Principal Components Extraction by Autoassociative Feed-Forward Networks

Hanseok Ko and R. H. Baran
Naval Surface Warfare Center, White Oak (U25/23)
Silver Spring, MD 20903-5000

Abstract: We describe a methodology for testing the ability of autoassociative, feed-forward, backpropagation networks to extract the principal components from a training set. Some simple examples, treated analytically and validated by numerical trials, suggest that such behavior may be typical.

A previous paper described some results of training three-layer, feed-forward networks to perform autoassociative mappings (Ko, Baran, and Arozullah, 1991). The N-H-N network has the structure shown in Figure 1. It maps N-component vectors \( X \) into N-component vectors \( Z \) through an internal representation \( Y \) which is the H-component vector of the hidden unit activations. The mapping is

\[
\begin{align*}
Z_k &= \sum_{j=0}^{H} W_{kj} Y_j, \quad k = 1, \ldots, N, \\
Y_j &= \sum_{i=0}^{N} V_{ij} X_i, \quad j = 1, \ldots, H,
\end{align*}
\]

where the bias units fully active as \( X_0 = 1 \) and \( Y_0 = 1 \) for every \( X \). Use \( S \) to denote the \( N \times H \) matrix whose columns are the patterns which comprise the training set, i.e.,

\[
S = [X(1) \ldots X(M)]
\]

when \( M \) is the number of patterns. Let \( Y(m) \) stand for \( Y(X(m)) \) and define

\[
T = [Y(1) \ldots Y(m)].
\]

The neural network is perfectly transparent to the patterns when the matrix equation

\[
S = W T
\]

is satisfied by the back end weight matrix \( W \) and the hidden layer activation vectors produced by multiplying the patterns by the front end weights \( V \) and passing the results through the no-memory nonlinear transformations of the usual form \( F(u) = 1/(1 + \exp(-u)) \). For future convenience let \( b(j) \), called an N-component basis vector, be the \( j \)th column of \( W \):

\[
W = [b(0) b(1) \ldots b(H)].
\]
1. Perfect Transparency

The existence and attainability of weights \((\mathbf{W}, \mathbf{W})\) that yield the desired autoassociative mappings depends, of course, on the characteristics of the pattern set and the size of the hidden layer. In this section we examine three disparate cases in which perfect transparency is attained.

1.1 Pattern Matching

A companion paper (Baran, Ko, and Arozullah, 1992) shows how, in general, the application of backpropagation will produce solutions to \((2c)\) when the number of statistically independent patterns is less than or equal to the number of hidden units (bias unit included), i.e., when \(M \leq H + 1\). There we imposed the added restriction \(H < N\) which is essential for data compression. Subject to these restrictions, there are generally many different factorizations of \(S\) having the form \((2c)\). The most obvious is to let \(X(1) = b(0), X(2) = b(0) + b(1), \ldots, X(M) = b(0) + b(H), H = M - 1\). Then \(Y_{0m} = 1\) for every \(m\) and \(Y_{im} = \delta_{im}\) in terms of the Kronecker delta. In this way the columns of \(W\) "match the patterns." But since it takes so long to send all the \(Y\)'s to their binary extrema, gradient descent usually leads to less obvious solutions, corresponding to linear transformations of these basis vectors, in which the activity of the hidden layer is concentrated in a few units having \(Y\) in the range 0.4 to 0.7.

1.2 Flat Lines

On the other extreme, when the patterns are all multiples of the same vector, the network can become transparent to an infinite number of such patterns. An example with \(H = 1\) will illustrate this point. Let each pattern be a vector of \(N\) identical components:

\[
X(m) = a_m \text{col}(1, 1, \ldots, 1),
\]

\(m = 1, 2, \ldots, M\), where \(-1 < a_m < 1\). Let there be a single hidden unit (in addition to the bias unit) and suppose that \(V_{1i} = v\) for every \(i\). Let \(W_{ki} = w_1\) and \(W_{k0} = w_0\) for every \(k\). Pick the \(i\)th component and drop the subscripts on the \(i\)th input and output nodes: \(z_i = z\) and \(x_i = x\). Then the mapping defined by \((1)\) reduces to

\[
z = w_1 F(Nx - V_{10}) + w_0.
\]

Assume that the argument of the activation function is small so that

\[
z = [1/2 + (1/4)(Nv_x - V_{10})]w_1 + w_0
\]

to first order. Set

\[
w_0 = -(1/2)w_1.
\]

Substituting \(z\) for \(x\), the solution is

\[
v = 4/(Nw_1) << 1
\]

with \(V_{10} = 0\).

