The n-tuple Classifier: Too Good to Ignore

Michał Morciniec and Richard Rohwer
Dept. of Computer Science and Applied Mathematics
Aston University
Birmingham, UK B4 7ET

December 2, 1994

Abstract

The n-tuple pattern recognition method has been tested using a selection of 11 large data sets from the European Community StatLog project, so that the results could be compared with those reported for the 23 other algorithms the project tested. The results indicate that this ultra-fast memory-based method is a viable competitor with the others, which include optimisation-based neural network algorithms, even though the theory of memory-based neural computing is less highly developed in terms of statistical theory.

1 Introduction

A popular style of neural computation is to apply optimisation techniques to suitably designed neural network models. This has the advantages of good performance and reasonably firm theoretical underpinnings, but often suffers from hefty computational requirements. An alternative style is based on memorisation of randomly selected features. Although the theory is less well-developed, such methods offer an overwhelming advantage in computation speed. One of the oldest memory-based methods is the n-tuple classifier of Bledsoe and Browning (Bledsoe & Browning, 1959). This method was tested on 11 data sets which had been previously used by the European Community ESPRIT StatLog project (Michie *et al.*, 1994) to test 23 other classification algorithms. The results add to a body of practical experience (Rohwer & Cressy, 1989; Tarling & Rohwer, 1993; Aleksander & Stonham, 1979), which lends weight to the view that more sophisticated methods often have no performance advantage to offer. Such methods include popular neural network methods with better theoretical foundations, such as the multi-layer perceptron and radial basis functions. It would appear that memory-based methods deserve more intensive study.

2 The n-tuple recognition method

The n-tuple recognition method is also known as a type of "RAMnet"[1] or "weightless neural network". It forms the basis of a commercial product (Aleksander *et al.*, 1984). It is a method for classifying binary patterns, which can be regarded as bit strings of some fixed length L. This is not an important restriction, because there is an efficient preprocessing method, tailored to the RAMnet's generalisation properties, for converting scalar attributes into bit strings. This method is defined in section 5. This section defines the n-tuple classification algorithm itself.

Several (let us say N) sets of n distinct[2] bit locations are selected randomly. These are the *n-tuples*. The restriction of a pattern to an n-tuple can be regarded as an n-bit number which, together with the identity of the n-tuple, constitutes a 'feature' of the pattern. The standard n-tuple recogniser operates simply as follows:

> *A pattern is classified as belonging to the class for which it has the most features in common with at least 1 training pattern of that class.* (1)

This is the $\theta = 0$ case of a more general rule whereby the class assigned to unclassified pattern $\boldsymbol{u}$ is

$$\underset{c}{\operatorname{argmax}} \left(\sum_{i=1}^{N} \Theta_\theta \left(\sum_{\boldsymbol{v} \in \mathcal{D}_c} \delta_{\alpha_i(\boldsymbol{u}),\alpha_i(\boldsymbol{v})} \right) \right) \tag{2}$$

where $\mathcal{D}_c$ is the set of training patterns in class c, $\Theta_\theta(x) = x$ for $0 \leq x \leq \theta$, $\Theta_\theta(x) = \theta$ for $x \geq \theta$, $\delta_{i,j}$ is the Kronecker delta[3] ($\delta_{i,j} = 1$ if $i = j$ and 0 otherwise.) and $\alpha_i(\boldsymbol{u})$ is the $i^{\underline{\text{th}}}$ feature of pattern $\boldsymbol{u}$:

$$\alpha_i(\boldsymbol{u}) = \sum_{j=0}^{n-1} u_{\eta_i(j)} 2^j. \tag{3}$$

Here u_k is the $k^{\underline{\text{th}}}$ bit of $\boldsymbol{u}$ and $\eta_i(j)$ is the $j^{\underline{\text{th}}}$ bit location of the $i^{\underline{\text{th}}}$ n-tuple.

