to familiar ones. In order to achieve such a goal, one must first demand a degree of reliability from the constituting networks. The higher levels of the system should not need to be bothered with the low-level functioning and supervision of the basic networks, but at the same time need to *assess* responses

[1]In: Proceedings of the 8th AISB conference on Artificial Intelligence, Leeds, 16–19 April, 1991

through the receipt of some indication of the validity of an answer or memory recall obtained from lower-level modules.

We view the operation and construction of such ultimate systems in a top-down sense, based on an architecture of hierarchical and interacting *modules,* which at the lowest level are comprised of neural networks. Each module should be designed as closely as possible to a *semi-closed system,* that hides the details of the network wetware but allows interaction with other network modules and transmission of "opinions" and the *validity* of these opinions on the questions being asked it to higher processing levels of the system hierarchy. In this paper we present the design of a particular Minos network module that may form part of a group of such modules comprising an Authority. An Authority has the capability of learning difficult mapping problems, or large numbers of memories (depending on the underlying network types), through the division of subtasks among the Minoses. It generates a confidence measure together with its reply to a question (from higher up the hierarchy) based on its degree of familiarity with the question.

Another aim of the Minos generation is to be able to vary the information learnt dynamically as the environment varies, and do it in a reliable way. Normally, experiments on the different types of network are carried out with an unchanging training set of possible environment situations, and the nets are adapted to respond correctly to these, and only these. Further learning on situations exclusive of this set is however often very destructive, with the previously learnt situations degrading since they are no longer being learnt. Such degradation is equivalent to "forgetting" and is a fundamental property of neural systems. However this would be an acceptable property of the system so long as one could *distinguish* "hazy" memories from clear ones. Progressive learning clearly presents the drawback of "recency" (the more recently encountered situations are more favourably remembered) but also the advantage of being able to vary the adaptation frequency to the situations' appearance frequency in the environment. In the functioning of Minoses the above characteristics of neural systems are recognized and used to the advantage of the system. Thus, it is good if, when new situations arise in an environment, the likelihood of acting as if the old cases applied reduced and that of the new increased.

By employing neural network module decomposition of problems, Authorities adopt the "divide and conquer" approach to solving difficult mapping problems. This is a way of attacking the neural network scaling problem [7, 6]: The more difficult a mapping problem (of a fixed input space size) the more hidden units that will be necessary in order to solve it [19, 17]. Furthermore, the more difficult the mapping function to be learnt, the more training examples that will be necessary for decent generalization [18, 1]. Numbers explode with input space dimension size. Instead of scaling single networks in the way suggested by such complexity scaling, the Minos approach is to keep networks as small as possible but use arbitrary numbers of them to solve the same problem, sharing out the training set in a particular way. This is a *modular* as opposed to a *neural soup* technique of producing systems capable of solving problems of higher complexity [8]. Such division often might be rejected on generalization grounds. Indeed, if as many networks as patterns are available, then absolutely no generalization should perhaps be expected. However, if it is the possibility of being able to generalize *at all* on the training set that is *used* to divide the set among the networks, then one might expect not to lose the desired generalization. Such assumptions about generalization are an important aspect of the Minos method of task division.

In this paper the fundamental ideas of division of tasks and module discrimination are explained, and alternative implementations discussed. First the previous influential "Pandemonium" work in this area is outlined, together with the mechanism of task division and discrimination. The need for module reliability is discussed, then improvements are proposed in the form of the Minos module, and its inclusion in an Authority system. Finally further extensions to the Authority system are briefly outlined.

PANDEMONIUM I AND II

In Selfridge's 1959 paper [15] he describes the Pandemonium system, which achieves efficiency in learning recognition problem domains through the utilization of several "demons", each of which responds in a certain way to a particular input. Thus the input to a layer of demons would be the same for each, but the weightings within the demons different, such that each demon computes different properties of the input. Thus each demon i outputs a scalar measure of the degree of "i-ness" in the input. During learning of the whole system, weights are adapted such that useful functions might be strengthened and produce larger outputs, and useless functions degraded or rejected completely. In normal operation the response of a complete Pandemonium system is that of the demon with the largest output.

