ESANN'2005 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 27-29 April 2005, d-side publi., ISBN 2-930307-05-6.

Applications of Multi-objective Structure Optimization

Alexander Gepperth and Stefan Roth*

Institut für Neuroinformatik, Ruhr-Universität Bochum,
Universitätsstraße 150, 44780 Bochum, Germany

Abstract. We present an application of multi-objective evolutionary optimization of feed-forward neural networks (NN) to two real world problems, car and face classification. The possibly conflicting requirements on the NN are speed and classification accuracy, both of which can enhance the embedding systems as a whole. We compare the results to the outcome of a greedy optimization heuristic (magnitude-based pruning) coupled with a multi-objective performance evaluation. For the car classification problem, magnitude-based pruning yields competitive results, whereas for the more difficult face classification, we find that the evolutionary approach to NN design is clearly preferable

1 Introduction

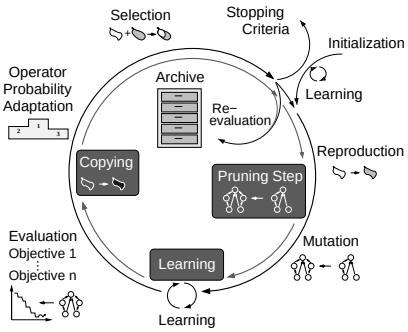
We apply two neural network (NN) optimization methods to two datasets from real-world classification problems. One method is an evolutionary multi-objective optimization (MOO) approach (see e.g. [4]), referred to as method (A). In order to assess the performance of this method, we compare the results to those of the second, greedy optimization method for NN known as magnitude-based pruning [5]. We evaluate this method, referred to as (B), in a MOO setting on an optimization problem for car classification which will be termed *car task*. The same comparison is performed on an optimization problem for face classification (denoted *face task*) [7, 8].

2 The optimization problems

The face and the car task are problems which arise in industrial applications. For a description and backgrounds of the face task we refer to [8]. The car task emerges when extending the car detection system described in [1] by a NN classifier for cars. It operates on pure back or front views of cars and allows for a more reliable traffic scene representation. The NN is trained on examples of vehicles and non-vehicles and is applied to all regions of interest (ROI) generated by an initial detection module with the purpose of rejecting ROI that do not contain cars.

In both tasks, two possibilities are suggested by application demands: reducing the classification error could save effort due to better suppression of incorrect

*email: {Alexander.Gepperth, Stefan.Roth}@neuroinformatik.rub.de. Stefan Roth is formerly mentioned in publications under his birth name Stefan Wiegand.



property	size(cars)	size(faces)	positives
D_{learn}	5000	3000	50%
D_{val}	5000	1400	50%
D_{test}	5000	2000	50%
D_{ext}	5000	2200	50%

usage	Pruning	Evolution
D_{learn}	Learning	Learning/Selection
D_{val}	Crossvalidation	Crossvalidation/Selection
D_{test}	Pareto dominace based archiving	
D_{ext}	Estimation of generalization loss	

Fig. 1: Left: Structure of the optimization loop for the hybrid evolutionary algorithm and the pruning algorithm (black labels). Right: Facts about the example datasets.

hypotheses, whereas reducing computation time would allow to apply the NN more times to neighbourhoods of ROI, thereby possibly recognizing objects that may otherwise have been missed. Of course, it is also conceivable that a compromise between these two requirements (which we expect to be at least partially conflicting) will give the best overall results; this is our motivation to improve the NN by MOO.

The two scenarios for MOO (car and face classification) are quite similar: the NN classifier receives input that is computed from a ROI. It consists of a *feature set*, an ordered collection of numbers representing key visual properties of the ROI (the computation of these properties is significantly different in both tasks). The network output is a quantity between 0.0 and 1.0, which is interpreted as a binary decision. In accordance with the procedure in [8], available examples are partitioned into four datasets termed D_{learn}, D_{val}, D_{test} and D_{ext}, see Fig. 1 (right).

In our implementation, the speed of a NN scales linearly with the number of connections n_{con}. We measure the classification error $\text{CE}(D)$ on some dataset D. The vector-valued function $f(\text{NN}) := (n_{\text{con}}(\text{NN}), \text{CE}_{\text{NN}}(D))$ is minimized.

3 Optimization methods

We investigate two optimization strategies: an evolutionary multi-objective method and magnitude-based pruning. In multi-objective optimization we approximate a set of Pareto-optimal trade-offs with respect to a number of simultaneous objectives and choose suitable solutions from this set *after* search. Evolutionary algorithms constitute established methods for the design of NN architectures (see [4] and references therein), especially recent evolutionary multi-objective approaches. The scheme used here might be regarded as a canonical evolutionary algorithm for network optimization as principally described in [7] using direct

encoding, nested gradient based learning, NSGA-II style selection [2] and online adaptation of the search strategy.

