Graph Isomorphisms Effect Structure Optimization of Neural Networks

Christian Igel
Institut für Neuroinformatik, Ruhr-Universität Bochum
44780 Bochum, Germany
Christian.Igel@neuroinformatik.ruhr-uni-bochum.de

Peter Stagge
Thinking Networks SI GmbH
Konrad-Zuse-Straße 4, 44781 Bochum, Germany
Peter.Stagge@thinking-networks.com

Abstract— Concepts from graph theory and molecular evolution are proposed for analyzing effects of redundancy induced by graph isomorphisms on structure optimization of neural networks. It is demonstrated that a graph database that considers isomorphisms can drastically reduce the number of evaluations in an evolutionary structure optimization process.

I. INTRODUCTION

The topology of a multi-layer perceptron network is crucial for its performance. The model selection process, defined here as the search for a suitable network structure, sets up a hard discrete optimization task, which is tackled successfully by means of global random search methods, in particular evolutionary algorithms (EAs) [1]. Usually, the search space in the model selection process consists of a subset of graphs (e.g., encoded by adjacency matrices) that are regarded as feasible networks. Considering graphs, it arises that there are isomorphic topologies capable of the same functions and having the same learning and generalization properties. Therefore, we identify each model (or structure) with an equivalence class of isomorphic network graphs (i.e., network topologies). As the search for suitable models is performed in a space of topologies, where different topologies can map to the same model, the search space is redundant. In this article, we point out some effects of this redundancy on the search process. We summarize, generalize, and extend previous work by the authors [2], [3], [4] that has been presented solely to the evolutionary computation community. In particular, we present new results showing that a graph database that considers graph isomorphisms can drastically reduce the number of evaluations in evolutionary structure optimization.

II. NEURAL NETWORKS AS GRAPHS

In this investigation, we concentrate on feed-forward neural networks (NNs). However, all concepts can be extended to recurrent systems, cf. [2]. Consider a network $\mathcal{N}$ consisting of n neurons $v_1, \dots, v_n$. Without loss of generality, we assume that $v_1, \dots, v_d$ are d external input neurons and that $v_{n-m+1}, \dots, v_n$ are m external output neurons. Additionally, there is an extra unit v_0 with constant output for the realization of bias (threshold) parameters. The topology of the NN can be described by a directed graph $G_\mathcal{N} = (V_\mathcal{N}, E_\mathcal{N})$ with vertices $V_\mathcal{N} = \{v_0, \dots, v_n\}$. There exists an edge $(v_j, v_i) \in E_\mathcal{N}$ iff the neuron v_i gets input from neuron v_j.

In the following, we make use of some concepts from graph theory, which are explained in [5], [6]. We associate a *coloring* $\pi_\mathcal{N}$ with each network graph. In general, a *coloring* π denotes an ordered partition of a vertex set V into disjoint, non-empty color classes. For example, for k different colors $\pi = (V_1, V_2, \dots, V_k)$. The order of the color classes is significant, but the order of the vertices within each class is not. A *color-preserving permutation* of a graph is a permutation of its vertices, where each vertex is mapped to a vertex of the same color. For a given network $\mathcal{N}$ we define

$$\pi_\mathcal{N} = (\{v_0\}, \{v_1\}, \dots, \{v_d\}, \{v_{d+1}, \dots, v_{n-m}\}, \\ \{v_{n-m+1}\}, \{v_{n-m+2}\}, \dots, \{v_n\}) \enspace .$$

It holds:

