Neural Network Initialization by Combined Classifiers

Martijn van Breukelen and Robert P.W. Duin
Pattern Recognition Group, Department of Applied Physics
Delft University of Technology, The Netherlands

Abstract

If a set of linear classifiers in the same feature spaces is combined by a linear output classifier and if each of these classifiers has a sigmoid output function then this set of classifiers has the same architecture as a feed-forward neural network. A combined set of classifiers, however, is trained in an entirely different way. In this paper it is shown that it can be advantageous to use such a set as an initialization for a neural network.

1. Introduction

Combining sets of different classifiers has been frequently studied recently and is shown to be an effective tool for improving classification results, e.g. see [1], [2], [4], [8] and [9]. The basic set of input classifiers can be based on different feature sets, different training sets, different realizations of the same classifier (e.g. a neural network) or can be a set of different classifier types (e.q. linear, quadratic, k-NN, neural net, see [7]). The output classifier combining the results of the input classifiers is often of a fixed type, e.g. mean, product, minimum, maximum or median of continuous classification outputs like posterior probabilities. Theoretical justifications can be given for such rules, [1] and [9].

In this paper we will focus on sets of linear input classifiers combined by a trained linear output classifier. In between is a mapping of the input classifier outputs on classifier conditional posterior probabilities, [7]. For this a logistic function or sigmoid is used. As a result the entire set of linearly combined classifiers can be mapped on a feed-forward neural network. Each linear classifier is transformed into a set of nodes, one for each class, in the neural network. There is one little difference between the resulting network and the combined set of linear classifiers: the class conditional posterior probabilities on the output of a linear classifier sum to one. Outputs of neurons pointing to different classes don't necessarily sum to one.

By the above training a combined set of linear classifiers is an alternative for neural network training rules like back-propagation or Levenbergh-Marquardt optimization. It is a point of research to determine whether and when this is faster and how performances might differ. In this paper we will show by a single set of experiments that by the combined classifier at least a good initialization is obtained for the neural network.

In the next sections some details on our combined set of linear classifiers are given, followed by a description of the data and the experiments. This paper is based on experiments more extensively described in [3].

2. The linear combination of linear classifiers

Suppose we have a multi-class problem with classes $\Omega = \{\omega_1, \omega_2, \ldots, \omega_c\}$. Each object $\mathbf{x}$ is given by k features: $\mathbf{x} = (x_1, x_2, \ldots, x_k)$. In order to obtain a set of n different classifiers the feature set is split in n different feature sets of k_i (i = 1,n) features. In the below experiments these sets are non-overlapping, so $\Sigma k_i = k$, but this is not essential.

Each multi-class classifier consists of a set of two-class classifiers between one of the c classes and all other c-1 classes. In each of the n feature spaces c linear classifiers $S_{ij}(\mathbf{x}_i)$ are computed (i=1,n; j=1,c) using Fisher Linear Discriminant $D_{ij}(\mathbf{x}_i)$. For each of these discriminants an optimal output sigmoid is found by optimizing a multiplicative coefficient α using the maximum likelihood rule over the training set, so

$$S_{ij}(\mathbf{x}_i) = \frac{1}{1 + \exp(-\alpha_{ij} D_j(\mathbf{x}_i))}$$

After all nc classifiers $S_{ij}(\mathbf{x}_i)$ have been computed a similar output classifier $C(\mathbf{x})$ consisting of c Fisher Discriminants $C_j(\mathbf{x})$ is trained on all the nc outcomes over the training set. This combined classifier is mapped on a neural network with k inputs, nc hidden neurons and c outputs. As for each

hidden neuron just weights are computed to k_i of the k input features, all weights for the other features are set to zero.

The neural networks obtained in this way are further trained using the extended backpropagation routines supplied in Matlab's Neural Network Toolbox [6]. These can also be based on the Nguyen-Widrow random initialization method [10] which we used as a reference.

3. The data set

We used the same dataset as in earlier reported experiments on combined classifier [2], [3], [4]. It consists of a set of handwritten numerals as used on dutch utility maps. They were scanned in 8 bits using 400 dpi. The grey value images were sharpened, normalized on size, deskewed and thresholded, resulting in 30 by 48 binary pixels. On these data several feature sets are computed. Here we use:

a. Karhunen - Loève: the 64 coefficients related to a Karhunen - Loève orthogonalization of the original set images representing the raw features.

b. Pixel features. The 30 x 48 pixels were divided in 15 x 16 tiles of 2 x 3 pixels. All these tiles were averaged, resulting in 240 features.

c. Zernike moments. These are 47 rotation invariant moments (by which almost each distinction between the characters '6' and '9' is lost). To this dataset 6 topological/ morphological features have been added like the number of endpoints derived from the skeleton. The total number of features is thereby 53.

