A Stochastic Search Algorithm to Optimize an N-tuple Classifier by Selecting Its Inputs

Hannan Bin Azhar[1] and Keith Dimond[2]

[1] Department of Electronics, University of Kent,
Canterbury, Kent, England
maha2@kent.ac.uk
[2] Master, Keynes College, University of Kent,
Canterbury, Kent, England
k.r.dimond@kent.ac.uk

Abstract. The N-tuple method [4] is a statistical pattern recognition method, which decomposes a given pattern into several sets of *n* points, termed "*N* tuples". The input connection mapping of the N-tuple classifier determines the sampling and defines the locations of the pattern matrix. Realizing the fact that the classification performance of the N-tuple classifier is highly dependant on the actual subset of the input bits probed [3][7], we have introduced an approach based on a **R**eward a**n**d **P**unishment (RnP) scheme to select input mappings of the classifier. We termed the classes with high error rates as critical classes. Different groups of tuples have been formed for different classes. The strategy was to employ more number of tuples to a critical class-group than an easily distinguishable class. In order to illustrate the capabilities of the RnP based measure the task of recognizing hand-written digits from NIST [10] database has been chosen.

Keywords: N-tuple classifier, Pattern recognition, Stochastic search.

1 Introduction

The N-tuple classifier [4] is one of the oldest pattern recognition methods and has been successfully used in many application domains [8][2]. The Random Access Memory (RAM) is the basic component of the classifier. A RAM node (tuple) can be viewed as implementing a look-up table, the entries of which are determined by the RAM inputs. Sampling of *n* specific data locations of the input constitutes a 'feature' of the pattern. A pattern is classified as belonging to the class for which it has the most features in common with at least one training pattern of that class.

The classification performance of the N-tuple classifier is highly dependant on the input bits probed [7][3]. The number of possible connections for a 32x32-bit pattern matrix is enormous. Let us consider an N-tuple classifier of 140 tuples with the tuple-size 8 bits. For an input binary image of 32 bits high and 32 bits wide the total number of pixels are 1024 bits. Now if we consider that the same pixel doesn't repeat in a single tuple then the possible number of tuples that can be formed is found by the formulae of combination, $M = {}^{1024}C_8 \approx 2.91 \times 10^{19}$. Now M tuples when divided into

A. Campilho, M. Kamel (Eds.): ICIAR 2004, LNCS 3211, pp. 556–563, 2004.

groups of 140 then total number of combinations will be $B = {}^{M}C_{140}$, which is a very big number. Therefore an exhaustive search for B mappings is impossible.

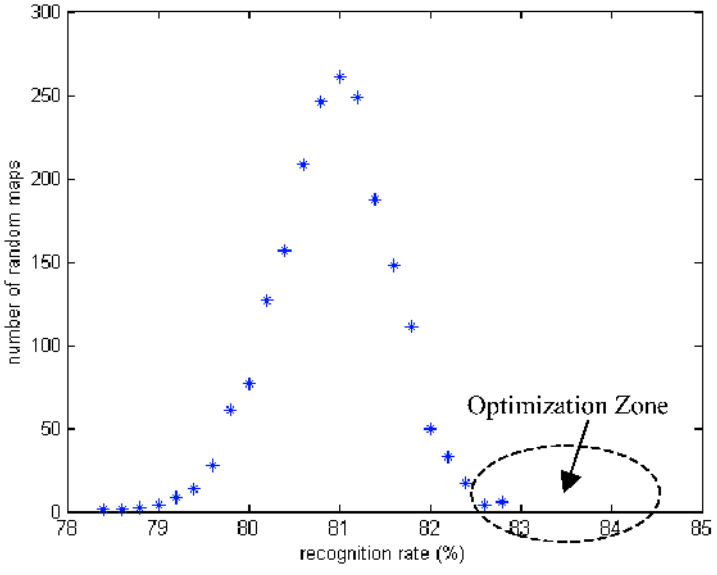

Fig. 1. Classification histogram with 2000 different maps to recognize handwritten characters from NIST Special Database 19 [10] with 140 tuples of tuple-size 8 bits.