Given two feed-forward neural networks $\mathcal{N}_1$ and $\mathcal{N}_2$, the following statements are equivalent:
1. $\mathcal{N}_1$ and $\mathcal{N}_2$ are isomorphic, i.e., they are equivalent concerning functionality, learning, and generalization and therefore describe the same model (assuming that the learning process is invariant under color-preserving permutations).
2. The colored graphs $G_{\mathcal{N}_1}$ and $G_{\mathcal{N}_2}$ with colorings $\pi_{\mathcal{N}_1}$ and $\pi_{\mathcal{N}_2}$, respectively, are isomorphic.
3. There is a color-preserving permutation that transforms $G_{\mathcal{N}_1}$ into $G_{\mathcal{N}_2}$.
4. It holds $\mathcal{C}(G_{\mathcal{N}_1}, \pi_{\mathcal{N}_1}) = \mathcal{C}(G_{\mathcal{N}_2}, \pi_{\mathcal{N}_2})$ for some canonical labeling map $\mathcal{C}$.

Roughly speaking, given a colored graph G, a canonical labeling map $\mathcal{C}$ maps G and all graphs isomorphic to G to a (canonical) graph that is isomorphic to G, see [6], [2], [3], [4] for details. The colorings of the networks are necessary to fix the input and output nodes. If different types of neurons are encoded, the coloring can also be used to group these neurons. For example, when a *sigma-pi* network is described, coloring can prevent from

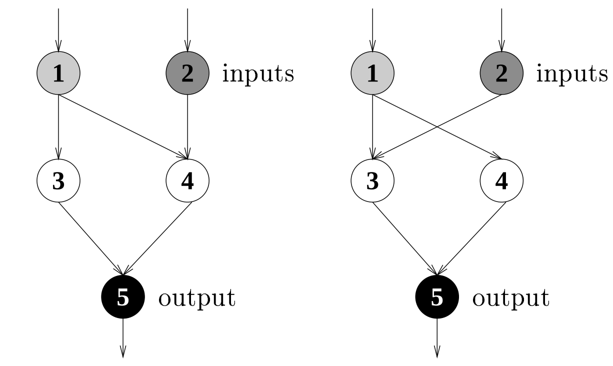

Fig. 1. Two isomorphic networks with colored input and output neurons, from [4].

mapping a sigma unit to a pi unit and vice versa. See Fig. 1 for an example of two colored isomorphic networks.

For the analysis of the effects of isomorphic topologies on model selection, we need a way to calculate the number of different isomorphic networks for a given NN. As explained in [3], [4] it holds:

The number of different adjacency matrices that code for graphs that are isomorphic to G is given by

$$\frac{1}{a(G)} \prod_{i=1}^{k} |V_i|! \ , \tag{1}$$

where $a(G)$ is the size of the automorphism group *of G.*

An *automorphism* of a graph G is a color-preserving permutation of the vertices that maps G onto itself. The size of the automorphism group of a graph G denoted by $a(G)$ is defined as the number of automorphisms of G.

The general graph isomorphism problem and therefore the canonical labeling is a potential member of the problem class of so called NP-incomplete problems [7]. However, efficient heuristic algorithms exist that can handle the canonical labeling for graphs representing NNs that can be regarded as tractable with today's learning algorithms. Hence, the costs of computing their canonical representation can almost be neglected compared to their learning time.

III. GLOBAL SEARCH FOR NEURAL NETWORK TOPOLOGIES AND REDUNDANCY

Apart from greedy heuristics like pruning or constructive methods (e.g., see [8] chapters 12 & 13), global random search algorithms are the state-of-the-art choice for model selection of NNs [1]. Such methods sample the new points according to a probability distribution on the search space, which is adapted based on the explored points. Although we restricted our experiments to evolutionary optimization, most of our results also apply to other global random search algorithms.

Let $\mathcal{G}$ denote a search space and $\Phi : \mathcal{G} \to \mathbb{R}$ a quality function. In case of NN structure optimization, Φ usually corresponds to the performance of the NN after learning. Let Γ be the function that maps an encoded network $g \in \mathcal{G}$ to the corresponding model (i.e., class of isomorphic network topologies), and let $\mathcal{S} = \{\Gamma(g) \mid g \in \mathcal{G}\}$ be the set of all models accessible by the search algorithm. When standard encodings of network topologies are used, Γ is not injective due to isomorphic graphs.