In Pandemonium the demon choice criterion was related to the usefulness of the function computed by the demon in this aspect of the recognition. It is this idea that is developed in the Minos modules.

The demons of Pandemonium are really equivalent to single units in a neural net. If each demon is however replaced by a neural network one arrives at a higher-level extension of the strength in numbers policy. The idea is to reduce the learning complexity of a network working on a particular task, through sharing naturally separable members of a training set among several networks, and choosing the output of the current expert in the normal (non-learning) operation of the system.

An initial attempt at such an extension of Pandemonium was undertaken in [16] with the Pandemonium II system (see also [9]). Made available to the system was a number of feed-forward neural networks of identical structure, but different initializations. Desired was to see the system use these networks to *divide* a problem up, resulting in each network being faced with an easier mapping task. There were certain important things to consider first. Namely, *how* is the problem to be split up, *what* is going to supervise this procedure and also *what* will decide who will answer the question in normal processing mode.

In Pandemonium II it was done in the following way. An input pattern to be learnt was presented to all the nets taking part and processed. The (feed-forward) net with its output nearest to the target output associated with the input pattern, as judged by the *Taskmaster* (loosely equivalent to Selfridge's "Decision Demon"), was allocated the learning of this pattern. Learning the pattern meant performing n back-propagation [13] steps. Thus the task domain was divided up in such a way that patterns were allocated to be learnt by those nets that were most suited to this aspect of the (possibly complicated) mapping problem, at this time. The problem domain was viewed as a pool of pattern-target pairs, and these pairs converged on the system with certain probabilities. Thus the patterns would be learnt to an extent depending on their frequency of occurrence in the world of the system.

The choice of network for learning the pattern was based on a sum of squared difference measure E_α for each net α:

$$E_\alpha = \sum_i (t_i - o_{i\alpha})^2. \tag{1}$$

$o_{i\alpha}$ represents the output i of net α, and the target for the pattern in question is t_i at output i. The summation is over all the output units of the network. The choice of network for normal processing was based on a "who shouts loudest must be most confident" principle (see below).

Patterns are present in the environment and the Taskmaster effectively controls the learning of such patterns, working on a frequency of appearance principle, and allocating to the nets such that specialization develops.

In the implementation of Pandemonium II on a simple (but divisible) problem domain the two fundamental requirements of such a system become clear. They are, (1) An allocation method, and (2) A discrimination method. The Taskmaster uses the allocation method to determine by which net a particular pattern at a certain time should be learnt, and the discrimination method to determine which net should be believed in normal processing. The methods used were

1. for allocation: choose the net that produces the lowest error for this pattern. The error measure is the same as that used in the gradient descent (back-propagation) procedure (i.e. as in (1)).

2. for discrimination: In the task examined in Pandemonium II the targets consisted of only ever one output unit required to be high, and all the others at zero. Thus the correct net was deemed to be that with the highest output unit activation (which was then used to construct a binary pattern). Hence the "he who shouts loudest" term used above.

The technique was successful for this instance of Pandemonium II, but would not always be expected to be successful. It is not necessarily the case that the highest node activation for a given input pattern, among all the output nodes and nets, is the one corresponding to the target node. It may well be found that a net responds positively to an input pattern that has not been learnt, because it may share a few input node similarities with a well-learnt pattern. In such a case the output could be taken to be a *generalization* of the pattern, but there is no way of knowing whether this is happening. Indeed, such generalized patterns could produce grossly erroneous results, which would seriously undermine the performance and reliability of the entire system. A related problem was the output representation itself. In the

Pandemonium II prototype the output consisted of single node activations, and not a complete n-bit pattern. In the latter case the "who shouts loudest" policy for network selection is not applicable. Required is some kind of general loudness measure so that it may be determined which network really should be believed: but there is no reason to suppose that the *intensity* of the overall output corresponds to the correctness of the response.

UPGRADING PANDEMONIUM II

In order to upgrade the simple operation of the Pandemonium II system the two conflict phases of allocation and discrimination need to be developed. They represent the two major problems confronting designers of modular systems, that can be summed up in the questions: How can I tell whether a module will be a good expert before it is trained? and: How do I know which expert to believe when I ask a question?

