Backpropagation Neural Nets with One and Two Hidden Layers

Jacques de Villiers and Etienne Barnard

Abstract— **The differences in classification and training performance of three and four layer (one and two hidden layer) fully interconnected feedforward neural nets are investigated. To obtain results which do not merely reflect performance on a particular data set, the networks are trained on various distributions, which are themselves drawn from a "distribution of distributions." Experimental results indicate that four-layered networks are more prone to fall into bad local minima, but that three- and four-layered networks perform similarly in all other respects.**

I. Introduction

BACKPROPAGATION networks [1] with feedforward connections have by now been established as highly competent classifiers [2]–[4], but much remains to be discovered concerning the optimal design of such networks for particular applications. Issues such as the appropriate choice of features for input to the network [5], the training methodology to be used [6], and the best network topology [7], [8] have all been identified, but completely satisfactory solutions have not been offered for any of these problems.

In this paper we investigate a particular aspect related to the topology of backpropagation networks, namely the layering of the neurons. It has become customary to arrange the neurons into an input layer, one or more hidden layers, and an output layer. The feedforward connections are typically only allowed to exist between successive layers. We investigate the effect of using only one hidden layer (which we shall refer to as a three-layered network), as opposed to using two or more hidden layers.

It is known that a backpropagation network with one or more hidden layers can form arbitrary decision boundaries if sufficiently many neurons are used in the hidden layers [9]. This implies that there is no difference in the ultimate separation capabilities of such nets; they might, however, differ in other aspects like the separation performance with a limited number of neurons, or the generalization made. The latter point is of particular importance for practical applications, since generalization to a separate test set is the true measure of a classifier's success. One of the parameters that influence generalization capability of a network is the complexity of the network [10]. We thus need to compare three and four layer networks with the *same complexity*. Network complexity is commonly measured by the *Vapnik-Chervonenkis (VC) dimension.* Baum and Haussler [10] have shown that the VC dimension is closely related to the number of weights in the architecture, and since the exact VC dimension is practically impossible to calculate for complex networks, the number of weights will be used as an approximate indicator of complexity in what follows.

This means that we can compare three and four layer networks with the same number of weights. The only difference between the nets we examined thus lay in the topology, and therefore in the parametrization made by the net.

The parametrization that a classifier makes has a definite influence on performance. Classifiers do better if the decision boundaries (as determined by the parametrization) are naturally related to the distribution of data in the input space. Consider a classifier which can only place three circles with constant radii in a two-dimensional input space (Fig. 1(a)) versus one that can place an ellipse (Fig. 1(b)). All data points to the interior of these figures are classified into one class, and the rest of the feature space is classified into the other class. For a given data set, which has an elliptical distribution, the latter will do a better classification even through it has fewer free parameters to adjust (center of ellipse, minor and major axis lengths, and rotation angle versus the coordinates of the centers of the three circles). Were both classifiers to be tested on a data set where one class consists of three disjoint circular clusters, with radii matching the circles in Fig. 1(a), the situation would be reversed.

As three and four layer networks have different parametrizations, we expect a difference in performance; both their generalization and their classification properties may differ. The latter issue has been addressed by Obradovic and Yan [7], who have shown that the classification boundaries of four-layered networks with size at most polynomial in the number of training samples, are strictly more general that those of such three-layered networks. For real world performance this is not the most important issue; as pointed out above, generalization to unseen examples is the true test of a classifier. (It was also claimed by Chester [8] that three-layered networks are inferior because of an inability to instantiate certain fundamental functions; this claim seems, to be mistaken, since Baum [11] has shown how those functions can be implemented in a three-layered net.) Several related issues (such as the effect of constrained weights [2], [12] on multilayered nets) will not be discussed here, since those issues are sufficiently dissimilar from the current topic to warrant a separate treatment.

