Nonlinear Autoassociation is not Equivalent to PCA

Nathalie Japkowicz
Faculty of Computer Science
Dalhousie University
Halifax, Nova Scotia
Canada
Nathalie.Japkowicz@Dal.Ca

Stephen José Hanson
Department of Psychology
Smith Hall
Rutgers University
Newark, NJ 07102
jose@kreizler.rutgers.edu

Mark A. Gluck
Center for Molecular &
Behavioral Neuroscience
Rutgers University
Newark, NJ 07102
gluck@pavlov.rutgers.edu

Abstract

A common misperception within the Neural Network community is that even with nonlinearities in their hidden layer, autoassociators trained with Backpropagation are equivalent to linear methods such as Principal Component Analysis (PCA). The purpose of this paper is to demonstrate that nonlinear autoassociators actually behave differently from linear methods and that they can outperform these methods when used for latent extraction, projection and classification. While linear autoassociators emulate PCA and thus exhibit a flat or unimodal reconstruction error surface, autoassociators with nonlinearities in their hidden layer learn domains by building error reconstruction surfaces that, depending on the task, contain *multiple* local valleys. This particular interpolation bias allows nonlinear autoassociators to represent appropriate classifications of nonlinear

multi-modal domains, in contrast to linear autoassociators which are inappropriate for such tasks. In fact, autoassociators with hidden unit nonlinearities can be shown to perform nonlinear classification and nonlinear recognition.

An autoassociator is a feedforward connectionist device whose goal is to reconstruct the input at the output layer. When used with a hidden layer smaller than the input/output layer and linear activations only, the autoassociator implements a compression scheme which was shown to be equivalent to Principal Component Analysis (PCA)—also known as *Singular Value Decomposition*—by [Baldi and Hornik1989]. Another interesting and related result was obtained by Bourlard and Kamp who claim, in their 1988 paper:

> "(...) for autoassociation with linear output units, the optimal weight values can be derived by standard linear algebra, consisting essentially in singular value decomposition (SVD) and making thus the nonlinear functions at the hidden layer completely unnecessary",
> [Bourlard and Kamp1988].

[Bourlard and Kamp1988] backed their claim using theoretical considerations while [Cottrell and Munro1988] arrived at the same conclusion, independently, using an experimental methodology.

Paradoxically, nonlinear autoassociators are also famous for their capacity to solve the encoder problem ([Rumelhart *et al.*1986]), a problem which cannot be solved by PCA because of the singularity of the principal components.[1] Moreover, recent research suggests that nonlinear autoassociators are well-suited for

[1]As a matter of fact, [Kruglyak1990] and [Phatak *et al.*1993] demonstrate further that for any N, the N-input, N-output encoder problem can be *represented* by an autoassociator of only two nonlinear hidden units. The fact that large encodings can actually be *learned* by two-hidden-unit autoassociators was demonstrated by [Zipser1989] who showed that such devices are capable of learning how to encode the position of a spot (or cluster) of normally distributed points appearing at random locations on a 10X10 squared array.

classification of certain types of *nonlinearly separable* and *multi-modal* domains ([Japkowicz *et al.*1995], [Petsche *et al.*1996]), another task for which PCA and linear autoassociation are not appropriate because of their linear restriction.

In spite of the obvious contradiction exhibited by these facts, there has been little effort at explaining the source of this inconsistency. The purpose of this paper is to demonstrate experimentally that [Bourlard and Kamp1988]'s claim does not necessarily hold and to analyze the differences between PCA and nonlinear autoassociators when it does not. In particular, we test the performance of nonlinear autoassociators relative to PCA and linear autoassociation on a nonlinear multi-modal classification problem inspired by the projected topology of a real-world domain. After showing that there is, indeed, a difference in the classification accuracy obtained by the linear and nonlinear devices on this domain, we explain this difference by comparing the reconstruction error surfaces constructed by each system over the test domain.

