Paying Attention to Relevant Dimensions: a Localist Approach

Mike Page,
MRC Cognition and Brain Sciences Unit,
15, Chaucer Road,
Cambridge,
CB2 2EF, UK.

Abstract.

Localist models of, for example, the classification of multidimensional stimuli, can run into problems if generalization is attempted when many of the stimulus dimensions are irrelevant to the classification task in hand. A procedure is suggested by which a localist model can learn prototype representations that focus on the relevant dimensions only. These permit good generalization which would be lacking in a simple exemplar-based model.

1. Introduction

The research described here represents the first stages of development of a localist neural network for supervised learning that improves its classification performance by paying attention to input dimensions relevant to the task at hand. The work started very much as an applied problem which, as will be seen, benefits from a more theoretical analysis than was attempted at first.

1.1 The Problem

The problem involved the classification of 50-dimensional vectors of reals which had previously been derived from gray-scale images of faces. The faces had been preprocessed using “morphing” techniques so as to standardize the images to a common face-shape. The 110 faces, each comprising 10 000 pixel values, were then subjected to a principal components analysis (PCA), which allowed the faces to be represented as a compressed vector of 50 numbers. Each number represents a coordinate along an axis corresponding to one of those 50 principal components (PC) with the highest eigenvalues. The details of this preprocessing, and the motivation behind it, are given in more detail in [1].

The face set comprised a number of subjects, each posing 7 different expressions, namely anger, disgust, fear, happiness, neutral, sadness and surprise. The numbers of each emotion were approximately balanced and were 17, 15, 15, 18, 14, 17 and 14 respectively. Each 50-dimensional vector could therefore be labelled by the identity of the subject and by the emotion posed. The task was to design a localist network to learn to classify the faces into categories defined by their emotional expression.

As will be seen, and as is perhaps intuitively obvious, this task is more difficult than classifying the faces by identity.

The decision to use a localist network was motivated by earlier work (e.g., [7] [8]) that highlighted the advantages of such models. Of course, given that the task is one of classification, it would have been possible to train, using the back-propagation (BP) learning rule, a standard three-layer (of units) network, with 50 input units and 7 output units each one representing a localist coding of the correct expression-category. Nonetheless, given the reservations expressed by myself and others (see [8] and accompanying commentary) with regard to the plausibility of BP learning, an alternative model was sought. Similarly, a simple two-layer network trained by the delta-rule was avoided in favour of a network constrained such that each category was represented by an output unit whose activation would be maximal for a prototypical category member — not a natural consequence of applying delta-rule learning. This constraint encouraged the use of a radial-basis-function (RBF) network, as will be described below.

2. A Naive First Step

A naive first step, that helped to clarify the nature of the problem, was to attempt a simple nearest-neighbour classification of a given test face-pattern. To be specific, each face was classified according to the emotional label of its nearest neighbour in 50-dimensional space. Performance was extremely poor for the following reason: because each subject posed each expression (with a few exceptions) it is likely that the nearest face to that of subject A posing expression 1 is that of subject A posing a different expression. In these circumstances, in which distance between different expressions posed by the same model is smaller on average than distance between different models posing the same expression, performance of a nearest neighbour classifier is guaranteed to be poor in the expression classification task. One might say that similarity between vectors is dominated by identity at the expense of expression. This is intuitively plausible: It is not difficult to imagine that, even in the full 10 000-dimensional face space, an image of one person posing surprise is more similar to an image of the same person posing, say, happiness than it is to one of another person posing surprise. We can see, therefore, that the task faced by the network is in some sense to learn to pay attention to that subset of the 50 dimensions which defines a subspace in which expressions of the same type do indeed cluster, regardless of the identities of the models.