The classification performance is a function of input mappings and it approximates to a normal distribution [1], where the majority of the mappings give average performance. Figure 1 demonstrates the classification histogram for a character database where two thousands maps were generated randomly and the frequency of the maps was plotted. If we generate all B mappings as we explained earlier, the tail on the right side of the histogram would go much farther as shown by the imaginary dotted area in Figure 1. To find the mappings in this dotted optimization zone is extremely challenging. The detection of mappings in this area by random search is governed by chance. Jung et al. [5] presented a method for selecting an optimal set of n-tuples in order to recognize classes of characters. The relative confidence [9] of a class discriminator [9] has been measured in [3] to find different sampling sequences for characters those are too similar. Jorgensen et al. [7] proposed a measure, which combines the concept of cross-validation and Shannon information theory to select the optimum connections needed to achieve a given performance.

Unlike others our inspiration was to develop an algorithm that would find maps in the optimization zone of Figure 1 and would improve the classification performance considerably. Our proposed method will measure the performance of an individual map based on a reward and punishment scheme and the best scored maps over a number of iterations will be selected for classification task to recognize handwritten characters from the NIST [10] database. The next section will describe our proposed stochastic search method and Section 3 will describe the reward and punishment

based performance measure to evaluate input connections. Section 4 will show the experimental outcomes and finally Section 5 will conclude the paper.

2 Tuple Search Algorithm

Our objective was to find an optimal set of input connections for N-tuple classifier that will give higher recognition rate than the random case. The performance of each connection map during search will be evaluated by a reward and punishment technique that will be explained in the next section. We termed the classes with high error rates as critical classes. As critical classes are giving low recognition rate, one strategy would be to optimize or tune a number of tuples so that they will give high scores to only one critical class. We distributed the total available tuples among classes proportionately to the error rates. This way classes with the higher error rates will get more tuples. This scheme ensures that more care has been taken to the critical classes by engaging tuples specific to a class. It has been found that class-specific tuples best describe a certain class by giving high scores evaluated by the objective function (Section 3) and might also give moderately high scores for few other classes. Thus tuples engaged to a specific class best describe the features of that class and also have the features for other classes to some extent.

The pseudo code of the search algorithm is given below:

```
LET i = 0;//class index
    Tf = 0;//number of successful tuples in any
           //iteration
    p'i = number of tuples responsible to best describe
         class Ci;
REPEAT
GENERATE ([P-Tf] set of tuples randomly);
FIND SCORE (P set of tuples based on class Ci);
RANK (P set of tuples based on their scores);
CARRY (Tf set of successful tuples to next iteration);
IF (Tf = p'i) THEN {
             SAVE(p'i set of tuples as mature tuples);
             SET (i = i + 1, Tf = 0);}
UNTIL (i < number of classes);
```

To understand the search algorithm to find class specific tuples, consider a class index *i* to identify a class '*Ci*'. Let us consider an objective function (Section 3) that gives scores for class-specific tuples. Let's say *P* is the total number of tuples in any iteration. Our target is to find *N* number of class-specific tuples in total. The distribution of *N* tuples among the classes is proportionate to the error rates. The more the class is critical, the greater number of tuples it gets. Let us assume *p'i* is the number of tuples that will be matured for class *Ci*. Thus the summation of all *p'i* ($\Sigma p'i$) will be equal to *N*. In every iteration, scores for *P* sets of tuples are evaluated according to the objective function based on *Ci.* So the RnP based objective function will give the scores only for the class *Ci.* Then *P* sets of tuples are ranked based on their scores for the class *Ci.* The number of tuples, say *Tf*, whose scores are higher than a predefined threshold are treated as the successful tuples for a certain iteration and they are being carried to the next iteration by virtue of their good scores. So in the next iteration only *P-Tf* tuples will be created randomly. Before moving to the next iteration a check has to be made if the number of successful tuples (*Tf*) have met the number of class-specific tuples (*p'i*) in the class *Ci*. If *Tf* = *p'i,* then the mappings for these successful tuples will be saved and these tuples will be treated as the matured tuples those are specific for the class *Ci.* The whole process repeats until all matured class-specific tuples ($\Sigma p'i = N$) for all classes have been sought and later these matured tuples will be used for the final recognition task.