For each of the 10 classes '0' - '9' 200 objects are available. At random a training set of 100 objects per class was generated for learning. The remaining objects are used for testing.

4. Description of the experiments

In each of the experiments two Fisher classifiers are trained (n = 2) on a single set of features. We used all ten classes and we randomly divided the features over the two classifiers. That way we introduced some randomness. Combined classifiers were trained as described in section 2.

We used the parameters of this combination as initial weights for the equivalent neural network. The combination and its equivalent neural network now produced the same outputs if the same inputs were applied to them. Since we used ten classes (c = 10), each of the Fisher classifiers corresponds to 10 neurons, see section 2. The combined classifier corresponds thereby with 20 hidden neurons and 10 output neurons.

After the initialization, the equivalent networks were trained with the back-propagation algorithm. For this training we used targets of 0.9 and 0.1, an initial learning rate of 0.01, a learning rate increase of 1.05 and a learning rate decrease of 0.7. During training the classification errors on

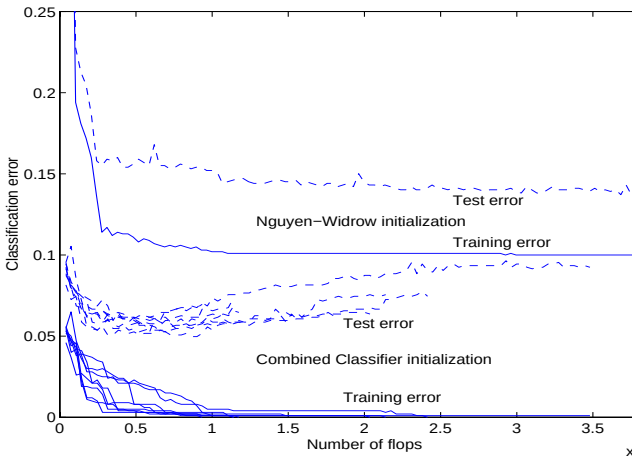

Fig. 1. Fig. 1 Learning errors and test errors during neural network training for the Karhunen-Loève feature set for the Nguyen-Widrow initialization (onl the converging network is shown) and for the 8 combined classifier initializations.

both, the test set and the learning set were determined. The results are compared with those of neural networks of the same architecture initialized with the Nguyen-Widrow method and trained with the same training parameters. Training was stopped when either the learning error became zero or the number of training iterations was more than 5000 and no improvements were made during the last half number of training iterations. That way the networks were trained for a rather long time and most networks ended over-trained. This was done to be able to evaluate the complete behavior of the network during training. Each experiment was repeated eight times, using the same learning and testing set each time.

5. Results

5.1 Karhunen-Loève feature set

The results of this experiment are shown in table 1 and in figure 1. For each of the 2 x 8 experiments this table shows:

$e_{l,o}$ the error on the learning set directly after initialization (for initialization with the combined classifier this is the error after its training).

$e_{l,min}$ the minimum error on the learning set during training

$e_{l,e}$ the error on the learning set at the end of training

$e_{t,o}$ the error on the test set directly after initialization (for initialization with the combined classifier this is the error after its training).

$e_{t,min}$ the minimum error on the test set during training

$e_{t,e}$ the error on the test set at the end of training

This table shows convergence problems for the networks initialized with Nguyen-Widrow, 7 of the 8 networks ended training with a learning error of 90%. The other network ini-

tialized with Nguyen-Widrow ended with a learning error of 10% and had a minimal test error of 13.6%. (Other experiments reported in [3] show that networks with 10 hidden units also, but less suffer ed from convergence problems).

The test and learning errors of the network initialized with the combined classifier did reduce during training and the networks were over-trained again when training ended. This can also be seen in figure 1. The test errors first reduced and after a while they increased again. The learning errors decreased until they ended with an error of 0.0 or 0.1%. Their minimal errors on the test set varied from 5.0-5.9%. This was much better than the Nguyen-Widrow initialized networks (and also better than the results obtained from the network with 10 hidden units).

