How parallel plug-in classifiers optimally contribute to the overall system

Wolfgang Utschick and Josef A. Nossek

Lehrstuhl für Netzwerktheorie und Schaltungstechnik
Technische Universität München
Arcisstr. 21, 80290 München

Abstract. The plug-in classification technique has been recently proposed as a new art of combining parallel classifiers. The classification of an input pattern succeeds if the output vector of all components is equal to a valid representation of the corresponding class-membership. A method is presented how each parallel component is optimally adapted for the benefit of the overall system. Instead of perfectly fitting desired target values during the training phase, the presented method performs a trade-off to the stability-plasticity dilemma of supervised learning schemes. Using the new approach the expressivity of a system for classification of handwritten characters has been improved.

Keywords: plug-in classification systems, supervised learning, optimal references, quadratic programming, character recognition.

1 Introduction

In general voting is based on combining various hypotheses as a result of multiple training runs from different random intitial conditions [1], or ones that combine multiple classifiers that have been constructed by different learning algorithms [2]. A new art of combining classifiers has been recently proposed [3]. The new approach reduces both the bias-error and the variance-error of a classification system [4]. The same training algorithm is applied to a set of different two-class classification problems which are the result of a decomposition of the original classification problem [5,3,6,7]. The set of two-class tasks implicitely corresponds to a set of binary target vectors in the layer of the classifier outputs. Each classifier is related to a single two-class problem.

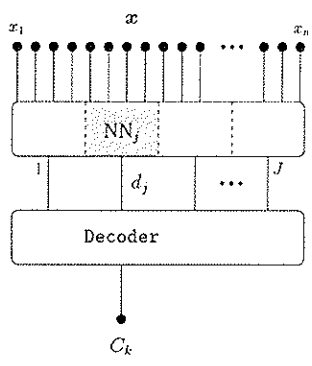

Fig. 1. A classification system of J parallel plug-in components. The decoder assigns the output vector d to one of the possible classes C_k by means of minimal distances to target vectors.

In this work, the classification system consists of J parallel neural networks and is applied to a K-class classification task [6,7] (see Fig. 1). Each neural network NN_j is

fully connected with the input vector $x \in \mathcal{R}^n$. The complete system performs a mapping from input space $\mathcal{R}^n$ into the decision space $\mathcal{R}^J$ constituted by the outputs of all parallel plug-in components NN_j. Finally, the decision rule is based on a vector quantizer approach. The decision making of the decoder is based on the minimal euclidian distance $\min_k ||d(x) - t_k||_2$ of a real-valued output vector $d \in \mathcal{R}^n$ to the binary target vectors $t_k \in \{-1, +1\}^J$ of all classes C_k, $k \in \{1, 2, \ldots, K\}$.

In a supervised training environment, the target vectors of classes provide the desired values for all network outputs according to the correct class membership of the input vectors. The classification of an input vector succeeds if the output vector is equal to a valid representation of the corresponding class. On the other hand, it is not mandatory that the output vector d perfectly fits the target vector t_k, especially not if the decision rule depends on minimal distances to target vectors. In the following, a method is proposed how simultaneous online learning of all parallel components optimally contributes to the overall system. If an output error is detected, each plug-in component is just adapted until the union of all networks significantly contributes to a correct decision of the current input vector, or in other words, until the output vector matches the so-called "optimal" target (see ahead). The presented method follows Widrow's principle of minimal disturbance [8,9] and may be discussed in terms of Grossberg's stability-plasticity dilemma [10].

2 Providing Optimal Targets

The presented training algorithm does not rely on an error-function approach. Instead of supervised training as stochastic approximation (see [11]), it is based on empirical risk minimization [12,13] by embedding of input vectors relative to decision boundaries in the domain of input patterns in order to remove the current classification error. In each training step, the algorithm adapts the synaptic weights of all neural networks until the output d_j of each component satisfies the desired output value for the given input vector [14]. In order to guarantee the optimal contribution of each plug-in component, a quadratic program is performed whenever a misclassified input vector is detected. Thereby an "optimal" target is determined which guarantees a correct classification under the constraint of Widrow's principle of minimal disturbance [8,9]. Hence, the search for the optimal target vector $t^{opt} = d + e$ is given by

$$\min_e \frac{1}{2} e^T C e \quad \text{subject to} \quad t^{opt} \in \mathcal{V}_k, \tag{1}$$