In order to formalize this idea, and taking an RBF-type approach, I next tried a simple two-layer classifier with 50 input nodes and 7 output nodes each representing a given expression. The weight, w_{ji}, incoming to the i^{th} output node from the j^{th} input node, represented the mean value along the j^{th} dimension of patterns in class i. On presentation of a given pattern, $\boldsymbol{p}$, for classification, the activation, A_i of the i^{th} output node was given by

$$A_i = K - f(\sum_j g(\alpha_{ji} d_{ji})) \tag{1}$$

where K is a constant, f and g are functions to be defined, α_{ji} represents the attention paid by the i^{th} output node to the j^{th} input dimension and d_{ji} is a measure equal to

$|w_{ji} - p_{ji}|$, the absolute value of difference between the j^{th} element of the input vector and the corresponding weight. This idea of attentional weighting is very similar to that found in Nosofsky's generalized context model ([4]) and Kruschke's ALCOVE model ([3])with one exception: in the model described here, each output node can allocate its attention differently. This seemed a useful development since the dimensions relevant to the classification of one emotion might well differ from those relevant to the classification of other emotions. In the work of Nosofsky and Kruschke, the attentional parameters have been envisaged as allowing the "stretching and "shrinking" of the input space to permit more appropriate classification. Here the aim is for each output node (i.e., each emotion classifier) to learn to stretch and shrink its input space in a manner such that patterns corresponding to that emotion class are well clustered.

The first network tested used the simplest version of (1), namely

$$A_i = K - \sum_j \alpha_{ji} d_{ji} \tag{2}$$

There were seven output nodes, one for each expression category, and the bottom-up weight vector to a given node was set to the 50-D mean vector for the corresponding category. Classification of a test vector was implemented by clamping the test vector at the input and activating the output nodes according to (2). The category of the test vector was assumed to be that corresponding to the most active of the output nodes.

Attentional weights, α_{ji} in (2), were initialized to unity so that the classifier starts as a 1-nearest-neighbour classifier in an undistorted input space. We then attempted to ameliorate the poor performance of this classifier (described earlier) by adjusting only the values of the attentional weights. At first, this was effected by a learning rule which can be qualitatively summarized as:

1. present pattern and classify by the nearest weighted prototype.
2. if classification is correct reduce attention to badly matching dimensions (i.e., those with high d_{ji}), increase attention to well matching dimensions.
3. if classification is incorrect increase attention to badly matching dimensions, decrease to well matching dimensions.

The intuition underlying this procedure was that if a test pattern was classified correctly despite a large mismatch (i.e., d_{ji}) along a given dimension, then a sensitivity to values along that dimension is not likely to be critical in calculating the activation of the node representing that category. Conversely, for an erroneously responding category node, more attention should be paid to badly matching dimensions.

Beyond an intuitive appeal, it can also be seen that such a procedure minimises a measure of error with respect to the attentional weights using a gradient-descent-based rule. From (2),

$$\frac{\partial A_i}{\partial \alpha_{ji}} = -d_{ji} \tag{3}$$

which indicates that if we wish to increase the activation of a given node in response to a particular test pattern, then we should subtract a value proportional to d_{ji} from the corresponding attentional weights. A large value of d_{ji} will lead to a large decrease in attention to that dimension, a small value of d_{ji} will lead to a small attentional decrease. The qualitative procedure enumerated above suggests an increase in attention to well-matched dimensions under these circumstances, rather than a small decrease. This can be achieved by renormalizing the attentional weights to a constant sum (or length) after each weight change.

The simple learning rule described above was tested extensively, using a leave-one-out crossvalidation regime. (This regime involves, for each pattern in the training set, training the network on all other patterns in the set and testing the resultant network's performance with the pattern itself – this gives a good test of generalization while maximizing the size of the training set in each case.) While it proved possible to increase performance on the training set using the attentional learning rule, generalization performance was poor. The performance never approached the 95% correct for training patterns and 78% correct for untrained (leave-one-out) test patterns that could be achieved using a standard linear discriminant analysis on this dataset.