Shouting vs. Confidence Discriminants

Discriminating on the basis of the form or amplitude of the output may not be the wisest method. For instance, large threshold weights could cause a particular network to respond with almost boolean outputs to patterns. What is needed is not so much a measure of how the outputs look to the taskmaster, but how the *inputs* look to the networks. Thus if a network likes the look of a question, and can express this in some measurable way regardless of its actual answer, then we may more or less confidently believe its answer. So how might the Pandemonium II networks be upgraded to possess a "confidence" output? We suggest the following five ways of incorporating such a discriminator:

1. The degree of confidence in what has been learnt by the network is a function of the *number of times* the pattern in question has been presented. So this assumes the reasonable hypothesis that the net that has been chosen most often for the learning of a particular pattern should be able to give the most accurate output.

2. The degree of confidence in a particular output pattern is a function of the *sharpness* of this output pattern. i.e. given that one is always learning to map to binary targets, a net output that looks most like a binary output is taken to be indicative of closeness to the desired output.

3. There exists a separate *supervisory network* that is trained on all the patterns and learns to partition them explicitly, getting the knowledge of this decomposition from observation of the network learning allocation. In future normal processing this network is consulted (or can gate the other networks) about the suitability of each network for advice on each pattern. This idea is very similar to the first one, but employs a network with a learning instead of a counting mechanism.

4. The network monitors its own progress by employing an adjacent companion network somehow to learn to map the *current error* of patterns. In this way we

know directly how accurate an answer is (assuming the mapping is achieved by this companion network).

5. The network uses an alternative companion adaptive network that indicates the degree of *familiarity* it has with the input pattern in question. This familiarity measure is dependent on the frequency with which the pattern has been allocated to the net and also the *recency* of the pattern's appearance.

Suggestion 1 is a non-neural solution, and is unappealing in that, given that a RAM is being used for every pattern to retrieve the counter, one may just as well do away with the learning net and code the answer instead of the counter. Suggestion 2 cannot work if we wish to reap the benefits of modular division of labour, for it means that not only do we wish the individual expert nets to learn well their allocated portions of the training set, but we need to ensure that they sufficiently "unlearn" sharp answers to questions not allocated. Enforcing such unlearning at this level will degrade the expertness of the modules and the effective higher complexity of the problem will degrade the scaling prospects of the networks. Suggestion 3 is a reasonably good solution to this problem, that was employed in [10] using a Kohonen net [4] to separate pattern clusters for learning among a set of modules, and the gating idea has also been employed in [3]. Such gating is viewed here as placing too great a restriction on the system as a whole. What is desired is a system consisting of a set of modules *independent* at the module level, whose learning can be affected only indirectly through target allocation, and that one may obtain reliable behaviour also when the learning task globally alters. Suggestion 4 allows such independence, and seems ideal in that it provides us with an obvious and believable confidence measure. However, when one considers learning of dynamically changing target patterns, one may also expect that the companion network might be more complex and error-prone than the other one. Thus one alleviates the strain of complex mapping learning only to be burdened by a set of complex companion network mappings.

The idea in suggestion 5 seems most appealing. It promises a variation of the monitoring of the net through degradation brought about by the reduction of a pattern's presentation frequency, and thus might be more suitable when the system is faced with a changing environment. Independence is retained by the nets. Such familiarity nets (later called "monitors"), and their integration into an Authority, will be described in the context of a Minos module.

Allocation Discriminants

In Pandemonium II there arose a problem with the allocation technique: i.e., which network should actually be chosen to do the learning. The interpretation of a fit between output and target using (1) results in a net that has been doing a little more learning than others to be automatically more suitable for learning the next pattern. To give an example, say all the networks start off with their uncommitted outputs at around 0.5. If the target values of the output are mostly going to be set to zero, then the first net that has (by chance) to learn a pattern will have an unfair advantage for all the other patterns, because most of the output nodes will be biased to the lesser side of 0.5. The next pattern to come along will then probably be best mapped by the same net.