Table 1. Results for the Karhunen Loève feature set.

	initialized with Nguyen-Widrow						initialized with combined classifier					
	$e_{l,0}$	$e_{l,min}$	$e_{l,end}$	$e_{t,0}$	$e_{t,min}$	$e_{t,end}$	$e_{l,0}$	$e_{l,min}$	$e_{l,end}$	$e_{t,0}$	$e_{t,min}$	$e_{t,end}$
1	90.1	10.0	10.0	90.6	13.6	14.2	5.2	0.1	0.1	8.8	5.9	9.2
2	93.1	90.0	90.0	92.8	90.0	90.0	5.6	0.0	0.0	9.5	5.8	7.4
3	92.5	90.0	90.0	90.4	90.0	90.1	5.3	0.0	0.0	8.2	5.8	7.0
4	91.2	77.9	90.0	91.9	76.9	90.1	5.5	0.0	0.0	9.4	5.1	6.4
5	92.1	80.3	90.0	91.8	81.1	90.0	4.6	0.0	0.0	8.7	5.3	7.6
6	88.5	88.5	90.0	88.2	88.2	90.0	5.5	0.0	0.0	9.5	5.0	7.5
7	90.0	90.0	90.0	90.1	90.0	90.0	5.4	0.0	0.0	9.1	5.6	6.5
8	90.6	85.4	90.0	90.5	87.0	90.0	5.3	0.0	0.0	9.2	5.3	6.9
mean	91.0	76.5	80.0	90.8	77.1	80.5	5.3	0.0	0.0	9.1	5.4	7.3

Table 2. Results for the pixel feature set.

	initialized with Nguyen-Widrow						initialized with combined classifier					
	$e_{l,0}$	$e_{l,min}$	$e_{l,end}$	$e_{t,0}$	$e_{t,min}$	$e_{t,end}$	$e_{l,0}$	$e_{l,min}$	$e_{l,end}$	$e_{t,0}$	$e_{t,min}$	$e_{t,end}$
1	87.9	87.9	90.0	89.3	89.3	90.0	2.3	0.1	0.1	7.9	5.4	6.8
2	88.4	20.1	20.3	89.1	23.7	25.0	2.0	0.0	0.0	7.8	5.2	6.1
3	89.2	0.0	0.0	89.3	4.8	5.8	2.1	0.0	0.0	7.6	5.7	5.9
4	90.2	0.3	0.3	90.6	6.4	10.4	2.0	0.0	0.0	7.4	4.9	5.5
5	83.3	83.3	90.0	84.4	84.4	90.0	2.4	0.0	0.0	7.9	5.6	6.1
6	93.1	0.4	0.4	92.6	6.4	8.0	2.2	0.3	0.3	7.8	4.9	5.4
7	88.3	10.1	10.1	88.8	16.2	16.8	2.2	0.2	0.2	7.8	5.0	5.9
8	92.7	10.3	10.4	92.3	15.1	16.2	1.9	0.0	0.0	7.8	5.3	6.0
mean	89.1	26.6	27.7	89.5	30.8	32.8	2.1	0.1	0.1	7.7	5.2	5.9

5.2 Pixel feature set

The results of this experiment are shown in table 2. The neural networks on the pixel feature set initialized with the Nguyen-Widrow method ended twice with a learning error of 90%. It ended three times with a learning error of 0-0.4%. Their corresponding minimal test errors are the lowest minimal test errors and vary from 4.8% to 6.4%.

The learn and test errors of the networks initialized with the combined classifier reduce further during training and, again, the networks ended over-trained. The minimal test errors varied from 4.9-5.7% which was much better than the minimal test errors of the networks initialized with the Nguyen-Widrow method.

5.3 Zernike feature set

See table 3 for the results.The networks initialized with the Nguyen-Widrow method did not converge four times. Only twice a network ended with a learning error smaller than 40%, these learning errors were 2.5 and 12.5%. The first of these networks somehow managed to find a way to discriminate most of the samples of class ‘6’ from those of class ’9’,

although most features are rotational invariant. The minimal test error corresponding to the minimal learning error of 2.5% was 13.3% which is quite a good result for a rotational invariant feature set.