In the framework of biological molecular evolution, the effects of redundancy have been studied by Schuster and co-workers, cf. [9]. We adopt their way of analysis for our study of structure optimization of NNs.

The redundancy $r(s)$ *of a model $s \in \mathcal{S}$ is the cardinality of its pre-image in the search space given by*

$$r(s) = |\{g \in \mathcal{G} \mid \Gamma(g) = s\}| \ .$$

The average redundancy of the encoding *is defined as*

$$\bar{r} = \frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} r(s) = \frac{|\mathcal{G}|}{|\mathcal{S}|} \ .$$

A model (phenotype) $s \in \mathcal{S}$ is called frequent *if its redundancy is above the average redundancy, $r(s) > \bar{r}$. A topology (genotype) $g \in \mathcal{G}$ is called frequent if $\Gamma(g)$ is a frequent model. A model or topology that is not frequent is* rare*.*

The fraction of frequent models $h_{\mathcal{S}}$ *and the* fraction of frequent topologies *(network graphs) $h_{\mathcal{G}}$ can be defined as*

$$h_{\mathcal{S}} = \frac{|\{s \in \mathcal{S} \mid r(s) > \bar{r}\}|}{|\mathcal{S}|}$$

and

$$h_{\mathcal{G}} = \frac{|\{g \in \mathcal{G} \mid r(\Gamma(g)) > \bar{r}\}|}{|\mathcal{G}|} \ .$$

Redundancy can be beneficial or problematic for the model selection process. Especially the positive aspects have raised a lot of interest among researcher in the field of natural and artificial evolution. For example, random drift on *neutral networks* [9], [10], [11] can be useful for leaving local optima and increasing the accessibility of phenotypes, i.e., models in our context. However, in this study we concentrate on redundancy in structure optimization stemming from graph isomorphism and assume that the (evolutionary) search is based on standard variation operators like addition or deletion of a connection or node. With respect to such operators, isomorphic graphs do not form neutral networks, and, more fundamental, have the same neighborhood in the structure space $\mathcal{S}$. Hence, important requirements for typical scenarios where redundancy can be beneficial are not fulfilled, cf. [12].

IV. RECENT FINDINGS

A. Different Codings Lead to Different Distributions of Redundancies

In [3], [4] it has been shown that the distribution of rare and frequent structures strongly depends on the encoding used. In this section, we consider feed-forward NNs without layer restriction as an example and study two very common encoding schemes. In one scheme, the search space consists of lower-triangular adjacency matrices coding for feasible NNs, in the other one all matrices representing feed-forward NNs are considered, i.e., the first search space is a subset of the second. We performed generalized random walks to estimate the distributions of rare and frequent structures for both encodings. We pursued random walks with different numbers of hidden neurons and fixed the number of hidden units l within each walk. There was a single output neuron and the number of input neurons d was set to $d = 4$ for all different values of l.

The results are shown in Fig. 2. In case of the triangular-matrix encoding, for increasing l the fraction of frequent models decreases and the fraction of frequent topologies increases. For $l = 15$, more than 80 % of the search space codes only for less than 20 % of the models. The results for the encoding via a full connection matrix show a completely different behavior. Nearly all models and all topologies are frequent. These results reveal that the choice of the encoding for a structure optimization task may have drastic side effects as it can introduce a strong bias in the structure space.

For example, a *fully connected* NN with one hidden layer and l hidden units is a topology that is represented only once irrespective of the number of hidden neurons; on the other hand, there are a lot of topologies with one hidden layer and l hidden units with redundancy $l!$ given that the number of input neurons d is not too small. A model selection process may thus favor specific models $s \in \mathcal{S}$ with high $r(s)$. For NNs this may also imply a bias towards specific learning and generalization characteristics because these features depend on the topology of the NN.

