

Evolutionary Optimization of a Hierarchical Object Recognition Model

**Georg Schneider, Heiko Wersing, Bernhard Sendhoff,
Edgar Körner**

2005

Preprint:

This is an accepted article published in IEEE Transactions on Systems, Man and Cybernetics, Part B: Cybernetics. The final authenticated version is available online at: [https://doi.org/\[DOI not available\]](https://doi.org/[DOI not available])

Special Issue on Learning in Computer Vision and Pattern Recognition

IEEE Transactions on Systems, Man and Cybernetics, Part B: Cybernetics

Evolutionary Optimization of a Hierarchical Object Recognition Model

Georg Schneider, Heiko Wersing, Bernhard Sendhoff, and Edgar Körner

Honda Research Institute Europe GmbH

Carl-Legien-Strasse 30, D-63073 Offenbach/Main, Germany

georg.schneider@honda-ri.de and heiko.wersing@honda-ri.de

Abstract— A major problem in designing artificial neural networks is the proper choice of the network architecture. Especially for vision networks classifying 3D objects this problem is very challenging, as these networks are necessarily large and therefore the search space for defining the needed networks is of a very high dimensionality. This strongly increases the chances of obtaining only suboptimal structures from standard optimization algorithms. We tackle this problem in two ways. Firstly, we use biologically inspired hierarchical vision models to narrow the space of possible architectures and to reduce the dimensionality of the search space. Secondly, we employ evolutionary optimization techniques to determine optimal features and nonlinearities of the visual hierarchy. Here, we especially focus on higher-order complex features in higher hierarchical stages.

We compare two different approaches to perform an evolutionary optimization of these features. In the first setting, we directly code the features into the genome. In the second setting, in analogy to an ontogenetical development process, we suggest the new method of an indirect coding of the features via an unsupervised learning process, which is embedded into the evolutionary optimization. In both cases the processing nonlinearities are encoded directly into the genome and are thus subject to optimization. The fitness of the individuals for the evolutionary selection process is computed by measuring the network classification performance on a benchmark image database. Here, we use a nearest-neighbor classification approach, based on the hierarchical feature output.

We compare the found solutions with respect to their ability to generalize. We differentiate between a first and a second order generalization. The first order generalization denotes how well the vision system, after evolutionary optimization of the fea-

tures and nonlinearities using a database A, can classify previously unseen test views of objects from this database A. As second order generalization we denote the ability of the vision system to perform classification on a database B using the features and nonlinearities optimized on database A. We show that the direct feature coding approach leads to networks with a better first order generalization, whereas the second order generalization is on an equally high level for both direct and indirect coding. We also compare the second order generalization results with other state-of-the-art recognition systems and show that both approaches lead to optimized recognition systems, which are highly competitive with recent recognition algorithms.

I. INTRODUCTION

The usage of neural networks for different image processing applications has grown rapidly in recent years. A key problem is still the development of a robust mechanism for invariant feature extraction and selection. The incorporation of prior knowledge is of fundamental interest in this context [1]. The correct recognition of presented visual objects should be robust under translation, scaling and rotation of the input stimuli. To integrate these properties into a neural vision system a proper architectural design of the network is essential.

Evolutionary algorithms provide a general and powerful method for system design optimization and their successful combination with neural networks has been shown in various applications

[2]. In the work presented here, we use evolution strategies to support the design process. In order to apply evolutionary algorithms to the design of neural systems their structure and parameters must be represented or encoded. Most approaches use the so-called *direct coding*, e.g., via a *connection matrix*, where each entry represents a connection between two neurons. The main disadvantage of this method is the bad scaling property, since the representation scales quadratically with the number of neurons. In neural networks used for vision tasks, the number of neurons needed is immense and, therefore, the direct encoding is difficult to apply. This makes *indirect coding* [3] approaches, where not every neuron with every connection is explicitly encoded in the chromosome, especially attractive for this application. The main idea behind this is that an efficient genotype-phenotype mapping [4] can reduce substantially the number of entries in the chromosome compared to a fully explicit description. The genotype-phenotype mapping can be realized by a fixed mapping from the low-dimensional genotype parameter space to the high-dimensional¹ phenotype space. This dimension reduction may, however, cause an inaccessibility of certain phenotypes using this mapping, therefore the representation might not be complete. Another problem is to ensure strong causality for this mapping, i.e., the neighborhood structure in genotype space is conserved under the mapping to the phenotype space. A more complex form of this mapping is to use a generating process instead of a fixed functional genotype-phenotype mapping. By using a predefined building process which allows the development of the phenotype, the only information which have to be encoded in the chromosome are potential process control parameters. The next step is not only to use a set of fixed rules for the development, but an active learning process for the indirect coding. Interesting approaches which focus on this combination of evolution and learning can be found in [6], [7] and will be discussed in more detail in the next section. This scheme of an indirect coding using local learning rules for the building

process of a complex neural system is biologically far more realistic. The simple fact, that the human genome contains not enough information which could determine every connection from one neuron to another, makes an indirect coding necessary for a biologically inspired design. This problem could be solved by coding simple construction rules into the genome which determine the growth of the neural system. However, neurobiological evidence tells us that the used indirect coding is far more than a fixed collection of rules how the neural system has to be composed. Rather it seems that the process of ontogeny – the development process from a fertilized egg cell to the full-grown individual – is strongly influenced by the environment. For example, prenatal activation patterns, which have been detected in the retina, are supposed to be a necessary input for the active structuring process of the visual cortex [8], [9].