As the example of Fig. 1 suggests, generalization performance is intimately related to the distribution of the data to be classified. For this reason a comparison of different types of parametrizations is bound to be to a certain extent data-dependent. Since all nonpathological pattern-recognition problems have certain characteristics in common, one can gain a good understanding of the effects of various parametrizations by taking these into account. The basic idea is explained in

Manuscript received June 14, 1991; revised March 14, 1992.

The authors are with the Department of Electronics and Computer Engineering, University of Pretoria, Pretoria 0002, South Africa.

IEEE Log Number 9200942.

1045-9227/93$03.00 © 1993 IEEE

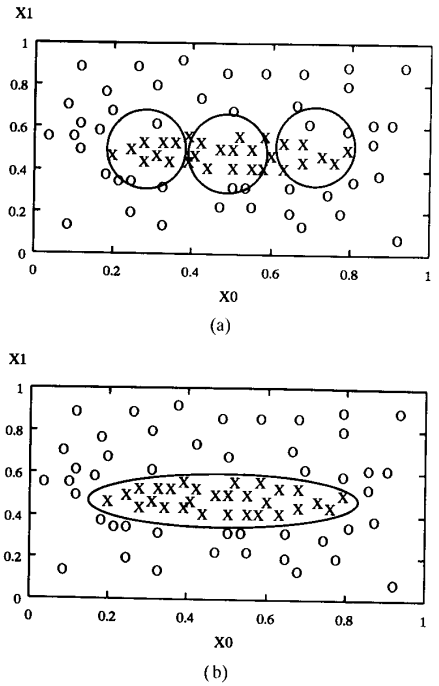

Fig. 1. Influence parametrization has on classifier performance: (a) a classifier places circles of constant radii in feature space, (b) a classifier that uses an ellipse.

Section II, and in Section III we describe a particular choice of data sets which instantiates it. Section IV contains the results of experiments conducted to compare three- and four-layered networks on the data set introduced in Section II. Section V contains our main conclusions.

II. A Distribution of Distributions

The fundamental assumption of classifiers suitable for pattern-recognition systems is that data samples which belong to the same class will somehow be similar to one another. (This similarity derives from the fact that the designer of the system would only place samples in the same class if they have certain characteristics in common, and that features which capture this commonality would be used by the designer.) Samples from the same class will therefore tend to cluster together.

This does not mean that the samples from a particular class will necessarily fall into a single cluster: various causes—especially context dependence—might cause a number of such clusters to develop for each class. Also, clustering might only occur along certain dimensions and not others—cars of the same brand will not necessarily be clustered in a color-based feature, for instance. One should thus think of a typical class in pattern recognition as a set of clusters in feature space, each of which can be more or less spread out, and which might involve some or all of the dimensions of the feature space.

To capture this intuition we introduce the concept of a *distribution of distributions:* probability density functions (distributions) which fit the above description well have a high probability of occurring, whereas distributions which are less clustered are less likely to occur. The distribution of distributions thus assigns a probability to each distribution which might occur.

This idea can be given mathematical content by parametrizing the class of distributions to be considered, and then constructing a conventional probability density function in this parameter space. Depending on the parametrization used, certain data distributions will be excluded *a priori* by this procedure, but for a sensible parametrization those distributions will be either very similar to distributions with nonzero probability, or pathological enough to warrant a probability of zero being assigned to them. Each density function with nonzero probability is described by a vector of parameters $\alpha = (\alpha_1, \alpha_2 \cdots \alpha_N)$, and the probability of a particular distribution $\boldsymbol{P}_\alpha(\boldsymbol{x})$ occurring is given by a probability density function, $\boldsymbol{q}(\alpha)$.

Questions about the suitability of different techniques for pattern-recognition problems can be phrased in terms of expectation values with respect to the distribution $\boldsymbol{q}$. For instance, if we wish to evaluate the ability of a particular technique to train as classifier, (i.e., the percentage of training samples that are classified correctly) we can compute the expectation value of this quantity with respect to $\boldsymbol{q}$. This expectation value then serves as an evaluation of the overall ability of the technique as a tool for training classifiers.

