Learning a Continuous Hidden Variable Model for Binary Data

Daniel D. Lee
Bell Laboratories
Lucent Technologies
Murray Hill, NJ 07974
ddlee@bell-labs.com

Haim Sompolinsky
Racah Institute of Physics and
Center for Neural Computation
Hebrew University
Jerusalem, 91904, Israel
haim@fiz.huji.ac.il

Abstract

A directed generative model for binary data using a small number of hidden continuous units is investigated. A clipping nonlinearity distinguishes the model from conventional principal components analysis. The relationships between the correlations of the underlying continuous Gaussian variables and the binary output variables are utilized to learn the appropriate weights of the network. The advantages of this approach are illustrated on a translationally invariant binary distribution and on handwritten digit images.

Introduction

Principal Components Analysis (PCA) is a widely used statistical technique for representing data with a large number of variables [1]. It is based upon the assumption that although the data is embedded in a high dimensional vector space, most of the variability in the data is captured by a much lower dimensional manifold. In particular for PCA, this manifold is described by a linear hyperplane whose characteristic directions are given by the eigenvectors of the correlation matrix with the largest eigenvalues. The success of PCA and closely related techniques such as Factor Analysis (FA) and PCA mixtures clearly indicate that much real world data exhibit the low dimensional manifold structure assumed by these models [2, 3, 4].

However, the linear manifold structure of PCA is not appropriate for data with binary valued variables. Binary values commonly occur in data such as computer bit streams, black-and-white images, on-off outputs of feature detectors, and electrophysiological spike train data. Binary neurons also have a long history in the field of neural networks, and their close association with Ising spin systems in statistical physics led to the Hopfield associative memory model based upon point attrac-

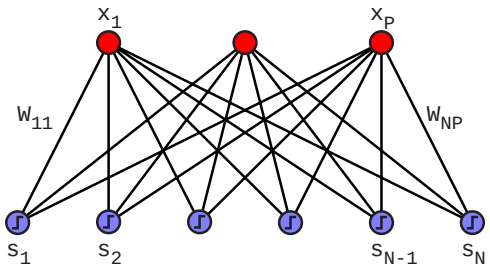

Figure 1: Generative model for binary data using a small number of continuous hidden variables.

tors [5]. The Boltzmann machine is a generalization of the Hopfield model that includes hidden binary spin variables. In principle, its learning algorithm enables the Boltzmann machine to model binary data with arbitrary spin correlations [6]. Unfortunately, the computational time needed for training a Boltzmann machine renders it impractical for most applications.

In this submission, we present a model that uses a small number of continuous hidden variables rather than hidden binary variables to capture the variability in binary valued data. The generative model differs from conventional PCA because it incorporates a clipping nonlinearity. The resulting spin configurations have an entropy related to the number of hidden variables used, and are also connected by spin flips. The learning algorithm is particularly simple, and is related to PCA by a scalar transformation of the correlation matrix.

Generative Model

Figure 1 shows a schematic diagram of the generative process. As in PCA, the model assumes that the data is generated by a small number P of continuous hidden variables x_i. Each of the hidden variables are assumed to be drawn independently from a normal distribution with unit variance:

$$P(x_i) = \exp(-x_i^2/2)/\sqrt{2\pi}. \tag{1}$$

The continuous hidden variables are combined using the feedforward weights W_{ij}, and the N binary output units are then calculated using the sign of the feedforward activations:

$$y_i = \sum_{j=1}^{P} W_{ij}\, x_j \tag{2}$$

$$s_i = \mathrm{sgn}(y_i). \tag{3}$$

Since binary data is commonly obtained by thresholding, it seems reasonable that a proper generative model should incorporate such a clipping nonlinearity. The generative process is similar to that of a sigmoidal belief network with continuous hidden units at zero temperature. The nonlinearity will alter the relationship between the output correlations and the weight matrix W as described below.