A number of authors have suggested approaches that use the principle of hierarchical processing for achieving invariant recognition [10]–[13]. The main idea is to use intermediate stages in a hierarchical network to achieve higher degrees of invariance over responses that correspond to the same object, thus reducing the combinatorial complexity effectively. There is also considerable biological evidence for hierarchical processing in the visual cortex [14]–[17]. Since the work of Fukushima [10], who proposed the Neocognitron as an early model of translation invariant recognition, two major processing modes in the hierarchy have been emphasized. Feature-selective neurons are sensitive to particular features which are usually local in nature. Pooling neurons perform a spatial integration over feature-selective neurons which are successively activated, if an invariance transformation is applied to the stimulus. The combined stages of local feature detection and spatial pooling face what could be called a stability-selectivity dilemma. On the one hand, excessive spatial pooling leads to complex feature detectors with a very stable response under image transformations. On the other hand, the selectivity of the detector is largely reduced, since wide-ranged spatial pooling may accumulate too many weak evidences, increasing the chance of accidental appearance of the feature. Therefore, a proper balancing of these antagonistic properties using the processing nonlinearities is one of the crucial

¹The assumption that the phenotype space is larger than the genotype space is not necessarily valid, a prominent example in biology is the mapping from RNA sequences (genotype) to RNA secondary structures (phenotype), where the genotype space is much larger than the space of RNA secondary structures [5].

problems that has to be solved when designing deep hierarchical networks.

Another crucial aspect in the hierarchy is the coding strategy that is implemented through the feature ensembles in subsequent hierarchical stages. Redundancy reduction has been suggested as a predominant strategy for visual processing [18]. Along this line, wavelet-like features have been derived which resemble the receptive fields of V1 cells either by imposing sparse overcomplete representations [19] or imposing statistical independence as in independent component analysis [20]. Common to all these approaches is that they use local unsupervised learning rules on an ensemble of visual inputs to obtain features that capture salient structures in the inputs. Therefore, we can use these local learning rules as a structure generating process for the composition of hierarchical visual networks.

In our work, we combine biologically inspired hierarchical networks with evolution strategies in a novel way to obtain powerful recognition architectures for general 3D object recognition. Our focus is a comparison of the advantages and disadvantages of direct versus indirect coding of the features in the visual hierarchy with regard to the generalization capabilities of the network. In addition to the features, we also optimize important nonlinearities of the vision model architecture, such as competition strengths between feature channels, processing thresholds and the widths of pooling kernels for spatial integration. The target value of the optimization is the classification performance of the vision network in an 3D object recognition task. Our vision model architecture is introduced in Section II. The details of the evolutionary optimization are described in Section III. We state the results, including a comparison to other state-of-the-art algorithms, in Section IV. In Section V, we discuss and compare the results of the first and second order generalization from the directly and the indirectly coded optimization and conclude in the last section.

A. Related work

In the following, we review related work with a focus on approaches for feature learning, structure optimization in hierarchical networks for pattern recognition, and indirect coding. The principles of redundancy reduction and sparse coding have

so far been mainly applied to the initial or early feature detection stages [19], [20]. Recently, the application to complex contour-coding cells has been shown in [21], but the issue of object recognition was not considered. Wersing & Körner have investigated the application of a nonnegative sparse coding approach with invariance constraints to obtain complex combination or so-called hypercomplex cells and have applied the system successfully to a number of complex recognition benchmarks [13], [22]. Fukushima [10] has introduced with the Neocognitron a principle of hierarchical processing for invariant recognition, that has been based on successive stages of local template matching and spatial pooling. The Neocognitron can be trained by unsupervised, competitive learning. However, applications like hand-written digit recognition have required a semi-supervised training procedure, where training patterns are selected manually. Serre et al. [23] have used a vector quantization approach to determine features optimized for face recognition in their hierarchical HMAX model.

Few works use evolutionary methods to optimize hierarchical vision systems. Pan et al. [24] have optimized features with manually designed patterns as targets in intermediate stages of the vision architecture. Shi et al. [25] have used genetic algorithms to build these patterns and use conventional supervised training methods of the Neocognitron [10] to get the features. Honavar and Uhr [26] have successfully constructed combination features guided by the misclassification of the network. As candidates for the new features they have used image patches where the activation is high. In addition to the new generation of features, they also have re-train existing weights via an error-back-propagation method. The task of the optimized network has been to perform pattern recognition of simple line drawings. Teo and Sim [27] have used design of experiments and orthogonal array methods to optimize a number of free parameters in the Neocognitron. In [28] they have compared this optimization with an optimization of a number of Neocognitron parameters using a genetic algorithm. The task of the network has been the classification of 10 handwritten digits. In summary, we find that works which have focused on biologically inspired recognition architectures and evolution have not yet been applied to 3D ob-

ject recognition and often have required additional manual tuning.

Few researchers have used a form of indirect coding for the evolutionary optimization of neural networks. Kitano [29] has used a graph generation grammar to indirectly code neural networks. He has optimized the network with a genetic algorithm and has compared the results with a directly coded network. He has shown that the indirectly coded networks exhibit a magnitude of speedup in convergence of the evolutionary optimization. Sendhoff and Kreutz [7] have included a developmental phase - a growth process - in the analysis of the dynamic interaction between genetic information and information learned during development. They have shown that a cycle of learning the weights of the neural system and growing the system according to a genetically defined rule increases the stability of the overall ontogenetic process. The grown network has been able to exploit the previously learned weights, although the structure of the network had changed. A strongly neurobiologically inspired approach to the combination of evolution and learning for the design of neural networks has been suggested by Rolls and Stringer [6]. They have referred to ontogeny as a fundamental concept in the evolution of neural systems. The optimized networks have been restricted to three canonic architectures: pattern association memory, auto-association network and competitive neural network. The evolutionary search process has had a variety of parameters to optimize, such as: the choice of different learning algorithms, network nonlinearities, the number of neurons of a special class and the relative number of neurons of a class which fire with a certain probability. Also in the field of genetic programming Topchy and Punch [30] have shown the successful use of coupling evolution and learning, working on a number of numerical test problems. In summary, most contributions which have focused on indirectly coded evolutionary optimization schemes have not approached complex object recognition tasks.