More specifically, our experiments demonstrate that nonlinear sigmoidal single-hidden layer autoassociators may sometimes interpolate sets of points differently from PCA. In particular, we show that, on the test domain considered, while PCA (or equivalently, linear autoassociation) exhibits flat or unimodal reconstruction error surfaces, nonlinear sigmoidal autoassociators build error reconstruction surfaces that contain multiple local valleys. As a matter of fact, the reconstruction error surfaces built by the sigmoidal autoassociators are closely related to the reconstruction error surface built by a radial basis function autoassociator, thus suggesting that when the sigmoidal autoassociator does not operate like PCA—computing a globalized solution to the interpolation problem—, then it operates like a radial basis function autoassociator—computing a localized solution to the interpolation problem.[2]

[2]Although both the sigmoidal and RBF autoassociators are nonlinear, all

We begin our study by discussing the limitations of [Bourlard and Kamp1988]'s claim. We then demonstrate experimentally that these limitations can indeed be exceeded by standard problems within the context of classification and we analyze these results, concluding with some speculations as to why sigmoidal autoassociators sometimes do and sometimes do not operate like PCA.

1 Limitation of the Conventional Wisdom

When considering the results of [Baldi and Hornik1989], [Bourlard and Kamp1988] and [Cottrell and Munro1988] and the evidence that nonlinear autoassociators can solve problems that cannot be solved by PCA ([Rumelhart *et al.*1986], [Japkowicz *et al.*1995], [Petsche *et al.*1996]), an important question comes to mind:

> Does the [Bourlard and Kamp1988] claim hold in all cases or does it depend on certain assumptions concerning the underlying domain or the autoassociator?

A careful look at the discussion laid out in [Bourlard and Kamp1988] reveals that the claim actually does depend on a particular condition. More specifically, let F be the nonlinear function present at the output of the hidden units of a nonlinear autoassociator (i.e., F is the function that makes the autoassociator nonlinear). The assumption made by [Bourlard and Kamp1988] in order to carry out their demonstration is that for small values of x, $F(x)$ can be approximated as closely as desired by the linear part of its power series expansion. However, this means that $x = h$, the vector of pre-synaptic hidden unit activations (i.e., the hidden unit activations prior to their transformation by F), must be in the

references to "nonlinear autoassociators" in the remainder of this paper correspond to references to sigmoidal autoassociators.

linear range of the squashing function. This remark suggests that nonlinear autoassociators do not necessarily emulate PCA when the net inputs are outside the linear range of the squashing function.

The remainder of the paper demonstrates experimentally that there are indeed differences between the linear and nonlinear schemes and that these differences have practical consequences for the task of classification.

2 Experiments

We now describe the two sets of experiments we conducted for this paper. We first present the four devices considered and compared in our study as well as the task on which they were compared. We then describe the test domain used for this comparison followed by our experiments and their results. These results are subsequently discussed in Section 3.

2.1 Specification of the Devices and Description of the Task

Devices The devices we used in our experiments are Principal Component Analysis (PCA) and three autoassociators. From an analytical point of view, the four devices differ in that two of them are purely linear while the other two have nonlinear capabilities. The three autoassociators are connectionist methods which differ from one another with regard to the type of autoassociator they use. In more detail, we compare a one-hidden-layer autoassociator whose hidden and output layers are both linear (L-L); a one-hidden-layered autoassociator whose hidden layer is nonlinear but whose output layer is linear (NL-L); and a one-hidden-layered autoassociator whose hidden layer and output layer are both nonlinear (NL-NL). In each of these devices, the function used in the nonlinear units is the usual logistic function.

Task The operation of the four devices just described was contrasted on the task of classification. A formal definition of the classification problem typically involves an input vector x, and a discrete response vector y which are such that the pair (x, y) belongs to some unknown joint probability distribution, P. The goal of classification is to induce a function $f(x)$ from a set $(x_1, y_1), (x_2, y_2)$, ... (x_N, y_N) of training examples, so that $f(x)$ predicts y. In this paper, we assume that y is a binary vector. Although classification by Neural Networks is typically performed using a discrimination-network[3], it can also be performed using a recognition approach implemented by PCA or the autoassociator. This approach is discussed in [Oja1983] for PCA and in [Hanson and Kegl1987] for autoassociators and operates as follows[4]: during the training phase, the autoassociator is taught how to reconstruct examples of the concept. Once training is completed, classification is performed on a new vector x_{Test} by comparing its

[3]Using such a scheme, y_i, the response variable associated with input vector x_i, takes on (in the simplest case) a value of "1" or "0" which is interpreted as to mean that x_i belongs or does not belong to the conceptual class of the problem. After training the network to approximate the function $f(x)$ from which the training examples are believed to have been generated, it is expected that, if the training set was appropriately designed, the network will be able to compute the appropriate label ("1" or "0") for any input vector of size d, the size of the input layer, even if that vector did not appear in the training set.