3. Theoretical Considerations

It was decided to make a more detailed theoretical analysis of the problem. In particular, classification was conceived of as a Bayesian maximum-likelihood decision. For a multidimensional Normal distribution centred on mean vector $\boldsymbol{m}$

$$p(\boldsymbol{x}) = \frac{1}{(2\pi)^{\mathrm{d}/2}\,|\boldsymbol{\Sigma}|^{1/2}} \exp\{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{m})^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{m})\} \tag{4}$$

where $p(\boldsymbol{x})$ is a probability density function and $\boldsymbol{\Sigma}$ is the covariance matrix. This implies that

$$\log p(\boldsymbol{x}) = \log K - \frac{1}{2}\log(|\boldsymbol{\Sigma}|) - \frac{1}{2}(\boldsymbol{x}-\boldsymbol{m})^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{m}) \tag{5}$$

where K is a constant. (It is often more convenient to deal with the logarithm of the probability rather than the probability itself, since the probability values can become very small. Looking for a class with the maximum log posterior probability is equivalent to seeking the class with the maximum posterior probability because the log function is monotonic increasing.) Various assumptions can be made about the covariance matrix for a given category. For example, if we assume that the off-diagonal elements are zero and that the on-diagonal elements (i.e., the variances of each dimension) are equal to $(1/\alpha_1, 1/\alpha_2, \ldots, 1/\alpha_n)$ (where the second subscripted index of α_{ji} has been dropped here because we are only talking about the distribution within a single category) then

$$\log p(\boldsymbol{x}) = \log K + \frac{1}{2}\sum_{j=1}^{n}\log\alpha_j - \frac{1}{2}\sum_{j=1}^{n}\alpha_j(x_j - m_j)^2 \tag{6}$$

where n is the number of input dimensions. Thus the log probability density at a given vector $\boldsymbol{x}$ is given by a constant minus a linear combination (i.e., attentional weighting) of the dimensionwise distances squared, this time with an additional normalizing factor $\frac{1}{2}\sum_{j=1}^{n}\log\alpha_j$. If this function (with the constant $\log K$ term dropped for convenience) is allowed to replace the previous activation function for output nodes in our RBF network then, given a test vector as input, the output nodes will respond with activations equal to the posterior probability of each class given that test vector (assuming uniform priors) providing the our two assumptions are true, that is, that the covariance matrix is diagonal, and that the attentional weight on a given dimension is equal to the reciprocal of the variance along that dimension. Picking the most active output node corresponds, therefore, to a maximum likelihood decision process that assumes uniform priors.

Looking at the partial derivative of $\log p(\boldsymbol{x})$ with respect to α_j gives

$$\frac{\partial \log p(\boldsymbol{x})}{\partial \alpha_j} = \frac{1}{2}(\frac{1}{\alpha_j} - d_j^2) \tag{7}$$

which is similar to the earlier partial derivative but with the additional $1/\alpha_j$ term. A learning rule based on this partial derivative has the correct form such that α_j tends towards the reciprocal of the within-class variance along the j^{th} input dimension. This is, of course, exactly as is required to satisfy the second of our assumptions above.

Unfortunately, the first assumption, that of diagonal within-class covariance matrices, is overly restrictive, even in cases when, as here, the fact that the 50-D vectors are themselves derived from a PCA ensures that the covariance matrix taken across the whole 110-pattern set is indeed diagonal. Of course it is possible to use a full covariance matrix and to perform the appropriate mathematical calculations to give the value of the posterior log probability of a given class but the neural network implementation becomes somewhat complicated due to the appearance of unwanted crossproduct terms. Another way around this problem, the one adopted here, is to “whiten” each of the classes, that is to preprocess the members of a given class such that their covariance matrix becomes diagonal. One way of doing this is to perform class-conditional PCAs, that is, for each class perform a PCA using only members of that class. For each class, this results in a number of linear components (at most one fewer than the number of patterns in the class) representing a rotated space of reduced dimensionality for which the class-conditional covariance matrix is indeed diagonal.