whereby $\mathcal{V}_k$ represents the corresponding volume element of the correct candidate class C_k in $\mathcal{R}^J$, constituted by the tesselation of the decision space by all target vectors and the minimal distance decision rule. A more applicable notation of the optimization problem is obtained by

$$\min_e \frac{1}{2} e^T C e \quad \text{subject to} \quad r_l - (d + e)^T n_l \leq 0 \tag{2}$$

for all $l \in \{1, 2, \ldots, K\}/\{k\}$ with $n_l := (t_k - t_l)/\|t_k - t_l\|_2$, and $r_l := ((0.5 + \eta) \cdot t_k + (0.5 - \eta) \cdot t_l)^T n_l$. Here, n_l is the normal vector on the decision boundary between the volume element $\mathcal{V}_l$ of an arbitrary target vector t_l and the volume element $\mathcal{V}_k$ of the candidate target vector t_k of the current input pattern x. The value $\eta(t_k - t_l)^T n_l$ represents the desired distance of an optimal non-binary target to the respective l-th decision boundary. It is determined by a fraction $\eta > 0$ of the distance between t_l and t_k. In the least

robust case, if $\eta := 0$, the required distance to the decision boundary vanishes. The matrix $C = C^T$ defines the metric for measuring disturbances Δt in the decision space. For the present, the matrix C is considered to be the identity matrix I. Using Lagrange multipliers $\lambda_1 \geq 0, \lambda_2 \geq 0, \ldots, \lambda_K \geq 0$, the Lagrange function of the optimization problem is given by

$$L(e, \lambda_1, \lambda_2, \ldots, \lambda_K) := \frac{1}{2} e^T C e + \sum_{\substack{l=1 \\ l \neq k}}^{K} \lambda_l \left(r_l - (d+e)^T n_l \right), \tag{3}$$

According to the Kuhn-Tucker conditions for a stationary point ($\frac{\partial}{\partial c_j} L = 0$, $\frac{\partial}{\partial \lambda_l} L = 0$) of (3), it follows

$$e := C^{-1} \sum_{\substack{\tilde{k}=1 \\ \tilde{k} \neq k}}^{K} \lambda_{\tilde{k}} n_{\tilde{k}}. \tag{4}$$

Therefore, with (4) in (3) the Lagrange function of the dual optimization problem $\max_\lambda L(\lambda)$ is equal to

$$L(\lambda) := \sum_{\substack{l=1 \\ l \neq k}}^{K} \lambda_l r_l - \frac{1}{2} \sum_{\substack{l=1 \\ l \neq k}}^{K} \sum_{\substack{\tilde{k}=1 \\ \tilde{k} \neq k}}^{K} \lambda_l \lambda_{\tilde{k}} n_l^T C^{-1} n_{\tilde{k}} - \sum_{\substack{l=1 \\ l \neq k}}^{K} \lambda_l d^T n_l, \tag{5}$$

with $\lambda_l \geq 0, l \neq k$. Any component $l \neq k$ of the gradient of the dual is given by

$$\frac{\partial L}{\partial \lambda_l} := r_l - n_l^T C^{-1} N \lambda - n_l^T d, \tag{6}$$

where $N = (n_1, \ldots, n_{k-1}, n_{k+1}, \ldots, n_K)$ is the $J \times (K-1)$ dimensional matrix of normal vectors to all decision boundaries of the volume element $\mathcal{V}_k$ and $\lambda = (\lambda_1, \ldots, \lambda_{k-1}, \lambda_{k+1}, \ldots, \lambda_K)^T$ is the $K-1$ dimensional vector of all feasible Lagrange multipliers. The solution of the optimal non-binary target vector is calculated by (4). The required Lagrange multipliers are iteratively obtained by

$$\Delta \lambda_l^{new} := \max \left(-\lambda_l^{old}, \ \sigma_l \frac{\partial L}{\partial \lambda_l^{old}} \right). \tag{7}$$

The maximum operator guarantees the consideration of the given box constraints $\lambda_l + \Delta\lambda_l \geq 0$ for all $l \neq k$. The optimal value of the step-size σ_l is derived by $\frac{dL(\sigma_l)}{d\sigma_l} = 0$ and is equal to $\sigma_l := \frac{1}{n_l^T C^{-1} n_l}$.