3 RnP Based Performance Measure

A trained classifier can either recognize or misclassify or reject a test pattern. In the reward and punishment (RnP) scheme a reward is associated with the correct recognition of the pattern and the penalties for misclassification and rejection. The whole pattern data are divided in three parts: training set, evaluation set and test set. Let us consider S_l is the total number of samples for training the classifier, S_e is the number of samples available for evaluation purpose and S_t denotes the number of samples in the test data set. If *S* is the total number of available samples then $S = S_l + S_e + S_t$. Now for optimization purpose the network is trained with S_l and evaluated with S_e dataset. For the final recognition task both the S_l and S_e are used for training the network and S_t is used for testing. Finally, the dataset S_e can be considered to have three parts: Sc_i, S_r and S_m. Sc_i is the number of samples recognized only as the class *Ci*, S_r is the number of samples rejected and S_m denotes the number of samples misclassified. Considering all these definitions the objective function for class *Ci,* denoted as *O(i)*, will be evaluated by the following equation:

$$O(i) = Sc_i \times R + S_r \times P_r + S_m \times P_m \tag{1}$$

Where, R = Positive Points associated for reward for recognition,
P_m = Negative Points associated for misclassification,
P_r = Negative Points associated for rejection.

The point scheme for the objective function (Equation 1) will be explained in the next section. As explained in the previous section, in every iteration the search algorithm selects the number of tuples as the successful tuples according to a threshold. Consequently a threshold function was developed for the system, which is shown in the Equation 2:

$$O(i) \geq O(i)max \times (1 - k(t)) \tag{2}$$

$$k(t) = \exp(-\tau / t) \tag{3}$$

In the Equation 2, *O(i)max* is the score of the tuple which is the maximum among all scores in the current iteration and the value of *k* defines the percentage of the *O(i)max* that will be set as a threshold to be required to become a successful tuple. To accelerate searching, *k* can be varied over time based on the Equation 3. In this equation τ is the time constant which should be chosen suitably to set the exponential decay of the threshold over the number of iterations. Like any other stochastic search, the RnP based search will give better results if more time is given for the search. So the value of τ should be carefully chosen and varied throughout the search as a tradeoff between the performance and the speed.

3.1 Point Scheme for RnP

The point scheme determines what values we should set for *R, Pm* and *Pr* in the Equation 1. In general a rejection is thought to be more favorable than a misclassification. It is equivalent to the system getting confused rather than making the wrong decision. To find out what should be the point to set for the reward consider a point for misclassification as *Pm* = –1. If *J* is the criterion deciding minimum number of samples of the class *Ci* that must be recognized by one individual tuple to maintain a predefined minimum score, say *Omin*, then the value of *R* can be found by the following formulae:

$$J \times S_v \times R + P_m \times S_v(1 - J) = Omin \tag{4}$$

In the above formulae if *Omin* is set to 500 and *J*=5%, S_v=500; then *R* comes out as 39. So rewarding points should be 39 for a recognition of a pattern when *Pm*=-1. As rejection is more favorable than misclassification *Pr* can be set empirically as –0.5. *J*=5% indicates that the pattern is so complex that even when the least 5% of the samples of the evaluation data set is recognized by a single tuple, the evaluation function (Equation 1) will give a minimum score of 500. When the tuple-size is larger, a single tuple can recognize more patterns of the evaluation set. So for the larger tuple-size higher percentage of *J* can be chosen. It is more convenient to choose lower *J* value as it works for both higher and lower tuple-size, on the contrary algorithm with higher *J* value works only for the larger tuple-size.