[4]We describe the method for autoassociators, taking into consideration that it is similar for PCA.

reconstruction error[5] to a threshold, and assigning it to the conceptual class if the reconstruction error is smaller than this threshold and to the other class otherwise. The idea behind this recognition-based classification scheme is that since the autoassociator is trained to compress and decompress examples of the conceptual class only, when tested on a novel data point, it will compress and decompress it appropriately if this example belongs to the conceptual-class, but it will not do so appropriately if the example does not belong to the conceptual class.

2.2 Specification of the Test Domain

In order to compare and explain the classification performance of the four devices considered in this study, two experiments were conducted for each classifier on a two-dimensional artificial domain. This problem was inspired from real-world data as shown in Figure 1 which illustrates the transition from real-world to artificial data. Specifically, Figure 1(a) displays a 2-dimensional representation of sonar target recognition data available from the U.C. Irvine Repository for Machine Learning. These data were compressed from sixty features to two, using Principal Component Analysis. In this plot, black triangles and white circles correspond to actual points (mines and rocks, respectively) and data clusters are indicated by polygons. Figure 1(b) shows the same two-dimensional representation of sonar target recognition data, but this time with means replacing the clusters[6]. Although this figure is just a two-dimensional projection of the original data set, it clearly indicates the typical kind of problem that the autoas-

[5]The reconstruction error corresponds to $\Sigma_{j=1}^{d}[x_{Test}^{j} - g(x_{Test}^{j})]^2$, where g is the function implemented by the autoassociator and x_{Test}^{j} and $f^j(x_{Test})$ represent input unit j and output unit j of the network, respectively.

[6]Clusters of fewer than three points, however, were ignored.

sociator has to deal with: *multi-modality.* We chose a simple two-dimensional abstraction of the multi-modality problem for benchmarking purposes and explaining the capabilities of autoassociators. This artificial domain is illustrated in Figure 1(c) and consists of four conceptual data clusters with means represented by black triangles and nine counter-conceptual data clusters with means represented by white circles. In more detail, the conceptual clusters are located at points $(.2, .2), (.2, .8), (.8, .2)$ and $(.8, .8)$, respectively, while the counter-conceptual clusters are located at points $(.1, .1)$, $(.1, .9)$, $(.2, .5)$, $(.5, .2)$, $(.5, .5)$, $(.5, .8)$, $(.8, .5)$, $(.9, .1)$ and $(.9, .9)$, respectively. Each cluster is composed of 50 points normally distributed around these means and with variance $\sigma^2 = .01$. The counter-conceptual clusters are further divided into internal and external clusters. Internal counter-conceptual clusters correspond to the counter-conceptual data that are located inside the convex hull defined by the conceptual data while external counter-conceptual clusters correspond to the counter-conceptual data located outside this convex hull.

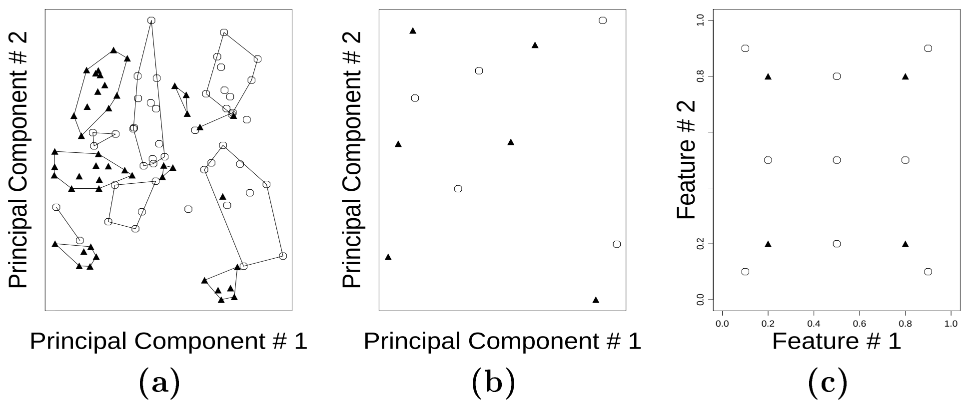


Figure 1: Transition from the sonar detection domain to the artificial non-linear domain.