Recent work by Krawiec and Bhanu [31] have not used a neural structure but a set of vision operators, like e.g. high- and low-pass filtering. These operators for feature extraction have been coded in a form of linear genetic programming into individuals. In a cooperative co-evolutionary

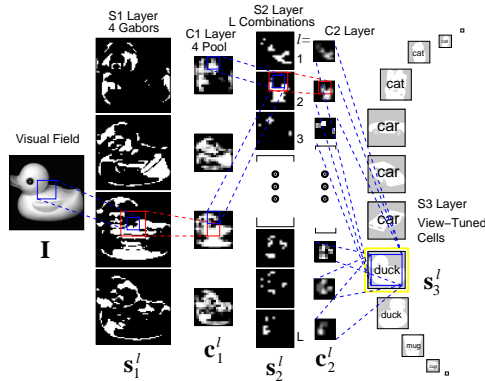


Fig. 1. Sketch of the hierarchical network. The input image is presented as a 64×64 pixel image. The S1 layer consists of 4 Gabor feature planes at 4 orientations with a dimension of 64×64 each. After convolution with the Gaussian kernel the C1 layer only uses every 4th pixel (i.e. sub-sampling by 4 to resolution 16×16). The S2 layer contains combination coding cells with possible local connections to all of the C1 cells. The C2 layer pools the S2 planes and sub-samples down to a resolution of 8×8 . The final S3 cells are tuned to representative views (covering the appearance variation), which are represented as the activity pattern of the C2 planes for an input image.

approach these individuals have evolved to an optimal feature generating procedure for the classification of synthetic aperture radar (SAR) images of real-world objects.

As a new method for an evolutionary optimization with an efficient and indirect coding of a biologically inspired neural vision network we suggest to use local unsupervised learning rules based on sparse coding concepts. These local learning rules have the benefit of not requiring any manual intervention during learning. We apply the optimized networks to a classification task of real-world 3D images.

II. THE NEURAL VISION SYSTEM FOR OBJECT RECOGNITION

In the following, we define the hierarchical model architecture that we will use for the evolutionary optimization. The model is based on a feedforward architecture with weight-sharing and a succession of feature-sensitive matching and pooling stages (see also [13] for a discussion on the general properties and biological relevance of this architecture). The model comprises three stages arranged in a processing hierarchy (see Figure 1).

The first processing stage consists of a convolution with 4 differently oriented first-order Gabor filters, a Winner-Take-Most (WTM) mechanism between these features and a final threshold function. We adopt the notation, that vector indices run over the set of neurons within a particular feature plane of a particular layer. To compute the response $s_1^l(x, y)$ of a neuron in the first layer S1, responsive to feature type l at position (x, y) , first the image vector \mathbf{I} is multiplied with a weight vector $\mathbf{w}_1^l(x, y)$ characterizing the receptive field profile:

$$q_1^l(x, y) = |\mathbf{w}_1^l(x, y) * \mathbf{I}|, \quad (1)$$

where the inner product is denoted by $*$, i.e. for a 10×10 pixel image, \mathbf{I} and $\mathbf{w}_1^l(x, y)$ are 100-dimensional vectors. All neurons in a feature plane l have the same receptive field structure, given by $\mathbf{w}_1^l(x, y)$, but shifted receptive field centers, as in a classical weight-sharing architecture [10]. In a second step, a Winner-Take-Most mechanism is performed with

$$r_1^l(x, y) = \begin{cases} 0 & \text{if } \frac{q_1^l(x, y)}{M} < \gamma_1 \text{ or } M = 0 \\ \frac{q_1^l(x, y) - M\gamma_1}{1 - \gamma_1} & \text{otherwise} \end{cases} \quad (2)$$

where $M = \max_k q_1^k(x, y)$ and $r_1^l(x, y)$ is the response after the WTM mechanism which suppresses sub-maximal responses and provides a model of latency-based competition [13]. The parameter $0 \leq \gamma_1 \leq 1$ controls the strength of the competition. The activity is then passed through a simple threshold function with a common threshold θ_1 for all neurons in layer S1:

$$s_1^l(x, y) = H(r_1^l(x, y) - \theta_1), \quad (3)$$

where $H(x) = 1$ if $x \geq 0$ and $H(x) = 0$ otherwise and $s_1^l(x, y)$ is the final activity of the neuron sensitive to feature l at position (x, y) in the S1 layer. The activities of the first layer of pooling C1-neurons are given by

$$c_1^l(x, y) = \tanh(\mathbf{g}_1(x, y) * s_1^l), \quad (4)$$

where $\mathbf{g}_1(x, y)$ is a normalized Gaussian pooling kernel with width σ_1 , identical for all features l , and \tanh is the hyperbolic tangent function. The features in the intermediate layer S2 are sensitive to local combinations of the features in the planes of the previous layer, and are thus called *combination neurons* in the following. We also use the term feature bank to denote the set of features of a

particular layer. We introduce the layer activation vector $\bar{\mathbf{c}}_1 = ((\mathbf{c}_1^1)^\top, \dots, (\mathbf{c}_1^K)^\top)^\top$ and the layer weight vector $\bar{\mathbf{w}}_2^l = ((\mathbf{w}_2^{l1})^\top, \dots, (\mathbf{w}_2^{lK})^\top)^\top$ with $K=4$. Here $\mathbf{w}_2^{lk}(x, y)$ is the receptive field vector of the S2 neuron of feature l at position (x, y) , describing connections to the plane k of the previous C1 neurons. The combined linear summation over previous planes is then given by $q_2^l(x, y) = \bar{\mathbf{w}}_2^l(x, y) * \bar{\mathbf{c}}_1$. The weights of these combination neurons are a main target of our evolutionary optimization. After the same WTM procedure with strength γ_2 as in (2), the activity in the S2 layer is given by $s_2^l(x, y) = H(r_2^l(x, y) - \theta_2)$ after thresholding with a common threshold θ_2 . The step from S2 to C2 is identical to (4) and given by $c_2^l(x, y) = \tanh(\mathbf{g}_2(x, y) * s_2^l)$, with Gaussian spatial pooling kernel $\mathbf{g}_2(x, y)$ with range σ_2 . The nonlinearity parameters $\gamma_1, \theta_1, \sigma_1, \gamma_2, \theta_2, \sigma_2$ will be subject to our evolutionary optimization.