So this kind of allocation will not lead to the automatic development of task-dependent experts. Instead the network that is by chance chosen first to do some learning will be chosen to learn the whole task domain. A rapid convergence with the vast majority of patterns being learnt by this network will certainly occur for problem domains consisting of mainly sparse target vectors (the most likely case). The problem is solved in an Authority in the following way. The absolute values of the output components from the network are not important, but their *relative values* are. An expert will be chosen on the basis of the promising state of the relative components of its output. One may look at it another way: the output vector from the module is a discretized wave; a promising undertrained output wave has a similar *form* to the desired (answer) wave.

Mathematically, the following is performed. The output vector $\vec{v}$ from a module is renormalized to $\vec{v}'$ such that its maximum component has the same value as the maximum component of the target vector $\vec{t}$, and its minimum component has the same value as the minimum component of $\vec{t}$. The module that possesses an output with the least value of $L1(\vec{t}, \vec{v}')$ (the L1 norm) is considered to be the best candidate for the expert for this pattern.

MINOS MODULES AND RELIABLE AUTHORITIES

So much for the techniques for upgrading Pandemonium II. It is appropriate at this point to discuss the objectives we have in mind, and the problems that may be attacked when such a Pandemonium II, or *Authority*, is available.

Reliability

Neural networks are notoriously unreliable. Indeed they derive their very advantages over symbol-pushing processing and RAMs from their inherent "fuzziness". That is, **generalization** on a topic of learning is a probabilistic process manifested in the neural network expressing some "leaning" towards a particular answer in preference to other possibilities. By the same token, the generalizations can be (probabilistically) very wrong—perhaps more often wrong than right. What is more, there is no way of determining whether the answer offered by the neural network to an enquiry should be strongly believed (because such an association formed part of the network's training schedule) or taken as a possible generalization (if not). Before we can hope to include sets of neural network "expert modules" in a general purpose learning system we must have some way of gleaning such critical information from these building blocks.

Thus the message is: all neural networks are in some respect unreliable, and 100% performance will like as not never be achieved from a flexible undedicated neural network module; the key is to *recognize* where the unreliability lies, gauge it and base consequent higher-level decision making on the degree of "fuzziness", or *error*, of the answer. After all, almost all decisions made by man are more or less approximate and have associated an understanding of their error, from mundane acts like walking

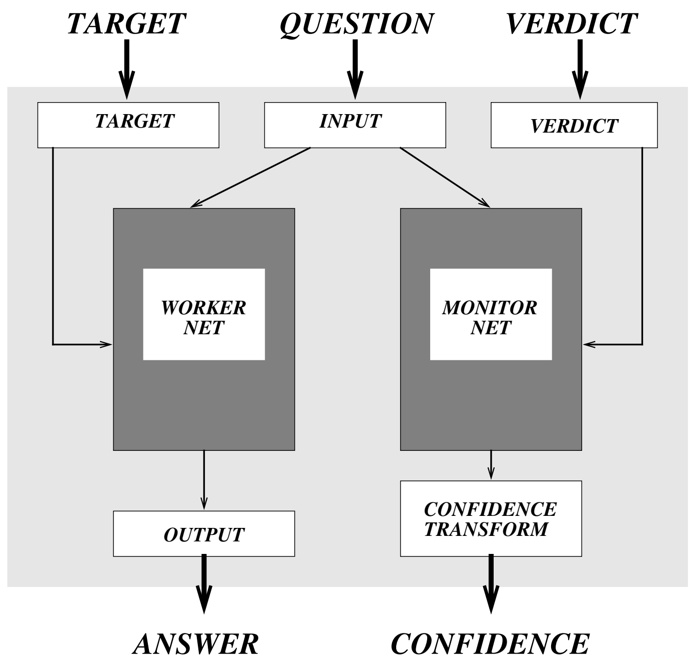

Fig. 1: The components of a Minos module.

or masticating food, to playing racket games or snooker, to the precision of scientific measurement. Once the disadvantages of the model have been recognized, or its failings, we can more confidently benefit from its advantages. This bodes too as a quicker alternative to constantly refining the performance of particular models by the odd percent.

A step in this direction is manifested in our Authority system. An Authority combines the sought-after property of a measure of confidence in the answer it provides—and through which its performance and consequently usefulness may be compared with other such Authorities—with the enticing prospect of reducing problem complexities through the Pandemonium divide-and-conquer principle.