2.3 Experiment Set # 1

The first experiment we conducted consisted of comparing the classification accuracy of the four systems on the test domain of Figure 1(c). After having been tuned to their optimal settings, the four devices were trained on conceptual data and a threshold was established by fitting the reconstruction error obtained on additional conceptual data to a Gaussian and setting a boundary corresponding to a pre-specified confidence interval. Subsequently, the systems were tested on a testing set containing both instances and counter-instances of the concept.

Tuning the Devices The number of hidden units used by the nonlinear connectionist systems was determined by running each network with 1, 2, 4, 8, 16, 32, and 64 hidden units on five cross-validation sets containing 25 points per cluster. It was concluded that in order to reach a good classification performance, the two nonlinear autoassociators had to be trained with 16 hidden units. It was also shown that the networks had converged by Epoch 2000 which was thus selected as a stopping point. Optimization, in this experiment, was performed using the backpropagation procedure with standard learning rate and momentum of 0.05 and 0.9, respectively. For PCA, since the test domain is two-dimensional, only one or two principal components could be used. It was determined that one principal component yields a better classification rate than two, and, therefore, the PCA experiment was conducted using a single principal component. Linear autoassociation performed better as well with a single hidden unit than with two or more; therefore, it was also tested with a single hidden unit. The stopping criterion, learning rate and momentum used by the linear autoassociator were the same as those used by the nonlinear devices. In the four systems, the threshold was set so as to allow for a 97% confidence interval.

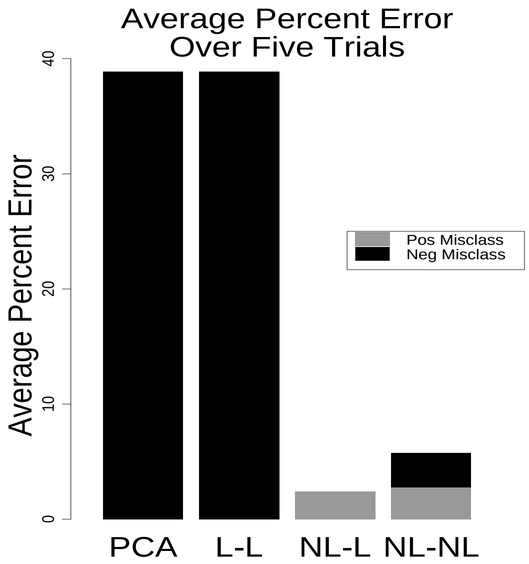


Figure 2: Classification Error Rate of the four classification schemes. Positive misclassification rates are indicated in gray while negative ones are indicated in black.

Accuracy Rates The results we obtained in this experiment are presented in Figure 2. This figure plots the average classification error rates over five trials obtained by the four systems on the test domain of Figure 1(c). The graph shows that PCA and L-L obtained very large classification error rates whereas the two nonlinear autoassociators obtained low error rates. This demonstrates that while PCA and L-L can technically be used for classification, they are not that useful on the problem of Figure 1(c) whereas both NL-L and NL-NL yield acceptable classification rates on that problem.[7] This result, thus,

[7]As a matter of fact, nonlinear autoassociation-based classification was previously used in practical settings involving CH-46 helicopter gearbox and motor monitoring ([Japkowicz *et al.*1995], [Petsche *et al.*1996]) and yielded accurate results.

confirms that there can be a difference in the computations performed by PCA (or linear autoassociation) and nonlinear autoassociation, and that this difference has practical consequences for the task of classification. In order to analyze this difference in more depth, we conducted the experiments presented in the next section.

2.4 Experiment Set # 2

The experiments described in this section seek to explain the results obtained in the previous set of experiments by computing and plotting the reconstruction error surfaces constructed by PCA and the three connectionist systems after being trained on the conceptual data (i.e., the data summarized by black triangles in Figure 1(c)). In other words, we are interested in finding out how the interpolation strategy used by the various devices considered in this study differ from one another.