III. Experimental Approach

To evaluate the performance of neural networks, we have to use the performance of *trained* networks in computing the above expectation value; training must, however, be done by iteration. To compute the expected value, we further need to evaluate integrals over two or three layers of nested sigmoidal nonlinearities. These factors combine to make the derivation of an analytic result extremely difficult; we have therefore decided to examine the differences between three- and four-layered networks by experiment. To do so, we have introduced a particular choice for $\boldsymbol{q}$. The parametrization of distribution functions that we use is based on the normal distribution: each class is described by a sum of weighted normal distributions (i.e., Gaussian mixture distributions [13]). The parameters are thus (i) the number of such clusters, (ii) the weighting of each cluster (i.e., the fraction of samples in the class which belong to each cluster), (iii) the statistical parameters (mean and standard deviation) of each cluster, and (iv) the *a priori* probability of each class.

We now detail the distributions chosen for each parameter. We chose to use either two or three cluster to each class, with equal probabilities. Furthermore, the weights of all these clusters were chosen to be equal. That is, if there are n clusters, the probability of a given sample belonging to any particular cluster was chosen to be $1/n$. Similarly, we chose to work with a two class problem, and each class had an *a priori* probability of 0.5.

The D-dimensional distribution of each cluster was chosen to be the product of D independent one-dimensional normal distributions. Let, for cluster i, $\mu_i = (\mu_{i1}, \mu_{i2}, \cdots, \mu_{iD})$ be

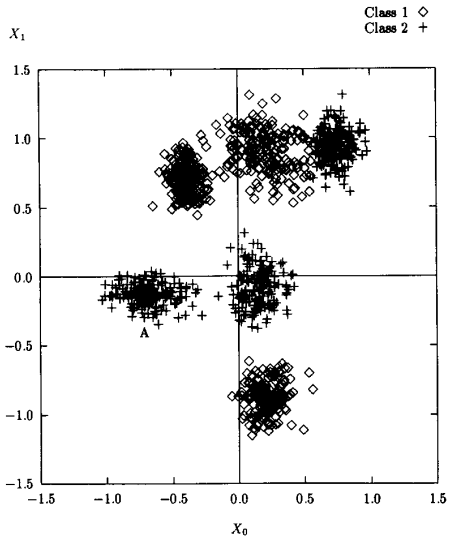

Fig. 2. An example of a training set with 1000 samples. The one-dimensional distributions in x_0 and x_1 of cluster A are detailed in Fig. 3.

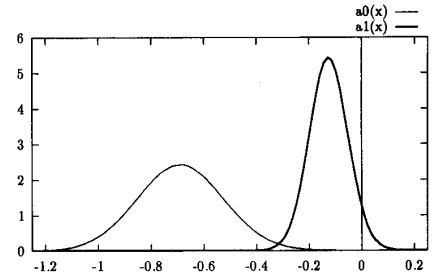

Fig. 3. An example of two normal distributions with means drawn from $m(x)$ and standard deviations drawn from $r(x)$ (See Section III). $A_0(x)$ is the one-dimensional distribution of cluster A (Fig. 2) in the x_0 direction; likewise, $A_1(x)$ is the distribution in x_1.

a vector of the one-dimensional normal distribution means; likewise let $\sigma_i = (\sigma_{i1}, \sigma_{i2}, \cdots, \sigma_{iD})$ be a vector of standard deviations for the same distributions. For each cluster, μ_{ij} was drawn from m, a uniform distribution over $[-1, 1]$ and σ_{ij} was drawn from a normal distribution r, with mean $\mu_r = 0.1$ and standard deviation $\sigma_r = 0.003$. (This implies that the probability of σ_{ij} being less than zero is exceedingly small, namely 4.29×10^{-4}). The chosen values of μ and σ give a reasonable spread in cluster shape and center.

A typical two-dimensional training set is plotted in Fig. 2. This set has two equiprobable categories, each with three clusters. Fig. 3 shows the one-dimensional normal distributions from which the (x_0, x_1) coordinates of points in cluster A are drawn.