Different coding schemes generally result in different characteristics as is evident in our examples. The encoding, together with the search operators, determines the success of an evolutionary setup, thus it is important to know the implications of a specific setting. Based on these findings, we propose that the search bias stemming from rare and frequent structures should be removed (see section IV-C), unless there is *a priori* knowledge that it improves the search process.

B. A Solution to the Competing Conventions Problem

For a fixed canonical labeling map $\mathcal{C}$ and for each network $\mathcal{N}$ there exists exactly one isomorphic *canonical*

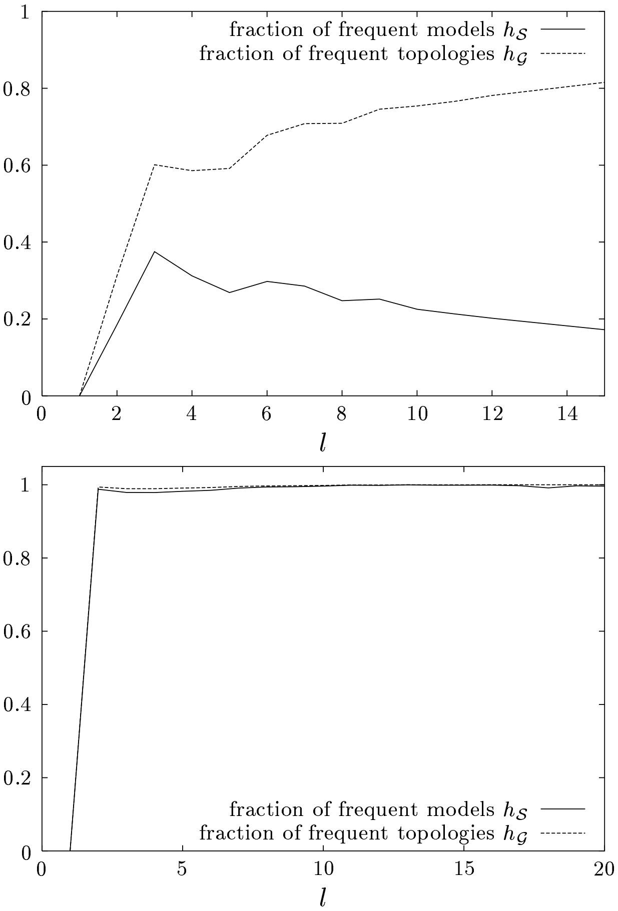

Fig. 2. Percentages of frequent models and frequent topologies for different numbers of hidden neurons l. The upper plot shows the results for the lower-triangular encoding, the lower one for the full-matrix encoding.

network that is uniquely defined by the graph $\mathcal{C}(\mathcal{G}_{\mathcal{N}}, \pi_{\mathcal{N}})$. This fact can be regarded as a way to solve the *competing conventions* problem, also called permutation problem [13], [14], [15], [16], [17], [18]. That is, many genotypes (i.e., elements of the search space) map to equivalent NNs, i.e., the corresponding network topologies (conventions) are isomorphic. It was argued that this has a harmful effect on evolutionary optimization, especially on recombination, as the outcome of combining two models depends on the very topologies they are represented by. If only canonically labeled networks are stored in the population, there is exactly one representation of each structure and the competing conventions problem taken literally can be regarded as solved. This requires that each topology which is created by the search operators is replaced by the genotype belonging to the canonical representative of the NN. Although this isomorphism-free representation ensures a single "convention", it does not necessarily lead to something like a "homologous" crossover between NNs.

C. Distribution of Redundancies Influences Search

In this section, we demonstrate that the presence of rare and frequent models is not only of theoretical interest, but considerably influence the model selection process. In order to remove the bias stemming from isomorphic network topologies, the probability $p(g)$ that a variation operator generates an offspring $g \in \mathcal{G}$ can explicitly consider the redundancy of g, e.g.,

$$p(g) \propto \frac{1}{r(\Gamma(g))} \quad .$$

Such an operator can increases the performance of the search process, because there may be well performing rare structures that are not likely to be detected by an operator which does not take the redundancies into account and therefore tends to explore mostly frequent structures.