With C classes to distinguish, the system can be implemented as a network of NC nodes, each of which is a random access memory (RAM); hence the term *RAMnet*. The memory content $m_{ci\alpha}$ at address α of the $i^{\underline{\text{th}}}$ node allocated to class c is set to

$$m_{ci\alpha} = \Theta_\theta \left(\sum_{\boldsymbol{v} \in \mathcal{D}_c} \delta_{\alpha,\alpha_i(\boldsymbol{v})} \right). \tag{4}$$

In the usual $\theta = 1$ case, the 1-bit content of $m_{ci\alpha}$ is set if any pattern of $\mathcal{D}_c$ has feature α and unset otherwise. Recognition is accomplished by summing the contents of the nodes of each class at the addresses given by the features of the unclassified pattern. That is, pattern $\boldsymbol{u}$ is assigned to class

$$\underset{c}{\operatorname{argmax}} \left(\sum_{i=1}^{N} m_{ci\alpha_i(\boldsymbol{u})} \right). \tag{5}$$

---

[1] *RAMnets* also include stochastic generalisations, *pRAMs*, to which the n-tuple recognition algorithm is not applied. These are not considered here.

[2] Relaxing the requirement that an n-tuple has n *different* bit locations amounts to introducing a mixture of differently sized n-tuples. Note the restriction does not disallow a single pattern component from being shared by more than one n-tuple.

[3] The comma is unconventional but is used here for extra clarity.

3 Discussion of the algorithm

The n-tuple classifier is a memory-based method akin to Kanerva's Sparse Distributed Memory (Kanerva, 1988). Such methods differ from optimisation-based methods, such as Back Propagation of error through multi-layer perceptrons, in two important ways. Firstly, "Hidden" representations (or "features") are selected randomly, and secondly, training is a simple memorisation task involving these features. These differences give memory-based methods an awesome advantage in training speed. Radial Basis Functions obtain part of this speed advantage by selecting features randomly (Broomhead & Lowe, 1988), and multi-layer perceptrons can often be trained faster with little or no loss of performance by using fixed random weights into the hidden layers (Gallant & Smith, 1987; Sutton & Whitehead, 1993). However, this does not give the speed and simplicity that training by mere memorisation provides.

There is some theoretical understanding of memory-based models, and the n-tuple method in particular (Aleksander & Stonham, 1979; Flanagan *et al.*, 1992; Bledsoe & Bisson, 1962; Ullmann & Dodd, 1969), but not at the level of statistical sophistication available for optimisation-based methods (MacKay, 1992). Although an n-tuple network can be trained using optimisation instead of memorisation (Luttrell, 1992; Rohwer, 1994), the statistical tools which can then be brought to bear have so far failed to comprehensively explain and quantify the success of memorisation alone. Perhaps the experimental results reported here will encourage further effort in this area.

It is interesting to note that their sheer simplicity may make memory-based methods less biologically implausible than optimisation-based methods. Of course, the n-tuple method itself uses features specialised to digital hardware, but the principle may apply just as well to more biologically computable features.

4 The architectural parameters

The adjustable parameters of the n-tuple recognition method are the n-tuple size n, the number of n-tuples N, and the threshold θ. These are architectural parameters, and as with many other neural network algorithms, there is a shortage of theoretically convincing prescriptions for setting them. One can argue on the basis of sampling fluctuations that results should improve towards an upper bound with increasing N. Practical experience shows values of 100 to 1000 to be adequate. The optimal settings for n and θ are related to (among less measurable things) the amount of training data used. If n and θ are both too small, then one can easily end up with "saturation", a condition in which the recogniser fails because $m_{ci\alpha} \equiv \theta$ everywhere (Tarling & Rohwer, 1993). Having n large is thought to be good in that correlations among several bits of the pattern might be relevant to class discrimination, but if the size of the address space for each node (2^n) is too large compared to the number of training patterns, then performance can decline due to overly sparse memory usage (Ullmann, 1969; Rohwer & Lamb, 1993). Experience with data sets of up to a few thousand patterns indicates that n should be between about 3 and 8, and performance is almost always best with $\theta = 1$. Although better theoretical work on the n-tuple recognition algorithm would be highly desirable, the algorithm's high speed makes it entirely practical to just run a few tests with a few plausible parameter settings to find suitable ones.