Test sets used in our experiments have the same statistics as the corresponding training sets, i.e., the same number of clusters, and clusters have the same weighting and statistical parameters. Five test sets (drawn from the same distributions as the training set) were generated for each training set.

All the networks we evaluated had less than 100 weights. Baum [10] gives the following heuristic: if one loads M examples onto a net with W weights (where $M \gg W$), one expects to make a fraction $\epsilon = W/M$ errors in classifying future examples drawn from the same distribution. Bearing in mind that we would like to know the generalization a net makes given a limited training set, we used sets with $M \in \{100, 1000\}$.

In all, we used the following parameters of $\boldsymbol{q}$ in our experiments: (i) two class problems with equal probability, (ii) D (the dimensionality of the input space) $\in \{2, 5\}$, (iii) for two-dimensional data sets we used all combinations of two and three clusters per class. For five-dimensional sets we used six and nine clusters for each class, and (iv) $M \in \{100, 1000\}$.

The networks were trained using conjugate-gradient optimization [14] with restarts. (This approach to neural-network optimization was chosen for a variety of reasons [15]: it converges quickly, does not require the choice of training parameters, and requires storage linear in the number of weights.) The output of neuron j in layer i is given by: $l_{ij} = \phi(\boldsymbol{l}_{i-1}^T \cdot \boldsymbol{w}_{ij})$, where $\boldsymbol{l}_{i-1}$ is the output of the previous layer (or the inputs to the net) and $\boldsymbol{w}_{ij}$ is the weight vector from layer $i-1$ to neuron j in layer i. The nonlinearity ϕ is the sigmoid $\phi(x) = 1/(1+e^{-x})$. A classification is considered to be correct if the output neuron corresponding to the correct class is most active. Networks of different topologies were constrained to have the same number of parameters or weights. The number of hidden nodes in the three-layered nets were chosen to give a total of 20, 40, or 60 weights; these nets were compared to four-layered nets with the same number (to within 5%) of weights. In our study, we calculated the expectation values of: (i) *average trainability* (T_{ave}), (ii) *optimal trainability* (T_{opt}), (iii) *average generalization performance* (G_{ave}), *optimal generalization performance* (G_{opt}), and (iv) *98% classification count* (T_{98}, G_{98}) of these networks with respect to the distribution $\boldsymbol{q}$. To enable us to draw valid statistical conclusions, each network was trained $r = 10$ times with random initial weights. All the weights were independently chosen, and all the weights leading to a particular neuron were uniformly distributed in the range $[-3/\sqrt{N}, 3/\sqrt{N}]$, where N is the number of weights leading to the neuron. The *average trainability* T_{ave} is defined as the average (over the r runs for each net) ability of a net to classify training samples: i.e., the percentage of training examples that are classified correctly. *Generalization performance* G_{ave}, is the percentage of *test* samples that are classified correctly. *Optimal trainability* T_{opt} is defined as the best classification of the training set a net made out of the r runs for that net. This is of practical importance—when we need a neural classifier, we usually train it a number of times and then select the trained network with the best performance. *Optimal generalization performance* G_{opt} is the performance on the *test* set of the network that delivered T_{opt}. We define the *98% classification count* as the number of networks that correctly classify more than 98% of the examples (either from the training set (T_{98}) or the test set (G_{98})).

In the four-layered networks, different combinations of the number of neurons in the first and second hidden layers can be found if the net is to have W weights. In order to be

unbiased in our conclusions, we compared three-layered nets with W weights to four-layered nets with different topologies: an optimal and suboptimal example (that is with respect to T_{ave}). The optimal four-layer topology for a network of given complexity was found by experiment.

IV. Results

All results in this section were evaluated at a statistical significance level of 5%.