The real Gaussian variables y_i are exactly analogous to the output variables in conventional PCA. They lie on a linear hyperplane determined by the span of the matrix W and their correlation matrix is given by:

$$C_{yy} = \langle yy^T \rangle = WW^T. \tag{4}$$

By construction, the correlation matrix C_{yy} has rank P which is much smaller than the number of components N. Now consider the binary output variables $s_i = \mathrm{sgn}(y_i)$. Their correlations can be calculated from the probability distribution of the Gaussian variables y_i:

$$(C_{ss})_{ij} = \langle s_i s_j \rangle = \int \prod_k dy_k \, P(y_k) \, \mathrm{sgn}(y_i) \, \mathrm{sgn}(y_j) \tag{5}$$

where

$$P(\mathbf{y}) = \sqrt{\frac{|C_{yy}^{-1}|}{(2\pi)^N}} \exp\left(-\frac{1}{2}\,\mathbf{y}^T C_{yy}^{-1}\,\mathbf{y}\right) \tag{6}$$

and

$$C_{yy}^{-1} = \lim_{\sigma \to 0} (WW^T + \sigma^2 I)^{-1}. \tag{7}$$

The integrals in Equation 5 can be done analytically, and yield the surprisingly simple result:

$$(C_{ss})_{ij} = \left(\frac{2}{\pi}\right) \sin^{-1}\left[\frac{(C_{yy})_{ij}}{\sqrt{(C_{yy})_{ii}(C_{yy})_{jj}}}\right]. \tag{8}$$

Thus, the correlations of the clipped binary variables C_{ss} are related to the correlations of the corresponding Gaussian variables C_{yy} through the nonlinear arcsine function. The normalization in the denominator of the arcsine argument reflects the fact that the sign function is unchanged by a scale change in the Gaussian variables.

Although the correlation matrix C_{ss} and the generating correlation matrix C_{yy} are easily related through Equation 8, they have qualitatively very different properties. In general, the correlation matrix C_{ss} will no longer have the low rank structure of C_{yy}. As illustrated by the translationally invariant example in the next section, the spectrum of C_{ss} may contain a whole continuum of eigenvalues even though C_{yy} has only a few nonzero eigenvalues.

PCA is typically used for dimensionality reduction of real variables; can this model be used for compressing the binary outputs s_i? Although the output correlations C_{ss} no longer display the low rank structure of the generating C_{yy}, a more appropriate measure of data compression is the entropy of the binary output states. Consider how many of the 2^N possible binary states will be generated by the clipping process. The equation $y_i = \sum_j W_{ij} x_j = 0$ defines a $P-1$ dimensional hyperplane in the P-dimensional state space of hidden variables x. This hyperplane divides the region where $s_i = +1$ from the region where $s_i = -1$. Each of the N spin variables will have such a dividing hyperplane in this P-dimensional state space, and all of these hyperplanes will generically be unique. Thus, the total number of spin configurations s_i is determined by the number of cells bounded by N dividing hyperplanes in P dimensions. The number of such cells is approximately N^P for $N >> P$, a well-known result from perceptron learning [7]. To leading order for large N, the entropy of the binary states generated by this process is then given by $S = P \log N$. Thus, the entropy of the spin configurations generated by this model is directly proportional to the number of hidden variables P.

How is the topology of the binary spin configurations s_i related to the manifold structure of the continuous variables y_i? Each of the generated spin states is represented by a polytope cell in the P dimensional vector space of hidden variables. Each polytope has at least $P+1$ neighboring polytopes which are related to it by a single or small number of spin flips. Therefore, although the state space of binary spin configurations is discrete, the continuous manifold structure of the underlying Gaussian variables in this model is manifested as binary output configurations with low entropy that are connected with small Hamming distances.