5 Preprocessing of scalar attributes

A RAMnet classifies bit strings, but the attributes of the patterns in the StatLog data sets are mostly real numbers or integers. It is known (Aleksander & Stonham, 1979) that the generalisation behaviour of RAMnets is based on a generalised Hamming distance between bit strings. Given that generalisation from numerical attributes should be related to arithmetic differences, it is important to transform numbers into bit strings in such a way that numerical proximity is transformed into Hamming proximity. A memory-efficient method tailored to the generalised Hamming distance underlying RAMnet generalisation has been provided by Allinson (Allinson & Kolcz, 1993), using a combination of CMAC and Gray coding techniques. The prescription for encoding integer x is to concatenate K bit strings, the $j^{\underline{\text{th}}}$ of which is $\frac{x+j-1}{x}$, rounded down and expressed as a Gray code. The Gray code of an integer i can be obtained as the bitwise exclusive-or of i (expressed as an ordinary base 2 number) with $i/2$ (rounded down). This provides a representation in aK bits of the integers between 0 and $(2^a - 1)K$ inclusive, such that if integers x and y differ arithmetically by K or less, their codes differ by Hamming distance $|x - y|$, and if their arithmetic distance is K or more, their corresponding Hamming distance is at least K.

In the experiments reported here, $K = 8$ and $a = 5$, giving 40-bit representations of the integers in $[0, 248]$. All scalar attributes were linearly rescaled and rounded to obtain integers in this interval. In the Letter data set (see table 1), where the attributes can take on only 16 values, it would be more reasonable to use a one-out-of-N encoding with strings of 16 bits, but the CMAC/Gray procedure was used anyway for the convenience of uniformity.

6 Selection and pre-processing of StatLog data sets

The European Community ESPRIT project 5170, the StatLog project, was designed to carry out comparative testing and evaluation of classification algorithms on large scale applications. About 20 data sets were used to estimate the performance of 23 procedures. These are described in detail in (Michie *et al.*, 1994). Each of the larger data sets (with many more than 1000 samples) were randomly split into training and testing partitions. Different methodologies (cross-validation and bootstrap) were applied to the smaller data sets. This study used the large data sets, which are summarised in table 1. There are 11 of these.

Name	Classes	Largest Prior	Attributes	Training Patterns	Testing Patterns	Description
BelgianII	2	0.924	57 real	2000	1000	Classify measurements on simulated large scale power system as leading to stable or unstable behaviour.
Cut50	2	0.941	50 real	11220	7480	50 measurements from a candidate segmentation point in joined handwritten text. Classify as suitable cut point or not. Commercially confidential data.
Cut20	2	0.941	20 real	11220	7480	Best 20 attributes (by stepwise regression) from Cut50.
Technical	91	0.230	56	4500	2580	Commercially confidential. Appears to be generated by a decision tree. Most attribute values are 0.
DNA	3	0.525	240 Boolean	2000	1186	Sequences of 60 nucleotides (4-valued) classified into 3 categories.
SatIm	6	0.242	36 integer	4435	2000	3x3 pixel regions of Landsat images. Intensities in 4 spectral bands. Classified into 6 land uses at central pixel.
Chromo	24	0.044	16	1250	1250	Images of Chromosomes, reduced to 16 features.
BelgianI	2	0.5664	28 real	1250	1250	As Belgian II with a smaller simulation. Attributes thought to be least informative omitted from simulation.
Tsetse	2	0.508	14 real	3500	1499	Classify environmental attributes for presence of Tsetse flies.
Letter	26	0.045	16 16-valued	15000	5000	Images of typed capital letters, described by 16 real numbers discretised into 16 integers.
Shuttle	7	0.784	9 real	43500	14500	Classification problem concerning position of radiators on the Space Shuttle. Noise-free data.

Table 1: Descriptions of data sets used.

7 Experiments

The threshold θ was set to 1 in all the experiments reported here, and N was set to 1000 n-tuples. Each experiment was repeated for a selection of small n-tuple sizes. The results reported are averages over 10 different random input mappings η for the best n. This involves using test data to set an architectural parameter, so strictly speaking, these experiments do not show generalisation performance purely and directly. However, the subsequent re-randomisation of the input mapping completely scrambles the network connectivity, completely re-defining the random features used for discrimination. Thus, the experiments do demonstrate generalisation from one input mapping to another, for a given n, and it is difficult to argue that new test data randomly drawn from the same distribution will have a more severe effect than selecting new features randomly from the same data. Hence this procedural expedient was felt justified.

Computation time requirements were insignificant in these experiments, which were carried out with a C++ program on a SUN Sparc workstation. For example, an 8-tuple network can be trained on the 2000 57-attribute training patterns of the BelgianII data set in about 49 seconds. Sixteen of these seconds are needed just to read in the data; another 4 to do the CMAC/Gray conversion of the floating point attributes; and the final 29 to train the RAMnet itself. Testing the same 2000 patterns takes slightly longer, 37 seconds instead of 29, because a loop over classes is needed within the loop over n-tuples. Detailed timing statistics are not published for the algorithms used in the StatLog project, but it is clear that popular neural network algorithms such as Back Propagation and even the relatively fast Radial Basis Functions are slow by comparison. The algorithm is highly parallelisable, so if it were important for the RAMnet to be even faster, special purpose parallel hardware could be designed or purchased (Aleksander *et al.*, 1984).