A. Four layer topology

To investigate the influence of topology on a four-layered network with W weights, we calculated the expectation values $E(T_{\text{ave}})$, $E(T_{\text{opt}})$, and $E(G_{\text{ave}})$, so as to compare the performance of various four-layered networks with similar weight counts but different neuron counts in the two hidden layers. Eight nets with $W = 40$ and ten nets with $W = 60$ were trained ten times each with random initial weights. The density functions described in Section III were employed.

The expectation value $E(G_{\text{ave}})$ (i.e., with respect to the test set) for a 40 weight network over different topologies is given in Fig. 4. No significant difference between $E(G_{\text{ave}})$ and $E(T_{\text{ave}})$ was observed. The expected value $E(G_{\text{ave}})$ is seen to peak where the number of neurons in the first and second hidden layers are more or less equal (Fig. 4(a)); the results indicate a preference for an architecture which gradually expands until the last layer contraction. Differences in the graph are statistically significant. In Fig. 4(b), we plot $E(G_{98})$ with respect to the test set. We recognize a distribution very similar to $E(G_{\text{ave}})$. Expectation values were computed over a set of 1200 networks. This set was constructed as a product over: (i) the topology of the network (8 examples), (ii) 10 different initial states of network weights, (iii) three training set cluster configurations, and (iv) five random training sets for each cluster configuration.

We have found no significant difference in $E(G_{\text{opt}})$ for different topologies—the exception occurs when there is a severe bottleneck in the net, when either the first or second hidden layer has only one neuron.

The expectation values for G_{ave}, G_{opt}, and G_{98} for $W = 60$ networks delivered similar results.

We have found that four layer backpropagation networks are easier to train (using conjugate-gradient optimization) if the number of neurons in the hidden layers are balanced. The optimal performance is not severely influenced by the network topology. This leads us to reason that the training algorithm is more likely to get stuck in a local minimum if the number of neurons in the hidden layers are not balanced.

B. Three versus Four layer networks

In our main experiment we compared three- and four-layered networks of the same complexity. The complexity (as defined by the number of weights W) was chosen with $W \in \{20, 40, 60\}$. In the case of four-layered networks, two topologies (where the distribution of neurons between hidden layers differ) were included in our experiment: examples labeled 4_{subopt} were chosen to be suboptimal in $E(T_{\text{ave}})$

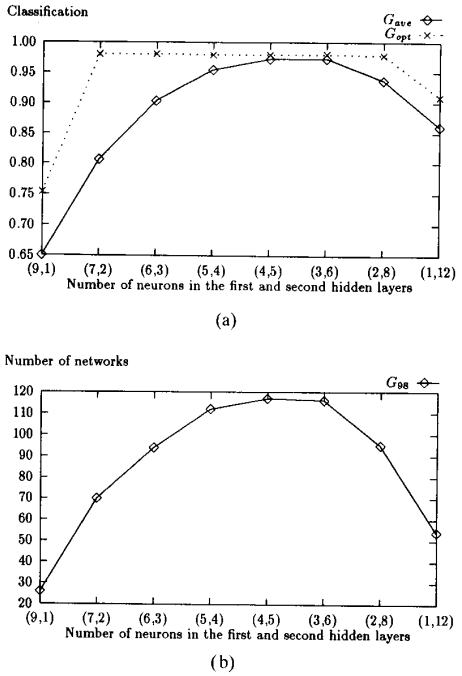

Fig. 4. Performance of a four-layered classifier as a function of network topology: (a) Generalization performance ($E(G_{\text{ave}})$), and optimal trainability ($E(G_{\text{opt}})$). (b) 98% Classification count ($E(G_{98})$).

(as determined in experiments reported in Section III) and 4_{opt} examples to be optimal. In the networks where $W = 20$ only one set of four layer nets were used—the only possible topology to satisfy our condition that the number of weights should not deviate from W by more than 5%. The expected values of T_{ave}, G_{ave}, T_{opt}, and G_{98} (Section III) were calculated for all the nets.