The solution t^{opt} of the quadratic programming is called optimal target vector of the current misclassified input pattern x. The vector t^{opt} is projected to the coordinate axes of the decision space $\mathcal{R}^J$, thus providing optimal non-binary target values $t_1^{opt}, t_2^{opt}, \ldots, t_J^{opt}$ for all parallel components.

The quadratic programming in (1) realizes the principle of minimal disturbance $\|e\|_2$ in the space of output values $\mathcal{R}^J$. However, the space of weight parameters (of NN_j) is considered to be the more relevant domain for applying the principle of minimal disturbances because all inference from the learning process is only memorized in the weights of the neural network layers. In order to apply the principle to the previous layers of the complete network, the disturbances of components in $\mathcal{R}^J$ are weighted by the factor c_{jj} which is introduced to represent the impact of possible disturbances in the decision space

to the previous layer. Therefore, a modified metric for the measure of distances is introduced in decision space. Then, the matrix C is defined by

$$c_{jj} = \left(1 + \frac{1}{\|w_j\|_2} \cdot \left\| \frac{\partial w_j}{\partial t_j} \right\|_2 \right)^2 . \tag{8}$$

The definition is derived from a local approximation $w_j + \frac{\partial w_j}{\partial t_j} \Delta t_j$ of synaptic weight parameters, whereby $\left\| w_j + \frac{\partial w_j}{\partial t_j} \Delta t_j \right\|_2 \leq \|w_j\|_2 \left\| \frac{\partial w_j}{\partial t_j} \Delta t_j \right\|_2$, according to required changes in decision space for fitting the current output vector with the desired target vector. The definition still holds for non-continuous decision functions of the individual parallel components. The value of c_{jj} is thus numerically derived from an additional previously performed training step based on the identity matrix, i.e. e and Δw_j are firstly calculated for $C = I$ and then c_{jj} is determined by

$$c_{jj} := \frac{1}{\|w_j\|_2^2} \cdot \left(\|w_j\|_2 + \left\| \frac{\Delta w_j}{\Delta t_j} \right\|_2 \right)^2 = \left(1 + \frac{1}{\|w_j\|_2} \cdot \left\| \frac{\Delta w_j}{\Delta t_j} \right\|_2 \right)^2 . \tag{9}$$

3 Theoretical Aspects

The proposed algorithm for simultaneous training of all parallel components of the system represents an optimal trade-off to the stability-plasticity dilemma [10] of neural network learning and combining. How can a learning system be stable enough to remember previously learned patterns, and yet plastic enough to learn new ones? Whereas supervised training by means of backprogagation methods answers the stability-plasticity question in terms of learning rates (see [11]), the presented approach realizes a compromise to the dilemma as follows: first, the minimal disturbance principle in the parameter space of neural components as well as the the correct embedding of the input vectors after each training step guarantee the stability of already learned responses. On the other hand, applying the principle to required changes of the output vector by searching for the optimal target vectors improves the plasticity of the overall classification system.

The improved expressivity of the classification system is also due to the shattering of the complete decision space by the volume elements of possible classes. On the contrary, for the original binary targets, the definition of volume elements $\mathcal{V}_k^{bin}$ follows the definition of the $\mathcal{V}_k$, however, r_j are zero and the normal vector $n_j \in \mathcal{R}^J$ is parallel to the j-axis of the cartesian decision space $\mathcal{R}^J$ and points into the direction of the target vector t_k. Therefore, volume elements in the binary case are only subsets of the volume elements based on the vector quantization approach. Therefore, the joint set of all volume elements $\mathcal{V}_k^{bin}$ cannot constitute a tesselation of the decision space.

A proof of convergence for the introduced algorithm is not provided. The online fashion of training and the overall use of non-continuous functions within the neural networks makes a proof practically impossible. However, if we employed an appropriate architecture of the neural plug-in classifiers, together with the designated backpropagation algorithm [11,15], the idea of a proof would be based on the different instantaneous error functions using the original binary target vectors or providing real-valued optimal targets. In both cases, under the assumption of the squared error of e, the increment or decrement of the synaptic weight parameters of a single component would be proportional to $(t_j - d_j) \cdot$

$\frac{\partial d_j}{\partial w_j}$ or $(t_j^{opt} - d_j) \cdot \frac{\partial d_j}{\partial w_j}$. Thus, for any misclassified output vector d, a feasibel robustness of $0 \leq \eta < 0.5$, and bounded output vectors, the gradients would only differ by a positive factor $\frac{t_j^{opt} - d_j}{t_j - d_j} \leq 1$. Obviously, providing optimal targets instead of original targets would preserve any convergence properties if applying a standard backpropagation algorithm.