For both methods, optimization is performed iteratively in T trials, see Fig. 1 (left). These constitute independent applications of an algorithm to an initial population $\mathcal{P}^{(t=0)}$ of NN. An iteration t includes reproduction, structure adaptation and learning with cross-validation (CV), and in case of (A) subsequent performance evaluation for selecting the next parental population. Structure adaptation is subject to the constraint that at least one input neuron of a NN is connected to the output neuron. A second performance evaluation is used to update the *archive* $\mathcal{A}^{(t)}$ which represents the outcome of a trial after its completion at $t = t_{\max}$. NN learning minimizes the mean squared error $\mathrm{MSE}(D_{\mathrm{learn}})$ for 100 epochs regardless of convergence, producing 100 weight configurations. The learning algorithm is an improved variant of the Rprop algorithm [3]. Model selection is conducted by CV using $\mathrm{MSE}(D_{\mathrm{learn}})+\mathrm{MSE}(D_{\mathrm{val}})$. The evolutionary algorithm of (A) uses $(n_{\mathrm{con}}, \mathrm{CE}(D_{\mathrm{learn}} \cup D_{\mathrm{val}}))$ for selecting the next parental population. In both methods, $(n_{\mathrm{con}}, \mathrm{CE}(D_{\mathrm{test}}))$ is used for updating $\mathcal{A}^{(t)}$. For a synopsis see Fig. 1 (left).

Due to the requirement of reducing the number of connections instead of nodes, method (A) differs slightly from the algorithm presented in [7]. We use different operators for the insertion and deletion of connections: the number of connections which are inserted or deleted depends linearly on the total amount of connections in the network (factors of proportionality are 0.05 for the deletion and 0.01 for the insertion of connections). There is no operator for the deletion of hidden nodes; this happens only when nodes no longer have any ingoing or outgoing connections. The operators that add or delete whole receptive fields are not used in the car task, and operators that jog weights are also omitted. EP-tournament selection is implemented on the basis of the NSGA-II style ordering of the population.

Weight elimination in method (B) is applied identically in both tasks: a percentage p of connections with the largest absolute weight is eliminated at each iteration. Reproduction simply copies the current population.

4 Multi-objective performance assessment

To compare the multi-objective results produced by two methods, we follow the procedures summarized in [7]: For each NN, an objective vector $\boldsymbol{z}$ consists of the qualities computed singly according to all objectives. We do not impose a straightforward way to compare two arbitrary objective vectors as we could do e.g. by defining a scalar quality measure since this determines a prior trade-off between objectives. The space of all objective vectors is referred to as *objective space* $\mathcal{O}$; its elements are partially ordered by the dominance relation $\succ$ ($\boldsymbol{z}$ dominates $\boldsymbol{z'}$) defined as

$$\boldsymbol{z} \succ \boldsymbol{z'} \in \mathbb{R}^n \quad \Leftrightarrow \quad \forall\, 1 \leq i \leq n:\ z_i \leq z'_i \ \wedge\ \exists\, 1 \leq j \leq n:\ z_j < z'_j\ . \qquad (1)$$

The *Pareto front* of a set $M \subseteq \mathcal{O}$, denoted P_M, is then defined as the subset of elements of M which are not dominated by any other element of M. We

characterize the outcome of an optimization trial by the Pareto front $\mathcal{A}^{(t=t_{\max})}$ using $\mathcal{A}^{(t)} := P_{\mathcal{A}^{(t-1)} \cup \mathcal{P}^{(t)}}$ with $\mathcal{A}^{(0)} = \emptyset$. A vector $z \in \mathcal{O}$ *dominates* a vector $\boldsymbol{z'} \in \mathcal{O}$ *weakly* ($z \succcurlyeq \boldsymbol{z'}$) iff z is not worse than $\boldsymbol{z'}$ in all objectives. Given two sets X and Y, *weak dominance* of sets ($X \rhd Y$) is defined as

$$X \rhd Y \text{ iff } X \neq Y \text{ and } \forall\, \boldsymbol{y} \in Y : \exists\, \boldsymbol{x} \in X : \ \boldsymbol{y} \text{ is weakly dominated by } \boldsymbol{x} \ . \quad (2)$$