Tuning the Devices Within the context of Experiment Set # 2, two different hidden unit settings were tested: *expansion* and *compression*. This was done so as to find out whether or not the systems operate qualitatively differently when used is a non-bottleneck or in a bottleneck fashion. In the expansion setting, the same 16 hidden unit setting was used for the nonlinear systems as those used in the first experiment while PCA and the linear autoassociator were tested with two principal components or hidden units. In other words, the optimal non-bottleneck setting for the nonlinear systems were compared to the only non-bottleneck setting possible for the linear systems.[8] All the other parameters remained the same as in Experiment Set # 1, except for the learning rate of L-L

[8]Note that for the linear autoassociator, hidden layers greater than two are actually possible, but they are equivalent to hidden layers of size two.

which had to be increased from 0.05 to 0.1 for full convergence to take place. In the compression setting, the number of principal components and hidden units were restricted to one for all the systems since it is the only bottleneck setting available given that the test domain is two-dimensional. All the other parameters remained the same as in Experiment # 1.

Expansion Results The results we obtained using the expansion setting of Experiment Set # 2 are presented in Figure 3 which displays 3-D plots of the error ratio surfaces constructed by NL-L and NL-NL, respectively. The results for PCA and linear autoassociation are not displayed since they exhibit a flat reconstruction error surface. In the graphs of Figure 3(a) and (b), the plots are

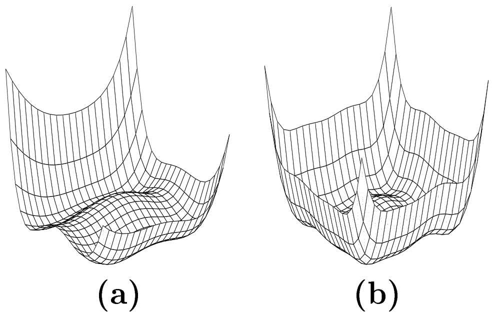


Figure 3: Reconstruction errors of NL-L (Figure (a)), and NL-NL (Figure (b)) with 16 hidden units on the artificial domain generated from Figure 1(c).

drawn along the x-, y-, and z-axes where x corresponds to the x-axis of the input domain, y, to its y-axis and z is the error ratio at every point considered in the space such that:

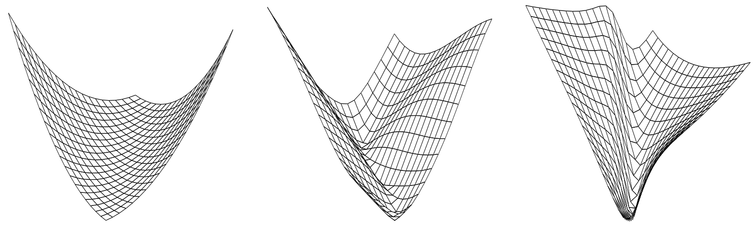
$$z(x, y) = \frac{1}{\gamma}((A(x) - x)^2 + (A(y) - y)^2) \qquad (1)$$

In the above formula, x and y represent the two dimensions of each data point, $A()$ is the function realized by the trained autoassociators, and γ is a modeldependent scale factor used for plotting purposes. The plots of Figures 3(a) and (b) are particularly helpful for understanding the nature of the solutions computed by the nonlinear autoassociators and contrasting them to the solution obtained by the linear autoassociator and PCA. In particular, they show that these error surfaces are qualitatively different from the ones computed by PCA and the linear autoassociator since, while the nonlinear autoassociators build multiple-local-valley representations of the underlying domain (Figures 3(a) and (b)), PCA and the linear autoassociator learn how to reconstruct the domain perfectly and exhibit a flat reconstruction error surface. This suggests that the solutions computed by autoassociators with nonlinearities in their hidden layer use a *multi-modal interpolation bias* which is not used by the linear methods.[9] Our results of Figure 3 can be used to explain the nonlinear autoassociators (NL-L and NL-NL) results of Figure 2 which show that nonlinear autoassocia-

[9] As expected from [Baldi and Hornik1989], our results thus show that the linear autoassociator is capable of emulating PCA. Note, however, that, as mentioned in the text, in order for the two paradigms to be equivalent, the linear autoassociator had to be trained with a learning rate of 0.1 (instead of the learning rate of 0.05 used by the nonlinear systems). In the case where a learning rate of 0.05 was used, the linear autoassociator got stuck in a saddle point (see [Baldi and Hornik1989]) and returned a constant output corresponding to the mean of the four training clusters. This observation is interesting in its own right because it underlines the difference between linear and nonlinear autoassociators even prior to the linear autoassociator's full convergence. Indeed, it suggests that the linear autoassociator tackles the reconstruction problem *globally*, using a *uni-modal* bias, whereas the nonlinear autoassociators use *local* strategies characterized by their *multi-modal* interpolation biases.

