Estimating the Reliability of Neural Network Classifications

Aarnoud Hoekstra, Servaes A. Tholen and Robert P.W. Duin

Pattern Recognition Group, Faculty of Applied Physics,
Delft University of Technology, Lorentzweg 1, 2628 CJ Delft, The Netherlands
e-mail: *aarnoud@ph.tn.tudelft.nl* *

Abstract. A method for quantifying the reliability of object classifications using trained neural networks is given. Using this method one is able to give an estimation of a confidence value for a certain object. This reveals how trustworthy the classification of the particular object by the neural pattern classifier is. Even for badly trained networks it is possible to give reliable confidence estimations. Several estimators are considered. A k-NN technique has been developed to compare these using a learning set based artificially generated validation set. Experiments show that applying the developed estimators on a validation set gives the same results as applying the estimators on an independent test set. The method was tested on a real-life application, human chromosome classification, and gave good results which indicate the applicability of our method.

1 Introduction

In pattern recognition applications one is not only interested in the performance of a pattern classifier but also in the reliability of such a classifier. It has been shown in many applications that especially in nonlinear problems neural networks can be a good alternative for traditional pattern recognizers. We investigated the situations in which one has a given, trained, neural network. Given a set of unlabeled objects to be classified, one cannot determine from the neural classifier output whether it performs correctly or not. If the classifier is trained badly the network's performance cannot be judged from the net output values. Only perfectly trained networks will give reliable output values, due to the fact that these networks approximate the a posteriori probabilities sufficiently well. An interesting point therefore is to define a reliability measure that still works in cases of badly trained networks. In this paper a method is proposed with which the reliability of a single classification of an object by a given neural classifier can be determined. This reliability measure, referred to as *confidence value*, enables further processing of an object or rejecting it.

The method presented here does not transform the network's output values in such a way that they fullfil the probability constraints, e.g. the network's ouput values constitute the aposteriori probabilities of the classes. The research started from the same point of view as the work by Denker and LeCun [2] but differs significantly

* Published in: Proceedings of the ICANN'96, pages 53-58, 1996

in the developed technique. In their work it is assumed that a network is trained is such a way that some second order constraints are fullfilled. We do not make any assumptions on the way the network is trained. Furthermore also a method for generating a validation set, e.g. a calibration set as mentioned in [2] will be presented in this paper. Denker and LeCun use such a set as an independent test set to collect statistics about the classification results of the network. However, they do not state how a validation set was inferred, whereas we propose a k nearest neighbour data generation method for the case that an independent set is not available.

The neural networks under investigation are the normal feedforward networks trained with backpropagation and having sigmoidal transfer functions. Furthermore only networks consisting of one hidden layer are considered, although the results can be generalized to more than one hidden layer. The outline of the paper is as follows. First the problem of confidence in neural networks will be introduced. The next section is dedicated to the calculation of confidence values. The third section treats the generation of validation sets from a given learning set. In the following section the application of confidence values on a real-life problem, the classification of human chromosomes, is presented. The paper ends with conclusions and some remarks.

2 Confidence Values

When a trained neural network classifies an unlabeled object, the operator would like to know what confidence one should have in the classification. Much has been published about the overall error rate for a discriminant function
[5, 3], but here the goal is to estimate the probability that a certain classification is correct, see also [2]. The confidence of a classification can be estimated using different methods and in different layers of the network. The confidence, and its corresponding estimator, will be introduced more formally.