4 Application of RnP Based Optimization to NIST Data Set

We applied our developed stochastic search method to recognize handwritten characters from the NIST database. We used the partition *hsf_{0}* [10] of the Special Database 3 for training and the partition *hsf_{4}* [10] from the Special Database 7 for testing. NIST recommends using images of *hsf_{4}* for testing as they are more difficult from other partitions and this ensures the heterogeneity between the training and testing set (see analysis in [10]), a fact which is reflected in our results. The numeric data set consisting only the digits (0,1…9) was used for the experiments. Each character is a binary image with the dimension 32 by 32. All digits are scaled into same dimension and centered. In the partition *hsf_{0}* there are 1000 images per class for training and in the partition *hsf_{4}* there are 1000 images (S_t) per class for testing. For the experiments the total train samples were again divided into two halves by the holdout method [6] and one part (S_l) was used to train the network in the evaluation phase and other part (S_e) to evaluate the RnP based objective function. After finding the mature tuples, all training images ($S_l + S_e$) of *hsf_{0}* were used to train the classifier and the images from *hsf_{4}* were used to test.

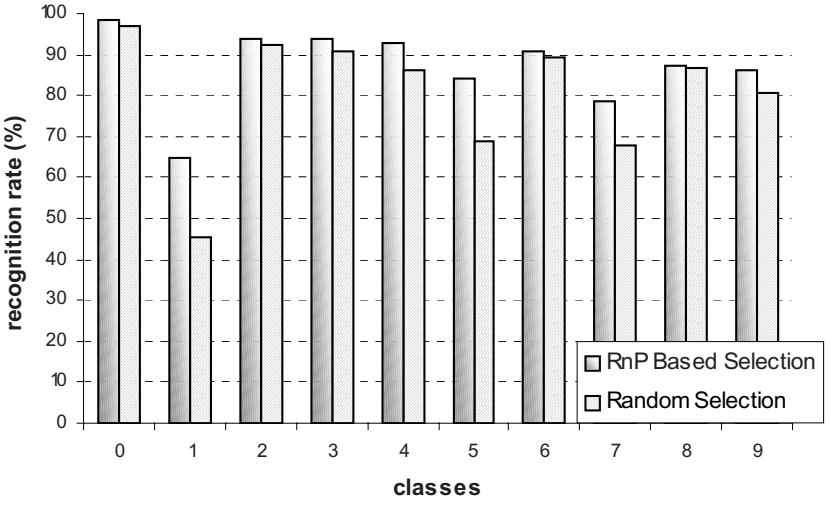

Fig. 2. Class-wise comparison of recognition rates between random selection and RnP based selection. For a class, in RnP based selection all tuples were tuned only for that class.

Two experiments were performed. The first one was to demonstrate how the recognition can be improved for a class when all the tuples in the network are tuned only for that particular class. In total 140 tuples were used for the network with the tuple-size 8 bits. The results are shown in the Figure 2. It can be seen that character 1 is the most critical class to be recognized in the NIST database. For the random case the recognition of class 1 was 45.26, which was the mean of 2000 runs. The RnP based optimization improves the recognition of class 1 by 19.66% (Fig. 2). It improves the recognition rate for class 5 by 15.56%, class 7 by 11.02% and all other classes by

several percents. The recognition rates found by RnP based search (Fig. 2) correspond to the mean of ten runs.

Table 1. Experimental Settings for RnP Optimization

Parameters	Values
Time constant (Eqn. 3), τ	100
Constant (Eqn. 4), J	5%
Point for misclassification, Pm	-0.5
Point for rejection, Pr	-1
Point for reward, R	39
Population size, P	200
For experiment 2: {*p`0,p`1, p`2,p`3, p`4, p`5,p`6,p`7,p`8,p`9* }	{2,42,6,7,10,24,8,25,11,15}

Table 2. Improved overall recognition rate by RnP based optimization

Methods	Average Recognition Rate (%)	Best Recognition Rate (%)
Conventional randomly selected N-tuple	80.93	82.83 (in 2000 runs)
RnP Based Stochastic approach	83.67	84.5 (in 10 runs)

