Neural-fuzzy Feature Detector : A New Approach

Harvey A Cohen
Achan (Software) Pty Ltd.
Melbourne, Victoria, Australia
email harveycohen@aanet.com.au

Abstract

A novel scheme for developing, at low computational cost, neural-fuzzy classifiers based on large-scale, model-based exemplars is outlined. The new method extends the approach that Bezdek applied to train a neural net (NN) Sobel edge classifier by training the NN on the complete population of 3x3 binary image prototypes scored to fuzzy values by a classical operator. We first show that, replacing the fuzzy values of edgeness of the exemplaers, by crisp defuzzified values vastly improved computational speed. A complexity analysis proves however that for operators based on larger windows, the use of complete binary exemplars sets will be computationally intractable. In the new scheme the NN classifier is trained over a hybrid set { selected binary image exemplars with crisp outputs | sampled pixels within a realistic image, these pixels being crisply scored by use of a classic operator.} We demonstrate the scheme by deriving a 5x5 neural fuzzy Plessy operator, far superior to the classic Plessy.

Keywords

Image processing, feature detectors, edge detector, corner, interesting points, fuzzy, crisp, label, Bezdek.

1.0 INTRODUCTION

Feature operators assign to the pixels in an image a label such as edgedness, (Plessy) cornerdness, or (Moravec) Specialness [5]. Following normalisation, classical feature detectors produce a value in some range which represents the extent to which a pixel can be said to be a member of the class under consideration. If this fuzzy value is thresholded the pixel is labeled crisply. The threshold level must be set so as to eliminate all but the clearly defined feature points. The classic operators therefore work well on image regions where there is a high contrast, such as a very sharp edge transition. In fact, these operators work very well within those regions of an image which may be converted to a binary image by simple thresholding. The classic operators perform poorly on low contrast features, such as an edge, which represents only a small grey scale jump. And classic operators are highly sensitive to image noise.

Our objective here is to develop a neural-fuzzy approach to point pixel features which will offer useful insights into the construction of more general feature detectors applicable to the analysis of medical and biological images. For all such notable features, whether point-pixel wise in machine vision, or of grosser character in biological images, there are always exemplars which can be readily scored; but how should one go from the class of exemplars to the general purpose operator? This paper extends an earlier attempt [1], by developing further the capability of arbitrary scale.

Bezdek and collaborators, in several papers [3][4] showed how a neural-fuzzy extension of the 3x3 Sobel operator could be developed by training a neural net over a (equal-weighted) population of all possible 3x3 binary windows, each examplar being scored by the classical Sobel operator. And most notably, Bezdek's neural-fuzzy Sobel outperformed the classic operator in realistic images. We attribute the limited success of this approach to the use of (binary) exemplars on which the classic operator (here the Sobel) gives 'good' values; but find there a definite deficiencies that must be addressed to determine a methodology applicable to features that relate to pixel values over larger (>3x3) windows.

We extend the Bezdek method through the use of a training set comprising: (a) a set of binary image exemplars with crisp outputs; (b) a set of pixels taken from a window within a realistic image, these pixels being crisply scored by use of a classic operator. Our method, which leads to relatively fast training, has the notable feature of being extensible over large windows and for any general window based feature detector.

1.1 THE SOBEL EDGE DETECTOR

In this section, we discuss NN counterparts of the classic Sobel edge detector, beginning with a discussion of the classic Sobel edge detector. The classic Sobel Edge detector [5][7] utilizes the two smoothed gradient operators:

$$D_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad D_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

in conjunction with a threshold T, so that an edge pixel is one for which:

$$E = (1/6)\{ | DX(i,j) | + | DY(i,j) | \} > T$$

Here we assume normalized pixel values in the range

[0 .. 1.0]. The scaling factor of (1/6) is chosen so that the output of the Sobel operator is also in the range [0 .. 1.0]. For later reference, we call E "edgednesss". Applied to a binary image with pixel values of 0 and 1, the 3x3 Sobel

operator returns one of 4 possible values 0, 1/3, 2/3 and 1. Standard texts give examples where the classic Sobel performs well.

