PERGAMON

Neural Networks 15 (2002) 459–469

Neural Networks

www.elsevier.com/locate/neunet

Contributed article

AANN: an alternative to GMM for pattern recognition

B. Yegnanarayana*, S.P. Kishore

Department of Computer Science and Engineering, Indian Institute of Technology Madras, Chennai 600 036, Tamil Nadu, India

Received 29 September 2001; revised 4 January 2002; accepted 4 January 2002

Abstract

The objective in any pattern recognition problem is to capture the characteristics common to each class from feature vectors of the training data. While Gaussian mixture models appear to be general enough to characterize the distribution of the given data, the model is constrained by the fact that the shape of the components of the distribution is assumed to be Gaussian, and the number of mixtures are fixed a priori. In this context, we investigate the potential of non-linear models such as autoassociative neural network (AANN) models, which perform identity mapping of the input space. We show that the *training error surface realized by the neural network model in the feature space* is useful to study the characteristics of the distribution of the input data. We also propose a method of obtaining an error surface to match the distribution of the given data. The distribution capturing ability of AANN models is illustrated in the context of speaker verification. © 2002 Published by Elsevier Science Ltd.

Keywords: Autoassociative neural network models; Training error surface; Annealing gain parameter; Speaker verification

1. Introduction

In a pattern classification problem, given the training samples in the form of feature vectors for each class, the objective is to capture the characteristics of the vectors unique to each class by a classification model. In the context of speech, speaker recognition, speech recognition and language identification are some of the problems, which involve development of pattern classification models from speech data. The models are typically intended to capture the characteristic features of each class from the given training data. The more constrained the model is, the more will be the deviation of the training data from the data predicted by the realized model. Within the constraints imposed by the model, one can interpret the surface formed by the error with the training data as characterizing the distribution of the data, and hence that of the class.

Traditionally Gaussian mixture models (GMMs) are used to represent the complex surface characterizing the distribution of the data (Redner & Walker, 1984). The simplest case of GMM is vector quantization (VQ), where the data distribution is represented by the set of codewords corresponding to the cluster centers (Gray, 1984). In the GMM, the distribution of the data points around each codeword is represented by a multivariate Gaussian distribution, and the relative weights of each of these Gaussian functions determine the overall surface characterizing the distribution. While the GMM appears to be general enough to characterize the distribution of the given data, the model is constrained by the fact that the shape of the components of the distribution is assumed to be Gaussian, and that the number of mixtures are generally fixed a priori. Parameters of the GMM are determined using an iterative algorithm like estimation maximization (EM) (Dempster, Laird, & Rubin, 1977).

Most practical data encountered in speech and other application areas has complex distribution in the feature space, and hence cannot adequately be described by a GMM, which uses only the first and second order statistics and mixture weights. The surface representing the distribution may be highly non-linear in the multidimensional feature space, and hence it is difficult to model the surface by simplified models. Fig. 1 shows 2D data with different levels of complexity of distribution. While the distribution of data in Fig. 1(a) can be represented by clustering using VQ, and the distribution of data in Fig. 1(b) by GMM, it is necessary to explore other methods to represent the distributions such as in Fig. 1(c). In this context, it is worthwhile to investigate the potential of non-linear models such as artificial neural networks (ANNs) (Yegnanarayana, 1999).

ANN models consist of interconnected processing units,

* Corresponding author. Tel.: +91-44-2354591/4458338; fax: +91-44-4458352.

E-mail addresses: yegna@cs.iitm.ernet.in (B. Yegnanarayana), kishore@speech.cs.iitm.ernet.in (S.P. Kishore).

Nomenclature

λ	gain parameter of the activation function
E_i	error for the data point i
e(·)	exponential function
f_i	represents exponential weighted error for the data point i
α	temperature parameter

where each unit represents the model of an artificial neuron, and the interconnection between two units has a weight associated with it. ANN models with different topologies perform different pattern recognition tasks (Yegnanarayana, 1999). For example, a feedforward neural network can be designed to perform the task of pattern classification or pattern mapping, whereas a feedback network can be developed to perform the task of pattern storage or pattern environment storage (Yegnanarayana, 1999). One can interpret the design problem as *loading* the characteristics of the complex distribution of the data into the ANN model. In the case of a feedforward network, the weights are adjusted so as to realize the global minimum of the total error for the training data in the weight space. The weights are adjusted to reduce the error, and hence the training vectors with large error will influence the adjustment to a large extent. The objective in these supervised learning tasks is to achieve generalization from the training data, so that the classification or mapping error for test data is low. For generalization, the number of patterns should be much larger than the number of weights of the network. No attempt is made explicitly to capture the characteristics of the distribution of the data for each class, although there are attempts to interpret the results of learning in terms of the distribution of the data (Haykin, 1999; Lippmann, 1989).

If the probability distribution of the data is given in the form of pattern vectors and their probability of occurrence, then one can design a feedback network, like Boltzmann machine, to capture the pattern environment specified by the given data (Yegnanarayana, 1999). In this case, the number of patterns will have to be far less than the number of weights in the network. The weights are determined to realize an energy landscape of the network, so that the states with minimum energy correspond to the patterns with highest probabilities. This can be viewed as another supervised task, where the desired probabilities are specified in terms of the distribution of the states of the network. Concepts of simulated annealing and annealing schedule are used to realize the desired energy surface from the feedback network. Feedback networks are also used to search for the global minimum of the energy landscape to determine a set of values for the variables that optimize an objective function (Yegnanarayana, 1999). In this case, the energy surface is fixed by the weights, which in turn are determined by the constraints of the given optimization problem. Here the network is used merely as a constraint satisfaction (CS) model. There is no concept of data and distribution of data in these CS models.