Assume a network finds a discriminant function $S(\mathbf{x}) : \mathbf{x} \rightarrow \hat{\omega}$. This discriminant function S assigns a class label $\hat{\omega}$ to a sample $\mathbf{x}$. For each sample $\mathbf{x}$ the classification based on S is either correct or incorrect, therefore a correctness function $C(\mathbf{x}, \omega)$ can be defined:

$$C(\mathbf{x}, \omega) = \begin{cases} 1 \text{ if } S(\mathbf{x}) = \omega \\ 0 \text{ if } S(\mathbf{x}) \neq \omega \end{cases}, \quad (1)$$

where ω is the true class label of sample $\mathbf{x}$. When an unlabeled sample is classified, the *confidence* in this classification is defined as the probability the classification is correct. Given a certain network, for an (in general unknown) object $\mathbf{x}$, the confidence is given by:

$$q(\mathbf{x}) = P(C(\mathbf{x}, \omega) = 1 | \mathbf{x}). \quad (2)$$

The classification confidence is thereby the posteriori probability for the estimated class. For known distributions and given $S(\mathbf{x})$ the errors due to class overlap as well as imperfect training influence $q(\mathbf{x})$, therefore $q(\mathbf{x})$ can be expressed as:

$$q(\mathbf{x}) = \frac{P_{\hat{\omega}} f_{\hat{\omega}}(\mathbf{x})}{f(\mathbf{x})}, \quad (3)$$

where $f_{\hat{\omega}}$ is the class density function for class $\hat{\omega}$ as a function of the feature $\mathbf{x}$ and $P_{\hat{\omega}}$ the a priori class probability. The function $f(\mathbf{x})$ denotes the joint probability for $\mathbf{x}$. For unknown distributions $q(\mathbf{x})$ has to be estimated. As there are several possibilities for estimating $q(\mathbf{x})$ (see the next section), a criterion function J is defined to compare the different estimators for $\hat{q}_i(\mathbf{x})$:

$$J_i = \frac{1}{N_{\mathcal{T}}} \sum_{\mathcal{T}} [C(\mathbf{x}, \omega) - \hat{q}_i(\mathbf{x})]^2, \quad (4)$$

where $\mathcal{T}$ is an independent test set and $N_{\mathcal{T}}$ is the number of samples in the test set, the index i is used to indicate the J for the i^{th} estimator $\hat{q}_i(\mathbf{x})$. The lower bound of J is usually not zero due to class overlap. However, when the underlying class densities are known J_{min} can be estimated by substituting (2) for each sample in (4). The aim is to minimize J over different estimators $\hat{q}_i(\mathbf{x})$ for a certain independent test set, yielding the best estimator for the confidence values.

3 Confidence Estimators

Since in real applications parameters such as distributions of the samples and a priori probabilities are unknown, there is a need for estimating confidence values. It was discussed in the previous section how the different estimators can be compared but nothing was said yet about which kind of estimators are used for $q(\mathbf{x})$. One can distinguish two kinds of estimations.

Let us first consider the layer in the network where the confidence is estimated. The confidence can be estimated in the input space, hidden unit space or output space. A feed forward network can be considered as series of data mappings, from the input feature space, being the input space of the network, followed by one or more hidden layers, towards the output space in which the final classification is done. These spaces differ in dimensionality and data configuration. Thereby, depending on the result of the training, it is open which space is the optimal one for confidence estimation. For sufficiently trained networks it can be advantageous to estimate in a different layer than the output space.

Second there are different methods for estimating the confidence value. We used three methods for estimating the value of $q(\mathbf{x})$, namely:

1. Nearest neighbour method
2. Parzen method
3. Logistic method

Each of the methods has its advantages and disadvantages. In the application to be presented next we restrict to the methods 1 and 3. Method 2 has been extensively tested, but appeared to be outperformed by the other methods. However, the Parzen method is suited for detecting outliers and therefore mentioned for the sake of completeness. Now each of these methods will be discussed.

3.1 Nearest Neighbour Method

Given a certain space (input, hidden or output) with a defined measure, the k-nearest neighbour (k-NN) method determines the k nearest learning samples to a test sample under consideration. For that sample the confidence is estimated as:

$$\hat{q}_{knn} = \frac{N_{\hat{\omega}}}{k}, \quad (5)$$

where $N_{\hat{\omega}}$ is the number of correctly classified samples among the k nearest neighbours. For this number we just used $k = \sqrt{N_{\mathcal{L}}}$, where $N_{\mathcal{L}}$ is the number of samples in the learning set. Note that equation 5 needs to be altered when the a priori class probabilities differ.