1) A specific case: We begin by describing the results obtained for a specific choice of parameters in $\boldsymbol{q}$; networks were trained and tested on a restricted $\boldsymbol{q}$: the dimensionality of the feature space was fixed at $D = 2$, the number of examples in the training set at 1000, the number of clusters in each class was set at three. Results pertaining to the expected values of our criteria for comparison between the networks are somewhat similar to the more general results obtained by utilizing the full $\boldsymbol{q}$. The values of $E(G_{\text{ave}})$ are shown in Fig. 5.

We have found a significant difference in $E(G_{\text{ave}})$ between the three layer and the 4_{subopt} examples of the four layer networks, with three layer nets always doing better. The difference between the three layer nets and the 4_{opt} examples are not significant at the 5% level. *No* significant difference whatsoever was found between the $E(T_{\text{opt}})$ values for the three- and four-layered networks. The training set classification performance $E(T_{\text{ave}})$ followed the same pattern as that of the generalization performance $E(G_{\text{ave}})$.

Here we only examined a case where the condition $M \gg W$ (Section III) holds. In the following section we look at the influence a training set with limited size has on classifier performance.

2) Influence of the training set size: The size of an available

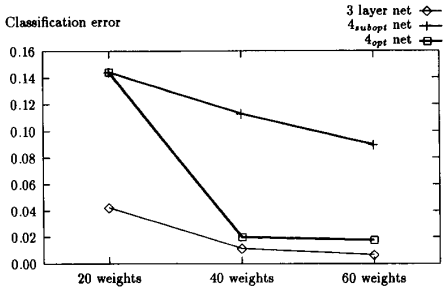

Fig. 5. Generalization performance ($E(G_{\text{ave}})$) for networks with a two-dimensional feature space, 1000 training samples and three groups in each class.

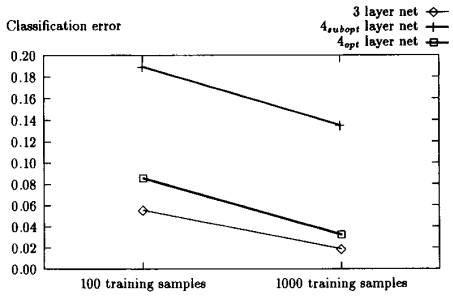

Fig. 6. Generalization performance of a 60 weight network ($E(G_{\text{ave}})$) as a function of training set size.

training set is an important parameter—in real problems training sets are hardly ever complete; the generalization a network makes based on a limited training set is therefore of some interest. We now present the results (expectation values) for three- and four-layered networks when trained on a restricted $\boldsymbol{q}$; the number of examples in the training set was fixed at $M = 100$ and then at $M = 1000$, expectation values of our criteria for the networks are then compared.

The results for $E(G_{\text{ave}})$ for a network with 60 weights are shown in Fig. 6. Three layer networks are seen to do better. The classifiers differ significantly in performance. The one exception is in the $W = 40$ networks, between the nets 3 and 4_{opt}, this difference is not significant at the 5% level. Training set results ($E(T_{\text{ave}})$) are mostly consistent with those of the test set ($E(G_{\text{ave}})$). We did, however, find one anomaly: the performance of the 3 and 4_{opt} nets show no significant difference for 1000 training set samples, for $M = 100$ the difference *is* significant. There are, once again, no difference in $E(G_{\text{opt}})$ between network 3 and the 4_{opt} or 4_{subopt} nets. (Note, also, that the larger training set always leads to better performance, as expected.)

We now move to our general comparison between three- and four-layered networks.

3) General results: Here we consider expectation values over the complete $\boldsymbol{q}$ (see Section III). We give $E(G_{\text{ave}})$ in Fig. 7(a) and $E(G_{\text{opt}})$ in Fig. 7(b). (As a baseline for the comparison, the Bayes optimal results, computed from the known probability density functions, are also indicated on these figures.) The expectation values of our criteria should give us a general indication of the differences between three- and four-layered networks.