It is useful to develop an ANN model which can capture the distribution of the given training data in the feature space for each class. Then the distribution surfaces may be used as signatures for the individual classes. Note that probabilistic neural networks (PNNs) and radial basis function neural networks (RBFNNs) are expected to capture the distribution of a given training data (Bishop, 1995; Wasserman, 1993). The PNN is an unconstrained model, and the RBFNN is a severely constrained model. Both may have limitations in capturing an arbitrary distribution of the training data. The question is whether it is possible to design a network to capture the characteristics of the distribution of the training data. In this context it is interesting to note that not much importance is given to the error surface of the training data in the feature space in the design of ANNs so far. The problem we would like to address here is, whether it is possible to design a training error surface in the feature space to match the given data, subjected to the constraints imposed by the network structure. In other words, is it possible to design a network that *scoops the error surface* in a desired manner? The desired manner implies that the surface relates to the distribution of the given data. A suitable architecture for this purpose is a feedforward network that performs an autoassociation task. These networks are called autoassociative neural networks (AANNs) (Ikbal, Misra, & Yegnanarayana, 1999; Yegnanarayana, Kishore, & Anjani, 2000). AANN models have also been used for several applications including dimension compression, non-linear principal component analysis, signal validation, numeral recognition,

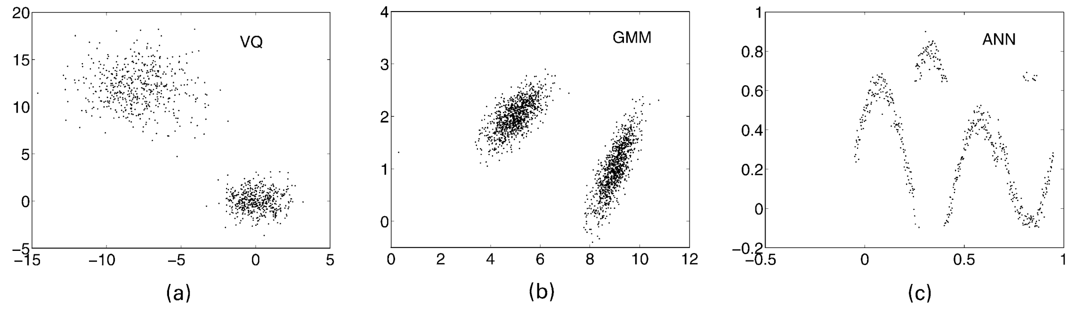

Fig. 1. Illustration of data with different levels of complexity of distribution.

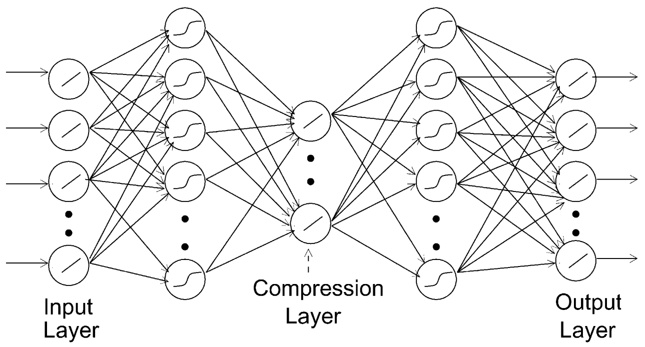

Fig. 2. An AANN model (only a few connections are shown for illustration).

etc. (DeMers & Cottrell, 1993; Gori, Lastrucci, & Soda, 1996; Hines, Uhrig, & Wrest, 1997; Kimura, Inoue, Wakabayashi, Tsuruoka, & Miyake, 1998; Kramer, 1991; Oja, 1991). In each case the nature of the data used as input and the interpretation of the results depend on the application for which AANN is used. The performance of the AANN for each of these applications are evaluated in the literature in comparison with non-neural approaches, especially signal processing and statistical methods. In this present case the AANN model, with a dimension compression layer in the middle, is used primarily for capturing the distribution of input vectors in the feature space.

This paper is organized as follows. In Section 2, we discuss the characteristics of AANN and their distribution capturing abilities. The design of an AANN for realizing the desired training error surface involves some form of annealing using the gain parameter of the activation function of a processing unit. This issue is discussed in Section 3. Interpretation of the training error surface in terms of probability distribution is described in Section 4. The results of this study in the context of speaker recognition are presented briefly in Section 5.

2. Autoassociative neural network models

An AANN is a feedforward network with the desired output being same as the input vector (Bishop, 1995). Therefore, the number of units in the input and output layers are equal. Fig. 2 shows the structure of a five layer AANN model with three hidden layers. The number of hidden layers and the number of units in each hidden layer depend on the problem. Typically, if the number of units in any hidden layer is less than the dimension of the input vector, then there will be a compression of the input vectors to a lower dimension, like in the principal component analysis (Diamantaras & Kung, 1996). The processing units in the first and third hidden layers are non-linear, and the units in the second hidden (compression) layer can be linear or non-linear (Kramer, 1991). As the error between the actual and desired output vectors is minimized, the clusters of points in the input space determine the shape of the hypersurface obtained by the projection onto the lower dimension space.

Fig. 3 shows the space spanned by the 1D compression layer for the input 2D data shown in Fig. 3(a) for two different network structures. The structures of the two networks are $2L4N1N4N2L$ and $2L10N1N10N2L$, where L denotes a linear unit and N denotes a non-linear unit. The non-linear output function for each unit is $\tanh(\lambda x)$, where λ is arbitrarily chosen to be equal to 0.66. The networks were

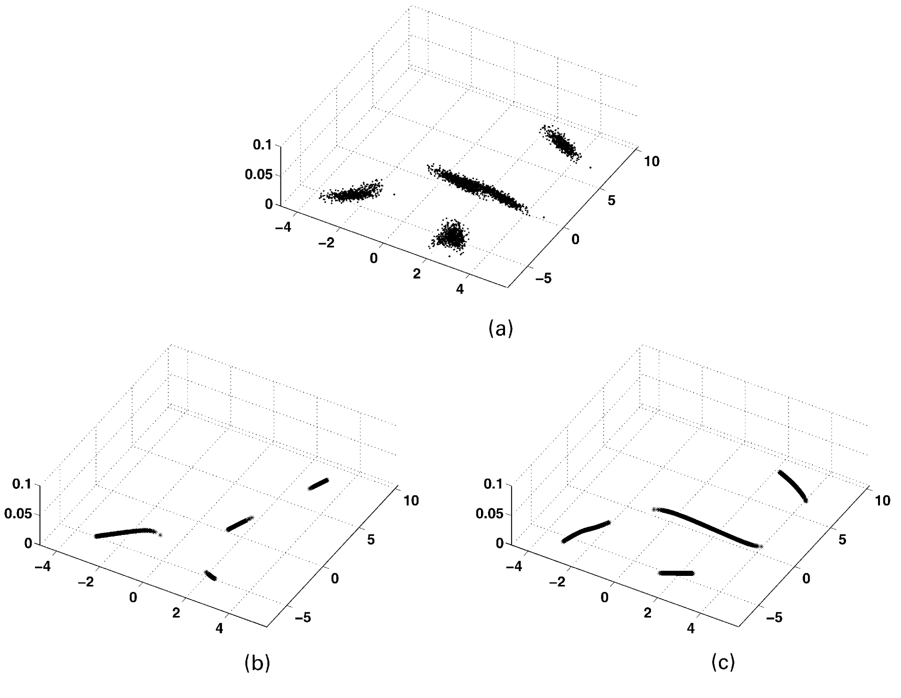

Fig. 3. (a) Artificial 2D data. (b) Output of AANN model with the structure $2L4N1N4N2L$. (c) Output of AANN model with the structure $2L10N1N10N2L$.