Classification of an input image with C2 output $\bar{\mathbf{c}}_2$ is done by nearest neighbor match to previously stored template activations $\bar{\mathbf{c}}_2^v$ for each training view v . This can be realized e.g. by view-tuned units (VTU) in an additional S3 layer with a radial basis function characteristics according to $s_3^v = \exp(-\|\bar{\mathbf{w}}_3^v - \bar{\mathbf{c}}_2\|^2)$ where $\bar{\mathbf{w}}_3^v = \bar{\mathbf{c}}_2^v$ is tuned to the training C2 output of pattern v . Classification can then be performed by detecting the maximally activated VTU.

III. EVOLUTIONARY OPTIMIZATION OF THE NEURAL VISION SYSTEM

A. Evolution strategies

In evolution strategies (ES) the essential variations during the evolutionary search are mutations which are realized by adding normally distributed random numbers $z_i \sim N(0, \sigma_i^2)$ to the objective variable $\mathbf{x} = (x_1, \dots, x_n)^\top$. The standard deviations σ_i of the normal distribution are called the strategy parameters and their values determine the width of the evolutionary search process. Therefore, the σ_i are also called *step-sizes*. The standard deviations have to adapt during the process to the local topology of the search space. This process of *self-adaptation* is a key principle of evolution strategies. It relies on a “second-order” or indirect selection of the strategy parameters which are part of each individual. The strategy parameters are also subject to mutations. Thus, the chromosome

of an individual consists of both the objective and the step-size vector, see e.g. Schwefel [32]. The standard evolution strategy with one global step-size σ for all objective parameters can be expressed as follows:

$$\sigma(t) = \sigma(t-1) \exp(\tau z') \quad (5)$$

$$\mathbf{x}(t) = \mathbf{x}(t-1) + \mathbf{z} \quad (6)$$

$$z' \sim N(0, 1), z_i \sim N(0, \sigma(t)^2),$$

where the parameter τ controls the adaptation-speed of σ and t denotes the number of the generation. In addition to the mutation operator, we also use two different recombination operators. The *discrete recombination* operator randomly selects two parent individuals and creates an offspring individual by randomly choosing a chromosome entry from one of the two parents for the chromosome entry of the newly created offspring. Therewith an offspring chromosome entry originates from one of the two parents. In the used *generalized intermediate recombination* the offspring chromosome entries are generated by a randomly chosen intermediate value of the parent chromosome entries.

An important aspect in the optimization is the proper choice of the genotype-phenotype encoding or mapping. In our case the phenotype is an instance of a hierarchical neural vision system with particular features and nonlinearities. Its classification performance is used as negative fitness for the corresponding genotype. In the genotype the characteristics of the vision system are described and this description is subject to evolutionary operators like mutation and recombination. As mentioned in the introduction, there are two fundamental approaches which we can distinguish: direct and indirect coding. In direct coding all properties of the phenotype are explicitly defined in the genotype, whereas this is not the case in indirect coding. In our investigations, we compare both approaches for the optimization of our vision system.

In the evolutionary optimization applied here, we used a semi-global step-size-adaptation with two different step-sizes, which turned out to be sufficient: one for the 6 nonlinearity parameters and one for the weights of the combination features, described in more detail in the following sections. In the case of the indirect coding, we need just one step-size since the combination fea-

tures are optimized by the local learning process. We used discrete recombination for the 6 nonlinearity parameters and the combination features. The strategy parameters were recombined by a generalized intermediate recombination [33]. In our studies, we used the “ES-typical” deterministic (μ, λ) selection, which means, that from μ parent individuals λ offsprings are produced and from these the best μ individuals are selected to form the parents of the next generation.

B. First and second order generalization

The target of the evolutionary optimization process is to find system nonlinearities and combination features which optimally enable the vision system to classify unseen test views of objects – after being trained by storing C2 activations of a few object training views. For the evaluation of the optimized vision systems we introduce the concept of first and second order generalization², which is displayed in Figure 2. The flow of the evolutionary optimization of the hierarchical neural vision system is the following: We code the vision system into the chromosome (directly in the first and indirectly in the second setting). This chromosome together with the chromosome of strategy parameters define an individual. Then we apply evolutionary operators like mutation and recombination to the population of individuals. Thereafter, we construct the offsprings – different vision systems – and train them: by storing a few training views from each object of the image database A in the C2 space. Then we test the systems with the classification of test object views from database A, not contained in the training set. We use the classification rate as the fitness for the following selection of the parents, which constitute the next generation. After a sufficient number of generations we get vision systems which are well structured and successfully classify objects of database A. We call this performance *first order generalization*. With *second order generalization* we denote the ability of the system optimized on database A, to successfully classify objects from

²The differentiation between first and second order generalization used here is different from the one proposed by Hüsken et al. [34], where second order generalization referred to changing between problems belonging to one class. Note also that the more common terms test and validation error are not suitable, since we are working on different databases and not on two subsets of one database.

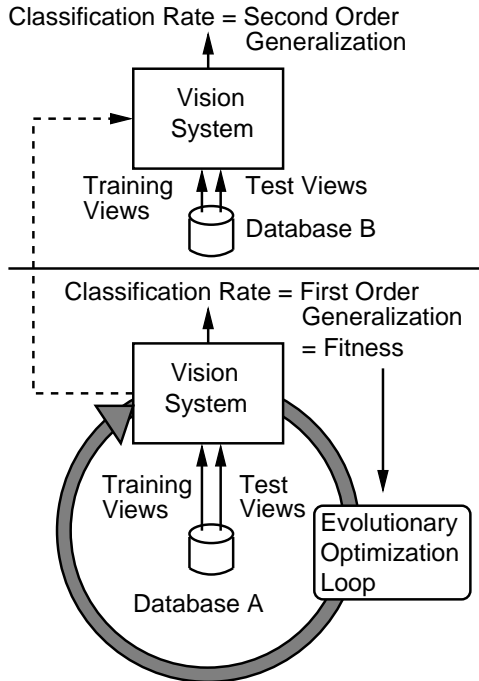


Fig. 2. Concept of the first and second order generalization.

a database B – after training with a few training views of each object of database B – but without any changes to features or nonlinearities.

C. Direct coding

In the representation of our vision system, we differentiate between system nonlinearities and the combination features. For the fitness of an individual we used always the first order generalization.

