
Proceedings Paper:
O'Keefe, S. E M orcid.org/0000-0001-5957-2474 and Austin, J. orcid.org/0000-0001-5762-8614 (1995) Image object labelling and classification using an associative memory. In: Fifth International Conference on Image Processing and its Applications, 1995. Fifth International Conference on Image Processing and its Applications, 04-06 Jul 1995 IET, GBR, pp. 286-290.

https://doi.org/10.1049/cp:19950666

IMAGE OBJECT LABELLING AND CLASSIFICATION USING AN ASSOCIATIVE MEMORY

S E M O'Keefe, J Austin

University of York, UK

ABSTRACT

An essential part of image analysis is the location and identification of objects within the image. Noise and clutter make this identification problematic, and the size of the image may present a computational problem. To overcome these problems, we use a window onto the image to focus onto small areas. Conventionally we still need to know the size of the object we are searching for in order to select a window of the correct size. We describe a method for object location and classification which enables us to use a small window to identify large objects in the image. The window focusses on features in the image, and an associative memory recalls evidence for objects from these features, avoiding the necessity of knowing the dimensions of the objects to be detected.

INTRODUCTION

The motivation behind this work was the desire to analyse large images, with a view to their classification. In particular, we are looking at the classification of document images according to their content. In order to determine the content of the images, we need to locate and identify objects within the image, and build up a description of the image in terms of these objects. The work discussed below allows for the identification of complex, arbitrarily shaped objects in an image, and allows for them being overlapped or partially obscured by other objects in the image.

Consider a two-dimensional object in an image. The object is composed of a number of features, not necessarily unique, lying in particular relationships to each other. If we can identify the features which comprise the object and determine whether they exist in the correct relationship to each other, then we are able to determine whether the object appears in any image to some level of confidence based on the certainty of correctly identifying features. This is the basis of the Generalised Hough Transform (GHT) (Ballard (1)), used to detect arbitrary shapes in images.

Features and their corresponding parameter values describing the object are stored in a table (in effect, a template). If the object in the image does not match the template of the object perfectly, then the probability of detecting and correctly labelling the object is reduced. Depending on the features used to describe the object and the accuracy with which they are measured very small changes in individual features can make a large change to how well the object fits the template, making this method very susceptible to corruption of the image by noise, or occlusion etc. Grimson and Huttenlocher (2) analyse the GHT, allowing for possible inaccuracies in measurement, and making assumptions about the distribution of possible values for each feature. The results of their analysis show how fragile the method is.

In the GHT, then, the object model is in the form of a template which describes the features comprising the object, and their relationship to each other. Matching a model to the image data involves a search through all available models to decide which model best fits the data. This search may present a computational problem when we have a large number of models, as would be the case if we wished to analyse a real image. We solve the computational problem by implementing the Generalised Hough Transform using a fast associative memory, ADAM (Advanced Distributed Associative Memory)(Austin and Stonham (3), O'Keefe and Austin (4)). ADAM is a neural network specifically designed for application to image analysis tasks (Austin et al. (5)). Simply put, the network allows for the association of input and output patterns in a compact form, so that the presentation of an input pattern stimulates the output of the associated pattern without a serial search of the pattern memory. Thus, the network is ideal for performing the sorts of template matching involved in the GHT. The network is able to generalise from the features it has been trained on, so that it is able to recognise noisy or corrupted versions of features, and returns a measure of confidence in the recognition of each feature.

IMPLEMENTATION

We have implemented an object recognition tool for use in the analysis of images, using the ADAM neural network to implement the GHT. This implementation offers several enhancements over a conventional GHT, and provides the basis for further enhancements of the method to improve the resilience of re-

cognition to noise and clutter in the image.

General Description

The recogniser follows the principles of the GHT. It is trained to associate object features with a parameterised description of the object. The recogniser searches the image for features that it has been trained to recognise, and recalls the associated parameter values. The evidence for the existence of the object in the image is accumulated as the image is searched. When the whole image has been scanned, the resulting accumulation of evidence at each location is analysed to determine the location and class of objects in the image.

Generalised Hough Transform

In this implementation, the parameterisation used to model the target objects is as an association between features and vector/label pairs describing the class of object and the position of the notional object "centre" relative to the features. The features which are used to describe the objects are $m \times m$ blocks of pixels selected from a training example of the object. These features are associated with the relevant vector/label pairs.

The label used for the class of object is an N-point code. that is, it is a string of binary elements of which exactly N are set to 1. The method of encoding object class allows different labels to be superimposed when evidence is accumulated, and allows the dominant label at each position to be retrieved reliably.

The position vector is encoded by approximating it to a position in a grid of points surrounding the feature position. Each position in the grid contains the labels for object classes which have appeared at this position relative to the feature during training. The feature is associated with this grid data structure.