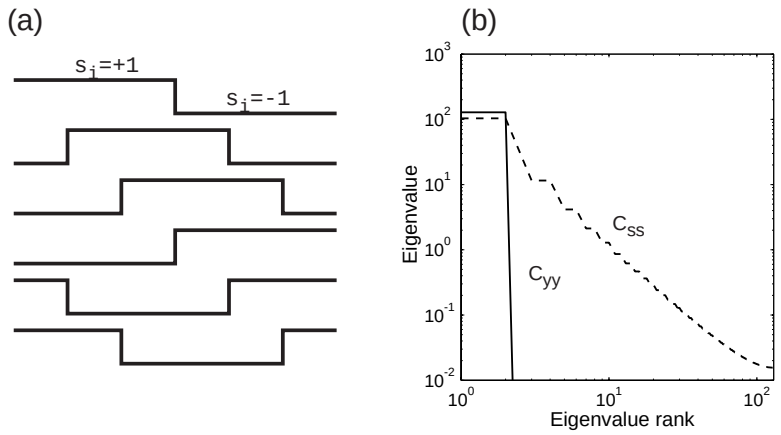

Figure 2: Translationally invariant binary spin distribution with $N = 256$ units. (a) Representative samples from the distribution. (b) Eigenvalue spectrum of C_{ss} and C_{yy}.

Translationally Invariant Example

Given some arbitrary binary data, maximum likelihood may be used with this generative model to learn the appropriate weights. However, Equation 8 suggests a simple learning rule that is much more computationally efficient, though it may not estimate the density as well as maximum likelihood. First, the binary correlation matrix C_{ss} is computed from the data. Then the empirical C_{ss} is mapped into the appropriate Gaussian correlation matrix using the nonlinear transformation: $C_{yy} = \sin(\pi C_{ss}/2)$. This results in a Gaussian correlation matrix where the variances of the individual y_i are fixed at unity. The weights W are then calculated using the conventional PCA algorithm. The correlation matrix C_{yy} is diagonalized, and the eigenvectors with the largest eigenvalues are used to form the columns of W to yield the best low rank approximation $C_{yy} \approx WW^T$. Scaling the variables y_i will result in a correlation matrix C_{yy} with slightly different eigenvalues but with the same rank.

The utility of this transformation is illustrated by the following simple example. Consider the distribution of $N = 256$ binary spins shown in Figure 2(a). Half of the spins are chosen to be positive, and the location of the positive bump is arbitrary under the periodic boundary conditions. Since the distribution is translationally invariant, the correlations $(C_{ss})_{ij}$ depend only on the relative distance between spins $|i - j|$. The eigenvectors are the Fourier modes, and their eigenvalues correspond to their overlap with a triangle wave. The eigenvalue spectrum of C_{ss} is depicted in Figure 2(b) as sorted by their rank. In this particular case, the correlation matrix C_{ss} has $N/2$ positive eigenvalues with a corresponding range of values.

Now consider the matrix $C_{yy} = \sin(\pi C_{ss}/2)$. The eigenvalues of C_{yy} are also shown in Figure 2(b). In contrast to the many different eigenvalues C_{ss}, the spectrum of the Gaussian correlation matrix C_{yy} has only two positive eigenvalues, with all the rest exactly equal to zero. The corresponding eigenvectors are a cosine and sine function. The generative process can thus be understood as a linear combination of the two eigenmodes to yield a sine function with arbitary phase. This function is then clipped to yield the positive bump seen in the original binary distribution.

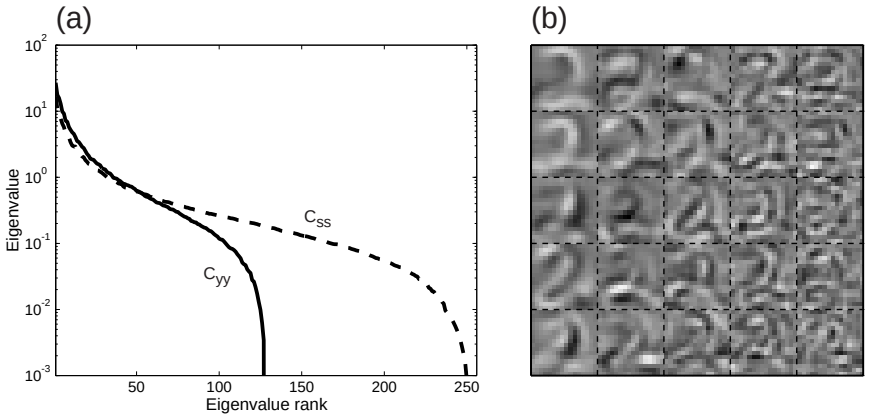

Figure 3: (a) Eigenvalue spectrum of the correlations in handwritten images of twos. (b) The 25 most significant eigenvectors for C_{yy}.