1) *System nonlinearities*: We selected 6 parameters which efficiently characterize the quality of the nonlinear processing steps of the system. These are:

- the WTM selectivities γ_1, γ_2 , which control the competition between the different features at the same image location within the same layer,
- the threshold parameters θ_1, θ_2 , which control the number of neurons firing,
- the pooling ranges σ_1, σ_2 , which control the sizes of the Gaussian pooling kernels used in layer C1 and C2.

The parameters $\gamma_1, \gamma_2, \theta_1, \theta_2, \sigma_1, \sigma_2$ are coded as real values into the chromosome. The values

are restricted to the following intervals : $\gamma_1, \gamma_2 \in [0, 1]$, with a value of 1 meaning that only the outputs of the strongest features are transmitted, whereas 0 means, that all signals from all features are transmitted without any reduction in strength. The normalization of the gray values of the images and used filters results in $\theta_1, \theta_2 \in [0, 3]$. For an adequate receptive field size for pooling we set $\sigma_1, \sigma_2 \in [0.0001, 7]$.

2) *Combination feature bank*: Additionally to the system nonlinearities the weights $\bar{\mathbf{w}}_2^l = (\mathbf{w}_2^{l1}, \dots, \mathbf{w}_2^{l4})$, which define the combination feature bank, are directly coded into the chromosome, $l = 1, \dots, L$, where L is the number of S2 feature planes.

The coding of the combination feature bank is organized as follows: We define the size of one feature of the combination feature bank $\bar{\mathbf{w}}_2^l \in \mathbb{R}^{36=4 \times 3 \times 3}$. Each of the 4 planes of layer C1 corresponding to four different local orientations in the image is convolved with a 3×3 filter. We define $w_{2i}^{lk} \in [0, 1]$, with $k = 1, \dots, 4$, and $i = 1, \dots, 36$ as the i th entry of \mathbf{w}_2^{lk} . The optimization was carried out with $L = 9$ and with $L = 50$ features. With 9 features $9 \times 36 = 324$ values have to be optimized. Thus the full optimization (including also the nonlinearities) took place in a 330-dimensional search space ($324 + 6 = 330$), with 50 features the space was 1806-dimensional ($50 \times 36 + 6 = 1806$).

D. Indirect coding

In our indirect coding approach, we still code the nonlinearities directly like described before but we use an unsupervised local learning process to determine the weights of the combination feature bank. The process uses 3×3 pixel patches of the C1 layer to learn a filter bank which can sparsely reproduce the input. This learning process is controlled by a sparsity factor which determines the trade-off between sparsity and reconstruction ability. We code this parameter in addition to the 6 nonlinearities into the chromosome and therefore perform the optimization in a just 7-dimensional search space. The price for this reduction is the time for the learning process, which we have to perform each time we want to build up an individual for assessing its fitness.

We apply the non-negative sparse coding learning rule [21] and generate for a given S1, C1

setting an ensemble of C1 activity vectors for 1440 COIL20 [35] images³. From the C1 layer activation patterns obtained from each image, we extract at a random position a local 3×3 patch⁴. Since we have four feature planes in the C1 layer, we obtain from each of these patches a 36-dimensional vector $\bar{\mathbf{c}}_1^{(p)}$ ($3 \times 3 \times 4 = 36$), where p is the patch index. The learning rule is defined by the minimization of

$$E = \sum_p \left\| \bar{\mathbf{c}}_1^{(p)} - \sum_l s_l^{(p)} \bar{\mathbf{w}}_2^l \right\|^2 + \lambda \sum_p \sum_l s_l^{(p)}, \quad (7)$$

with $l = 1, \dots, L$;
 $L =$ number of combination features,

jointly in the combination features $\bar{\mathbf{w}}_2^l$ and coefficients $s_l^{(p)}$, subject to the non-negativity of both the components of $\bar{\mathbf{w}}_2^l$ and the $s_l^{(p)}$. The left part of equation (7) measures the error of reconstructing the input C1 activity patch $\bar{\mathbf{c}}_1^{(p)}$ from a set of (non-orthogonal) basis features $\bar{\mathbf{w}}_2^l$, while the right part enforces sparse activation of the coefficients $s_l^{(p)}$. The parameter λ controls the strength of the sparsity constraint⁵. After random initialization of the $\bar{\mathbf{w}}_2^l$, the optimization is performed as a two-stage gradient descent process [19], [21]: First, the $\bar{\mathbf{w}}_2^l$ are fixed, and a local minimum of equation (7) is found in $s_l^{(p)}$ for each patch p , using an asynchronous, fast fixed-point search. In the second step, an average gradient step in $\bar{\mathbf{w}}^l$ is performed with $s_l^{(p)}$ set from the first step. Both steps are repeated till convergence. The chosen seed for the random initialization is constant for a complete optimization run. Thus the fitness of each individual is reproducible. Since E is bounded from below and tends to infinity for large $\bar{\mathbf{w}}_2^l, s_l^{(p)}$, the gradient descent always converges to a local minimum of E . Although a number of local minima exist, they generally represent functionally equivalent feature sets (see also [20]–[22]).

A disadvantage of the indirect coding is the increased computation time caused by the unsupervised training process: On a Pentium III

850MHz PC one fitness evaluation takes about 90s in the indirect coding compared to 4s in the direct case.

In the following, we briefly summarize the procedure of our indirect coding scheme from the genotype to the phenotype to the fitness evaluation:

For each genotype in the population (parameters that we optimized in the evolutionary process are boldfaced)

- 1) Construction of the phenotype up to the C1 layer.
- 2) Generation of C1 layer activations using the database A.
- 3) Collecting 3×3 -patches of the activated C1 layer.
- 4) Use unsupervised local learning for the generation of the combination feature bank using the patches and the **sparsity**.
- 5) Construction of the complete phenotype – the vision system – with all **nonlinearities** and the combination feature bank.
- 6) Training of the vision system with training views of database A (storing C2 activations as a VTU for each training view)
- 7) Calculation of the classification rate using test views of database A in a nearest-neighbor classification based on C2 feature output. The result is the fitness of the individual.