In Fig 2, we give an example of failure of Sobel, applied to the Kosh image in Fig 1. Visually, the edges of the original Kosh image are quite apparent.

Figure 1 . 316x500 Kosh image

2.0 NEURAL NET FEATURE DETECTORS

In general, a feature detector can be defined as a computational process which assigns a numeric label to each pixel in a colour or gray-scale image. The label specifies the presence or absence of a feature characteristic such as 'edgeness', 'cornerness', 'interesting' etc. We deal solely with window based techniques, where the pixel is classified based on the pixels in a small surrounding region. Each pixel in the image is thus classified with this 'sliding window' approach. Results are presented below for the Sobel operator.

The simplest NN edge detector was that proposed by Weller [2] who intuitively scored a mere 20 examples of edge-situations in a 3x3 window, these 20 examples serving as the total training set for a feed-forward / back-propagation (FF/BP) neural network. The approach of Weller ignores altogether the capabilities of the classic operators.

A more contemporary approach was proposed by Bezdek and co-workers over several papers [3][4]. Bezdek's approach combines the training of a FF/BP neural network with a labeling scheme based on fuzzy membership values. The key feature of the Bezdek approach is the use of a training set based on a square window in a binary image.

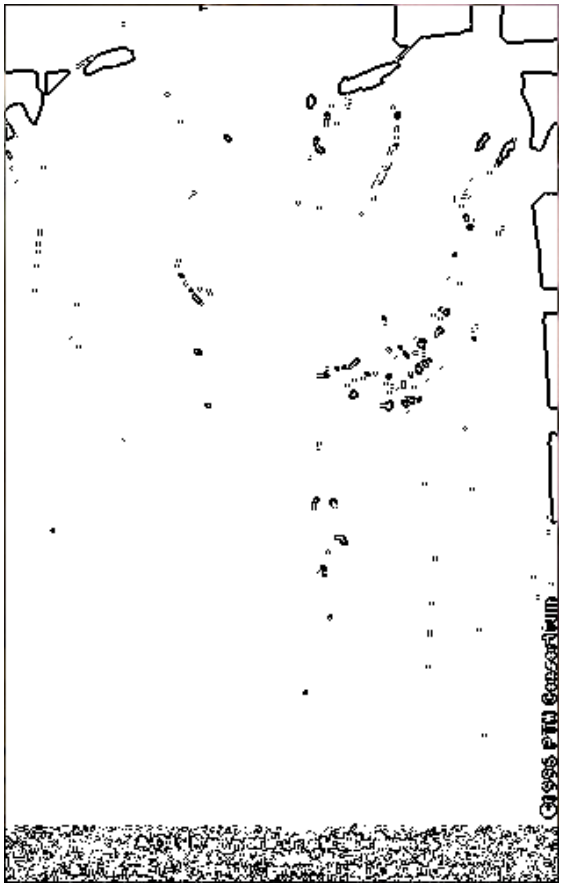

Figure 2. 316x500 Kosh: Output of Sobel edge detector applied on luminance The original image has well defined edges involving small changes in grayscale.

2.1 BEZDEK'S NN COUNTERPART OF SOBEL

The Sobel operator, applied to a binary window and with a suitable scaling factor, has four possible output values : 0, 1/3, 2/3, 1. Bezdek took as a training set all possible 3x3 binary windows, with the desired output for each example being scored by the Sobel operator. This led to a training set of 256 examples with four possible output values, which was used to train a FF/BP network.[1] In Bezdek's scheme the edgeness, E, is considered as a fuzzy membership value of the set of edge points. The neural net is then

[1] The neural networks used in the experiments described here were feed forward networks, trained by back-propagation of errors, configured as follows::

3x3 windows: 9:7:2:1

5x5 windows: 25:10:2:1

trained to give the appropriate value of E for each window. The neural network, although trained on binary windows, is actually applied to a normalized grayscale image, with pixel values scaled so as to range from 0 to 1.0. The output of the trained NN ranges from 0.0 to 1.0 at any pixel, due to the sigmoid activation function (Sigmoid function $1/(1 + e^x)$) of the output unit. Since the NN is applied to a gray-scale image the output is not restricted to the four values of the binary case. A process of defuzzification, equivalent to the choice of a threshold for the classical Sobel, is then applied to the (single) output of the NN.