As with the GHT, each feature may occur any number of times, and in many different classes of objects. The feature is therefore associated with any number of vector/label pairs. The template for the object search is the set of all the features and their associated parameter values.

To locate an object in an image, we search for the $m \times m$ blocks of pixels constituting the features we have learned. When a feature is found in the image, the corresponding vector/label pairs are found from the template. From the position of the feature, and the recalled position of the object centre, we find the location of the putative object in the image (figure 1). At this position, we accumulate the label for the object. In fact, we do this for all vector/label pairs that we recall for the feature. This is repeated for every feature we find in the image.

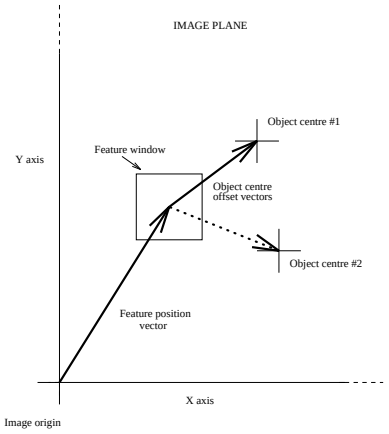

Figure 1: Calculation of object position from feature position and recalled parameters

When the whole image has been scanned for features, we threshold the accumulated evidence. When an object is actually present in the image, and all or most of its features are found, a large number of labels are accumulated for that object at the approximate position of the object centre. Thus, on thresholding the labels accumulated at each point on the image, we can determine what the dominant label is at each position, and the confidence attached to that dominant label.

ADAM neural network

The GHT is implemented using ADAM. ADAM is a fast, high-capacity associative memory, designed for use in image analysis tasks.

ADAM uses binary correlation matrix memories (CMMs) to learn and recall associations between input and output arrays of binary data. An example of such a matrix is shown in figure 2. A is the input array and B is the output array. The matrix M represents the outer product of A and B. All elements of M are set to 0 initially. During training, the elements of M record the intersections of the rows corresponding to the bits set to 1 in A, with the columns corresponding to the bits set to 1 in B. At these intersections the elements of the arrays are set to 1. Each subsequent pattern is written over existing patterns, forming the logical OR of the patterns.

To recall an associated pattern from the memory, the test pattern is placed in A and a matrix multiplication is performed, with the result placed in B. The result will be a (integer) array, which must be thresholded to give a binary result. The thresholding method used is that developed by Austin (3), and is referred to as N-point thresholding (sometimes termed L-max (Casasent and Telfer(6))). In this method, every original pattern in B has N bits

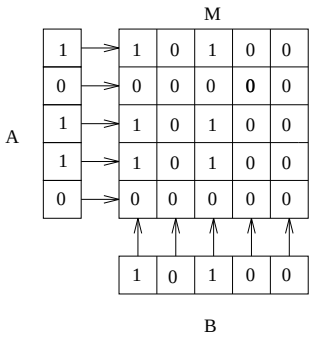

Figure 2: Binary correlation matrix M, with input A and output B

set. On recall, the N elements of the array which have the highest value are set to 1, and all others are set to 0.

ADAM consists of two of these CMMs. Because the input and output patterns may be very large, a single CMM associating input and output would be impractical. ADAM uses an intermediate code which is much smaller than either of the input or output arrays. The first CMM learns associations between the contents of the input array and some intermediate code, selected to be unique for that input, and the second CMM learns the association between the intermediate code and the output. In this way, the total size of the matrices is reduced.

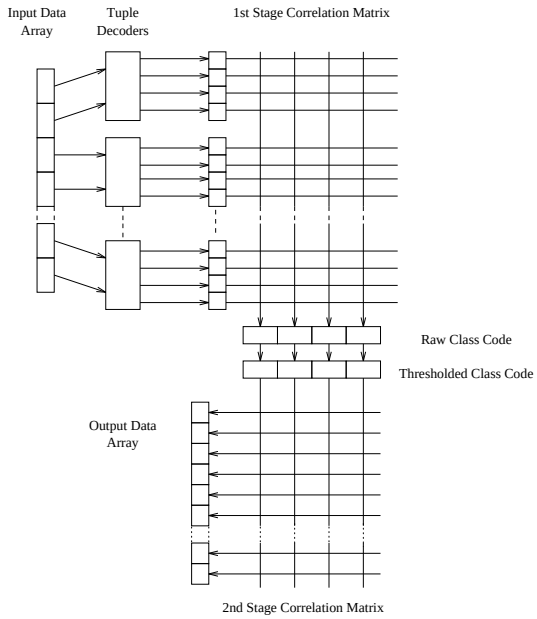

Figure 3: The ADAM associative memory

The layout of ADAM is shown in figure 3. The input array is shown with tupling. This preprocessing of the input splits it into groups of binary elements ("tuples"). Each tuple decoder interprets the values in its tuples as representing a state. The decoder has a separate output for each state, and sets to one the output corresponding to the state of the tuple. As a consequence of this tupling of the input, linearly inseparable patterns may be classified; the input to the first CMM is much more sparse than the original data, thereby reducing saturation of the memory and increasing capacity; and the number of bits set in the input to the CMM is fixed, making thresholding of the output from the CMM simpler.