IV. RESULTS OF THE OPTIMIZATION

For the evolutionary optimization of the combination features and nonlinearity parameters we used the object database COIL20 [35]. This database contains 20 different objects with 72 images of varying angles of view, reaching from 0 to 360 degrees in 5 degree steps. After the vision system is generated according to the parameters in the chromosome, it is trained with 3 views⁶(0, 120 and 240 degrees) of each object, by simply storing the highest order C2 feature activation vectors of each training view. In the test phase the vision system has to classify 24 views at the angles $\alpha = 5 + i \cdot 15$ for $i = 0, 1, 2, \dots, 23$. Thus the

³This means we use the COIL20 as database A.

⁴Note that we used for each individual the same set of patches.

⁵In the indirect coding the parameter λ will also be coded into the genome and thus will be object to optimization.

⁶Note that for the final evaluation of the second order generalization against other approaches we also use more training views (database B), even though the system was optimized for 3 training views (database A) only (see Figure 2).



Fig. 3. Objects of the COIL100 database together with the three training views of two objects and three of 24 test views.

training views are not included in the test views and the test views are equally distributed between 0 and 360 degrees (see Figure 3). These test views are matched in a nearest-neighbor fashion to the stored training vectors. We note that other classifiers like linear discriminators could also be applied, (see [13]), but the nearest-neighbor approach has the advantage of not requiring an additional weight adaptation on the view-tuned units. The target of the optimization is the determination of the nonlinearities and the combination features in a way that the system will have a minimal classification error after training, i.e., that the first order generalization of the hierarchical vision system is maximal.

A further test for the system is the second order generalization. For this test we use a subset of the COIL100 [35]⁷ database which contains 100 objects, also with 72 images of continually varying rotation angle, see Figure 3. We have to note that 17 objects of the COIL20 are also objects of the COIL100 database. Although the images

⁷We converted the color images of the COIL100 database to gray value images and scaled them down to 64x64 pixels.

are not identical and the objects are photographed under different illumination conditions, we excluded these objects to ensure a proper measure of the second-order generalization. We call this reduced database of 83 objects *COILselect*. The setting of training the system with 3 training views of the database and testing it with 24 other views, stays the same like before.

In the following tests, we optimized each setting 10 times for 400 generations using a (7,19)-ES⁸, which we identified to be a good setting in preliminary studies, considering the trade-off between evaluation time and performance.

A. Direct coding

We set the number of features contained in the combination feature bank to $L = 9$ features, which showed the best performance in preliminary evolutionary optimization runs testing also $L = 6, 7, 8, 10, 14, 20$ [36]. We also carried out an optimization run with $L = 50$ filters to allow a comparison to results published in [22], using a manually tuned version of the vision architecture. In the upper part of Table I we display the first order generalization ability, which corresponds to a low misclassification rate on the COIL20 database, and the second order generalization on the COILselect database. We show the best result of the 10 runs, the average and the standard deviation. Comparing the cases $L = 9$ and $L = 50$ we note that an increased number of features does not improve the classification on the COIL20 data. Only the best individual is slightly better for $L = 50$. On the contrary, the second order generalization improved significantly⁹. This indicates that, in spite of the larger number of free parameters, no overfitting has occurred.

In Figure 4, we display the 6 nonlinearity values of the best individuals of each optimization run for the $L = 50$ case. The best 4 individuals are plotted with solid lines and the remaining 6 individuals are plotted with dashed lines. One can see, that the best 4 individuals obtained similar values¹⁰ for $\gamma_1, \theta_1, \sigma_1$ and γ_2 , indicating that all four optimization runs identified a similar optimum

⁸Which means: $\mu = 7$ and $\lambda = 19$.

⁹With respect to the Student-t test at a confidence level of 0.01.

¹⁰Note that the highest possible value for the WTM nonlinearities γ_1 and γ_2 is 1.0.

| | | error COIL20 [%] (1st order generalization) | | |
|-----------------|----|--|-------------|-----|
| | L | b | m | s |
| direct coding | 9 | 7.9 | 8.6 | 0.5 |
| | 50 | 7.3 | 8.6 | 0.8 |
| indirect coding | 9 | 9.0 | 10.0 | 0.6 |
| | 50 | 8.8 | 9.5 | 0.6 |
| | | error COILselect [%] (2nd order generalization) | | |
| direct coding | 9 | 24.2 | 27.6 | 3.0 |
| | 50 | 23.2 | 24.8 | 1.5 |
| indirect coding | 9 | 24.1 | 26.5 | 1.6 |
| | 50 | 21.7 | 24.2 | 1.4 |

TABLE I

RESULTS OF EVOLUTIONARY OPTIMIZATION. L=NUMBER OF FEATURES, B=BEST RESULT, M=MEAN, S=STANDARD DEVIATION OF 10 RUNS.

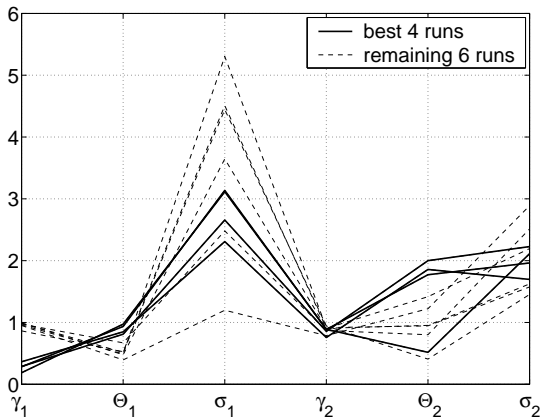


Fig. 4. Values of the nonlinearity parameters for the best individuals of each of the 10 different optimization runs for direct coding with $L = 50$.

for the nonlinearities. These individuals combine low selectivity due to low WTM strength γ_1 with high selectivity due to a high threshold θ_1 . The second group chooses the opposite strategy. While all individuals have a consistently high WTM competition γ_2 for the combination layer, a range of possible values for θ_2 and σ_2 can be observed. We note that a high value of γ_2 causes a strong reduction of activity in the S2 and C2 layers. The results for the case $L = 9$, which we do not display here, show essentially the same behavior as explained above.