Figure5. 316x500 Kosh: Output of NN edge detector trained on Sobel edgedness, after defuzzification.

2.2 Problems with Bezdek's methodology[2]

Bezdek's approach is noteworthy in that it does in fact succeed in producing an excellent edge detector. The fully trained NN agrees with the Sobel operator on binary images, but has much greater power than Sobel in the detection of low contrast grayscale edges (See Figs 2 and 6). Bezdek et al also examined a related approach, using the Takagi-Sugeno fuzzy reasoning paradigm. There are, however, two basic problems with this approach which prevent the training of general feature point detectors.

Bezdek's method is not readily extensible to larger scales. For a 3x3 window the training set consists of 2^9 = 512 proto-types, and training is readily achieved in a matter of minutes. If we extend the scale to a mere 5x5 window then using this method we have a training set of 2^25 = 33.55 x 10^6 binary proto-types. Assuming training times are linear in the number of inputs we compute as follows: Training times for the 3x3 NN based operators take of the order of minutes, whereas for 4x4 the corresponding time would be of order of 2^7 = 128 minutes. But for a 5x5 operator training times would take of the order of 2^16 = 64K minutes = 45 days. In fact the linearity is not reasonable, and combinatorial explosion would be far worse.

The second problem arises when we attempt to train a NN on a finely partitioned output space. Bezdek's approach to training a neural net for edge detection involved partitioning the output space of the training set into 4 levels - 0, 0.33, 0.67, 1. The reason for this approach is to ensure that the neural networks output corresponds to a fuzzy membership value between 0 and 1. Once again, this approach is not readily extensible to different scales or even to some different small scale feature detectors. In the case of the Sobel operator we have four discrete levels, but for a general feature detector we may have many more. This training approach has three detrimental effects: a) Increased training times. b) Decreased sensitivity of final network. and c) More points returned with no increase in descriptive power.

In section three we demonstrate these effects by comparison with a NN trained on crisp values. Training to fuzzy values ignores the fact that the final system will be applied not to a binary image, but to a gray scale image. The output level results from applying a sigmoid function to the weighted sum of inputs at the output unit. Clearly, any large inhibitory (negative) input will be mapped to zero, and any large positive value will be mapped to one. The intermediate values between 0 and 1 occur only when there is a degree of uncertainty as to the correct output, i.e. if the net input does not swing either to large negative or positive values. Training a network so that it must hover around these intermediate values results in more uncertainty during classification of gray level images.

Training a network to output either a 0 or 1 when training on *binary* data allows the weight vectors to stabilize and saturate the sigmoids to clearly defined levels. When such a network is applied to a gray scale image, however, ambiguities in classification will result in the sigmoids entering this uncertain region once again. This is in fact the desired response. If we wish the network to output fuzzy membership values, then these values should come as close as possible to 0 or 1 when we are completely certain that a pixel either does or does not belong to the fuzzy set. Membership values should stray into the gray area only when membership level is uncertain.

[2] This analysis extends the discussion presented in [1]

3.0 A NEW 3X3 NN SOBEL

This section discusses a solution to the second problem outlined in Bezdek's method. The strategy we propose involves training to crisp values. Two methods are presented. In the first method we use all possible binary exemplars, so that non-crisp values have to be "defuzzified", assigned to values 0 or 1, using a threshold T = 0.5. The results for the Lena image in Fig's 7, 8 and 9 are striking: Training on binary prototypes with crisp outputs has resulted in a reduction in the number of pixels required to represent the edge image. But the new NN operator has detected features, such as the reflection of the top half of Lena's hat, which are missing in Fig 7.