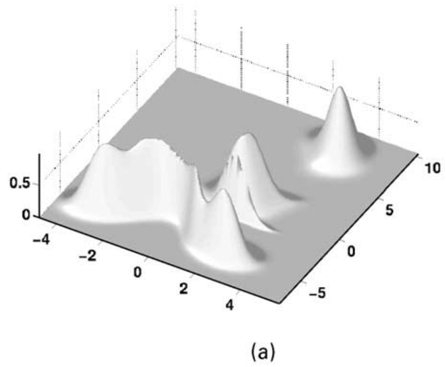
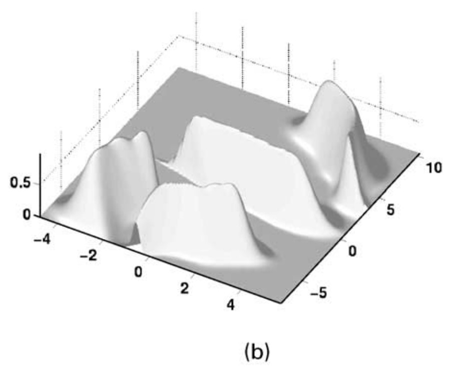

Fig. 4. Probability surfaces realized by two different network structures: (a) 2*L*4*N*1*N*4*N*2*L*; (b) 2*L*10*N*1*N*10*N*2*L* for the 2D data shown in Fig. 3(a).

trained using backpropagation algorithm (Haykin, 1999; Yegnanarayana, 1999). The solid lines shown in Fig. 3(b) and (c) indicates mapping of the given input points due to the 1D compression layer. The second network having more hidden units seem to represent the data through the 1D compression better, as shown in Fig. 3(c) compared to Fig. 3(b) for the network with fewer units in the hidden layers. Thus one can say that the AANN captures the distribution of the data points depending on the constraints imposed by the structure of the network, just as the number of mixtures and Gaussian functions do in the case of GMMs.

In order to visualize the distribution better, one can plot the training error for each input data point in the form of some probability surface as shown in Fig. 4 for the cases in Fig. 3(b) and (c). The training error E_i for data point (i) in the input space is plotted as $f_i = \mathrm{e}^{-E_i/\alpha}$, where α is a constant. Note that f_i is not strictly a probability density function, but we call the resulting surface as *probability surface*. The plot of the probability surface shows larger amplitude for smaller error E_i, indicating better match of the network for that data point. The justification for this plot is given in Section 4. The constraints imposed by the network can be seen by the shape the error surface takes in both the cases. One can use the probability surface to study the characteristics of the distribution of the input data captured by the network. Ideally, one would like to achieve the best probability surface, best defined in terms of some measure corresponding to low average error.

The aim is to obtain a set of weights of the network that gives the least average error for the given data within the constraints of the network. But the average error will be low if the error is small for the most frequently occurring input vectors. In order to achieve the best probability surface, one has to train the network in such a way that the network does not get stuck in the local minima of the average training error in the weight

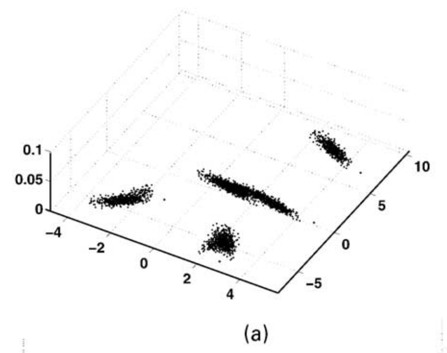
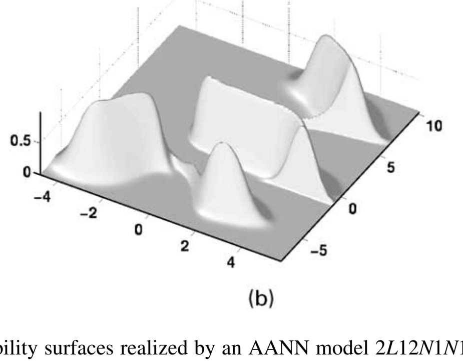
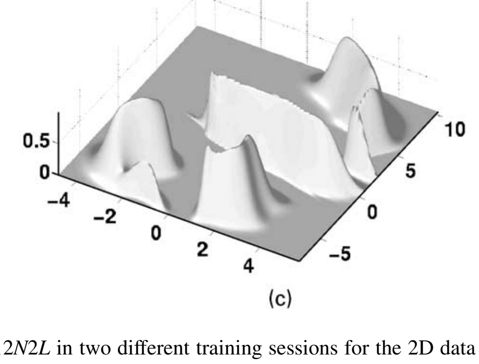

Fig. 5. Probability surfaces realized by an AANN model 2*L*12*N*1*N*12*N*2*L* in two different training sessions for the 2D data shown in (a).