The structure of a network for implementing this class-conditional whitening is shown in Fig. 1. Each class has a group of preprocessing nodes which are dedicated, via PCA (or similar), to the representation of members in that class in a whitened space. There are a number of self-organizing networks which can perform PCA ([2, 5, 6, 9]), and these might be employed in the learning of the layer 1 to layer 2 connections for each class. This was not done for the preliminary tests presented here. A standard PCA algorithm was run separately for each class to produce directly the values of the corresponding network weights. Because of the restrictions of the PCA algorithm available and the fact that the smallest of the classes only contained 14 faces, the PCA was run only on the first 13 values of the original 50-dimensional pattern set (i.e., those 13 with the highest eigenvalues), to give 13-dimensional vectors at the output of each of the class-specific preprocessing modules. The layer 2 to Layer 3 weights then encode the mean vectors for each of the seven classes, taken across the preprocessed patterns for that class.

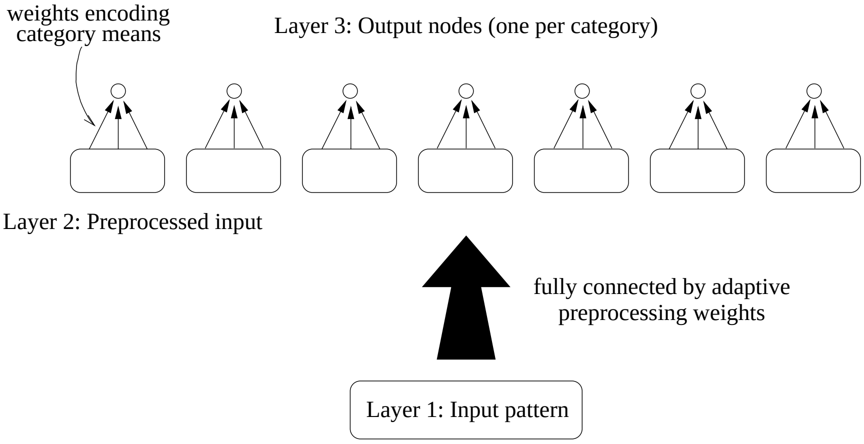
weights encoding category means
Layer 3: Output nodes (one per category)
Layer 2: Preprocessed input
fully connected by adaptive preprocessing weights
Layer 1: Input pattern

Figure 1: The structure of the network

The classification of a test face now involves the following: the reduced 13-dimensional pattern corresponding to the face is clamped at layer 1; each of the class-specific preprocessing modules then processes that vector, resulting in seven different 13-D output vectors, one for each class; these vectors are then processed further using attentional weights which are set to the reciprocal of the within-class variances in the whitened class-specific spaces. Again, in the preliminary testing presented here, these weights were calculated rather than learned. They should, however, be learnable via the rule described in (7). Each output node activates to an extent equal to the log posterior probability of membership of the corresponding class. Picking the most active of the output nodes implements a maximum likelihood decision. Prior probabilities can be incorporated into the model by adding the relevant bias to each of the output units.

4. Results

Preliminary results were encouraging. Using only the first 13 of the 50 dimensions in the original input pattern set, and the preprocessing strategy described above, training pattern performance of 95% correct was achieved. Cross-validation performance using a leave-one-out method, resulted in 93% correct. As a caveat regarding this latter figure, we note that all the training patterns were used in performing the class-specific PCAs from which the weights in the preprocessing stages were derived; likewise, all patterns were used in the setting of the class-specific attentional weights(i.e., reciprocal variances). The left-out pattern was not, however, used in the calculation of the class-conditional mean vectors for the preprocessed patterns. 93% correct is thus likely to be an overestimate of the crossvalidated performance of the network. Nonetheless, for the reduced 13-D input patterns, a linear discriminant analysis gives only 65% of training patterns correct and a crossvalidated (leave-one-out) performance of only

42% correct. This suggests that the ability of the two-stage network effectively to model fully general within-class covariance matrices, confers a considerable performance advantage, though at the cost of increased network complexity.