Figure 6. 256x256x8-bit Lena: Output of NN edge detector trained to duplicate "fuzzy" Sobel edgedness on 256 binary proto-types. Number of edge pixels = 7770.

Retaining only those exemplars in the training set for which a crisp decision is available results in greater sensitivity and swifter training times. Figure 9 shows the results of applying a NN trained with this method to Lena. The top half of Lena's hat is also picked up with this method, and the other edges show greater definition.

Figure 7. 256x256x8-bit lena: output of nn edge detector trained to duplicate "de-fuzzified" sobel edgeness on 256 binary proto-types. Number of edge pixels = 5895

The improvement of the new approach is even more patent in comparing training times. To train over the 256 prototypes, for fuzzy edgedness values, as used by Bezdek et al, 30,000 passes through the data were required. But for the "defuzzified" Sobel outputs, training required only 2000 passes. Training on data for which a "crisp" decision was available further reduced the training time to a mere 500 passes. It is important to note that the output values in each case have a similar distribution across the range [0.0 .. 1.0]. Effectively, this means that we can train a NN on crisp exemplars and still validly interpret the output as a fuzzy membership function of the feature class. Figure 10 shows the distribution of output values for figures 7 and 8.

Figure 8. 256x256x8-bit Lena: Output of NN edge detector trained to duplicate Sobel edgeness on 40 "crisp" proto-types. Number of edge pixels = 6852

4.0 INCREASING THE SCALE

As we discussed above, it would not be possible on a conventional workstation to train by the Bezdek method an operator based on large window sizes. The following presents a method of training an arbitrary feature detector at an arbitrary scale. We present as an example a Fuzzy-NN analogue of the Plessey Operator for a 5x5 window.

4.1 The Plessey Corner Finder

Corner points are more difficult to define than edge points. Corner point techniques tend to find L-structures and points of high curvature, i.e. when an edge changes direction sharply that is a corner. Ideally a corner point detector should ignore isolated points and return only corners, but noisy images can be a problem with this class of techniques. Corner detection attempts to locate points of high curvature in an image, returning strong results for L-structures. Many different approaches have been taken to corner detection, ranging from heuristic techniques to template based techniques to methods based on derivatives. [5] This paper presents NN counterparts to the Plessy corner finder. In this section we discuss the classic Plessey corner

finder.The algorithm is given by Noble [5] a, using a (n*n) window slid over the entire image is as follows::

1> Find I_x and I_y using (n * n) first-difference approximations to the partial derivative.

2> Using a Gaussian smoothing kernel of standard deviation σ, compute the weighted average means $<I^2_x>$, $<I^2_y>$ and $<I_xI_y>$

3> Evaluate the eigenvalues μ_1 and μ_2 of the matrix

$$A = \begin{bmatrix} <I_x^2> & <I_xI_y> \\ <I_xI_y> & <I_y^2> \end{bmatrix}$$

4> compute the 'cornerness' C_p as the ratio:

C_p = Trace(A) / Det(A)

= ($<I^2_x>$ + $<I^2_y>$)/($<I^2_x><I^2_y>$ - $<I_xI_y>^2$)

4> If both C_p is small. μ_1 and μ_2 are both 'large' declare a corner.

Noble [5] has shown that the Plessey operator is suitable only for L corners, as its behavior is unpredictable for higher order structures. Fig 10 shows the result of applying the Plessey operator to the Barb image (Fig 9.) that contains both sharp corners and smoothly varying curves.

Figure 9. 512x512x8-bit Barb

The Plessey operator clearly misses some important features, such as the mouth, the hairline and the right arm. Another problem of the Plessey operator is the number of pixels returned. Where a feature such as a corner can be represented by a single pixel, often the Plessey operator returns multiple pixels to represent a feature. That is to say that pixels neighboring a feature point are often returned as a feature point, leading to many small 'clumps'. This is undesirable both in terms of efficiency of representation and accurate location of feature points.