tors are capable of classifying the nonlinearly separable and multi-modal domain of Figure 1(c). Indeed, since the reconstruction error surfaces of the nonlinear autoassociators include several local valleys centered at each of the positive clusters (Figures 3(a) and (b)), the error ratios of the counter-conceptual data can all be appropriately high relative to their conceptual counterparts whether they are located in the interior, the sides, or the exterior of the square underlying the conceptual components. Thus all the counter-conceptual data get appropriately differentiated with respect to the conceptual data. The classification results obtained by PCA and linear autoassociation in Figure 2 will be explained in the next paragraph since they were obtained with bottleneck devices.

Compression Results The results obtained in the previous set of experiments demonstrate that nonlinear autoassociators are not equivalent to linear autoassociators or PCA in the case where the number of hidden units exceeds the number of input units, or when the number of principal components is equal to the domain dimensionality. Autoassociators, however, are most typically used as compression devices with a number of hidden units smaller than the number of input units. We now discuss whether the same result also holds in this case. Figures 4(a), (b) and (c) display the reconstruction error surfaces obtained by PCA or L-L (Figure 4(a)), NL-L (Figure 4(b)) and NL-NL (Figure 4(c)) using a single principal component or hidden unit. These reconstruction errors were also obtained using equation (1). The reconstruction error surfaces obtained by these devices show that, first of all, none of them are appropriate for full classification of the test domain of Figure 1(c) since they can only classify all (or most of) the data (whether conceptual or counter-conceptual) in one diagonal of the input space as conceptual and all (or most of) the other data (whether conceptual or counter-conceptual, again) as counter-conceptual. In other words,



(a) **(b)** **(c)**

Figure 4: Reconstruction errors of PCA or L-L (with one principal component or one hidden unit), NL-L, and NL-NL (with one hidden unit) on the artificial domain generated from Figure 1(c).

because they do not interpolate the training set fully, these devices are not useful for classification and this explains the high classification error rate obtained by the linear systems in Figure 2.[10] Nevertheless, these results are interesting since, like in the non-bottleneck case, they do illustrate the difference between the linear and the nonlinear schemes. Indeed, while the linear systems display an undistorted surface with no saddle points (Figure 4(a)), the nonlinear ones have saddle points (Figures 4(b) and (c)). Thus, the compression results demonstrate that nonlinear autoassociation is not equivalent to linear autoassociation or PCA even in the case where the number of hidden units or principal components is

[10]Note that while for the test domain used in this study, classification can be achieved only if using a number of hidden units greater than the number of input units, in the work of [Japkowicz *et al.*1995] and [Petsche *et al.*1996], the number of hidden units is smaller than the number of input units. This suggests that the hidden-to-input unit ratio does not have any bearing on the behavior of nonlinear autoassociators in classification tasks, since what really matters is their capacity to interpolate the training set regardless of how many hidden units that may take.

smaller than the number of input units.

3 Discussion

We will now analyze carefully the results we obtained in the previous section and explain why our observations depart from the expectations derived from the past literature. In addition, we will compare the results of the nonlinear autoassociators to the result of another well-known paradigm, radial-basis functions, and speculate on what nonlinear autoassociators compute and why they do so.

3.1 Nonlinear Autoassociation versus PCA and Linear Autoassociation

The difference between the linear and nonlinear systems can be explained by the fact that, for the particular test domain considered[11], [Bourlard and Kamp1988]'s assumption has been violated. Indeed, as mentioned in section 1, [Bourlard and Kamp1988]'s discussion was shown to hold only in the case where the inputs to the nonlinear activation function of the hidden units remain in the linear range of the squashing function. An observation of the pre-synaptic activations of the 16 hidden units of NL-NL reveals that, on average, these values are equal to $\mu = 0.6953$ with variance $\sigma^2 = 0.1919$. Similarly, for NL-L, we found that $\mu = .3222$ and $\sigma^2 = .0548$. Although for input values in the [-1,1] interval, the sigmoid function is close to linear, our results seem to suggest that a small violation of the [Bourlard and Kamp1988] assumption can make a big difference in certain contexts: while this difference might be marginal and thus overlooked when considering the hidden layer alone, it is magnified in the output layer, as suggested by the plots of Figures 3(a) and (b) which do not display a

[11] For other classes of domains, this might not be the case.

flat surface of the sort computed by PCA or autoassociation. This remark also holds for the encoder problem of [Rumelhart *et al.*1986] which can be solved by an autoassociator with nonlinear hidden units but not by a linear one: in this problem, nonlinearities in the hidden layer are also shown to have an impact.