A program was written to plot the learning curves of \(N-H-N\) backpropagation networks as they cycle through training setups to 10 thousand times. Figure 2 shows some results for 8-1-8 nets using \((3d)\) and \((3e)\) to pre-set the weights together with a typical learning curve of a network that was started with random initial weights \([W_{ij} \sim U(-2, 2), V_{ij} \sim 0]\). Both trained on a set of \(M = 16\) patterns made with \((3a)\) taking the \(a_m\) i.i.d \(U(-5, 5)\). In trials with \(w_1 \geq 5\), the mean absolute error started out below the
level which is the lower limit of the chart. As \( w_1 \) is made smaller, the plateau error level increases. In contrast, the network with random initial weights requires thousands of training cycles (with a learning rate of 0.1) to attain good performance. Examination of the back end weights after the error drops to one percent shows that \( b(0) \) and \( b(1) \) have converged to constant levels to within 0.1%. Thus the slow progress of backpropagation is due to the difficulty of "flattening" the front end weights to make them consistent with (3e).

![mean abs. error](image)

**Figure 2.** Learning curves of some 8-1-8 networks training on 16 flat line patterns. Uppermost curve is for random initial weights; others for indicated values of \( w_1 \) in equation (3d).

1.3 Straight Wires

At the risk of belaboring trivial cases, consider the problem of making the net perfectly transparent to every \( X \) comprised of real components that are bounded above and below. The "straight wires" solution is an N-N-N net with

\[
V_{ij} = \epsilon \delta_{ij}, \quad i = 1, \ldots, N,
\]

\[0 < \epsilon < < 1, \text{ with the input bias unit off } (X_0 = 0). \text{ At the back end, set}
\]

\[
W_{kj} = (1/\epsilon) \delta_{ij}, \quad k, j \in \{1, \ldots, N\}, \text{ and } W_{k0} = -1/(2\epsilon).
\]

Then

\[Z_k = (1/\epsilon)[1/2 + (1/4)(\epsilon X_k)] \cdot 1/(2\epsilon) = X_k.
\]

Figure 3 shows the learning curves of two 6-6-6 nets training on sets of \( M = 24 \) patterns with i.i.d. \( U(-.5, +.5) \) components. The lower curve begins with weights close to (4). The upper curve starts with random initial weights. Examination of the back end weights corresponding to the upper curve as the 0.01 error level is reached shows that they are superficially the same as (4b). Again the observation is that backpropagation works slowly as it "closes in" on the trivial solution.
Figure 3. Learning curves of some 6-6-6 networks training on 24 random patterns. Upper curve is for random initial weights; lower curve is for initial weights close to equations (4).

2. Principal Components Extraction

In many situations the patterns \( X(m) \) can be regarded as consecutive blocks of a weakly correlated time series or as adjacent blocks which collectively partition an image. It will generally be impossible to achieve transparency for all \( M \) patterns when \( H < M \). Therefore the objective is to fit the patterns as closely as possible subject to the constraint that the number of basis vectors is \( H \). Understanding goodness-of-fit in the mean squared error sense, the optimal mapping \( (\mathbf{1}) \) minimizes the sum over all the \( m \) of the squared magnitude of the difference \( D(m) = X(m) - \mathbf{Z}(X(m)) \). Recasting the problem in terms of \( (2b-d) \), the well known solution procedure begins with finding the basis vectors \( \mathbf{b} \) which satisfy

\[
\mathbf{A} \mathbf{b} = \lambda \mathbf{b}, \tag{5a}
\]

where \( \mathbf{A} \) is the \( N \times N \) covariance matrix obtained from the outer product,

\[
\mathbf{A} = E[SS^T], \tag{5b}
\]

in which \( (\mathbf{1}) \) denotes transposition. The expectation is taken over all realizations of a wide sense stationary source of the patterns or it may be approximated as an average over the given pattern set. The solutions \( \mathbf{b} \in \{\mathbf{c}(n), n = 1, 2, \ldots\} \), sometimes identified as principal components, are the eigenvectors of \( \mathbf{A} \). Substituting the \( \mathbf{c} ' s \) for the columns of \( \mathbf{W} \) in \( (2b) \), we see that \( \mathbf{U} \) in \( (2c) \) contains the coefficients of a principal components decomposition of the pattern set.