In all cases we have found a significant difference in $E(T_{\text{ave}})$, with the three layer network always doing better. The $E(G_{\text{ave}})$ results are similar. We have found no statistical difference in $E(T_{\text{opt}})$ and $E(G_{\text{opt}})$.

Our final data set was obtained from a real pattern-recognition problem, namely the classification of the modulation types of transmitted signals. After preprocessing a set of 21 features was extracted to describe each signal. Using these features each signal had to be classified as belonging to one of eight modulation types. Since these data were obtained empirically, the underlying density function is not known, and the Bayes rate is not defined. The results of

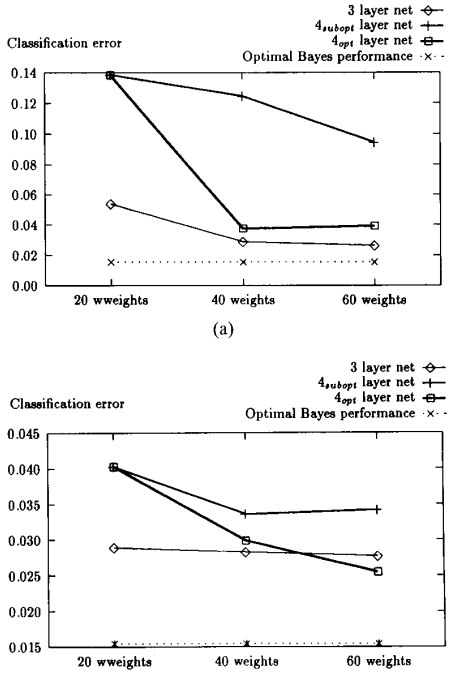

Fig. 7. Generalization performance (a) $E(G_{\text{ave}})$ and (b) $E(G_{\text{opt}})$ of three- and four-layered network classifiers with respect to $\boldsymbol{q}$.

this experiment are given in Fig. 8. We found no statistically significant differences in the performance of three- and four-layered networks on this dataset, thereby supporting the main conclusion found with the artificial "distribution of distributions."

V. Conclusion

Three and four layer neural networks have different parametrizations. Through the use of a data set which consists of a distribution of distributions and Monte Carlo simulations, we can draw initial conclusions with regard to the effects these differences have on various aspects of classifier performance.

The experiments we have described have shown that, for networks constrained to have the same number of weights, that is, comparable complexity:

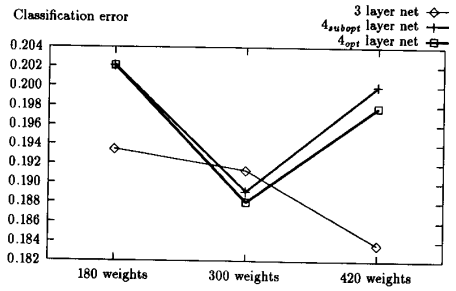

Fig. 8. Generalization performance $E(G_{\text{ave}})$ for a real-world data set.

- There is no statistically significant difference between the optimal performance ($E(T_{\text{opt}})$ and $E(G_{\text{opt}})$) of three- and four-layered nets;
- Three layer nets do a better classification on average ($E(T_{\text{ave}})$ and $E(G_{\text{ave}})$).
- Four-layered networks train easier if the number of neurons in the two hidden layers are more or less equal.

We have found no difference in the optimal performance of three- and four-layered networks. This leads us to believe that the networks have similar capabilities in classifying examples drawn from the most common distributions that we find in pattern recognition. If we consider the second result, that is, three layer networks do better on average, we have to conclude that four-layered nets are more difficult to train; the initial weight choices have a more pronounced influence on final network performance, but if we train enough networks we will most likely find a net with performance similar to that of an optimal classifier. A probable cause for this behavior is the local minima problem. We thus conclude that four layer networks are more prone to the local minima problem during training (using backpropagation and conjugate-gradient optimization).