4 Results and Conclusion

Both methods of distributing supervised training of the overall system to single plug-in components were thoroughly compared. The first approach used the original binary target vectors for the training of all components. In the second case, the presented algorithm for providing optimal targets was applied for each misclassified input pattern. The definition of target vectors $t_k \in \{-1, +1\}^{10}$ with $k \in \{1, 2, \ldots, 10\}$ was based on the one-per-class approach (1-out-of-10). The target vectors represented 10 classes of input vectors corresponding to extracted features $x \in \mathcal{R}^{40}$ or $x \in \mathcal{R}^{194}$ of 10×2000 handwritten characters.

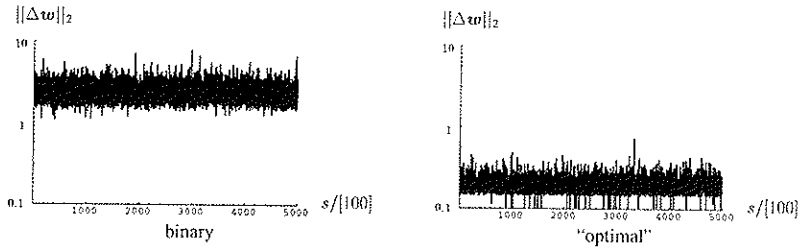

Fig. 2. Weight disturbances during the training process for binary targets by t_k and based on optimal non-binary targets, where c_{jj} is constantly equal to $c_{jj} = 1$. The distances r_l of the optimal targets from relevant decision boundaries were determined by $\eta = 0.1$.

Fig. 2 shows the disturbances of synaptic weight parameters during the training process. Providing optimal targets reduces the required changes of the weight parameters Δw about a factor 10 in each online iteration step compared with using binary targets. Hence, the new algorithm more supports the principle of minimal weight changes.

A further comparison of both methods is based on the plasticity or expressivity of the classification system applied to the given classification task. Therefore, 30 different sets of training patterns $x \in \mathcal{R}^{40}$ of handwritten characters with increasing number of elements from $370 \ldots 440$ and $540 \ldots 610$ were generated. The classifier was assembled by 10 parallel two-layer neural networks each consisting of 3 neurons in the first layer and overall 120 weight parameters. For any number of training samples the supervised training was repeated 20 times starting from different initial conditions. Using the one-per-class coding of target vectors, the capacity of both methods of providing targets is presented in Fig. 3. The capacity of the classifier was defined by $\alpha := \frac{m_{P=0.5}}{123}$. The value $m_{P=0.5}$ equals the cardinality of the set of training vectors for which a perfect solution of the complete system exists with probability of $P = 0.5$. The number 123 denotes the number of free parameters in each parallel component. The capacity of both cases is given by $\alpha_{bin} = 3.41$

and $\alpha_{opt} = 4.63$. Hence, the expressivity of the classifier is approximately $1.4\times$ larger providing non-binary targets instead of binary ones.

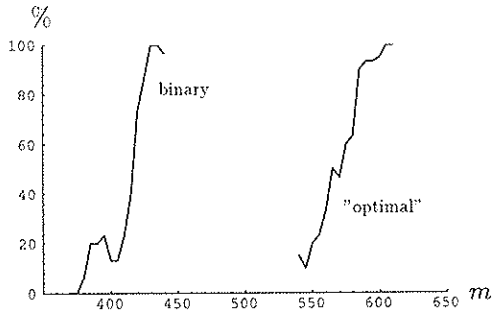

Fig. 3. Frequency of error-free training runs versus the number of elements m in the training set. The curves are based on averaged values from 20 independent training runs.