We compared the performance of an EA (similar to the one presented in the following section) with standard operators and with the corresponding bias-free operators. The task was to evolve a structure which was known to be comparatively rare. It turns out that the considered task is solved more than three times more often with the fair operator being used than with the standard operator [4].

V. GRAPH DATABASE

In the following, we describe another application of the introduced concepts, which reduces the computational complexity of the structure optimization process.

A. Graph Database

During topology optimization, each generated NN can be stored in a database. If the same network is sampled again from the search distribution, there is no need to determine its quality, because the outcome of the time consuming learning process (or even several learning processes, cf. [19]) can be looked up as proposed in [20]. In [2] we presented the idea to store and look up the canonically labeled graphs in the database. As isomorphic networks stand for the same model, the quality of isomorphic NNs is the same. Before the quality of a new individual is determined, it can be tested whether this network or an isomorphic one was evaluated before by looking up the canonically labeled graph in the database. In that way, more evaluations can be saved. The percentage of fitness evaluations that can be saved depends on the search method used and especially on the search space.

B. Example Problem

The task is to find a NN with as few as possible weights that solves the classification task given by the Boolean function $f : \{0,1\}^3 \rightarrow \{0,1\}$ defined by

$$f(x_1, x_2, x_3) = x_1 x_2 x_3 \vee \overline{x}_1 \overline{x}_2 \overline{x}_3 \quad .$$

The quality function is given by

$$\Phi(\mathcal{N}) = \nu \cdot \mathcal{E}_{\text{class}}(\mathcal{N}) + |E_{\mathcal{N}}| + \mathcal{E}_{\text{mse}}(\mathcal{N}) \quad ,$$

where $\mathcal{E}_{\text{class}}$ is the classification error and $\mathcal{E}_{\text{mse}}$ is the mean squared error. The number of edges $|E_{\mathcal{N}}|$ corresponds to the degrees of freedom of the model. The weighting factor ν is set to 10^6 in our experiments, hence $\mathcal{E}_{\text{class}}(\mathcal{N}) \gg |E_{\mathcal{N}}| \gg \mathcal{E}_{\text{mse}}(\mathcal{N})$.

Two hidden neurons (corresponding to two decision boundaries) getting input from all input nodes are sufficient to solve this problem. Therefore, the task is regarded as being solved iff the overall error drops below 9, i.e., all input patterns are classified correctly and there are only 8 connections. This problem has been chosen because we know in advance that the desired structure is not very frequent [4].

We use a simple EA, which is designed to be instructive and not to be as efficient as possible, to find a suitable NN topology. The number of hidden neurons is fixed to 5, only one hidden layer is allowed. Every genotype, i.e., encoded topology, can be regarded as a bit vector of length $\ell =$ (number of input neurons+1)$\cdot$(number of hidden neurons+1). The bits indicate whether (i) a hidden neuron is connected to an input, (ii) a hidden or the output neuron utilizes a bias parameter, and (iii) a shortcut connection exists. Additionally, for each connection the corresponding weight value is stored. A *bitflip* operator that changes the status of each connection from present to not present and vice versa with probability $p_{\text{flip}} = 1/\ell$ is employed[1]. The connections from the hidden neurons to the output are added and removed automatically: a hidden neuron is connected with the external output iff it has at least one connection to an external input. This encoding allows for a search without a bias in $\mathcal{G}$. The population is initialized with topologies having a random number of 15 to 35 connections. The population consists of 20 individuals and EP-tournament selection is used [21]; evolution is stopped after 200 generations. Each newly created individual is trained for 50 cycles using an improved version of the Rprop learning algorithm [22]. We would like to stress that the evolutionary algorithm is not intended to be efficient, but to be as simple and instructive as possible.