The Minos Module

The key component of the Authority is the Minos module. It is called a module because it may be considered as a kind of black box with identifiable leads emerging from it, capable of performing a certain type of processing. Its initial design is depicted in Fig. 1. There are two networks, a *worker* and a *monitor*. The worker performs the processing and determines the capabilities of the module, while the monitor performs the job of informing one of the degree of confidence the module has in the validity of its answer, with respect to a certain input pattern. Our choice for the form of the monitor is discussed in a following section.

The inputs to a Minos are the input pattern ("question"), its actual "target" mapping (if this is necessary) and a "verdict" from the Authority (see next section); outputs include the suggested association ("answer") and its confidence.

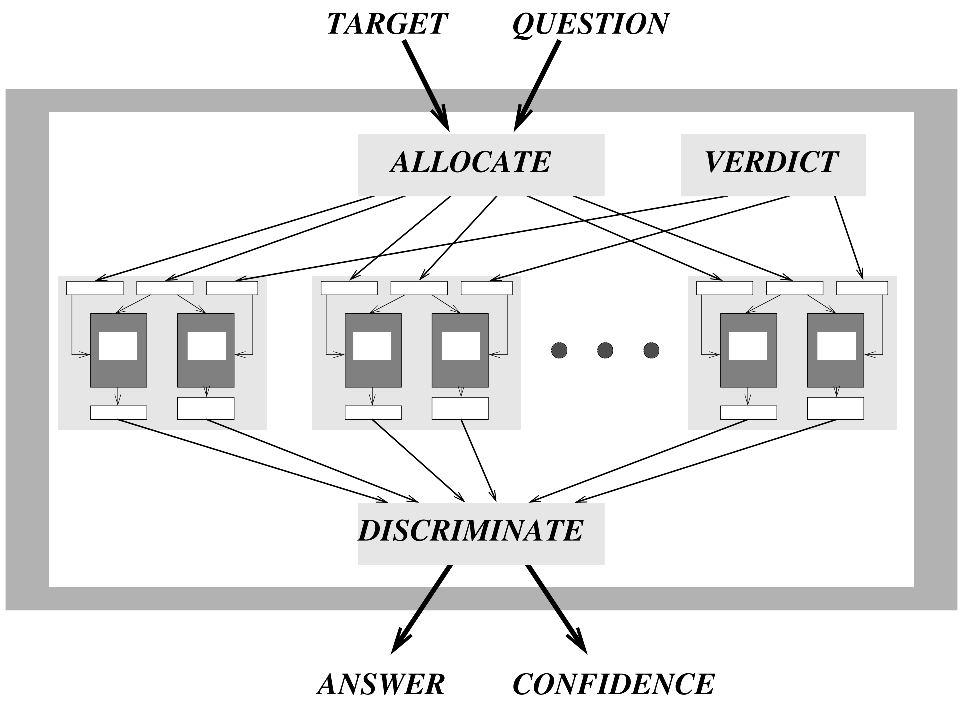

Fig. 2: The structure of an authority

Operation of an Authority

An Authority controls a number of Minoses, as depicted in Fig. 2. An Authority is also a module, and has similar leads coming in and going out.

When it receives a pattern to *learn* it allows all the Minoses to process the input, assesses their outputs (i.e. the worker net outputs, although see "Future Developments" for a refinement of this idea) based on the allocation discriminant discussed above, and *assigns* the pattern to be learnt by the Minos most successful or promising in such mappings. Of the possible learning schedules the simplest and conceptually most comfortable is for the chosen Minos now to learn this mapping (having received the target from the Authority). In our implementation we allow the Minos to learn the mapping up to *at least* a certain standard, but *no more than* a decided acceptance criterion (to prevent premature collapse of the mapping function). The monitor networks of all the Minoses simultaneously learn their mappings (see later), also to a certain standard. The Minoses receive the information of who is to learn the pattern through the "verdict" connection. If this is positive the Minos should accept a target and allow its worker and monitor to learn, if not the worker is idle and only the monitor learns.

In normal processing mode (i.e. a question/answer session) the Authority will receive the question part of the pattern and it has to output a suggested answer. It must now determine which of the expert Minoses to believe. This is simply done through choosing the Minos that has the highest confidence output. The answer and confidence are passed on to higher levels by the Authority.