Figure 5 displays the time course of the misclassification rate, the two strategy parameters $\sigma_{nonlin}, \sigma_{comb}$ and the nonlinearities of the best

optimization run for direct coding with $L = 50$ features. The misclassification rate is used as negative fitness in the evolutionary optimization. The strategy parameters $\sigma_{nonlin}, \sigma_{comb}$ are the standard deviations of the mutation operators applied to the nonlinearities and the combination features, respectively. One can observe that within about the first 40 generations the population can reduce the misclassification rate quite rapidly. After reaching a misclassification rate of about 10% the optimization becomes more difficult and therefore the progress slows down. At the same time, the strategy parameters $\sigma_{nonlin}, \sigma_{comb}$ decrease. This seems sensible since after a first “exploration” phase fine tuning of the parameters is likely to occur. This transition is controlled by the concept of self-adaptation, the key concept of evolution strategies explained in Section III. Regarding the nonlinearities one can observe the transition of γ_1 from a quite high competition (high values) between different planes to relatively low competition (low values). At the same time, one can see the inverse transition: from a low threshold θ_1 to high values, which in contrast to the transition of γ_1 means an increase in the selectivity. During these adaptation processes the step-size of the combination features σ_{comb} is nearly constant. After the transition of the nonlinearities it decreases, thus the changes in the combination features exhibit a close interaction with the nonlinearities.

B. Indirect coding

For the comparison with the results stemming from the optimization with direct coding we set the number of features used in the combination feature set also for 10 optimization runs to $L = 9$ and to $L = 50$. This reduces the evolutionary search space to 7 dimensions: the 6 nonlinearities and the sparsity factor λ . The results are displayed in the lower part of Table I. Here we see a similar trend as for the direct coding case: While the first order generalization is roughly the same for $L=9$ and $L=50$, the second order generalization is better for a larger number of features.

In Figure 6, we display the 6 nonlinearity values and the sparsity factor of the best individuals of each optimization run for the $L = 50$ case. The best 4 individuals are plotted with solid lines and the remaining 6 individuals are plotted with

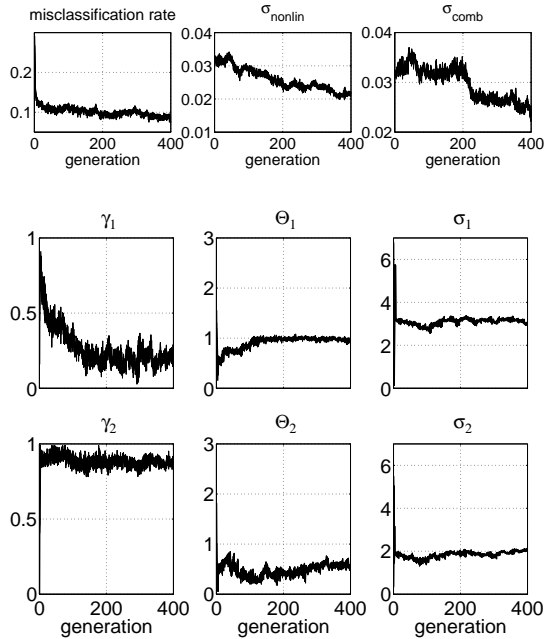


Fig. 5. Misclassification rate (negative fitness), the strategy parameters σ_{nonlin} , σ_{comb} and the nonlinearities of the 7 best individuals during the best optimization run for direct coding with $L = 50$ as a function of time. The 7 best individuals, which will be the parents in the next generation, are overlaid in the graphs and thus visualize the variance.

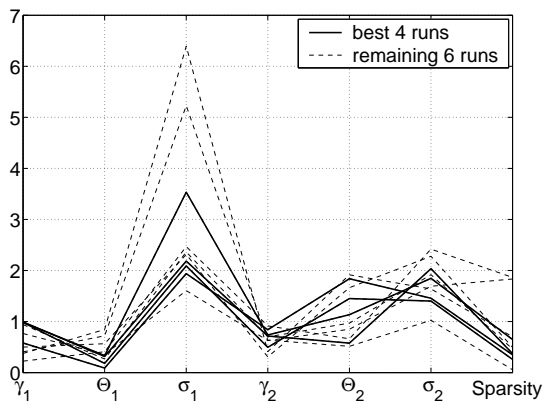


Fig. 6. Nonlinearity values for the best individuals of each of the 10 different optimization runs for indirect coding with $L = 50$.

dashed lines. Compared to the case with direct coding we have less clearly separated strategies. However, we can see, that in Figure 6 the best 4 individuals follow a nearly opposite strategy than before: The values for γ_1 are mostly high, which means a high selectivity in the beginning; θ_1 is quite low, causing a lower selectivity in the second step. Both strategies can reach a similar good trade-off between selectivity and generalization, but the strategy of having a high γ_1 and a low θ_1 seems to be slightly better in conjunction with the indirectly coded combination features, whereas the opposite seems to be better for the case of the directly coded combination features. Similarly to the direct coding, we again obtain a strong WTM competition for γ_2 . Like for the direct coding, no significant difference can be observed for the few feature case with $L = 9$ (not shown).

Figure 7 displays the time course of the misclassification rate, the strategy parameter σ_{nonlin} , the nonlinearities and the sparsity factor of the best optimization run for indirect coding with $L = 50$ features. In the indirectly coded setting the strategy parameter σ_{nonlin} additionally controls the mutation strength applied to the sparsity factor. One can observe that, similar to the direct coding, within the first 65 generations the population can reduce the misclassification rate quite fast. However, after reaching a misclassification rate of about 9.4%, the optimization slows down even more and achieves in the remaining 335 generations just an improvement of about 0.6%. This behavior can be explained with the introduction of local learning, which now cares for the optimization of the combination features¹¹. In contrast to the directly coded optimization run discussed before, the strategy parameter σ_{nonlin} increases first before it settles down after about 200 generations. This indicates that in this optimization run an increased exploration phase in the beginning is beneficial for the optimization. Concerning the time course of the nonlinearities and the sparsity factor one can observe that the majority of the adaptation dynamics takes place in the first quarter of the optimization run. Thereafter, the adaptation slows down. This is closely related to the step-size σ_{nonlin} , which decreases in the second half

¹¹This effect of combining evolution and learning, i.e. that the selection pressure on the fine-tuned improvement in later generations is reduced due to the learning process has first been reported by Hinton and Nowlan [37].