5. Conclusion

The ideas and results presented here are only preliminary, but they suggest a way in which standard unsupervised learning procedures can be combined to give a network whose generalization abilities are much improved over simple localist alternatives, such as unrefined nearest-neighbour techniques. The enhanced classification relies on using a subnetwork dedicated to each category, which produces as its output the posterior probability of that category given the test stimulus. In doing so, the classification network concentrates its “attention” on tranformed dimensions which show low within-class variance. Put another way, the subnetwork dedicated to a given class examines the test stimulus for evidence of invariant patterns that characterize that class.

In this preliminary work, many implementational shortcuts have been taken, such as the external calculation of class-specific PCs and subsequent within-category variances. Ideally, further work would build such functionality into the framework of a fully self-organizing network. One interesting point to note is that traditional PCA pulls out first those linear combinations that “soak up” the most variance in the input. Because of their high variance, these are the components to which the subsequent classification process pays least attention. It might be better to seek to extract first (nonzero) linear combinations with low variance, since these best characterize what is invariant about a given class, and are the dimensions to which most attention will subsequently be paid.

The network trained as described above, is able to perform classification of faces into emotional categories equivalent to a Bayesian maximum-likelihood decision rule. It assumes that all faces in a given category come from a single normal distribution centred on the category-mean vector. This assumption might not be appropriate – there may be more than one different “type” of happy face. This suggests that the network might usefully be extended by performing an early unsupervised clustering of faces from a given class, with whitening and calculation of the log probability performed separately for each cluster rather than just assuming that each class is equivalent to a single cluster. In the example presented here, however, this procedure does not seem necessary to permit good classification performance. Whichever procedure is used, it is clear that the preprocessing networks generated for the various emotional classes will not be appropriate for other classication tasks, of the faces into, say, identity or gender classes. (Compare the inappropriateness for a given task of hidden units in a BP net trained on another task.) If other classifications are required, then training can proceed as before, with preprocessing networks added accordingly. The resulting network might be described as modular, with separate subnetworks dedicated towards the recognition of emotions and identities. The double dissociations that have been found in patient populations, between emotion and identity recognition, and between the recognition of different emotions, might be seen as supportive of this modular structure.

References

1. Calder A. J., Burton A. M., Miller P., Young A. J., and Akamatsu S. (submitted) A Principal Component Analysis of Facial Expressions. *Vision Research*.
2. Földiák P. (1989) Adaptive Network for Optimal Linear Feature Extraction. In *International Joint Conference on Neural Networks (Washington)*. IEEE New York, Vol.1 401–405.
3. Kruschke J. K. (1992) ALCOVE: An Exemplar-based Connectionist Model of Category Learning. *Psychological Review*, 99(1):22–44.
4. Nosofsky R. M. (1986) Attention, Similarity and the Identification-Categorization Relationship. *Journal of Experimental Psychology: Learning, Memory and Cognition*, 115(1):39–57.
5. Oja E. (1982) A Simplified Neuron as a Principal Component Analyzer. *Journal of Mathematical Biology*, 15:267–273.
6. Oja E. (1989) Neural Networks, Principal Components and Subspaces. *International Journal of Neural Systems*, 1:61–68.
7. Page M. P. A. (1998) Some Advantages of Localist over Distributed Representations. In Bullinaria J. A., Glasspool D. W., and Houghton G., editors, *4th Neural Computation and Psychology Workshop, London, April 1997*. Springer-Verlag London, 3–15.
8. Page M. P. A. (2000) Connectionist Modelling in Psychology. *Behavioral and Brain Sciences*, 23:443–467.
9. Sanger T. D. (1989) Optimal Unsupervised Learning in a Single-layer Linear Feedforward Neural Network. *Neural Networks*, 2:459–473.