THE MONITOR NETWORK

The most important property of a Minos is clearly its self-monitoring capability, without which discrimination is not possible, and upon which the assessment relies. We have chosen the simplest form for the monitor net: a YES/NO classifier network. The reasons for its choice are the following. Required is a net that can be easily taught to recognize questions that have been seen before and likewise not to recognize questions either not seen before or ones that were previously seen but have since been allocated to a better Minos. Such learning is (as our earlier experiments made clear to us) far too complex for an encoder net for example (besides, the scaling is not promising). The simplest net is going to have just the one output node, and the target for this node is easily determined by the Minos (remember we want a target that is independent of the worker target). We also insisted on a back-propagation net. The reason is merely that our experiments involved the worker net using back-propagation, and so we desired similar decay and learning speed characteristics so that the two nets might more or less keep up with one another in the learning. We may use the slow (but well generalizing) back-propagation algorithm in the worker net with a degree of confidence, since it is responsible for the monitor net learning too.

One might however envisage other forms of monitor network, such as Kohonen net [4] or ART nets [2]. One might even come up with a recurrent form for use with recurrent worker networks.

The simple confidence measure used in our initial implementations is identical to the output of the monitor net.

Thus the Minos causes every pattern question it has to learn to be mapped (to a certain standard) to 1 at the monitor network. If the question is not to be learnt then the monitor network is trained to map it to 0 (to a certain standard). We note an advantage in simultaneous learning in the worker and monitor networks: the recency effects are not eradicated in progressive learning, but accepted and manifested in higher (lower) confidences from the expert (non-expert). Furthermore, both nets should be subjected to similar decay rates for the "being-forgotten" patterns.

Note too that the complexity of a two-set classification reduces as the disparity in sizes of the the two sets increases. Thus the mapping to be learnt by the monitors becomes easier the more experts there are. This is desirable since we want to keep the monitor's job as easy as possible, given that the worker's job is alleviated through the divide-and-conquer policy anyway.

FUTURE DEVELOPMENTS

This paper is intended as a brief description of the important ideas in the construction of the Authority Minos system, and space limitations prevent detailed assessment of initial simulations. Preliminary results showed that a three Minos Authority did well in expert division for random mappings, shift experiments and handwritten digit recognition (although this latter application requires more analysis). Results will appear in a later publication. We now mention a few interesting modifications that may improve the Authority reliability further.

Inclusion of mild "unlearning" for the idle workers has been tested and seems promising (and no extra cost in parallel implementations). It involves pulling bits of the current output for the non-allocated pattern further away from the target.

Other improvement possibilities include the addition of a "rubbish" Minos module consisting of solely a monitor net to recognize the patterns that are really not very promisingly mapped by the experts. The experts would ignore this pattern because to learn it would mean probable serious disruption of the successfully learnt set of patterns by the Authority. In such an eventuality higher-level decision making processes should shift the learning to an alternative Authority system. The Authority discovers the unsuitability of this pattern through observing the monitor net output during the allocation phase. If no Minos module is terribly promising in terms of both the allocation discriminant and confidence the pattern could represent a new "concept" or task, and the Authority should be able to avoid learning the pattern and/or tell higher levels of the system.

A related development is control of the "expert decision regions". If one imagines the input space as being separated into a number of expert (Minos) regions through the monitor mappings, then one might want to prevent overlaps between monitor nets, since this will cause ambiguity in discrimination. Such occurrences are identified when there are two or more comparable top values of confidence. In order to communicate this the Authority may output an "ambiguity" measure. This is no problem. However, if the decision boundaries are too sharp the ambiguity may never be noticed. One may deal with this by constructing a "no-man's land" of ambiguity, and forbidding strong learning of mappings falling in it. These patterns may also be assigned to the rubbish net, or rejected outright by the Authority.

CONCLUSION

In this paper we have outlined the components of a neural net module that is intended to provide an environment for the multiple use of a system of networks. We have developed the idea of self-supervised Taskmaster learning from Pandemonium II, and the divide-and-conquer policy of Pandemonium I, into a more specific realization, and intend it to be used as a general mechanism for modular network system construction.