3.2 Parzen Method

Here we use a Parzen estimator to estimate the confidence, by estimating the class densities and kernels. Experiments show that the Parzen estimator never outperforms the k-NN estimator ([7]). Denker and LeCun already indicated that the Parzen method would fail due to the high dimensionality of the space [2]. This was confirmed by our experiments. Therefore the Parzen estimator was not used in the application which is described in the next section.

3.3 Logistic Method

This method was chosen because of is simplicity, by which this method can be applied fast. The parameters can be calculated quite easily and need to be determined only once. It is assumed that the conditional class densities can be approximated by a sigmoidal function [1]. For a two-class problem, with classes A and B, this means that:

$$P(B|\mathbf{x}) = \frac{1}{1+\exp(\boldsymbol{\beta}\mathbf{x}+\beta_0)}, \quad (6)$$

under the assumption that P_A equals P_B. The number of parameters that have to be estimated in equation 6 equals $d+1$, the dimension of the space to be investigated plus one bias term. These parameters can be estimated using a learning set and an optimization method. In the experiments the parameters for the logistic model were optimized using maximum likehood estimation. The method is extended to more classes by taking one class as base class and estimate all parameters of the other classes with respect to this base class (g). By using our definition for confidence values, the logistic estimator $\hat{q}_{logistic}$ becomes:

$$\hat{q}_{logistic} = \{1 + \sum_{i=1}^{g-1} \exp(-\boldsymbol{\beta}_{gi}\mathbf{x}')\}^{-1}. \quad (7)$$

4 Validation sets

In the previous section three estimators for $q(\mathbf{x})$ have been introduced. Each of these estimators should be tested using an independent test set. Usually such independent test sets are not available, just one set is available from which both the learning set and test set are drawn. One could take the learning set for testing the estimators, however this is a very unhealthy situation. Both, the network and the estimator were created using the same set, which makes the estimator useless. To overcome this in practical situations a k-NN data generation method [4] was used to create a *validation set*. Such a validation set has roughly the characteristics of the learning set but is distinctive enough to be used a test set.

The k-NN data generation method is a fairly simple way to create a new dataset from a given one. In this method a learning set is used as starting point for a new set. Each point in the original set is used for generating an arbitrary number of new points by adding Gaussian noise in such a way that it follows the local probability density estimated from its k nearest neighbours. For k the dimensionality + 2 was used.

Note that a validation set will **not** be able to replace an independent test set entirely, but it is an acceptable alternative. Experiments show that the k-NN data generation method generates a validation set which is reliable enough to predict the same estimators and the network layer as an independent test set would do. In the application to be described next both a validation set and a test set were used. Many experiments not reported here can be found in [7]. These experiments show the validity of our data generation method for badly trained neural networks.

5 Application: Human Chromosome Classification

The problem of human chromosome classification has been under extensive investigation in our research group. Due to the increasing interest in neural networks in the recent years, also neural networks have been applied to chromosome classification. A normal human cell contains 46 chromosomes, which can be divided into 24 groups. There are 22 pairs of so called autosomes and two sex chromosomes. The chromosomes have different features that can be used for classification: *length*, *centromere position* and *banding pattern*. In this classification problem we only take the banding pattern into account. This banding pattern is sampled and then fed to a neural network for classification.

The sampling of a chromosome-band resulted in 30 features, ie. thirty samplings per band, which are used for classification of a chromosome. Earlier studies indicate that this is a reasonable value to take [6]. Consequently a neural network consisting of 30 input neurons, and 24 output neurons (one for each class) was used. The number of hidden units, however, is still arbitrarily chosen and was set to 100, a number derived from earlier experiments [6]. For training the network a total set of 32,000 training samples was used, the same number of samples was used for testing the network. Furthermore each sample was scaled between 0 and 1.