We store all NNs created in two graph databases, with and without transforming them into a canonical graph first, i.e., for each newly created network $\mathcal{N}$ the graph $G_{\mathcal{N}}$ is stored in one database and $\mathcal{C}(G_{\mathcal{N}_1}, \pi_{\mathcal{N}_1})$ in the other. The basic scheme of the evolutionary algorithm is depicted in Fig. 3.

[1] This operator does not alter the individual with a probability of $(1-1/\ell)^\ell$, which is approximately $e^{-1} \approx 0.37$ for large ℓ. However, this does not introduce a bias in our investigation.

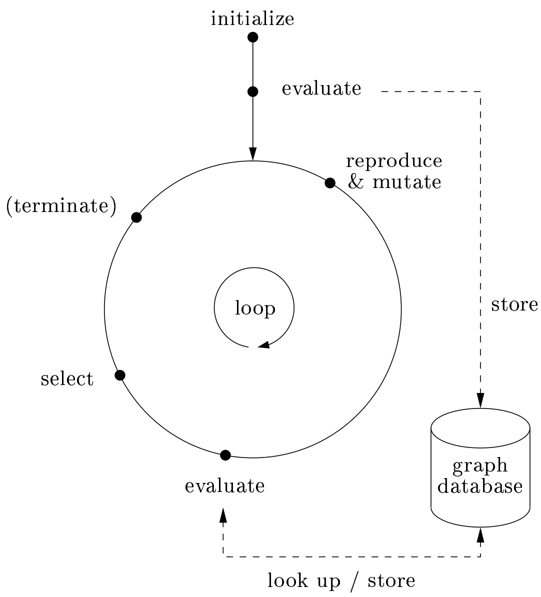

Fig. 3. Loop of the evolutionary algorithm using a graph database.

C. Results

We conducted 25 independent trials. On average, there were ≈ 1426 different entries in the standard database, and only ≈ 1124 in the one that stores the canonical representatives. As each entry corresponds to one network evaluation, this implies that more than 21 % evaluations could additionally be saved. This is in contrast to the findings in [2], where a very different EA for structure optimization of recurrent networks was used and only less than 3 % of the evaluations could additionally be saved.

VI. Conclusion

We have shown that redundancy induced by graph isomorphism has a remarkable effect on the search space in NN model selection: In particular, rare and frequent structures can be observed—similar to findings in natural evolution. Hence, selected models may be biased towards frequent structures inheriting their specific approximation, learning, and generalization properties. A method to deal with these effects of redundancy has been proposed, which is based on concepts from graph theory and molecular biology. When analyzing NN search spaces, it turns out that the distribution of rare and frequent structures crucially depends on the encoding. It has been shown that a considered choice of the encoding with respect to redundancy can be beneficial for the search process. Further, we have demonstrated the use of a graph database that takes graph isomorphisms into account and thereby drastically decreases the number of evaluations needed in the model selection process.

Acknowledgments

Christian Igel acknowledges support by the DFG under grant Solesys SE 251/41-1.

References

[1] X. Yao, "Evolving artificial neural networks," *Proc. of the IEEE*, vol. 87, no. 9, pp. 1423–1447, 1999.

[2] P. Stagge and C. Igel, "Structure optimization and isomorphisms," in *Theoretical Aspects of Evolutionary Computing*, L. Kallel, B. Naudts, and A. Rogers, Eds., Natural Computing, pp. 409–422. Springer-Verlag, 2001.

[3] P. Stagge and C. Igel, "Neural network structures and isomorphisms: Random walk characteristics of the search space," in *IEEE Symposium on Combinations of Evolutionary Computation and Neural Networks (ECNN 2000)*, X. Yao and D. B. Fogel, Eds. 2000, pp. 82–90, IEEE Press.