The storage requirements were moderate in most cases. In the most extreme case (Shuttle) 128kB of RAM per class was needed.

8 Results

The classification results for each algorithm attempted with each data set are presented in figure 1. Table 2 gives a brief description of each algorithm with the symbol used to represent it in the figure. The classification error rates increase from left to right, and are scaled separately for each data set, so that they equal 1 at the error rate of the trivial method of always guessing the class with the highest prior probability, ignoring the input pattern.

As remarked in section 7, the results plotted for the n-tuple recognition algorithm are averages over 10 randomly selected input mappings. If the corresponding standard deviations were plotted as error bars in figure 1, they would be obscured by the dots representing the means.

On its best 7 data sets, the RAMnet gave performance broadly comparable to most other algorithms, including the popular neural network methods of Back Propagation and Radial

RAMnets.

(●) n-tuple recogniser.

Discriminators.

(♣) Back Propagation in a 1-hidden-layer MLP.
(♠) Radial Basis Functions.
(♡) Cascade Correlation.
($\oplus$) SMART (Projection persuit).
($\otimes$) Dipol92 (based on pairwise linear discriminators).
($\ominus$) Logistic discriminant.
($\oslash$) Quadratic discriminant.
($\odot$) Linear discriminant.

Methods related to density estimation.

(α) CASTLE (Probabilistic decision tree).
(β) k-NN (k nearest neighbors).
(γ) LVQ (Learning Vector Quantisation).
(δ) Kohonen topographic map.
(ε) NaiveBayes (Estimate assuming independent attributes).
(ζ) ALLOC80 (Kernal function density estimator)

Decision trees.

(a) NewID (Decision Tree)
(b) AC^2 (Decision Tree)
(c) Cal5 (Decision Tree)
(d) CN2 (Decision Tree)
(e) C4.5 (Decision Tree)
(f) CART (Decision Tree)
(g) IndCART (CART variation)
(h) BayesTree (Decision Tree)
(i) ITrule (Decision Tree)

Table 2: Synopsis of Algorithms with symbols used in Figure 1.