The first steps were concerned with solution of the problem of discrimination among a set of expert modules so that a Pandemonium structure might be used. In solving this another major development of allowing a confidence measure to be propagated through a hierarchical system has been achieved. Thus with these two ideas progress has been made in the direction of robust neural network components and also in the direction of problem complexity reduction, and therefore more promising scaling possibilities through divide-and-conquer among relatively independent modules.

Future steps involve sophistication of the systems and development through allowing more information to be read from the networks and shaping learning schedules and allocation among Authorities according to their current capabilities. All this should allow more intelligent use of neural network based systems such that their real advantages may overshadow their inevitable drawbacks.

REFERENCES

[1] J. Denker, D. Schwartz, B. Wittner, S. Solla, R. Howard, L. Jackel, and J. Hopfield. Large automatic learning, rule extraction and generalization. *Complex Systems*, 1(5), 1987.

[2] S. Grossberg. Adaptive pattern classification and universal recoding II: Feedback, expectation, olfaction and illusions. *Biological Cybernetics*, 23:187–202, 1976.

[3] R. A. Jacobs, M. I. Jordan, and A. G. Barto. Task decomposition through competition in a modular connectionist architecture: The what and where vision tasks. Technical Report COINS TR 90-27, Dept. of Computer & Information Science, University of Massachusetts, Amherst, MA, March 1990.

[4] T Kohonen. *Self-Organization and Associative Memory, 2nd. edition.* Springer–Verlag, Berlin, 1988.

[5] Y. le Cun. Medical diagnosis using neural networks. In *Proceedings of Cognitiva*, Paris, 1985.

[6] M. Minsky. *The Society of Mind.* Simon and Schuster, New York, 1985.

[7] M. Minsky and S. Papert. *Perceptrons.* MIT Press, 1988. see particularly the Epilogue.

[8] H. Mühlenbein. Limitations of multilayer perceptrons – steps towards genetic neural networks. *Parallel Computing*, 14(3):249–260, 1990.

[9] H. Mühlenbein and J. Kindermann. The dynamics of evolution and learning – towards genetic neural networks. In R. Pfeifer, Z. Schreter, F. Fogelman, and L. Steels, editors, *Connectionism in Perspective*, Amsterdam, 1989. Elsevier. Proceedings of the International Conference Connectionism in Perspective, University of Zürich, 10–13 October 1988.

[10] Y. Nishikawa, H. Kita, and A. Kawamura. NN/I: a neural network which divides and learns environments. In *Proceedings of the 2nd IJCNN*, Washington D.C., 1990.

[11] B.A. Pearlmutter. Learning state space trajectories in recurrent neural networks. *Neural Computation*, 1(2):263–269, 1989.

[12] N. Qian and T. J. Sejnowski. Predicting the secondary structure of globular proteins using neural network models. *Journal of Molecular Biology*, 202:865–884, 1988.

[13] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. *Nature*, 323(533), 1986.

[14] T. J. Sejnowski and C. R. Rosenberg. NETtalk: A parallel network that learns to read aloud. *Complex Systems*, 1(1), 1987.

[15] O.G. Selfridge. Pandemonium: a paradigm for learning. In *The Mechanisation of Thought Processes: Proceedings of a Symposium Held at the National Physical Laboratory, November 1958*, pages 511–527, London: HMSO, 1959.

[16] F.J. Śmieja. Evolution of intelligent systems in a changing environment: I. First steps with a structured brain. Technical report, Gesellschaft für Mathematik und Datenverarbeitung, August 1988.

[17] F.J. Śmieja. The significance of underlying correlations in the training of a layered net. In *Abstracts of the First Annual Meeting of the INNS, supplement to Neural Networks, Vol 1*, Boston, Mass., September 1988. Available as Edinburgh University preprint 88/447.

[18] F.J. Śmieja and H. Mühlenbein. The geometry of multilayer perceptron solutions. *Parallel Computing*, 14:261–275, 1990.

[19] G. Tesauro and B. Janssens. Scaling relationships in backpropagation learning. *Complex Systems*, 2:39–44, 1988.

[20] G. Tesauro and T. Sejnowski. A parallel network that learns to play backgammon. *Artificial Intelligence*, 1988. in press.