The *hypervolume indicator* H_X [9] measures the percentage of objective space weakly dominated by X. Other performance indicators measure how likely the outcome X_i of a trial i is to weakly dominate any outcome Y_j of another trial, or to be incomparable to it. Let $V \subseteq \mathcal{O}$ be the smallest cuboid enclosing all objective vectors and m a measure. Performance indicators are defined as

$$\mathcal{P}_{X_i \rhd Y} := |\{(X_i, Y_j) : X_i \rhd Y_j,\ 1 \leq j \leq T\}| \cdot {}^1\!/_T \ , \quad (3)$$

$$\mathcal{P}_{X_i || Y} := |\{(X_i, Y_j) : X_i \not\rhd Y_j \wedge Y_j \not\rhd X_i,\ 1 \leq j \leq T\}| \cdot {}^1\!/_T \quad (4)$$

$$H_X := \left\{ m(\{z \in V | \exists z' \in X : z' \succcurlyeq z\})/m(V) \right\} \in [0,1]. \quad (5)$$

In the following, A_i and B_i, $1 \leq i \leq T$ will always be used for trial outcomes using method (A) and (B) respectively. For the purposes of comparison, we calculate the quantities H_{A_i}, H_{B_i}, $\mathcal{P}_{A_i||B}$, $\mathcal{P}_{B_i||A}$, $\mathcal{P}_{A_i \rhd B}$ and $\mathcal{P}_{B_i \rhd A}$, their median and median absolute deviation (mad).

5 Experimental setup

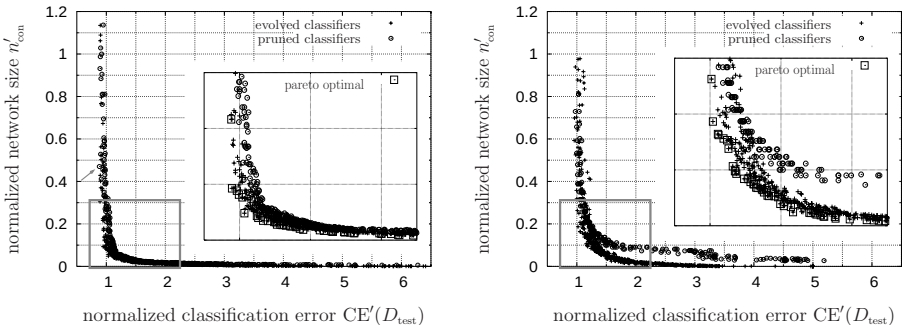
The following statements hold for methods (A) and (B): All NN have one hidden layer, activation functions are of logistic sigmoidal type. We simulate $T = 10$ trials. For each trial we set $|\mathcal{P}^{(t=0)}| = 25$. In the car task, each NN in $\mathcal{P}^{(t=0)}$ is fully connected, has between 20 and 25 neurons in its hidden layer and all forward-shortcuts and bias connections in place. At each iteration t, $\mathcal{P}^{(t)}$ is initialized with small random weight values between -0.05 and 0.05. We refer to this architecture as the *car reference topology*. In the face task, $\mathcal{P}^{(t=0)}$ is instantiated with 25 copies of the 400-52-1 architecture of [6], the *face reference topology*, each of which is randomly initialized like the car reference topology at $t = 0$.[1] All trials of method (B) are performed for 90 iterations at $p = 10\%$.[2] All method (A) trials are performed for 200 iterations.

We train the car and the face reference topologies 100 times for 2000 iterations using the improved Rprop learning procedure on D_{learn} and select the network a_{ref} with the smallest classification error $\text{CE}(D_{\text{val}} \cup D_{\text{test}})$. In the following, all results are normalized by the performance of a_{ref}.[3] For example, the normalized classification error of a NN a is given by $\text{CE}'_a(D) = \text{CE}_a(D)/\text{CE}_{a_{\text{ref}}}(D)$ and the normalized number of connections by $n'_{\text{con}}(a) = n_{\text{con}}(a)/n_{\text{con}}(a_{\text{ref}})$.

[1] For $t > 0$ the weight values of the predecessor are used for initialization prior to variation.

[2] No regular NN were ever produced afterwards. Reducing the pruning percentage p to the point where 200 valid NN iterations could be produced did not change results.

[3] Keep in mind that the reference topologies are not arbitrary, but tuned extensively by hand. They produce results that are highly competitive to other approaches in the literature.



performance indicator ($i = 1 \dots T$)	median ± mad
$\{H_{A_i}\}$	0.991±0.002
$\{H_{B_i}\}$	0.982±0.001
$\{\mathcal{P}_{A_i \triangleright B}\}$	0.1±0.15
$\{\mathcal{P}_{A_i \| B}\}$	0.9±0.15
$\{\mathcal{P}_{B_i \triangleright A}\}$	0.±0.0
$\{\mathcal{P}_{B_i \| A}\}$	0.8±0.15

performance indicator ($i = 1 \dots T$)	median ± mad
$\{H_{A_i}\}$	0.958±0.012
$\{H_{B_i}\}$	0.905±0.001
$\{\mathcal{P}_{A_i \triangleright B}\}$	0.15±0.222
$\{\mathcal{P}_{A_i \| B}\}$	0.85±0.222
$\{\mathcal{P}_{B_i \triangleright A}\}$	0.0±0.0
$\{\mathcal{P}_{B_i \| A}\}$	0.7±0.148