4.2 Fuzzy Plessey Operator

The key problem is the construction of an appropriate training set. The approach chosen was to develop a hybrid training set consisting of data from two sources. The majority of the data was sampled from an image and scored by a feature detector. For the example presented here this consisted of a set of 5x5 windows at 1000 pixel locations (on a regular grid) within a 256x256x8-bit grayscale Lena image. Each of the windows within the set was scored using a normalized Plessey operator, and then thresholded to produce a "crisp" decision.

For certain feature types, such as corners, the frequency of occurrence in an image is extremely low. Consequently, the data set produced by image sampling contains an overwhelming majority of negative examples. If this were the whole of the training set then

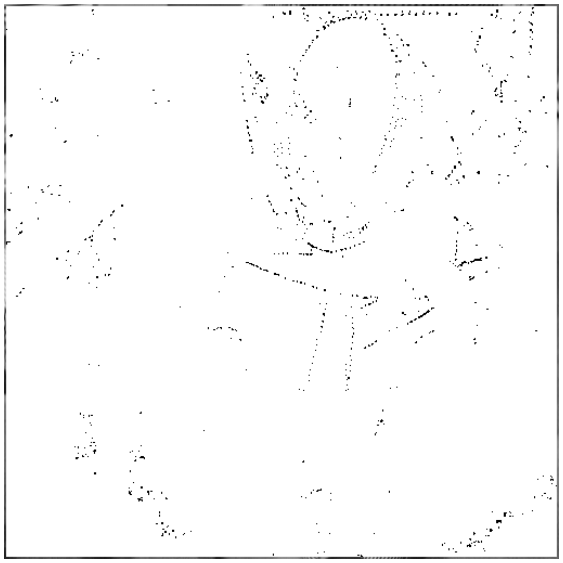

Figure 10. 512x512 Barb: Output of 5x5 Plessey Operator.

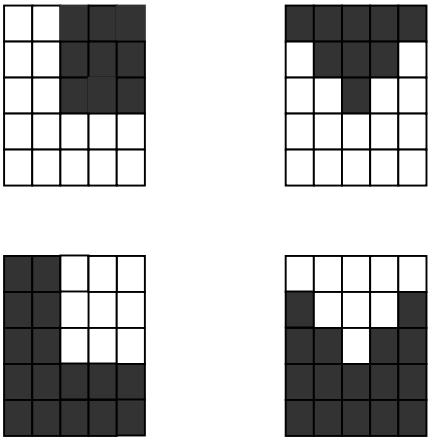

Figure 11 . The 16 binary 'corner' exemplars used in the hybrid training of a neural Plessey operator comprised the four rotational variants of these four 5x5 regions.

the NN thus trained would simply classify every pixel as not a corner. The solution adopted was to include a selection of "hand-ranked" positive examples to balance the training set. The solution adopted was to include a selection of "hand-ranked" positive examples to balance the training set. (It is in this regard that the new method supsedes the approach of [1].

In our example, 16 positive examples of corners were included in the training set. The resulting "hybrid" training set was then used to train a FF/BP NN. The results of applying this network to the barb image are shown in Fig 12.

For the Barb image, the NN corner point detector offers a far more effective abstraction of the original Barb than the Plessey operator (Fig 10). . The Hybrid trained NN picks up many features which are simply missed by the Plessey operator, such as the books in the top left corner.

The other noteworthy difference is the reduction or elimination of pixel "clumping". In Figure 10, most interesting points are marked by more than one pixel, often three or four. The resulting representation is far from efficient. In figure 12, we can see that the NN interesting point detector does not suffer from this problem.