The generality of our results remains to be discussed. Clearly, the choice we have made for $\boldsymbol{q}$ is too simplistic to be truly representative of probability densities encountered in pattern recognition. On the other hand, it is hard to imagine how our choice could have biased the experimental results—it is most likely that the same conclusions would have been reached if a more complete $\boldsymbol{q}$ had been used (this would require considerably increased computational expense, though!) Similarly, our choice of conjugate-gradient optimization does not compromise the generality of our results: any reliable local optimizer would have found similar minima, and global optimization of neural network criterion functions is surely unrealistic. The main simulations reported here have been restricted to low-dimensional input spaces, and it remains to be seen whether an increase in D will effect out results. No systematic variation with increasing D was observed in our experiments, and this agrees with our intuition that input dimensionality is not an important factor in determining the success of a particular parameterization. Also, the single high-dimensional ($D = 21$) data set we used supports the general conclusion of this work. However, further work on this issue is required to give a conclusive answer. Finally, our choice of initial weights may be thought to bias our results, but a separate study [16] shows that this choice is close to optimal for all topologies.

The above points lead us to conclude that there seems to be no reason to use four layer networks in preference to three layer nets in all but the most esoteric applications.

REFERENCES

[1] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, pp. 533–536, 1986.

[2] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. J. Lang, "Phoneme recognition using time-delay neural networks," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 37, pp. 328–339, Mar. 1989.

[3] E. Barnard and D. Casasent, "A comparison between criterion functions for linear classifiers, with an application to neural nets," *IEEE Trans. Syst., Man, Cybern.*, vol. 19, pp. 1030–1041, Sept./Oct. 1989.

[4] D. J. Burr, "Experiments on neural net recognition of spoken and written text," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-36, pp. 1162–1168, July 1988.

[5] E. Barnard, R. A. Cole, M. P. Vea, and F. Alleva, "Pitch detection with a neural-net classifier," *IEEE Trans. Signal Processing*, vol. 39, pp. 298–307, Feb. 1991.

[6] R. A. Jacobs, "Increased rates of convergence through learning rate adaptation," *Neural Networks*, vol. 1, no. 1, pp. 295–308, 1988.

[7] Z. Obradovic and P. Yan, "Small depth polynomial size neural networks," *Neural Computation*, vol. 2, pp. 402–404, winter 1990.

[8] D. Chester, "Why two hidden layers are better than one," in *Proc. Int. Joint Conf. Neural Networks*, Washington, DC, IEEE, Jan. 1990, pp. I-265–I-268.

[9] M. Stinchcombe and H. White, "Universal approximation using feedforward networks with non-sigmoid hidden layer activation functions," in *Proc. Conf. Neural Networks*, Washington, DC, IEEE, June 1989, pp. I—613–I-618.

[10] E. Baum and D. Haussler, "What size net gives valid generalization?" *Neural Computation*, vol. 1, pp. 151–160, spring 1989.

[11] E. B. Baum, "When are k-nearest neighbor and back propagation accurate for feasible sized sets of examples?" in *Neural Networks—EURASIP Workshop 1990*, L. B. Almeida and C. J. Wellekens, Eds. Berlin, Germany: Springer-Verlag, 1990, pp. 2–25.

[12] Y. L. Cun, J. Denker, and S. Solla, "Optimal brain damage," in *Advances in Neural Information Processing Systems 2*, D. Touretzky, Ed. San Mateo, CA: Morgan Kaufmann, 1989, pp. 622–629.

[13] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*. New York: Wiley, 1973.

[14] E. Barnard, "Optimization for training neural nets," *IEEE Trans. Neural Networks*, vol. 3, pp. 232–240, Mar. 1992.

[15] M. J. D. Powell, "Restart procedures for the conjugate gradient method," *Mathematical Programming*, vol. 12, pp. 241–254, Apr. 1977.

[16] L. Wessels, E. Barnard, and E. van Rooyen, "The physical correlates of local minima," in *Proc. Int. Neural Networks Conf.*, Paris, France, July 1990, p. 985.