3.2 What do Nonlinear Autoassociators Compute? —A Comparison with RBF Autoassociators—

We will now attempt to establish the nature of the computations taking place in the nonlinear autoassociators by comparing their reconstruction error surfaces to the reconstruction error surface obtained by a radial basis function (RBF) autoassociator with Gaussian activations in the hidden units and linear output activations. The reconstruction error surface obtained by such a device on the domain of Figure 1(c) is plotted in Figure 5. The particular RBF Network that yielded this figure has a capacity of 4 hidden units and the variance of the Gaussian functions are fixed at and set to 2.8. The reconstruction error surface of this

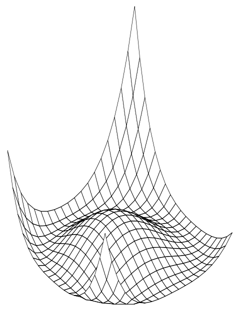


Figure 5: Reconstruction errors of the RBF autoassociator on the artificial domain generated from Figure 1(c). The network has a capacity of 4 hidden units and the variance of its Gaussian functions is fixed and set at 2.8.

autoassociator was computed in the same fashion as that of the other autoassociators, using equation (1). The similarity between the reconstruction obtained by the non-bottleneck, nonlinear autoassociators (especially NL-L, which also has linear output activations) and the RBF autoassociator suggests that, like the RBF network, the nonlinear autoassociators use their 16 hidden units to put centers down over the four clusters representing the training data, and evaluate the distance of the testing data to these centers.[12]

This result together with the results of [Bourlard and Kamp1988] suggests that nonlinear autoassociators can resort to one of two modes of operation: either they operate like PCA, seeking a globalized solution to the reconstruction problem they are trying to solve or they operate like an RBF autoassociator, seeking a localized solution.

3.3 On the Duality of Nonlinear Autoassociators: Our Speculation

The question we now ask is: when does a sigmoidal autoassociator resort to one or another of the two modes of action described in the previous section? Because of the current lack of technology available for analyzing feedforward neural networks fully, we can only speculate as to when each phenomenon will take place.

In particular, we suggest that the sigmoidal autoassociator will make use of its nonlinearities when the data set it attempts to reconstruct is multi-modal and the different clusters that compose it are very spread out.[13] Indeed, if the data set is unimodal or if it is multi-modal but can easily be confused for unimodal data, then every point in the training set activates hidden unit values located in the same vicinity and which increase or decrease monotonically. Such data

[12]We would like to thank Gary Cottrell for suggesting this comparison.

[13]Support for this speculation can be found in [Japkowicz1999] which shows a clear decrease in classification accuracy by NL-NL as the conceptual clusters of the domain in Figure 1(c) are moved closer to one another.

can, thus, be approximated linearly, in the same fashion as they would be by a linear autoassociator. On the other hand, if the data set is multi-modal and if the different clusters constituting it are very spread out, then approximation of their hidden unit activation values by a linear function will fail because of the discontinuities introduced by the multi-modal nature of the domain. In such cases, the sigmoidal part of the activation function is very useful since it permits the local processing of separate clusters in the same fashion as RBF autoassociators. Indeed, by using a sigmoidal function, a hidden unit can map all the data contained in different clusters to approximately the same default activation value (of 0 or 1) while it can map the data in some other clusters—which fall in the linear part of the sigmoidal function—to more "interesting" values. Each hidden unit can then use the same strategy for different clusters and, thus, a global solution can be found by the overall network by compounding the localized solution computed by each hidden unit.

Since, in the encoder problem, the autoassociator is expected to reconstruct data points located far away from each other—since they are located at extreme "corners" of the input space—this speculation also explains why the nonlinear autoassociator resorts to a localized solution not available to PCA in that case.