Network training was done for 200 epochs with momentum and adaptive learnrate, using the Matlab neural toolbox. The momentum term was set to 0.7, the initial learnrate was set to 0.1. Each time the mean squared error of the network increased by 10% the learnrate was halved. After training the network performance was 85% on the learn set and 83.5% on the test set (see also table 1). Two estimators plus the output of the network were used for the estimation of the confidence values. Table 1 states the different values of J which were found for the different estimators. Furthermore the k-NN and the logistic estimator were also applied in different parts of the network in order to estimate the confidence.

used set	learn set with l.o.o. est.	validation set	test set
estimator	J	J	J
k-NN			
input	0.167	0.165	0.161
hidden	0.119	0.119	0.119
output	0.086	0.088	0.095
logistic			
input	0.140	0.141	0.141
hidden	0.692	0.692	0.694
output	0.147	0.150	0.158
netout	0.482	0.479	0.474
avg. sd(J)	0.001	0.001	0.001
class perf	85.0%	84.5%	83.5%

Table 1. The criterion function values for all estimators, on the learning set, validation set and the test set.

The abbreviation l.o.o. means leave one out. Table 1 shows that the k-NN estimator in the output space has the best score over all the different sets.The logistic estimator performs worse in the output space than in the input space. Furthermore the network's output is not very useful as an estimator for the confidence. This is due to the fact that the network has only trained for 200 epochs. The data generation method was also tested, but it shows no advantage over using the learn set. However, other experiments showed that in some cases the validation set method is preferred over the learn set, especially in overtrained networks (see [7]).

6 Conclusions

In this paper an alternative method for calculating the confidence in classifications made by a neural network has been presented. This way uncertainties due to network training and class overlap are found. For the computation of the confidence three methods have been proposed: the nearest neighbour, the Parzen and the logistic method. From experiments (not reported here but to be found in [7]) we selected two methods for application on a real-life problem. These methods has been applied in the classification of human chromosomes. Moreover the estimates were made in different parts of the network. The results obtained from the chromosome experiments were very promising. From the experiments it was concluded that the k-NN method in the output space performs the best.

The method of generating a validation set has some shortcomings and needs therefore to be refined. A problem arises when the learning set contains some outliers. Since the data generation method is based on the learning set it will also generate new points around the outliers. However, in a test set these outliers will not be present at all. In this case the method fails. Another way to apply confidence values is to use them as a stopping criterion during training. At the moment the best confidence estimator is in the output layer the network can be expected to have an optimal generalization.

7 Acknowledgements

This research was partially sponsored by the SION (Dutch Foundation for Computer Science Research) and SNN (Foundation for Neural Networks, Nijmegen). Furthermore the authors would like to thank Martin Kraaijveld (Shell Research, Rijswijk) for his helpful discussions.

References

1. Anderson, J.A. *Logistic discrimination*, volume 2 of *Handbook of Statistics*, pages 169–191. North-Holland, 1982.
2. Denker, J.S. and Le Cun, Y. Transforming neural-net output levels to probability distribution. In R.P. Lippmann, J.E. Moody, and D.S. Touretzky, editors, *Advances in Neural information Processing Systems 3*, pages 853–859. Morgan Kaufmann, 1991.
3. Duda, R.O. and Hart, P.E. *Pattern classification and scene analysis.* John Wiley and sons, 1973.
4. Duin, R.P.W. Nearest neighbor interpolation for error estimation and classifier optimization. In *Proceedings of the 8th SCIA tromsø*, 1993.
5. Fukanaga, K. *Introduction to Statistical pattern recognition.* Academic Press, 1990.
6. Houtepen, J.D.J. Chromosome banding profiles: how many samples are necessary for a neural network classifier ? Master's thesis, Delft University of Technology, October 1994.
7. Tholen, S.A. Confidence values for neural network classifications. Master's thesis, Delft University of Technology, April 1995.