Table 1 presents a comparison based on the training results and the generalization of both approaches. The overall number of weight parameters depends from the input dimension and the system architecture and varies from 2050 to 5850. The robustness of the embedding of patterns by the online algorithm was given by $\eta = 0.25$. In the case of $n = 40$ inputs and $h = 5$ neurons, the generalization ability of providing optimal targets was superior (8.01%) to binary targets (9.61%). In the case of 5850 free parameters ($n = 194$ inputs and $h = 3$ neurons) the best result was achieved by the binary modus of providing targets (2.69%). The non-binary case was only marginally worse (2.75%) because of overtraining effects that were caused by the extended expressivity of the classifier which had been applied to a constant number of training vectors.

Table 1. Error rates (no rejection) after the training of 10×2000 training patterns (10×2000 test patterns). The system configuration is defined by $J \times K = 10 \times 10$. Each parallel component consists of $h = 5$ or $h = 3$ perceptrons of 40 or 194 inputs. The results present the averaged values of 10 independent training runs (200×1000 training epochs $\times$ iteration steps). The final error rates on the training data are given in parenthesis.

# of parameters	binary	optimal
$2050 = 10 \times 5 \times 41$	9.61% (5.98%)	8.01% (3.70%)
$5850 = 10 \times 3 \times 195$	2.69% (0.00%)	2.75% (0.00%)

Finally, Table 2 presents the number of required training steps for the proposed methods. Results are only presented for the second case of Table 1 because in the case of 40-dimensional feature vectors the training does not terminate within the range of the maximal number of training steps. The entries in Table 2 are equal to the averaged number of required iteration steps which the training algorithms takes to eliminate any error in the training set. In each step of the training process, the algorithm solves a quadratic optimization problem. Because of the simplicity of the quadratic problem in (2) the solution takes only a few vector-matrix operations in each step.

Table 2. Number of required training steps ($\times 1000$) until any error in the set of traing samples is eliminated.

Version of $2050 = 10 \times 5 \times 41$ parameters	binary	optimal
$\varnothing$ of steps	22.7	14.8

The new method of providing optimal representative targets during the supervised training of the combined system obviously improves the plasticity of the classifier. In cases where the expressivity of the system is satisfactory to the given task, the principle of minimal disturbances of synaptic weight parameters and the perfect embedding in each iteration step guarantee the generalization of the infered decision rule.

Literature

1. L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.
2. J.R. Quinlan. Combining instance-based and model-based learning. In *Proceedings of the 10th International Conference on Machine Learning*, pages 236–243,. Morgan Kaufmann, 1995.
3. J. Gareth and T. Hastie. Error coding and PaCT's, 1997. Winning paper in the ASA student paper competition for the Statistical Computing Section, available at http://playfair.Stanford.EDU/~trevor/ASA/winners.html.
4. E.B. Kong and T.G. Dietterich. Error-Correcting Output Coding Corrects Bias and Variance. In *Proceedings of the 12th International Conference on Machine Learning*, pages 313–321. Morgan Kaufmann, 1995.
5. T.G. Dietterich and G. Bakiri. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, 2:263–286, 1995.
6. W. Utschick, H.-P. Veit, and J.A. Nossek. Non-trivial codes for polychotomous classification in pattern recognition. In *Proceedings of the International Conference on Engineering Applications of Neural Networks*, pages 25–28, 1997.
7. W. Utschick. A regularization method for non-trivial codes in polychotomous classification. *International Journal of Pattern Recognition and Artificial Intelligence*, 1998. to appear.
8. B. Widrow. ADALINE and MADALINE. In *First International Conference on Neural Networks*, pages 143–158. IEEE Computer Society Press, 1987.
9. B. Widrow and M.A. Lehr. 30 years of adaptive neural networks: perceptron, madaline, and backpropagation. *Proceedings of the IEEE*, 78(9):1415–1441, 1990.
10. S. Grossberg. *Studies of Mind and Brain*. Reidel, 1982.
11. B. Kosko. *Neural networks and fuzzy systems: a dynamical systems approach to machine intelligence*. Prentice Hall, Englewood Cliffs, 1992.
12. V.N. Vapnik. *Estimation of Dependences Based on Empirical Data*. Springer, 1982.
13. V.N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, 1995.
14. W. Utschick and J.A. Nossek. Hybrid optimization of feedforward neural networks for handwritten character recognition. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, pages 147–150. IEEE Computer Society Press, 1997.
15. C. M. Bishop. *Neural Networks for Pattern Recognition*. Clarendon Press, 1995.

How parallel plug-in classifiers optimally contribute to the overall system