Interesting is that the learning error of the networks initialized with the combined classifier did not end with an error smaller than 10%, so, they did not find a way of distinguishing the samples of class `6' from those of class `9'. Apparently the networks initialized with the combined classifier did not find a global minimum for the data. The minimal test errors varied from 14.0-15.6% which was somewhat worse than 13.3%, but much better than all other results of the networks initialized with Nguyen-Widrow. The mean of the minimal test errors was 14.9%

6. Conclusions

Initialization by a combined classifier always resulted in good performing neural networks. The networks initialized with Nguyen-Widrow were occasionally somewhat better but never much. They often suffered from convergence problems leading to much worse performances.

In general it was possible to reduce the learning error and the test error by training with back-propagation, after initializing a neural network with a combined classifier. Almost all networks were over-trained at the end, but this was partly due to the deliberately long training periods.

These experiments have shown that the combined classifier can be a powerful tool for initializing neural network. Moreover, they show that further training is worthwhile resulting in an overall procedure in which first the initial layers and neurons are trained individually and then in their neural network combination.

7. References

[1]J. Kittler, M. Hatef, and R.P.W. Duin, Combining classifiers, ICPR13, Proc. 13th Int. Conf. on Pattern Recognition (Vienna, Austria, Aug.25-29) Vol. 2, Track B: Pattern Recognition and Signal Analysis, IEEE Computer Society Press, Los Alamitos, 1996, 897-901.

[2]M. van Breukelen, R.P.W. Duin, D.M.J. Tax, and J.E. den Hartog, Combining classifiers for the recognition of handwritten digits, in: P. Pudil, J. Novovicova, J. Grim (eds.), Proc. 1st International Workshop Statistical Techniques in Pattern Recognition (Prague, June 9-11, 1997), 1997, 13-18.

[3]M. van Breukelen, Improving performance by combining classifiers, Master thesis, Delft University of technology, November 1997, pp. 1-64.

[4]D.M.J. Tax, M. van Breukelen, R.P.W. Duin, and J. Kittler, Combining multiple classifiers by averaging or by multiplying?, submitted, september 1997.

[5]R.P.W. Duin, PRTools, A Matlab toolbox for pattern recognition, version 2.1, 1997,
see ftp://ph.tn.tudelft.nl/pub/bob

[6]Matlab Neural Net

[7]R.P.W. Duin and D.M.J. Tax, Classifier conditional posterior probabilities, STIPR98, submitted, November 1997

[8]T.K. Ho, J.J. Hull, and S.N. Srihari, Decision combination in multiple classifier systems, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 16, no. 1, 1994, 66-75.

[9]K. Tumer and J. Ghosh, Theoretical foundations of linear and order statistics combiners for neural pattern classifiers, IEEE Transactions on Neural Networks, March 1995.

[10]D. Nguyen, B. Widrow, Improving the learning speed of 2-layer neural networks by choosing initial values of the adaptive weights, International Joint Conf. of Neural Networks, vol. 3, pp. 21-26, July 1990.

Table 3. Results for the Zernike feature set.

	initialized with Nguyen-Widrow						initialized with combined classifier					
	$e_{l,0}$	$e_{l,min}$	$e_{l,end}$	$e_{t,0}$	$e_{t,min}$	$e_{t,end}$	$e_{l,0}$	$e_{l,min}$	$e_{l,end}$	$e_{t,0}$	$e_{t,min}$	$e_{t,end}$
1	97.3	2.5	2.5	98.2	13.3	19.3	12.8	11.9	12.7	16.4	15.4	16.2
2	89.9	40.1	40.1	89.9	41.4	41.7	13.0	12.5	13.9	16.1	15.1	16.1
3	90.3	81.4	90.0	90.9	81.0	90.1	13.5	13.1	14.1	15.6	14.9	15.3
4	87.8	12.5	12.5	87.4	22.5	26.8	13.4	13.4	14.1	16.3	14.6	15.5
5	90.2	80.6	90.0	90.0	80.4	90.1	12.8	11.1	11.4	16.8	14.4	14.8
6	91.8	80.1	90.0	92.6	80.1	90.0	13.3	12.6	13.4	15.8	14.0	14.8
7	90.1	63.3	80.2	91.5	63.2	81.0	14.0	13.0	13.3	16.2	15.6	16.7
8	90.0	90.0	90.0	90.0	90.0	90.0	12.8	12.7	13.6	16.2	15.3	16.0
mean	90.9	56.3	61.9	91.3	59.0	66.1	13.2	12.5	13.3	16.2	14.9	15.7