Does the repeated application of backpropagation to the \( N-H-N \) network cause it to learn a principal components decomposition of the pattern set? Since backpropagation tries to minimize the mean squared error, it would not be too surprising if such a functional equivalence could be shown. To answer the question we propose numerical experiments which measure two properties of the back end weights \( \mathbf{W} \) as training proceeds. First is the closeness with which the columns \( (\mathbf{b}) \) of \( \mathbf{W} \) match the patterns.
Second is the closeness with which they match the components \((\mathbf{c})\). We define the measures exactly as follows. Define an \(M \times H\) array of pattern-to-basis correlation coefficients

\[
\rho_{mj} = \frac{\mathbf{X}(m) \cdot \mathbf{b}(j)}{||\mathbf{X}(m)|| \cdot ||\mathbf{b}(j)||}
\]

and an \(L \times H\) array of component-to-basis correlation coefficients

\[
\eta_{nj} = \frac{\mathbf{c}(n) \cdot \mathbf{b}(j)}{||\mathbf{c}(n)|| \cdot ||\mathbf{b}(j)||}
\]

using the dot \((\cdot)\) to denote the inner product of \(N\)-component vectors. Generally \(L \leq M\). Note \(0 \leq \rho_{mj} \leq 1\). Taking

\[
\rho_j = \sup_m (\rho_{mj})
\]

finds the pattern with which \(\mathbf{b}(j)\) is most strongly correlated and sets \(\rho_j\) equal to that correlation coefficient. Similarly,

\[
\eta_j = \sup_n (\eta_{nj})
\]

measures the similarity of \(\mathbf{b}(j)\) and the principal component that it most strongly resembles. Finally

\[
\begin{align*}
P &= \frac{(P_1 + \cdots + P_H)}{H} \\
Q &= \frac{(Q_1 + \cdots + Q_H)}{H}
\end{align*}
\]

measure the overall similarity of the network's basis set to the pattern set and to the component set, respectively.

### 2.1 Pattern Matching

When the patterns are statistically independent as in section 1.1 above, the covariance matrix \(\mathbf{A}\) is given by a constant factor times the \(N \times N\) identity. Thus the eigenvectors of (5a) are somewhat arbitrary. It can be argued that any \(\mathbf{W}\) which solves the problem has the principal components for its columns, so that \(Q \) approaches unity in the course of training.

### 2.2 Flat Lines

When (3a) produces the patterns, \(E(X_iX_i') = \varepsilon a^2\) for any pair \((i, i')\). Thus \(\mathbf{A}\) is an \(N \times N\) array one 1’s and the only nonzero eigenvector in (5a) is proportional to \(\mathbf{c} = \text{col}(1, \ldots, 1)\). We saw in section 1.2 above that the first basis vector found by the \(N-1-N\) network is \(\mathbf{b}(1) = w_1 \text{col}(1, \ldots, 1)\). Thus the net extracts the principal component. Since \(P = P_{11} = 1\) and \(Q = Q_{11} = 1\), the motion across the \((P, Q)\)-plane, starting from random initial weights that are uncorrelated with any constant level, is from \((0,0)\) toward \((1,1)\) as sketched in Figure 4.

### 2.3 Straight Wires

Once again the covariance matrix approaches the \(N \times N\) identity. Now the number of principal components is \(N\): \(c_j(n) = \delta_{jn}\). Except for a permutation of the \(j\) indices, the basis vectors found in section 1.3
above were the same as these components. Thus $q_{nj} = \delta_{nj}$ and $Q_j = 1$ for each $j$ when the solution is attained. On the other hand,

$$p_{mi} = \frac{\hat{E}_I(m)}{E_I^2} = 0;$$

and the basis vectors are uncorrelated with the patterns themselves. Thus the progress of backpropagation leads the trajectory $(P, Q)$ from the origin to the point $(0, 1)$ as shown again in Figure 4.

3.0 Conclusion

In the general case of weakly correlated patterns we expect to find the kind of behavior suggested by the (less heavily traced) trajectory in Figure 4, which leads from the origin into the upper triangle of the unit square in the $(P, Q)$-plane, signifying the network's discovery of essentially the same principal components as those which would be obtained classically. Although we have yet to finish a program which would actually compute such trajectories, we have examined the back end weights of N-H-N nets trained on larger sets of patterns made by, eg., adding white noise to a smaller set of sinusoidal waveforms. When the number of hidden units equals the cardinality of the smaller set, the basis vectors $\{\hat{p}\}$ are essentially sinusoidal. A variety of similar experiments lead to the same kind of result.

We offer the hypothesis that N-H-N autoassociative networks tend naturally to discover principal components in pattern sets subject to mild constraints. It seems probable that the time required by the network to extract good approximations of the principal components is short compared to the time that might be spent trying to overfit the patterns and that the neural net method might be competitive with the classical procedure in real time adaptive processing applications.

REFERENCES