Implementation

The implementation of the GHT-based recogniser is shown in figure 4. The recogniser is trained by teaching the ADAM to associate features ($m \times m$ blocks of pixels) with object labels and positions. Once the ADAM has been trained, the recogniser can search for the object in an image. An $m \times m$ window is stepped over the test image. At each position, the $m \times m$ block of pixels is extracted. If the ADAM recognises this feature, it recalls the associated vector/label pairs. Recognition of a feature is determined by the confidence with which the first CMM produces the intermediate code — a high confidence is taken recognition of a feature, and low confidence implies the feature is not recognised. The vector/label pairs are recalled in the form of a grid. Object labels appear on the grid points corresponding to the object position relative to the feature.

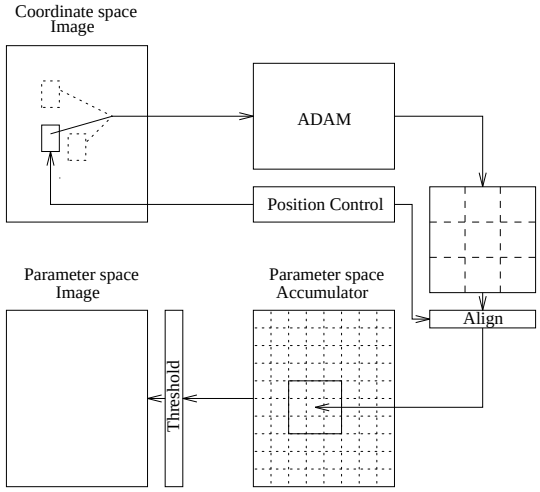

Figure 4: Implementation of the recogniser

The contents of this recalled grid are added into an accumulator grid. Thus, at each position in the accumulator we will get some number of labels accumulated, corresponding to putative objects in the image, for which there is some feature evidence belonging to the object. Once the whole image has been scanned for features, and the evidence has been accumulated, we need to determine where objects are. This we do by looking at every cell in the accumulator and de-

termining whether the accumulated evidence in that cell exceeds the threshold for an object. Assuming that the ADAM has been trained to recognise more than one object, then each cell of the accumulator may hold labels for more than one class of object. This is illustrated in figure 5.

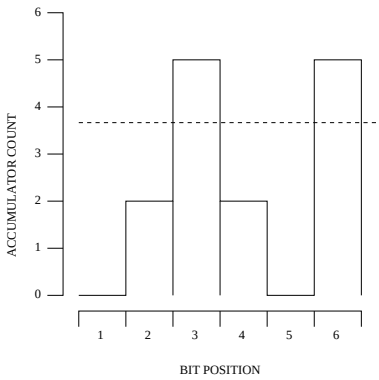

Figure 5: Labels for more than one class of object accumulated in one cell

To determine which is the dominant class of object at this location, we N-point threshold the contents of the accumulator cell. That is, we select the N largest elements in the cell. This gives us the object label "0 0 1 0 0 1". To determine the level of confidence we have in this label, we look at the difference between the "strength" of this label and the other labels in the cell which we are treating as noise, and compare this with the response we expect from the object model.

RESULTS

To demonstrate the capabilities of the recogniser, it has been applied to the problem of recognising objects in fax images. These images present difficulties because of their size, and the presence of noise in the images.

The recogniser has been trained on two of the large characters in the section of image shown in figure 6, specifically on the "A" and the "C". The feature size is 15×15 pixels, and the position of the feature relative to the object is approximated by an 21×21 grid, with seven pixels between grid points. The image is then presented to the recogniser, which must locate these objects in the image. The recognition process takes of the order of twenty seconds, running on a Hewlett Packard Apollo workstation.

Figure 7 shows the results of this initial processing. Each map shows the confidence with which the class of objects has been located at each point in the image. The higher the peak on the map, the higher the confidence that the object is at that point (confidences are scaled to a maximum of 256). It can be seen that object "C" has been located, and both in-

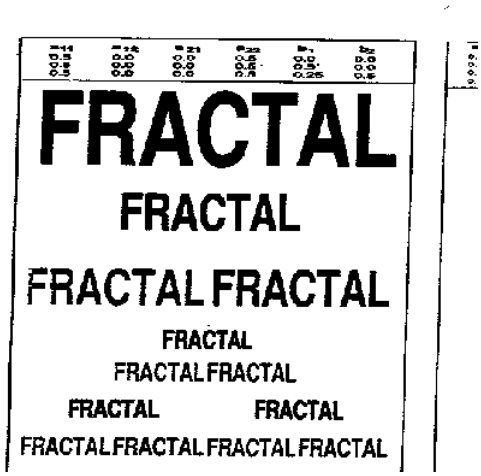

Figure 6: Test image (fax image)

stances of object "A". Note that this separation of the confidence maps for the two classes of objects is merely for convenience in visual interpretation of the results. The accumulator actually holds the information about all objects in the image as overlapped codes. In practice, it is intended that further analysis of the image would make use of the information in the accumulator without the necessity of filtering out particular objects.