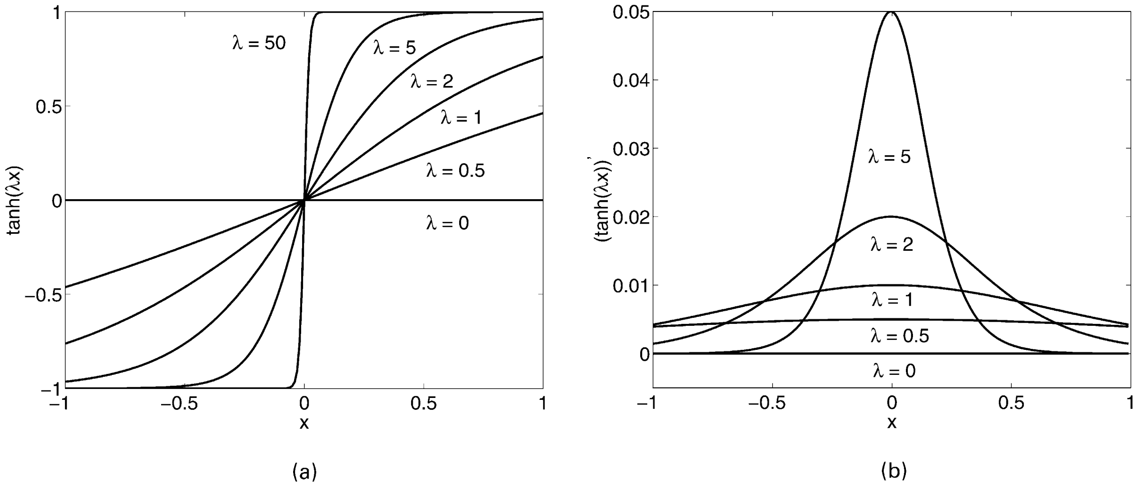

Fig. 6. For different values of gain parameter λ (a) activation function and (b) its derivative.

space. Fig. 5 shows the probability surfaces for two different set of weights with practically the same average error. It may possible to obtain better probability surface corresponding to a lower overall error. Note that during training, the weights are adjusted according to the backpropagation learning, which uses the derivative of the activation function, besides the error and the learning rate parameter. The value of the derivative depends on the gain parameter λ of the activation function $f(\lambda x)$. The activation function and its derivative are shown for different values of the gain parameter λ in Fig. 6. If a low value of λ is used, then the resulting probability surface may not represent the distribution well. If a large value of λ is used, the probability surface will reflect the distribution of the training error better, provided the weights do not get stuck at a local minimum of the total or average error. Fig. 7 shows the probability surfaces for two different λ's. In order to obtain the weights corresponding to the best probability surface, it is better to use annealing with the gain parameter λ during training, like simulated annealing in the Boltzmann machine (Yegnanarayana, 1999).

3. Annealing with gain parameter λ

The performance of the network for capturing the distribution of the data through the probability surface is best when the gain parameter λ is large, since there will be provision for the probability surface to show the dynamic range of the distribution. But, it is likely that the choice of large λ might result in a local minima problem in the weight space. The training error is unlikely to get stuck in a local minima when λ is small, since the error itself will be large. The behavior of the average error for different λ as a function of number of iterations is shown in Fig. 8. The error is usually large for small values of λ.

By using a low value of λ during the initial stages, and then increasing the λ in steps may help in realizing a better probability surface. This is like annealing in the Boltzmann machine. In order to implement the annealing with the gain parameter, one needs to have an annealing schedule, i.e. the change of λ at each step. For each λ, the weights are adjusted until some asymptotic error value is reached, after the transient part of the error dies down. The error for different iterations is as shown in Fig. 9, where the annealing schedule consists of five steps for $\lambda = 0.26$,

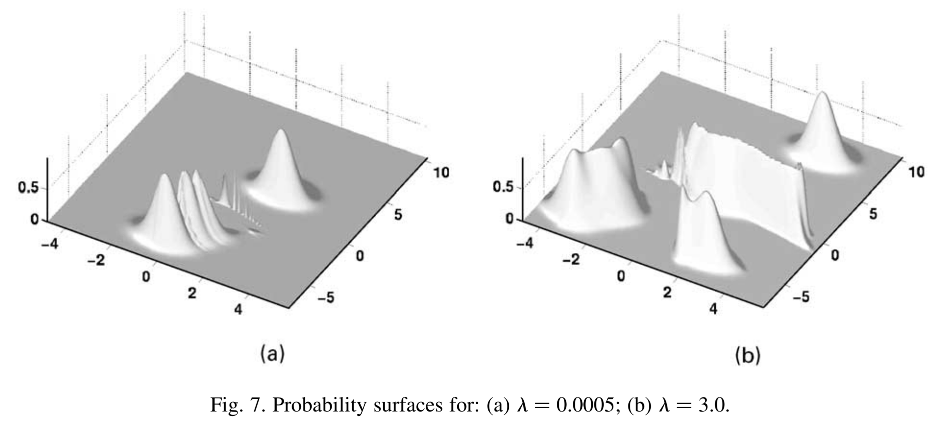

Fig. 7. Probability surfaces for: (a) $\lambda = 0.0005$; (b) $\lambda = 3.0$.

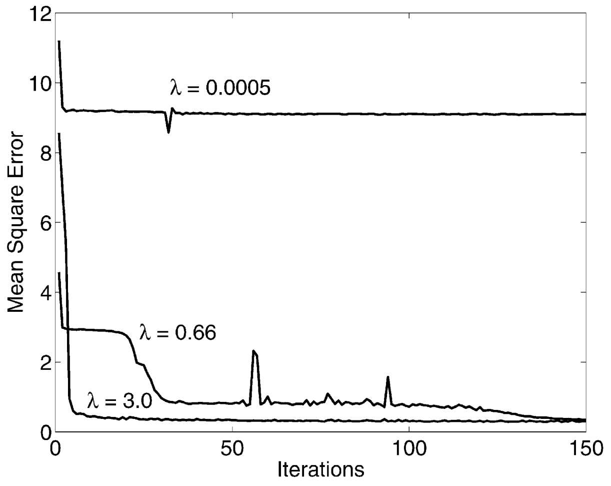

Fig. 8. Training error for different values of the gain parameter λ.

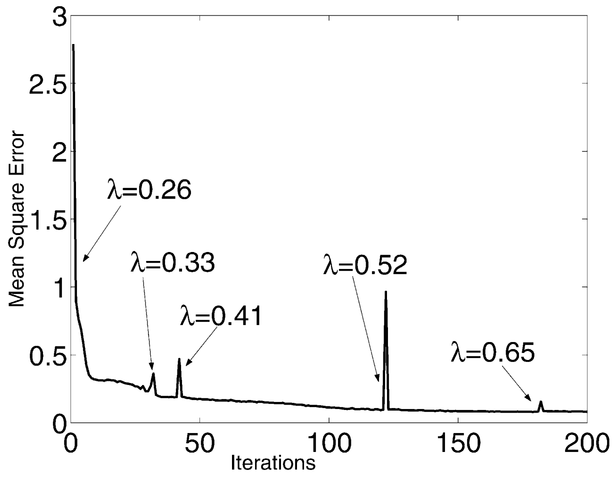

Fig. 9. Error as a function of the number of iterations for a given annealing schedule.