[4] C. Igel and P. Stagge, "Effects of phenotypic redundancy in structure optimization," *IEEE Trans. on Evolutionary Computation*, 2002, in press.

[5] B. D. McKay, "Practical graph isomorphism," *Congressus Numerantium*, vol. 30, no. 30, pp. 47–87, 1981.

[6] B. D. McKay, "*nauty* user's guide (version 1.5)," Tech. Rep. TR-CS-90-02, Australian National University, Computer Science Department, 1990.

[7] M. R. Garey and D. S. Johnson, *Computers and Intractability, A Guide to the Theory of NP-Completeness*, W. H. Freeman and Company, New York, 1979.

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

[9] P. Schuster, "Molecular insights into evolution of phenotypes," in *Evolutionary Dynamics—Exploring the Interplay of Accident, Selection, Neutrality and Function*, J. P. Crutchfield and P. Schuster, Eds., Santa Fe Institute Series in the Science of Complexity. Oxford Univ. Press, 2002.

[10] M. Conrad, "The geometry of evolution," *Biosystems*, vol. 24, pp. 61–81, 1990.

[11] M. Kimura, "Evolutionary rate at the molecular level," *Nature*, vol. 217, pp. 624–626, 1968.

[12] M. Toussaint and C. Igel, "Neutrality: A necessity for self-adaptation," in *Congress on Evolutionary Computation (CEC 2002)*, 2002.

[13] R. K. Belew, J. McInerney, and N. N. Schraudolph, "Evolving networks: Using genetic algorithm with connectionist learning," Tech. Rep. CS90-174 (revised), Computer Science & Engg. Dept., Univ. of California at San Diego, USA, 1991.

[14] J. D. Schaffer, D. Whitley, and L. J. Eshelman, "Combinations of genetic algorithms and neural networks: A survey of the state of the art," in *International Workshop on Combinations of Genetic Algorithms and Neural Networks (COGANN-92)*, L. Darrell Whitly and J. David Schaffer, Eds., 1992, pp. 1–37.

[15] P. J. B. Hancock, "Genetic algorithms and permutation problems: A comparison of recombination operators for neural net structure specification," in *International Workshop on Combinations of Genetic Algorithms and Neural Networks (COGANN-92)*, L. Darrell Whitley and J. D. Schaffer, Eds. 1992, pp. 108–122, IEEE Press.

[16] D. Whitley, "Genetic algorithms and neural networks," in *Genetic Algorithms in Engineering and Computer Science*, J. Periaux and G. Winter, Eds., chapter 11. John Wiley & Sons, 1995.

[17] X. Yao and Y. Liu, "Evolving artificial neural networks for medical applications," in *Proc. of the Australian-Korea Joint Workshop on Evolutionary Computation*, Taejon, Korea, 1995, pp. 1–16.

[18] D. Thierens, "Non-redundant genetic coding of neural networks," in *Proc. of the 1996 IEEE International Conference on Evolutionary Computation*. 1996, pp. 571–575, IEEE Press.

[19] P. Stagge, "Averaging efficiently in the presence of noise," in *Parallel Problem Solving from Nature (PPSN V)*, A. E. Eiben, T. Bäck, M. Schoenauer, and H. P. Schwefel, Eds. 1998, number 1498 in Lecture Notes in Computer Science, pp. 188–197, Springer-Verlag.

[20] C. M. Friedrich and C. Moraga, "An evolutionary method to find good building-blocks for architectures of artificial neural networks," in *Sixth International Conference on Information Processing and Management of Uncertainty in Knowledge Based Systems (IPMU '96)*, 1996, pp. 951–956.

[21] D. B. Fogel, *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence*, IEEE Press, 1995.

[22] C. Igel and M. Hüsken, "Improving the Rprop learning algorithm," in *Proc. of the Second International ICSC Symposium on Neural Computation (NC 2000)*, H. Bothe and R. Rojas, Eds. 2000, pp. 115–121, ICSC Academic Press.