Fig. 2: Left: results from the car task. Right: results from the face task. Shown on top are the unions of all trial outcomes; members of their Pareto fronts, called *meta Pareto fronts*, are shown in the magnifications. The only pruned NN in the meta Pareto front of the car task is indicated by an arrow. The performance indicators (tables at the bottom) are explained in the text.

6 Results

The normalized results of the car and the face task are shown in Fig. 2. One perceives the surprisingly similar performance of the two methods when applied to cars. While the evolutionary method performs better, the differences are small and the errors of the generated NN are similar in similar regimes of n_{con}. In contrast, the differences between the two methods are quite pronounced when applied to the face task: here, the evolutionary MOO is clearly superior. In both tasks, the distributions of the H_{A_i} and the H_{B_j} differ in a statistically significant way.[4] All results persist when considering $(n_{\text{con}}, \text{CE}(D_{\text{ext}}))$ instead of $(n_{\text{con}}, \text{CE}(D_{\text{test}}))$, showing that no significant overfitting has occured.

7 Discussion

We interpret the result of the car task as an indication that the problem class is intrinsically easier[5] than the face task. Therefore the simpler optimization method can yield competitive performance. For the more difficult face task, a

[4] Wilcoxon Rank Sum Test, $p < 0.001$

[5] w.r.t. the magnitude of the classification error of the best conceivable NN.

sophisticated (here: evolutionary) optimization strategy is clearly favorable.

For the support of our interpretation about the difficulty of both tasks, we observe that the (absolute) error $\mathrm{CE}(D_{\mathrm{test}})$ of the car reference topology is 3.5 times smaller than $\mathrm{CE}(D_{\mathrm{test}})$ of the face reference topology. As the results plainly show, optimization is unable to improve classification accuracy greatly compared to the reference topologies which constitute approximate optima in this respect. Therefore this difference in classification errors should be considered meaningful. Furthermore, optimization in the car task produced NN without a hidden layer which nevertheless had an (absolute) classification accuracy of about 80%. We take this as a hint that the problem is almost linearly separable and therefore can be considered "easy".

The embedding of structure optimization within an evolutionary MOO setting leads to a notable advantage compared to the single-objective formulation of the problem [7, 8]. There, a certain trade-off between partially conflicting objectives must be determined prior to search, thus disregarding certain types of solutions. In MOO, the best attainable set of incomparable solutions is generated first, and choice of specific solutions happens afterwards. We have demonstrated that simple structure optimization heuristics like pruning can easily be incorporated in the framework of MOO. While this does not improve the optimization results themselves, one can profit from the advantages of MOO that were discussed previously.

References

[1] T. Bücher, C. Curio, H. Edelbrunner, C. Igel, D. Kastrup, I. Leefken, G. Lorenz, A. Steinhage, and W. von Seelen. Image processing and behaviour planning for intelligent vehicles. *IEEE Transactions on Industrial Electronics*, 90(1):62–75, 2003.

[2] K. Deb. *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons, Chichester, UK, 2001.

[3] C. Igel and M. Hüsken. Empirical evaluation of the improved Rprop learning algorithm. *Neurocomputing*, 50(C):105–123, 2003.

[4] C. Igel and B. Sendhoff. Evolutionary framework for the construction of diverse hybrid ensembles. In *13th European Symposium on Artificial Neural Networks (ESANN 2005)*, 2005.

[5] R. D. Reed and R. J. Marks II. *Neural Smithing*. MIT Press, 1999.

[6] H. A. Rowley, S. Baluja, and T. Kanade. Neural network-based face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(1):23–38, 1998.

[7] S. Wiegand, C. Igel, and U. Handmann. Evolutionary multi-objective optimisation of neural networks for face detection. *International Journal of Computational Intelligence and Applications*, 4(3):237–253, 2004.

[8] S. Wiegand, C. Igel, and U. Handmann. Evolutionary optimization of neural networks for face detection. In M. Verleysen, editor, *12th European Symposium on Artificial Neural Networks (ESANN 2004)*, pages 139–144. Evere, Belgium: d-side publications, 2004.

[9] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. da Fonseca. Performance assessment of multiobjective optimizers: An analysis and review. *IEEE Transactions on Evolutionary Computation*, 7(2):117–132, 2003.