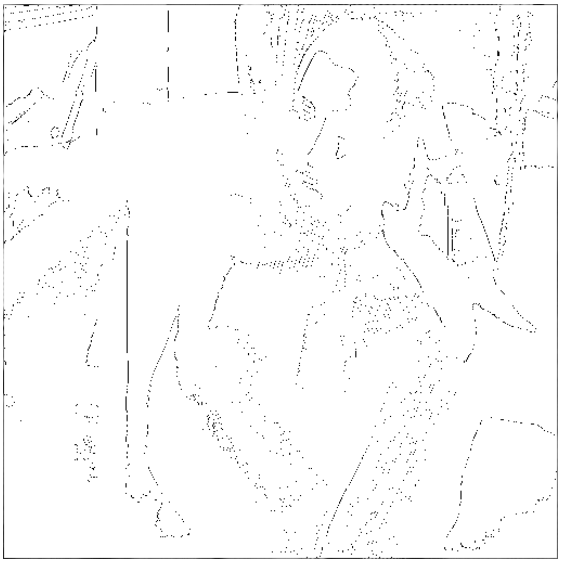

Figure 12.: 512x512x8-bit Barb : Output of NN corner-point detector trained with hybrid training set consisting of hand-ranked exemplars and Plessey scored samples from a realistic image.

4.0 CONCLUSIONS

The ability to generalise from supplied knowledge, and in some cases even modify an approach, is one of the often claimed strengths of neural networks. [6] Certainly, in this study, the fuzzy NN edge detector we developed from the Sobel is clearly superior to the classic Sobel edge detector.

We derived our approach from that of Bezdek,[3][4] where neural networks are trained to replicate fuzzy valued outputs. In [1] two new ideas were developed for training to crisp outputs. The first involved de-fuzzifying the exemplars, resulting in far more rapid training times, increased sensitivity and a more powerful representation with fewer pixels required. The second idea involved retaining only those examples in the training set for which a crisp decision was clearly available and discarding the other examples. This approach resulted in even more rapid training times, with comparable representational power to the de-fuzzified approach.

We found that the distribution of output values, in the range [0.0 .. 1.0], were similar for training both on fuzzy output values and on crisp values. Essentially, this means that an interpretation of the output as a fuzzy membership value of the feature set is equally valid in both cases. When we consider this in conjunction with the highlighted advantages of training to crisp outputs, we believe there is a strong argument in favor of crisp valued training sets.

In this paper, extending well beyond the discussion in [1] we presented a method of generalizing to arbitrary scales and feature detectors. The example given was a hybrid neural-Plessey operator for the detection of corner points. The training set was composed of two parts. The first part was 1000 5x5 windows taken from a realistic image (Lena), scored with the Plessey operator and thresholded. The second part balanced the low frequency of corner points in an image and consisted of 25 hand-ranked corner templates.

The superiority of this approach over the classical Plessey was clearly apparent (see Fig. 10 & 12), where the neural net analog of Plessey detected many critical features missed by the (classic) Plessey corner operator and did so with fewer pixels per feature.

6.0 REFERENCES

[1] Harvey A. Cohen, Craig McKinnon, and J. You, "Neural Fuzzy Feature Detectors", DICTA-97, Aukland, N.Z., Dec 10-12, pp 479-484. Available at

<http://homepage.cs.latrobe.edu.au/image/papers/NeuralFuzzyFeatureDetectors.pdf>

[2] S. Weller, "Artificial Neural Net Learns the Sobel Operators (and More)", (S K Rogers ed) Applications of Artificial Neural Networks II , SPIE Proceedings Vol SPIE-1469 pp 69-76, Aug 1991.

[3] J.C. Bezdek and M. Shirvaikar, "Edge detection using the fuzzy control paradigm", Proc 2nd European Congress on Intelligent Techniques and Soft Computing, Verlag der Augustinus Buchhandlung, Aachenn, Germany, Vol 1, pp 1-12, 1994

[4] J.C. Bezdek and D. Kerr, "Training Edge Detecting Neural networks with Model-Based Examples", Proc 3rd International Confererence on Fuzzy Systems, FUZZ-IEEE'94, Orlando, Florida, USA, pp 894-901, June 26 - 29, 1994.

[5] A. K. Jain, Fundamentals of Digital Image Processing, Prentice-Hall Intenational Editions, pp. 384-389, 1989.

[6] C.G. Looney, Pattern Recognition Using Neural Neworks, Oxford University Press, New York, 1997.