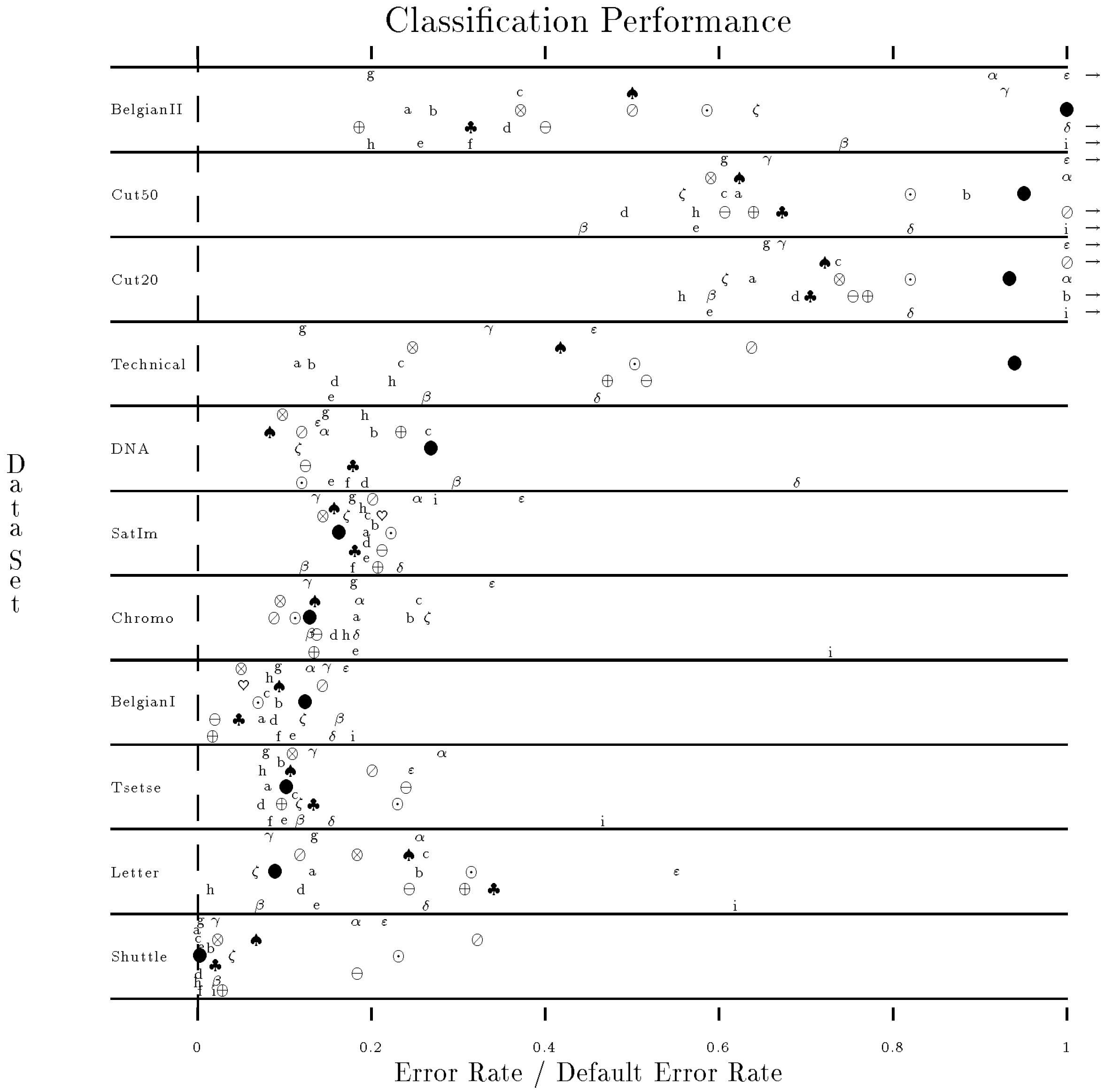

Figure 1: Results for N-tuple (●) and other algorithms. Algorithm codes appear in Table 2. Classification error rates increase from left to right, and are scaled separately for each data set, so that they equal 1 at the error rate of the trivial method of always guessing the class with the highest prior probability, ignoring the input pattern. The arrows indicate the few cases in which performance was worse than this.

Basis Functions. Sometimes it did rather better and sometimes rather worse, but never by an alarming margin. The relative performance of the other methods jumps around to a similar extent, as well as the eye can judge. A more sophisticated attempt to find systematic performance differences between the StatLog algorithms (Michie *et al.*, 1994) did not turn up much which cannot be gathered from such judgements by inspection.

The worst 4 data sets for the RAMnet tell a different story. Most algorithms did poorly on these data sets, but the RAMnet failed spectacularly. It did scarcely better than the method of assigning every test pattern to the *a priori* most probable class. The only good news is that these results suggest the hypothesis that if the RAMnet is used on a new type of data, it can be relied upon to either do reasonably well, or perform so badly that its unsuitability will be obvious.

Further research is needed to determine precisely what sort of data a RAMnet can handle, but these results suggest a couple of guesses. One possibility is that a highly skewed distribution of class priors is problematic, because the BelgianII, Cut50, and Cut20 data sets have most probability concentrated on 1 of their 2 classes, and the Technical data set concentrates a quarter of its probability on just 1 of its 91 classes. But the RAMnet did well on the Shuttle data, which is also rather skewed. Another possibility is that poorly informative attributes are a problem. A comparison between the BelgianI and BelgianII results particularly suggests this, because BelgianI uses a subset of the attributes of BelgianII, selected according to an expert's opinion of their informativeness.

9 Conclusions

It would be inappropriate to use the results in figure 1 to draw unequivocal conclusions such as "RAMnets perform better than multi-layer perceptrons on the Letter data set", because there is usually scope for improving any method by fussing with its parameters and implementational details. But a glance at this figure does seem to justify the conclusion that RAMnets often perform well compared to other methods. Consequently it would seem foolish to embark on an hours-long Back Propagation run before spending a minute with a RAMnet simulation.

The results suggest a few hypothesis which might be considered by future theoretical and experimental work. These are that the n-tuple recognition method fails completely and obviously when it does fail, that it fails when the pattern attributes are relatively uninformative, and that highly skewed class priors can be a contributing factor to failures.

