GREY SCALE N TUPLE PROCESSING

James Austin  
Dept. Computer Science  
University of York, York, UK

ABSTRACT

This paper describes a generalisation of the binary N tuple technique originally described by Bledsoe and Browning (1). The binary N tuple technique has commonly been used for the classification (2) and pre-processing (3) of binary images. The extension to the method described here allows grey level images of objects to be classified using the N tuple method without first having to convert the image to an intermediate binary representation. The paper illustrates the methods use in image preprocessing.

Introduction

The binary N tuple pattern recognition process was originally described by Bledsoe and Browning (1), and has been applied to a number of image processing tasks such as character recognition (4), face recognition (2), and scene analysis (5). The N tuple process may be seen as a simple perceptron (8) with a non-linear pre-processing transform. Thus, it is an adaptive classifier, which must be trained on a subset of the patterns to be recognised. It has two major advantages over the perceptron, first, it learns each training pattern quickly and, secondly, it is able to classify patterns that fall into the category of 'exclusive or' problems (9).

However, the N tuple method has always been limited to processing binary images; grey scale images may only be processed by the method if they are first converted to a binary representation (see 3). The method described here requires no intermediary binary representation of the grey scale data.

Although the binary N tuple process has been shown to be adequate in a number of applications there are a number of cases where it is insufficient, requiring the full grey scale information to be used in the classification process. For instance, in the classification of edge features where it is necessary to determine the angle and 'sharpness' or slope of an edge. This information is only present in the grey scale domain.

Binary N tuple process

The binary N tuple process as described by Bledsoe and Browning (1) and Aleksander (2) may be seen as a two stage process. The pre-processing stage performs a non-linear transform on the input image. The resultant image is then processed using a simple perceptron with binary weights. The addition of the pre-processor allows the perceptron (a linear classifier) to classify non-linearly separable data (i.e the 'exclusive or' or parity problem).

The binary N tuple process takes as its input a binary image. From this image a set of tuples are formed. Each tuple is made up of N elements from the image. The origin of each pixel in the image used to make up each tuple is defined once. Normally the origin of each pixel is selected in a random manner, each pixel only contributing to one tuple. The optimal size of N depends on the characteristics of the data and the
Each tuple may be denoted \( D_{k,N} \). For each image there are \( N_{\text{max}} \) number of tuples where \( N_{\text{max}} = \frac{i}{N} \), \( i \) being the number of elements in the image.

---

*Fig 1: One N tuple processing unit, \( U \).*

*Fig 2: An N tuple classifier.*

Fig. 1 shows one tuple and the processing associated with it (\( N = 3 \)). The first stage \( B() \) assigns one state, \( S \), to the tuple in accordance with table 1*. The tuple size, \( N \), equals 3 in this case, giving a total of \( 2^3 \) possible states for each tuple.

The next stage in fig. 1 shows a set of switches. Each switch is used to record the occurrence of a particular state, \( S \), assigned by \( B() \). There are \( 2^N \) switches to record all the possible states of the tuple. The output from the switches are OR'ed by the summing unit to produce an output \( \alpha \). For each of the \( N_{\text{max}} \) tuples there is one of the processing units, \( U \), shown in fig 1. Fig. 2 shows how the outputs of all the units are summed to produce an output \( R \), the response for the classifier.

There are two phases of operation of the system, teaching and testing. Teaching causes the system to learn an input pattern so that during subsequent presentation and testing it may be recognised.

When a pattern is presented for teaching, the image is broken into \( N \) tuples and each tuple assigned a

---

*The function \( B() \) may be defined formally. If the \( N \) tuple, \( D_{k,N} \), is represented as a binary (base 2) number, the state of the tuple is given as its decimal (base 10) equivalent.*