4 Summary

The purpose of this paper was to address the paradox raised by the claim of [Bourlard and Kamp1988] that, in autoassociation, nonlinearities in the hidden units are completely unnecessary, given that nonlinear autoassociators are known to perform tasks that cannot be solved by PCA or linear autoassociation. The paper begins by demonstrating that, although both PCA (or linear autoassociation) and nonlinear sigmoidal autoassociation can be used for classification, the linear systems are not appropriate for certain types of multi-modal and non-linear domains whereas the nonlinear systems are capable of classifying such domains accurately. An explanation of this result is provided by the observation of the interpolation surfaces constructed by the linear and nonlinear systems when trained on the conceptual examples of our classification test domain.

The difference in the interpolation and, as a result, in the classification scheme used by the linear and nonlinear devices thus contradicts the claim by [Bourlard and Kamp1988] initially considered. This contradiction is resolved by extracting the assumption on which the claim is based and showing that this assumption is not necessarily always verified.

In a last step, we note that when the nonlinear autoassociator does not operate in a linear fashion, it emulates a radial basis function autoassociator and thus computes a localized solution to the problem it attempts to solve. Speculations as to when nonlinear autoassociators do resort to a linear operative mode and when they do not do so are then formulated and conclude the paper.

Acknowledgments

We would like to express our gratitude to Gary Cottrell and an anonymous reviewer for their extremely helpful comments. We would also like to thank Inna Stainvas for useful discussions about autoassociators and for her careful reading of a previous version of this manuscript. This research was supported by the Office of Naval Research through grant N0014-88-K-0112 to Mark A. Gluck. We are also grateful to the School of Mathematical Science of Tel-Aviv University for its financial support and the Cognitive Science Laboratory of Princeton University for the use of their computing facilities.

References

[Baldi and Hornik1989] Pierre Baldi and Kurt Hornik. Neural networks and principal component analysis: Learning from examples without local minima. *Neural Networks*, 2:53–58, 1989.

[Bourlard and Kamp1988] H. Bourlard and Y. Kamp. Auto-association by multilayer perceptrons and singular value decomposition. *Biological Cybernetics*, 59:291–294, 1988.

[Cottrell and Munro1988] Garrison W. Cottrell and Paul Munro. Principal components analysis of images via back propagation. In *Invited Paper in Proceedings of the Society of Photo-Optical Instrumentation Engineers*, Cambridge, MA, 1988.

[Hanson and Kegl1987] Stephen J. Hanson and Judy Kegl. Parsnip: A connectionist network that learns natural language grammar from exposure to natural language sentences. In *Proceedings of the Ninth Annual Conference on Cognitive Science*, 1987.

[Japkowicz *et al.*1995] Nathalie Japkowicz, Catherine Myers, and Mark Gluck. A novelty detection approach to classification. In *Proceedings of the Fourteenth Joint Conference on Artificial Intelligence*, pages 518–523, 1995.

[Japkowicz1999] Nathalie Japkowicz. *Concept-Learning in the Absence of Counter-Examples: An Autoassociation-Based Approach to Classification.* PhD thesis, Rutgers University, October 1999.

[Kruglyak1990] Leonid Kruglyak. How to solve the n bit encoder problem with just two hidden units. *Neural Computation*, 2:399–401, 1990.

[Oja1983] Erkki Oja. *Subspace Methods of Pattern Recognition.* John Wiley & Sons, 1983.

[Petsche *et al.*1996] Tom Petsche, Angelo Marcantonio, Christian Darken, Steve J. Hanson, Gary M. Kuhn, and Iwan Santoso. A neural network autoassociator for induction motor failure prediction. In *Advances in Neural Information Processing Systems 8.*, MIT, Cambridge, MA, 1996. MIT Press.

[Phatak *et al.*1993] D.S. Phatak, H. Choi, and I. Koren. Construction of minimal n-2-n encoders for any n. *Neural Computation*, 5:783–794, 1993.

[Rumelhart *et al.*1986] David E. Rumelhart, Geoff E. Hinton, and R. J. Williams. Learning internal representations by error propagation. In David E. Rumelhart and J. L. McClelland, editors, *Parallel Distributed Processing*, pages 318–364. MIT Press, Cambridge, MA, 1986.

[Zipser1989] David Zipser. Programming neural nets to do spatial computations. In Noel E. Sharkey, editor, *Models of Cognition: A Review of Cognitive Science*, chapter 10. Ablex, Norwood, 1989.