0.33, 0.41, 0.52, and 0.65. The peak regions correspond to the transient part.

Once a proper annealing schedule for λ is followed, the resulting probability surface will be nearly the same, irrespective of the initial weights. Optimization of the annealing schedule is not considered in this study. Fig. 10 shows the probability surface realized for two different trials on the data. Comparing with the surfaces in Fig. 5, these two surfaces are more similar, and hence reflect the distribution of data better.

4. Interpretation of optimal error surface

Fig. 11 gives the probability surfaces for two different distributions, showing the relation between the training error surface and the data distribution. Having obtained the error surface to match the distribution of the given input data, it is preferable to have a measure that gives directly an idea of the probability distribution. While it is clear that lower error should correspond to higher probability to achieve minimum average error, the nature of the relation between the error surface and the probability distri-

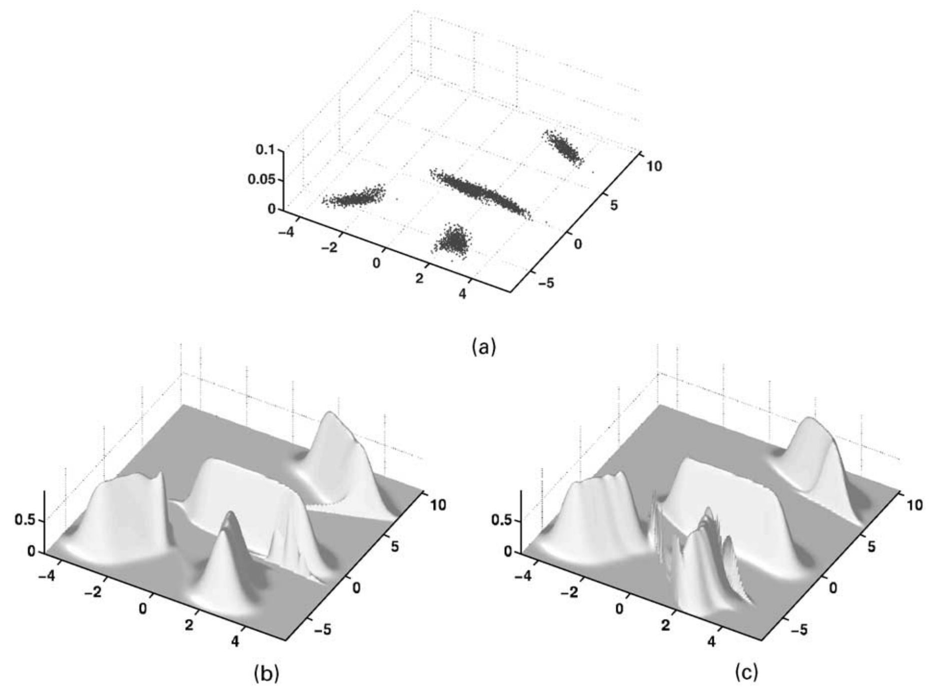

Fig. 10. (a) Artificial 2D data. (b) Probability surface realized by the AANN model for one training session. (c) Probability surface realized by the AANN model for another training session. In both the training sessions the gain parameter is annealed. The structure of the network is $2L12N1N12N2L$.

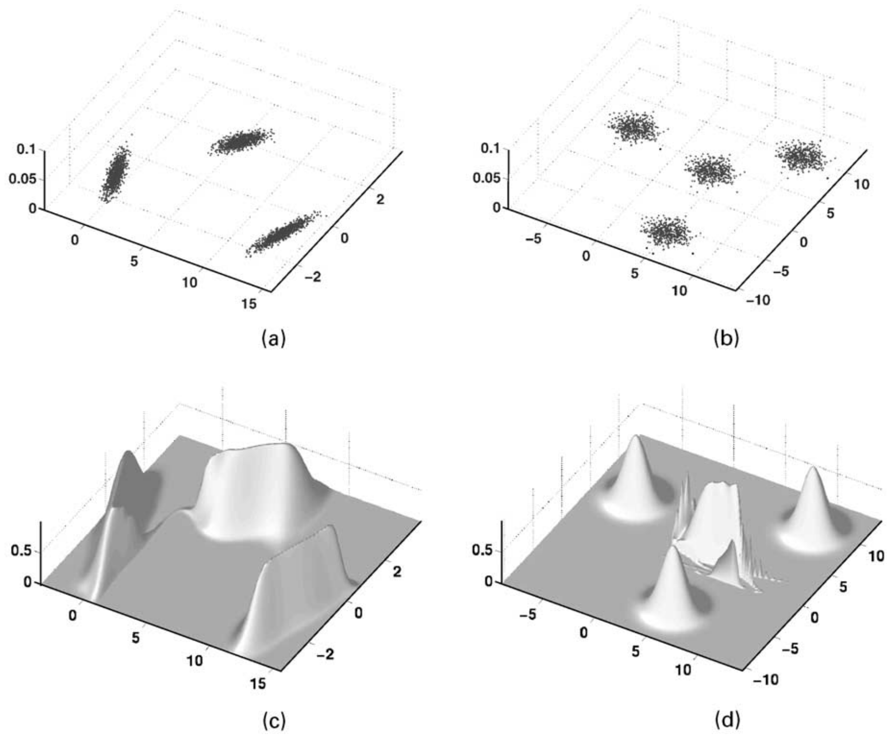
Fig. 11. Probability surfaces for two different distributions.

bution is not obvious, as in the case of Boltzmann machine, where the energy surface and the stationary probability distribution of states follow the Boltzmann–Gibb's law (Yegnanarayana, 1999). However, we will use a similar relation to plot the probability distribution and see if it reflects the distribution of the data. That is, we assume that the probability of a data point (i) is proportional to $e^{-E_i/\alpha}$, where α is like the temperature parameter in the Boltzmann–Gibb's law (Yegnanarayana, 1999). Fig. 12 shows the probability surface using the proposed relation