In comparison with the eigenvalues of C_{ss}, the eigenvalue spectrum of C_{yy} makes obvious the low rank structure of the generative process. In this case, the original binary distribution can be constructed using only $P = 2$ hidden variables, whereas it is not clear from the eigenvalues of C_{ss} what the appropriate number of modes is. This illustrates the utility of determining the principal components from the calculated Gaussian correlation matrix C_{yy} rather than working directly with the observable binary correlation matrix C_{ss}.

Handwritten Digits Example

This model was also applied to a more complex data set. A large set of 16×16 black and white images of handwritten twos were taken from the US Post Office digit database [8]. The pixel means and pixel correlations were directly computed from the images. The generative model needs to be slightly modified to account for the non-zero means in the binary outputs. This is accomplished by adding fixed biases to the Gaussian variables y_i before clipping:

$$y_i = y_i^0 + \sum_{j=1}^{P} W_{ij}\, x_j. \tag{9}$$

The means of the binary outputs can then be related to the biases y^0 through the expression:

$$\langle s_i \rangle = \mathrm{erf}\left[\frac{y_i^0}{\sqrt{2(C_{yy})_{ii}}}\right]. \tag{10}$$

This allows the biases to be directly computed from the observed means of the binary variables.

Unfortunately, with non-zero biases, the relationship between the Gaussian correlations C_{yy} and binary correlations C_{ss} is no longer the simple expression found in Equation 8. Instead, Equation 5 gives rise to the following double integral for C_{ss}:

$$(C_{ss})_{ij} = \int\int dt_i dt_j P(t_i, t_j) \mathrm{sgn}(t_i + y_i^0) \mathrm{sgn}(t_j + y_j^0) \tag{11}$$

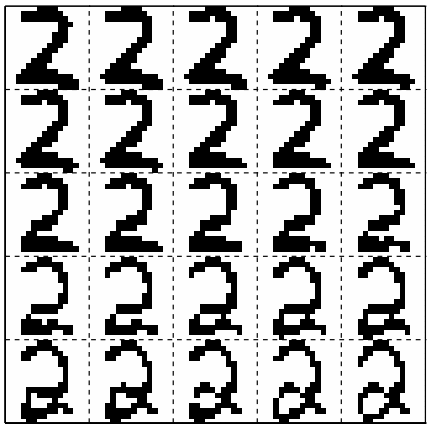

Figure 4: Morphing between two different instances of a handwritten two. The original digit examples are the upper left and bottom right images in the montage.

where

$$P(t_i, t_j) \propto \exp\left\{ -\frac{1}{2} \begin{bmatrix} t_i \\ t_j \end{bmatrix}^T \begin{bmatrix} (C_{yy})_{ii} & (C_{yy})_{ij} \\ (C_{yy})_{ij} & (C_{yy})_{jj} \end{bmatrix}^{-1} \begin{bmatrix} t_i \\ t_j \end{bmatrix} \right\}. \tag{12}$$

Equation 11 can be reduced to a single Gaussian integral of a product of error functions.

Given the empirical pixel correlations C_{ss} for the handwritten digits, the integrals in Equation 11 are numerically solved to yield the appropriate Gaussian correlation matrix C_{yy}. The correlation matrices are diagonalized and the resulting eigenvalue spectrums are shown in Figure 3(a). The eigenvalues for C_{yy} again display a characteristic drop that is steeper than the falloff in the spectrum of the binary correlations C_{ss}. The corresponding eigenvectors of C_{yy} with the largest positive eigenvalues are depicted as images in Figure 3(b). These eigenmodes represent common image distortions such as rotations and stretching and appear qualitatively similar to those found by the standard PCA algorithm.