It is clear from figure 7 that the recogition of objects is complicated by noise and clutter in the image. The next development will be to introduce feedback into the recognition process. The initial identification of objects in the image is used to recall, for each object, the type and location of the features in the object. This information is combined with the original feature information and passed through the recogniser again, until we are able to confidently label objects.

Another step in the development is the transition to specialised hardware. The C-NNAP processor (Austin et al (7)) is designed to implement ADAM directly in hardware, and to run ADAMs in parallel. Migration to this hardware will give two orders of magnitude increase in the speed with which images are analysed.

CONCLUSIONS

We have implemented an object recogniser, using a GHT-like algorithm, and using ADAM to provide the fast associative look-up. The system can search for multiple objects in parallel — features in the image prompt the recall of the associated object or objects directly, without a serial search of the memory. The search of the image for features is also an obvious candidate for parallelisation, since the detection of a feature and the recall of the object information at any location is independent of feature detection in

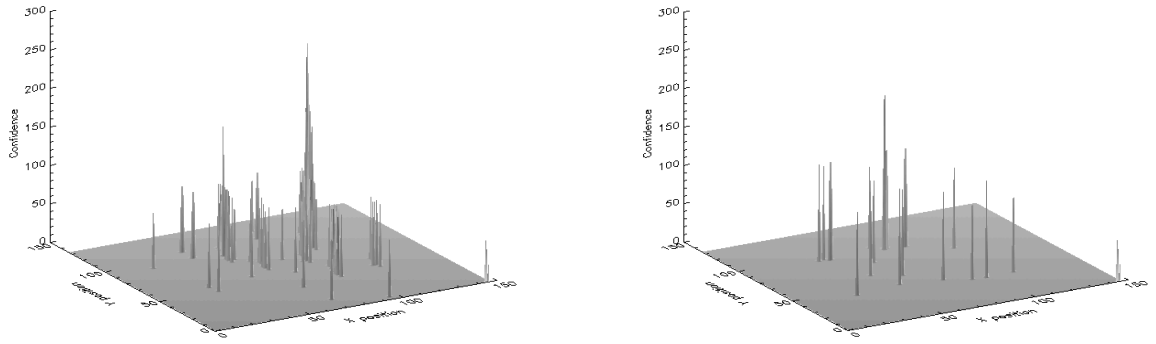

Figure 7: Results of processing. Each map shows the confidence with which the class of objects has been located at each point in the image. Confidences are scaled to a maximum of 256.

the rest of the image. When the recogniser is implemented using the C-NNAP hardware, rather than simulating this in software, we expect a considerable increase in speed.

The drawbacks of the system at the moment relate to the cost of implementing ADAM in software. Even for small numbers of object models with relatively few features, we still need large correlation matrix memories, so the benefits of this implementation are not apparent for small problems. The system also suffers from noise and clutter in the image, and the next stage of development is the inclusion of feedback, so that the recogniser can iterate towards a solution, labelling objects confidently and suppressing the effects of noise.

References

1. Ballard, D. H., 1981, "Generalising the hough transform to detect arbitrary shapes", Patt Rec, 12, 111-122

2. Grimson, W. E. L. and Huttenlocher, D. P., 1990, "On the sensitivity of the Hough Transform for Object Recognition", IEEE Trans. P.A. and M. I.,12(3),255-274

3. Austin, J. and Stonham, T. J., 1987, "The ADAM associative memory", T.R. YCS 94, Department of Computer Science, University of York

4. O'Keefe, S. E. M. and Austin, J., 1994, "Application of an Associative Memory to the Analysis of Document Fax Images", Proceedings, British Machine Vision Conference, BMVA Press,

5. Austin, J., Brown, M., Kelly, I. and Buckle, S., 1993, "ADAM neural networks for parallel vision", JFIT Technical Conference 1993

6. Casasent, D. and Telfer, B., 1992, "High capacity pattern recognition associative processors", Neu. Net., 4(5), 687-698

7. Austin, J., Turner, A., Buckle, S., Brown, M., Moulds, A. and Pack, R., 1994, "The Cellular Neural Network Associative Processor,C-NNAP", T.R. YCS 242, Department of Computer Science, University of York