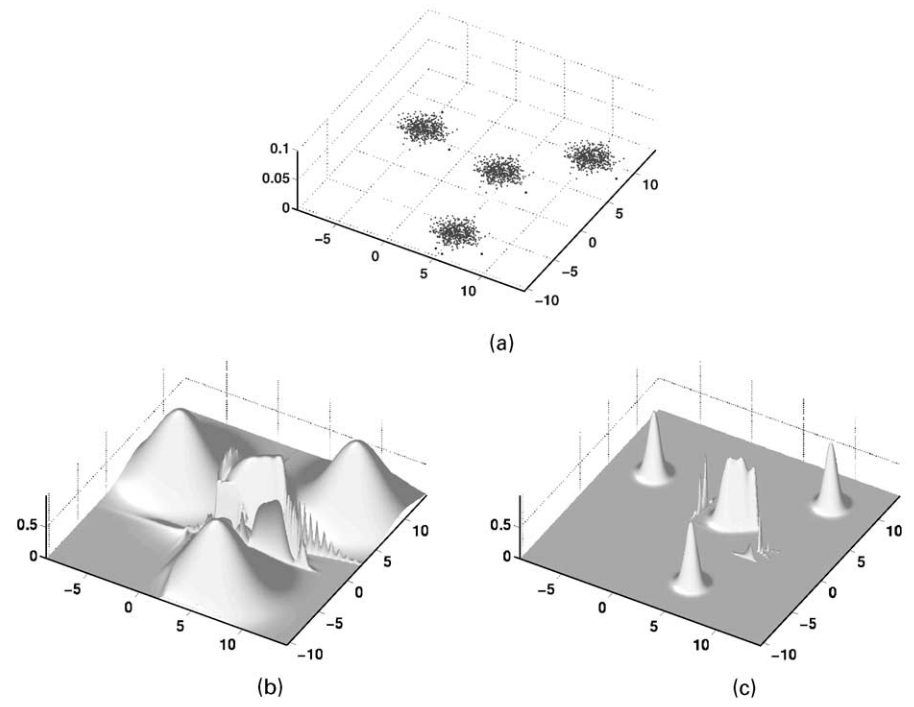
Fig. 12. (a) Artificial 2D data. (b) Probability surface for the data for the temperature parameter $\alpha = 10$. (c) Probability surface for $\alpha = 0.5$.

between error and probability. This distribution clearly shows lower probability for larger error, and vice versa. The shape of the probability surface plots also show that the constraints of the network, such as dimension reduction, forces a shape in the lower dimensional space. If the number of units in the dimension compression layer is changed, then the shape of the probability distribution also changes. Since it is difficult to visualize these distributions in a multidimensional data situation, the distribution plots can be observed in some reduced dimension space, although the effect of constraints cannot be seen clearly in the reduced dimension plot (Rumelhart, Smolensky, McClelland, & Hinton, 1986).

5. Application for speaker verification

AANN models can be derived using the feature vectors extracted from the speech data, one for each speaker. The model of each speaker captures the distribution of data for that speaker, which is expected to be unique for each speaker. That is, the training error surface or the corresponding probability surface is unique for each speaker. Typically the test feature vectors will be closer to the feature vectors near the error minima, and hence the probability is high. For an imposter, the feature vectors will have low probability with respect to this model. Here AANN is used primarily as feature extractor for speaker recognition task. The distribution of the input vectors in the feature space is used as feature representing each speaker, whereas, most of the other neural network models such as MLP and RBF are mainly used as discriminative classifiers. The implementation of text-independent speaker verification system using AANN models is explained in Sections 5.1–5.4.

5.1. Description of the speech database

Speech corpus used in this study consists of SWITCHBOARD-2 database of National Institute of Standards and Technology (NIST). These databases are used for the NIST-99 official *text-independent* speaker recognition evaluation (NIST, 1999). The phase-2 SWITCHBOARD-2 database is used for background modeling (described in the following Sections 5.2–5.4), and hence referred to as *development data*. Performance of the speaker verification system is evaluated on the phase-3 database, which is referred to as *evaluation data*. The development data consists of 500 speakers (250 male and 250 female), and the evaluation data consists of 539 speakers (230 male and 309 female). Data provided for each speaker is conversational telephone speech sampled at 8000 samples/s. The training data consists of 2 min of speech data, collected from the same phone number. The use of the same phone number results in passing the speech data over the same handset and communication channel. Two different types of microphones (also referred as handsets) are used for collecting the speech data. They are carbon-button and electret.

All the studies reported in this paper are performed on the male subset of 230 speakers with 1448 male test utterances of the evaluation data. Each test utterance has 11 claimants (of the same gender), where the genuine speaker may or may not be one of the claimants. Thus, 1311 genuine tests and 14 617 impostor tests are performed to evaluate the performance of the speaker verification system. The duration of each test utterance varies between 3 and 60 s. Performance of the speaker verification system is studied for the following three conditions:

(a) *Matched condition*. The training and testing data are collected from the same phone number.
(b) *Channel mismatch condition*. The training and testing data are collected from different phone numbers, but it is ensured that the same handset type is used in both the cases. The use of different phone numbers results in passing the speech signal over different communication channels.
(c) *Handset mismatch condition*. The training and testing data are collected with different handset types.

5.2. Generation of speaker models

Speech signal provided for each speaker is pre-emphasized using a difference operator. The differenced speech signal is segmented into frames of 27.5 ms using a Hamming window with a shift of 13.75 ms. The silence frames are removed using an amplitude threshold (Kishore, 2000). A 16th order linear prediction analysis is used to capture the properties of the signal spectrum (Makhoul, 1975). The recursive relation between the predictor coefficients and the cepstral coefficients is used to convert the 16 predictor coefficients into 19 cepstral coefficients (Furui, 1981). The cepstral coefficients obtained for each frame are linearly weighted to emphasize the peaks in the spectral envelope (Rabiner & Juang, 1993). Linear channel effects are compensated to some extent by removing the mean of the time trajectory of each cepstral coefficient (Atal, 1974; Furui, 1981).