The second experiment was required to demonstrate the improvement of the overall recognition rate by the stochastic search method. The overall rate was the average of all recognition rates of all classes. The total number of tuples for the experiment was 150 with the tuple-size 8 bits. Tuples were distributed among classes proportionately to the error rates of the classes. Thus being the most critical class, character 1 gets 42 tuples out of 150. The number of tuples for other class-groups (*p`i* in Section 2) can be found in Table 1. The randomly selected network was run for long enough (2000 iterations) to give it a chance to find better input maps that could be comparable with the maps found by our stochastic search. After long simulations we found the best overall recognition rate by the random network to be 82.83% and the average rate of 80.93%. In case of stochastic optimization (Table 2) the average overall recognition was 83.67%, which was 2.74% superior to the random case. We used a T-test with 99.9% confidence to determine if results between algorithms were significantly different. Our calculated t-value exceeded the tabulated value at the level of significance of 0.001. This shows that the increase in recognition rate by RnP based approach over traditional N-tuple method is statistically "very highly significant". The results were obtained with the code running under Windows XP on a 2 GHz Pentium 4 machine.

5 Conclusions

In this article we focused on a stochastic search technique to select an optimal set of n-tuples. We achieved only 2.74% of improvement in recognition rate by RnP based approach over conventional randomly selected N-tuple method [4]. This small improvement in recognition was due to the fact that in this paper we didn't consider any type of "diversity" while forming tuples through the stochastic process. Without diversity a pixel may copy itself several times while forming a tuple. Repetition of same pixels in a tuple can obstruct the exploration of new features in the image. Thus some sort of diversity control could benefit the system to achieve even better recognition rate than the one we achieved in this paper. We will try to explore this diversity issue in our follow-up work in future. This article will be a good reference to realize the underlying methodology of the RnP based stochastic process for N-tuple classifier. From the application point of view we expect our RnP based approach will help to reduce the requirement of hardware resources considerably as it can achieve a given level of recognition rate with less number of tuples.

References

1. Aleksander, I., and Stonham, T.J., "Guide to Pattern Recognition using random –access Memories," Computers and Digital Techniques,2,29-40 (1979)
2. Aleksander, I., W. V. Thomas, and P. A. Bowden, WISARD: A radical step forward in image recognition. Sensor Review 4 (3), 120—124. (1984)
3. Bishop, J.M., Crowe, A.A., Minchinton, P.R. and Mitchell, R.J., "Evolutionary Learning to Optimise Mapping in n-Tuple Networks", IEE Colloquium on "Machine Learning", Digest 1990/117. (1990)
4. Bledsoe, W. and Browning, I. Pattern recognition and reading by machine, Proceedings of Eastern Joint Computer Conference, pp.225-232, Birmingham (1959)
5. Dz-Mou Jung, M. S. Krishnamoorthy, George Nagy, Andrew Shapira: N-Tuple Features for OCR Revisited. IEEE Transactions on Pattern Analysis and Machine Intelligence 18(7): 734-745. (1996)
6. Hand, D.J., "Recent Advances in Error Rate Estimation," Pattern Recognition Letters, 4, 335 -346. (1986)
7. Jorgensen, T. M., S. S. Christensen, and C. Liisberg, Crossvalidation and information measures for RAM based neural networks. In D. Bisset (Ed.), Proc. of the Weightless Neural Networks Workshop (WNNW95), University of Kent at Canterbury, UK, pp. 87-92. (1995)
8. R. Rohwer and D. Cressy. Phoneme classification by boolean networks, Proceedings of the European Conference on Speech Communication and Technology, pages 557--560, (1989)
9. R J Mitchell, P R Minchinton : "Optimising memory usuage in n-tuple networks", Mathematics & Computers in Simulation, 40, pp: 549-563, (1996)
10. Wilkinson, R., Geist, J., Janet, S., Grother, P., Burges, C., Creecy, R., Hammond, B., Hull, J., Larsen, N., Vogl, T., and Wilson, C., The first census optical character recognition systems conference. Technical Report NISTIR 4912, National Institute of Standards and Technology, Gaithersburg, USA.(1992)