It would appear that memory-based learning algorithms in general, and the n-tuple classifier in particular hold great promise as ultra-fast systems giving competitive performance for many types of data. This lends some urgency to the problem of strengthening the theoretical foundations of these techniques.

10 Acknowledgements

The authors are grateful to Louis Wehenkel of Universite de Liege for useful correspondence and permission to report results on the BelgianI and BelgianII data sets, Trevor Booth of the Australian CSIRO Division of Forestry for permission to report results on the Tsetse data set, and Reza Nakhaeizadeh of Daimler-Benz, Ulm, Germany for permission to report on the Technical, Cut20 and Cut50 data sets.

References

Aleksander, I., & Stonham, T. J. 1979. Guide to pattern recognition using random-access memories. *Computers and Digital Techniques*, **2**, 29–40.

Aleksander, I., Thomas, W. V., & Bowden, P. A. 1984. WISARD: A Radical Step Forward in Image Recognition. *Sensor Review*, **4**, 120–124.

Allinson, N.M., & Kolcz, A. 1993. Enhanced N-tuple approximators. *Weightless Neural Network Workshop*, 38–45.

Bledsoe, W. W., & Bisson, C. L. 1962. Improved Memory Matrices for the n-Tuple Pattern Recognition Method. *IEEE Trans. Electronic Computers*, **11**, 414–415.

Bledsoe, W. W., & Browning, I. 1959. Pattern recognition and reading by machine. *Pages 232–255 of: Proceedings of the Eastern Joint Computer Conference.*

Broomhead, D. S., & Lowe, David. 1988. Multi-variable functional interpolation and adaptive networks. *Complex Systems*, **2**, 321–355.

Flanagan, C., Rahman, M. A., & McQuade, E. 1992. A Model for the Behaviour of N-Tuple RAM Classifiers in Noise. *Journal of Intelligent Systems*, **2**, 187–224.

Gallant, S., & Smith, D. 1987. Random Cells: an idea whose time has come and gone... and come again? *Pages II–671–II–678 of:* Caudill, & Butler (eds), *IEEE International Conference on Neural Networks.* San Diego: IEEE.

Kanerva, P. 1988. *Sparse Distributed Memory.* Cambridge, MA: MIT Press.

Luttrell, S. P. 1992. Gibbs distribution theory of adaptive n-tuple networks. *Pages 313–316 of:* Aleksander, I., & Taylor, J. (eds), *Artificial Neural Networks, 2.* Elsevier.

MacKay, D. 1992. Bayesian Interpolation. *Neural Computation*, **4**, 415–447.

Michie, D., Spiegelhalter, D. J., & Taylor, C. C. (eds). 1994. *Machine Learning, Neural and Statistical Classification.* Prentice-Hall.

Rohwer, R. 1994. *Two Bayesian Treatments of the n-tuple recognition method.* Tech. rept. NCRG/4342. Dept. of Computer Science, Aston University, Birmingham, UK. ftp: cs.aston.ac.uk (134.151.52.106) /pub/docs/rambayes.ps.Z (UNIX compressed postscript).

Rohwer, R., & Cressy, D. 1989. Phoneme Classification by Boolean Networks. *Pages 557–560 of:* Tubach, J. P., & Mariani, J. J. (eds), *Proceedings of the European Conference on Speech Communication and Technology.* Paris: CEP, 26-28 Albany St., Edinburgh, Scotland.

Rohwer, R., & Lamb, A. 1993. An exploration of the effect of super large n-tuples on single layer RAMnets. *Pages 33 – 37 of:* Allinson, N. (ed), *Proceedings of the Weightless Neural Network Workshop '93, Computing with Logical Neurons.*

Sutton, R. S., & Whitehead, S. D. 1993. Online Learning with Random Representations. *In: Tenth International Conference on Machine Learning (ML 93).*

Tarling, Roland, & Rohwer, Richard. 1993. Efficient use of training data in the n-tuple recognition method. *Electronics Letters*, **29**(24), 2093–2094.

Ullmann, J. R., & Dodd, P. A. 1969. Recognition Experiments with Typed Numerals from Envelopes in the Mail. *Pattern Recognition*, **1**, 273–289.

Ullmann, J.R. 1969. Experiments with the n-tuple Method of Pattern Recognition. *IEEE Transactions on Computers*, **18**(12), 1135–1137.