Each speaker model is built by training an AANN model with the feature vectors extracted from the utterance of the speaker in the evaluation data. The structure of the AANN model is $19L38N4N38N19L$, where L refers to a linear unit and N to a non-linear unit. The integer value indicates the number of units in that particular layer. The number of units in layers 2 and 4 is chosen empirically to be double that of the input dimension. The use of four units in the dimension compression layer is based on the systematic study conducted by Kishore (2000). The network is trained using backpropagation learning algorithm in pattern mode.

5.3. Verification procedure

During testing phase the feature vectors extracted from the test utterance (as described in Section 5.2) are given to the claimant model to obtain the claimant score. The score

Table 1
Performance comparison of speaker verification system for different values of α measured in terms of EER

Environment between training and testing	EER (%)				
	$\alpha = 0.01$	$\alpha = 0.05$	$\alpha = 0.1$	$\alpha = 0.2$	$\alpha = 5$
Matched	19.73	8.74	6.73	6.40	6.91
Channel mismatch	29.65	17.31	15.32	15.26	15.33
Handset mismatch	33.97	21.89	20.51	19.72	21.54

of a model is defined as

$$\frac{1}{\ell}\sum_{i=1}^{\ell} \mathrm{e}^{-D_i/\alpha}, \qquad D_i = \frac{\|\boldsymbol{x}_i - \boldsymbol{y}_i\|^2}{\|\boldsymbol{x}_i\|^2},$$

where $\boldsymbol{x}_i$ is the input vector given to the model, $\boldsymbol{y}_i$, the output given by the model, α, the temperature parameter and ℓ is the number of feature vectors of the test utterance. Since, the score of the claimant model is sensitive to the linguistic content of the utterance and to the variations of the speaker's voice, normalization of background model is used (Reynolds, 1997). Two approaches to represent the background model for AANN-based speaker verification system are compared by Kishore and Yegnanarayana (2000). The two approaches are universal background model and individual background model (IBM). The approach of IBM is used for AANN-based speaker verification system because of its simplicity in choosing the parameters and effectiveness in performance (Kishore, 2000; Kishore & Yegnanarayana, 2000).

The approach of IBM involves generation of pseudo-claimant models (also called as IBMs). To generate pseudo-claimant models, a set of 92 speakers is selected from the *development data*. Each pseudo-claimant model is derived by training an AANN with the feature vectors extracted from the utterance of a speaker in the chosen set. This subset of 92 speakers belong to male set of the development data. The utterances of 46 speakers are from an electret handset, and the utterances of remaining 46 speakers are from a carbon-button handset. These speakers are picked arbitrarily without any selection criteria such as cohorts (Finan, Sapeluk, & Damper, 1997). In the testing phase, the feature vectors of the test utterance are given to the claimant model and to the set of 46 IBMs whose utterances are from the same handset type as that of the claimant (Kishore, 2000). The position (R) of the claimant model score in the descending list of the scores from IBMs is used for accepting/rejecting the speaker.

Before obtaining the value of R, the scores of the claimant and pseudo-claimant models are normalized with the score obtained by their respective models for a set of impostor data (Kishore, 2000). If S_ξ denotes the score obtained by a speaker model ξ for a given test utterance, then the normalized score $N_\xi = S_\xi/\bar{I}_\xi$, where $\bar{I}_\xi$ is the mean of the scores obtained by the same model ξ for a set of impostor data. The normalized score indicates the closeness of S_ξ to $\bar{I}_\xi$. In mismatch condition, the value of S_ξ may be large enough to reject the genuine speaker. But the value of S_ξ in these cases may not be too close to $\bar{I}_\xi$ obtained by the same model, and hence the value of N_ξ may be a better measure to accept or reject the claim. Using a set of 25 speakers' utterances of NIST-97 database (the duration of each utterance is half-minute), this normalization procedure is applied to the scores of the claimant and pseudo-claimant models. The set of 25 speakers data used for normalization is randomly selected from the NIST-97 database. The use of NIST-97 database ensures that these utterances do not belong to any one of the claimant or pseudo-claimant models.

Table 2
Performance of one of the GMM-based speaker verification systems reported by NIST (1999)

Environment between training and testing	EER (%)
Matched	5.61
Channel mismatch	10.00
Handset mismatch	21.00

5.4. Results

False acceptance (FA) and false rejection (FR) are the two errors that are used in evaluating a speaker verification system. The tradeoff between FA and FR is a function of the decision threshold. Equal error rate (EER) is the value for which the error rates of FA and FR are equal (Oglesby, 1995). Performance of the AANN-based speaker verification system measured in terms of EER for 230 speakers of NIST-99 data is shown in Table 1. It can be observed that performance of the speaker verification system degrades for the mismatch conditions due to the channel and handset effects.

When there is noise or channel effects, then there will be a shift of the probability surface for the test data from the distribution of the training data. Due to deep minima in the error surfaces, there will be differences in the probability surfaces for training and test data, and hence the probability is low even for slight changes in the feature vectors. In such a case, the temperature parameter α can be increased to make the probability surface flatter. Performance of the AANN-based speaker verification system for different values of α is shown in Table 1. Improvement in the performance of the speaker verification system can be seen when α is large. But the discrimination among speakers is reduced due to flattening of the probability surfaces and hence there is an increase in EER for $\alpha = 5$. The EER results obtained from the standard GMM are given in Table 2 (NIST, 1999). The results of AANN are comparable with GMM, although the performance of GMM is slightly better. We feel that this

Table 3
Performance of the AANN-based speaker verification system with the gain parameter λ being annealed in the training phase

Environment between training and testing	EER (%)
Matched	6.92
Channel mismatch	14.43
Handset mismatch	20.54

is primarily in the way the fine tuning is done at the scoring level in the GMM case (NIST, 1999, 2000). The fine-tuning in done at fixing the threshold, compensating the channel and handset effects, etc. The objective of the present study is to show that AANN models can be used for capturing the distribution of the feature vectors for speaker verification. Using AANN models, the number of parameters can be as few as 1847 (weights + bias of the network) to capture the distribution of the feature vectors of a speaker, as opposed to 9728 parameters (256 mixture components using diagonal covariance matrix of 38D feature vectors) used by the GMM-based approach.

