Evolutionary learning to optimise mapping in n-Tuple networks

CONFERENCE PAPER · JULY 1990

Source: IEEE Xplore

John Mark Bishop
Goldsmiths, University of London

R.J. Mitchell
University of Reading

Evolutionary Learning to Optimise Mapping in n-Tuple Networks

Dr J M Bishop, A.A.Crowe, P.R.Minchinton, Dr R J Mitchell

Department of Cybernetics, University of Reading

Abstract

The use of n-Tuple networks as pattern recognition devices is well known[1]. Networks are able to recognise and discriminate between different classes of data if each class is taught into a separate discriminator. If the different classes are too similar, however, the system can have difficultly discriminating between the classes. This paper describes a novel method of optimising the learning strategy to minimise this problem.

Introduction

The n-Tuple classification method, first proposed by Bledsoe and Browning[2], and implemented in hardware by Aleksander and Albrow[3], can be used to produce a pattern recognition device. The system is first taught the data to be recognised, by processing various versions of the data set. This is achieved by dividing each data pattern into groups of bits, called tuples, and remembering these, by storing them in a memory called a discriminator. This process is repeated for all members of the data set. To see if the system can recognise a data pattern, that pattern is divided into tuples as before and a count made of the number of tuples stored in the discriminator.

Even if the test pattern is not in the training set, but is similar to patterns in the set, it is likely that a majority of tuples will have been remembered, and hence the system can be said to have recognised the pattern.

If the system is to be able to recognise and discriminate between different classes of data, more discriminators are needed, one for each class. Each class is taught into its own discriminator, in the manner described above. In the recognition stage, the test pattern is processed as before, the number of tuples recognised by each discriminator is found, and the pattern classified as being of the class taught into the discriminator with the highest count of recognised tuples.

However, if the classes of data are too similar, the system may have difficulty discriminating between data patterns from these classes. This is because the tuples formed when analysing a particular pattern may have been stored in the discriminators associated with the different classes. The object of the method described here is maximise the differences between similar data sets, that is to increase the orthogonality of response of each discriminator[4]. This is achieved by finding the optimum mapping of the data patterns, that is, the best method of forming the tuples.

Mapping

The mapping of the data patterns is the process whereby the pattern is sampled so as to form the tuples. The mapping can have a crucial effect on the discriminatory ability of the system. To illustrate this, consider figure 1, which shows two 16-bit data patterns which differ by four bits, and assume that 4-bit tuples are used. If each tuple was formed from each column, three of the tuples are identical to both patterns, so 75% of the patterns appear to be the same. However, if the tuples were formed by taking each row, none of the tuples are identical.

Figure 1. Effect of Mapping of Tuples

In practice, tuples are formed by sampling the patterns in a random manner, rather than the sequential methods illustrated above. However, the same effect could be achieved by different random mappings. The object of the method described here is to deduce the optimum mapping strategy. This is achieved using an evolutionary method.

Evolutionary method

The basic evolutionary technique is as follows[5]. A mutation of the current method is generated (in this case a different mapping used to form tuples). The performance of the system with this mutation is evaluated. If this is better than the current method, then the mutation is adopted. In this case the system is trained with the data classes, using the proposed mapping, and the orthogonality of the discriminator response is used to determine if the new mapping should be adopted. The mutation is generated by swapping addresses used to sample the data patterns so as to form the tuples.

The evolutionary technique is designed to provide rapid traversal of the search space. There is a danger, however, that the technique will find a local optimum, and not the global optimum: the best solution. To surmount this problem, the simulated annealing technique is used[6].

Simulated Annealing

In this technique, at early stages in the algorithm large random jumps across the search space are allowed. This allows the whole search space to be probed so as to find the general area of the optimum position. As the algorithm continues, however, the size of the largest possible jump is reduced exponentially, so that the search converges on the optimum. Thus initially most of the addresses which form each tuple are changed, but later fewer addresses are changed. The initial number of address lines swapped, and the rate of the exponential decay, should be chosen suitably: the former should ensure that the whole search space is probed after relatively few initial iterations; and the latter should be sufficiently low that the system converges on the global optimum, but the lower the value, the slower the algorithm.

Algorithm

The algorithm is as follows:

Choose arbitrary mapping

FOR i := 1 TO TotalNumberOfIterations

t := I Exp (-i/τ)

Select random discriminator from classifier, and find ϑ

Swap t connections in discriminator, relearn, and find new ϑ

IF new ϑ > current ϑ THEN restore connections in discriminator

(where I = Initial number of addresses swapped, τ is the time constant of the exponential decay, and the orthogonality measure, ϑ, is given by:

$$\vartheta = \sum_i \sum_j \mathrm{Disc}(i, j) \qquad \text{and } i <> \text{discriminator being remapped}$$

where Disc(i,j) is the response of the jth pattern of class i. Note, a low value of ϑ indicates that the chosen discriminator has a high orthogonality rating).

Experiments and Results

The technique has been used in the learning of images of characters, and has been tested on a system acting as a document reader. As the evolutionary algorithm takes some time to find a solution, it was not used on all characters. Instead, the method was used only on similar characters, for example the letters 'e' and 'c', and for 'i' and 'l', etc. Thus, each discriminator has its own method of mapping of the data patterns.

Specific test characteristics:

Image size of each character	16*16 pixels
Tuple size	4 bits
Initial value for number of swaps	16

(this ensures the whole search space probed after 4 iterations)

Time constant of decay of number of swaps	100 iterations
Total number of iterations of test	3000

Encouraging results have been found. On average, the orthogonality measure of discriminators of similar characters is increased by 50%, and this leads to an increase in discrimination (that is the difference between the number of recognised tuples) of about 10%.

The system was taught characters printed using one particular font. However, when shown characters in a different font, it was still able to correctly identify these characters. Thus the system can still generalise, that is it is able to recognise patterns similar, but not identical, to those in the training set.

Conclusion

An evolutionary method has been successfully applied to the problem of discriminating between similar data sets using n-Tuple networks.

References

1 Aleksander,I. and Stonham,T.J. *A guide to pattern recognition using random access memories* IEE Journal Computers and Digital Techniques 2(1), 29-40, 1979.

2 Bledsoe,W.W and Browning,I. *Pattern recognition and reading by machine* Proceedings of the Eastern Joint Computer Conference, 1959, pp 225-232.

3 Aleksander,I and Albrow,R.C. *Pattern recognition with adaptive logic elements* IEE-NPL conference on pattern recognition, 1968.

4 Badii,A., Binstead,M.J., Jones,A.J., Stonham,T.J and Valenzuela,C.L. *Applications of n-Tuple sampling and genetic algorithms to speech recognition* in Neural Computing Architectures, Ed Aleksander, I., Ch 10., North Oxford Academic Publishers, 1989.

5 Holland,J.H. *Adaption in natural and artificial systems* Ann Arbor: The University of Michigan Press, 1975.

6 Kirkpatrick,S., Gelatt,C.D.Jr. and Vecchi,M.P. *Optimization by simulated annealing* Science, 220, 672-680, 1983.