The correlation matrix C_{yy} also happens to contain some small negative eigenvalues. Even though the binary correlation matrix C_{ss} is positive definite, the transformation in Equation 11 does not guarantee that the resulting matrix C_{yy} will also be positive definite. The precise physical meaning of these negative eigenvalues is not clear at this point; however, their presence indicates a shortcoming of the clipped Gaussian generative process for modelling this particular data set.

Although the eigenvalue spectrum of C_{yy} exhibits a more rapid decay than C_{ss}, as with conventional PCA there is still some freedom in choosing the number of eigenvectors to model the data. Here a generative model with weights W composed of the $P = 25$ eigenvectors shown in Figure 3(b) is used to fit the handwritten twos. Whether this model quantitatively fits the original digit distribution better than conventional PCA still needs to be determined, but the utility of the nonlinear generative model is illustrated by Figure 4. The first and last image in the figure are two distinct examples of a handwritten two, and the generative model is used to morph between the two examples. The hidden values x_i for the original images are first determined using an iterative approximation, and the intermediate images in the morph are constructed by linearly interpolating in the vector space of the hidden units. Because of the clipping nonlinearity, this induces a nonlinear mapping in the

outputs with binary units being flipped in a particular order as determined by the generative model. In contrast, morphing using conventional PCA would result in a simple linear interpolation between the two images, and the intermediate images would not look anything like the original binary distribution [9].

These examples show the value of using the clipped generative model to learn the correlation matrix of the underlying Gaussian variables rather than using the correlations of the outputs directly. The clipping nonlinearity is convenient because the relationship between the hidden variables and the output variables is particularly easy to understand; however, this nonlinear model can also be generalized for other types of data that is not binary in nature. The learning algorithm differs from other nonlinear PCA models and autoencoders because the inverse mapping function need not be explicitly learned [10, 11, 12]. Instead, the correlation matrix is directly transformed from the observable variables to the underlying Gaussian variables. The correlation matrix is then diagonalized to determine the appropriate feedforward weights. We are currently exploring methods by which the appropriate nonlinear generative function can also be adapted to yield the most efficient representation.

We acknowledge the support of Bell Laboratories, Lucent Technologies, and the US-Israel Binational Science Foundation. We also thank H. S. Seung for helpful discussions.

References

[1] Jolliffe, IT (1986). *Principal Component Analysis.* New York: Springer-Verlag.

[2] Bartholomew, DJ (1987). *Latent variable models and factor analysis.* London: Charles Griffin & Co. Ltd.

[3] Hinton, GE, Dayan, P & Revow, M (1996). Modeling the manifolds of images of handwritten digits. *IEEE Transactions on Neural networks* **8**, 65–74.

[4] Tipping, ME & Bishop, CM (1997). Mixtures of principal component analysers. *Fifth International Conference on Artificial Neural Networks*, 13–18.

[5] Hopfield, JJ (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences of the U.S.A.* **79**, 2554-2558.

[6] Ackley, DH, Hinton, GE, & Sejnowski, TJ (1985). A learning algorithm for Boltzmann machines. *Cognitive Science* **9**, 147–169.

[7] Minsky, ML & Papert, S (1969). *Perceptron.* Cambridge, MA: MIT Press.

[8] LeCun, Y *et al.* (1989). Backpropagation applied to handwritten zip code recognition. *Neural Computation* **1**, 541–551.

[9] Bregler, C, & Omohundro, SM (1995). Nonlinear image interpolation using manifold learning. *Advances in Neural Information Processing Systems* **7**, 973–980.

[10] Kramer, MA (1991). Nonlinear principal component analysis using autoassociative neural networks. *AIChE Journal* **37**, 233–243.

[11] Hastie, T and Stuetzle, W (1989). Principal curves. *Journal of the American Statistical Association* **84**, 502–516.

[12] Demers, D, & Cottrell, G (1993). Nonlinear dimensionality reduction. *Advances in Neural Information Processing Systems* **5**, 580–587.