We also studied the effect of annealing the gain parameter on the performance of the AANN-based speaker verification system. The annealing schedule consists of three steps for $\lambda = 0.2$, 0.5, and 0.8 at the epoch intervals of 1, 10, and 20, respectively. The performance of the AANN-based speaker verification system for annealing the gain parameter is shown in Table 3. The value of temperature parameter $\alpha = 0.2$ is used in the verification procedure. From Tables 1 and 3, we can observe that there is slight improvement in the performance of the speaker verification system for channel mismatch condition. However, further studies need to be conducted to exploit the full potential of the annealing approach in the case of speaker verification.

6. Conclusion

We have proposed a method to exploit the training error surface to derive the characteristics of the distribution of the given data. This will enable us to design an AANN model for each class which captures the distribution of data for that class. The gain parameter may be effectively used to derive a network that yields the best probability surface, best defined in terms of average error. If there are fewer training samples, then the probability surface may have to be derived for the compressed data (dimension reduction layer).

The concept of annealing the gain parameter is described, which enables the learning process to avoid the local minima problem. The significance of the annealing schedule is to be explored for different problems. Also, faster annealing procedures need to be developed along the lines of mean field annealing in Boltzmann machine (Haykin, 1999; Yegnanarayana, 1999). The error surface can be interpreted in terms of probability distribution by transforming the surface suitably. Here the idea is similar to the Boltzmann–Gibb's law. That is, the probability is proportional to $e^{-E/\alpha}$. This helps us to derive a probability distribution which represents the distribution of the data, subject to the constraints imposed by the network in terms of number of units and the gain parameter. The probability distribution derived from the error surfaces has been used for speaker verification studies. The dependence of the probability surface on the temperature parameter α needs to be exploited to compensate for the mismatched channel conditions between the test and training data.

References

Atal, B. S. (1974). Effectiveness of linear prediction characteristics of the speech wave for automatic speaker identification and verification. *Journal of Acoustical Society of America*, *55*, 1304–1312.

Bishop, C. M. (1995). *Neural networks for pattern recognition*, New York: Oxford University Press.

DeMers, D., & Cottrell, G. (1993C). Nonlinear dimensionality reduction. In S. Hanson, J. Cowan & C. Giles, *Advances in neural information processing systems 5* (pp. 580–587). California: Morgan Kaufmann.

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of Royal Statistical Society Series B (Methodological)*, *39*, 1–38.

Diamantaras, K. I., & Kung, S. Y. (1996). *Principal component neural networks: Theory and applications*, New York: Wiley.

Finan, R. A., Sapeluk, A. T., & Damper, R. I. (1997). Imposter cohort selection for score normalization in speaker verification. *Pattern Recognition Letters*, *18*, 881–888.

Furui, S. (1981). Cepstral analysis technique for automatic speaker verification. *IEEE Transactions on Acoustics, Speech and Signal Processing*, *29* (2), 254–272.

Gori, M., Lastrucci, L., & Soda, G. (1996). Autoassociator-based models for speaker verification. *Pattern Recognition Letters*, *17*, 241–250.

Gray, R. M. (1984). Vector quantization. *IEEE Acoustics, Speech and Signal Processing Magazine*, 4–29.

Haykin, S. (1999). *Neural networks: A comprehensive foundation*, New Jersey: Prentice-Hall.

Hines, J. W., Uhrig, R. E., & Wrest, D. J. (1997). Use of autoassociative neural networks for signal validation. *Proceedings of NEURAP 97 Neural Network Applications.*

Ikbal, M. S., Misra, H., & Yegnanarayana, B. (1999). Analysis of autoassociative mapping neural networks. *Proceedings of the International Joint Conference of Neural Networks*, Washington, DC.

Kimura, F., Inoue, S., Wakabayashi, T., Tsuruoka, S., & Miyake, Y. (1998). Handwritten numerical recognition using autoassociative neural networks. *Proceedings of the International Conference on Pattern Recognition.*

Kishore, S. P. (2000). *Speaker verification using autoassociative neural network models.* MS Dissertation, Department of Computer Science and Engineering, Indian Institute of Technology, Madras.

Kishore, S. P., & Yegnanarayana, B. (2000). Speaker verification: Minimizing the channel effect using autoassociative neural networks models. *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, Istanbul.

Kramer, M. A. (1991). Nonlinear principal component analysis using autoassociative neural networks. *AlChE Journal*, *37* (2), 233–243.

Lippmann, R. P. (1989). An introduction to computing with neural nets. *IEEE Acoustics, Speech and Signal Processing Magazine*, *4*, 4–22.

Makhoul, J. (1975). Linear prediction: A tutorial review. *Proceedings of IEEE*, *63* (4), 561–580.

NIST (1999). Speaker recognition workshop notebook. *Proceedings of NIST 1999 Speaker Recognition Workshop*. USA: University of Maryland.

NIST (2000). Speaker recognition workshop notebook. *Proceedings of NIST 2000 Speaker Recognition Workshop*. USA: University of Maryland.

Oglesby, J. (1995). What's in a number? Moving beyond the equal error rate. *Speech Communication*, *17*, 193–208.

Oja, E. (1991). Data compression, feature extraction and autoassociation in feedforward neural networks. In T. Kohonen, K. MaK kisara, O. Simula & J. Kangas, *Artificial neural networks* (pp. 737–745). Amsterdam: Elsevier.

Rabiner, L. R., & Juang, B. H. (1993). *Fundamentals of speech recognition*, New Jersey: Prentice-Hall.

Redner, R. A., & Walker, H. F. (1984). Mixture densities, maximum likelihood and the EM algorithm. *SIAM Review*, *26*, 195–239.

Reynolds, D. A. (1997). Comparison of background normalization methods for text-independent speaker verification. *Proceedings of EURO-SPEECH*, Greece.

Rumelhart, D. E., Smolensky, P., McClelland, J. L., & Hinton, G. E. (1986). Schemata and sequential thought processes in PDP models. In PDP Research Group, J. L. McClelland & D. E. Rumelhart, *Parallel distributed processing: Explorations in the microstructure of cognition* Cambridge: MIT Press, Chapter 14.

Wasserman, P. D. (1993). *Advanced methods in neural computing*, New York: Van Nostrand Reinhold.

Yegnanarayana, B. (1999). *Artificial neural networks*, New Delhi: Prentice-Hall.

Yegnanarayana, B., Kishore, S. P., & Anjani, A. V. N. S. (2000). Neural networks models for capturing probability distribution of training data. *Proceedings of the International Conference on Cognitive and Neural Systems*, Boston.