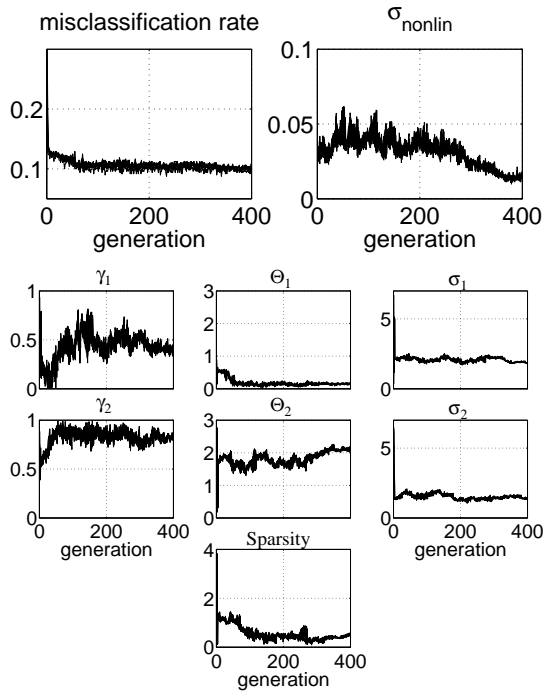


Fig. 7. Misclassification rate, the strategy parameter σ_{nonlin} , the nonlinearities and the sparsity factor of the 7 best individuals during the best optimization run for indirect coding with $L = 50$ as a function of time. The 7 best individuals, which will be the parents in the next generation, are overlaid in the graphs and thus visualize the variance.

of the optimization run.

C. Scalability of the optimized vision system

The recognition time for one of the 64×64 pixel images is below 0.1 second on a Pentium III 850MHz PC. When we feed in the images with a size of 128×128 pixels the recognition time is below 0.5 seconds. When we use the best setting found on the COILselect, which was optimized on 64×64 pixel images and use it now with 128×128 pixel input images the classification performance degrades only slightly from 21.7% to 24.1% misclassification rate. This illustrates the structural stability of the found solution. Note also that the found strategy using low thresholds for the initial local orientations detection implies high robustness with regard to low contrast images. In our experiments, we assumed the presence of a single object in the receptive field of a view-tuned cell and thus only a single object can be

recognized. The architecture can be extended to the recognition of multiple sequentially segregated objects by using a grid-like array of VTUs (see [13]), with local receptive fields covering the larger input image.

D. Comparison with the state-of-the-art

To assess the performance of the best result of our indirectly coded evolutionary optimization, we have performed a comparison to a previous manually tuned version of the vision system [22], and to other state-of-the-art systems. Here we use the results of Roobaert & van Hulle [38], who performed an extensive study on the COIL100 database, comparing support vector machines (SVM), and the eigenspace-based system of Nayar et al. [35] (denoted Columbia in the table). For the comparison we used the best vision system, with respect to the second order generalization, which has been found in the optimization runs so far ($L = 50$ indirect coding). This system was again “trained” by storing few training views of each object of the COIL100. For comparison reasons with [22] the classification stage was now performed using a linear discriminator, instead of a nearest-neighbor classifier like before.

The results are shown in Table II, where the number of objects and the number of training views are varied.¹² We see, that the evolutionary optimization could effectively improve the performance of the manually tuned vision system (mtVS) from [22]. Compared to other classifiers the optimized vision system (optVS) is highly competitive and shows superior performance especially in the most difficult cases of the task, where only few training views are available and a high number of objects have to be classified. The results of the nearest-neighbor classifier based on the plain image data (NNC) illustrate the baseline similarity of the images in the database. Note that the advantages of all recognition systems are only visible for the interesting cases of having very few training views. For few training views or adding slight disturbances like shifts and size variations, the performance of NNC degrades quickly (compare [13]).

¹²For less than 100 objects the first n objects are taken.

TABLE II
COMPARISON OF MISCLASSIFICATION RATES ON THE
COIL100 DATABASE.

| Method | 30 Objects | | | 4 Training Views | | |
|----------|----------------|-----|------|-------------------|------|------|
| | Training Views | | | Number of Objects | | |
| | 36 | 8 | 2 | 10 | 30 | 100 |
| NNC | 0 | 7.5 | 29.5 | 13.5 | 18.2 | 29.9 |
| Columbia | 0 | 4.4 | 32.9 | 7.9 | 15.4 | 23.0 |
| SVM | 0 | 4.8 | 29.0 | 9.0 | 15.1 | 25.4 |
| mtVS | 0 | 7.3 | 28.3 | 18.4 | 15.8 | 23.9 |
| optVS | 0 | 4.4 | 22.9 | 12.4 | 12.9 | 20.2 |

V. DISCUSSION

An important result is the fact that despite the huge dimensionality of the search space for 50 features (1806-dim.) the evolutionary optimization is able to find good results with a good convergence behavior. Considering second order generalization, the results are even better than the ones for only 9 features (330-dim.). We believe that the reason for this lies in the structural stability of the biologically inspired hierarchical architecture, where a lot of different combination feature banks are equally well suited for classification. For the more difficult COILselect database, 50 filters for the combination feature bank seem more adequate [22] and the drawback of a harder optimization is compensated by the enhanced representational capability of the network.

Comparing direct and indirect coding, we find that the direct evolutionary optimization yields significant better results in the first order generalization than the indirect optimization. When we compare the results with equal numbers of features the direct coding outperforms the indirect one with respect to the Student-t test at a confidence level of 0.005. At the same time, not just the mean misclassification rates of the direct coding are better, but also the best performance is reached. For the second order generalization, a slight improvement of the indirect over the direct coding case can be observed, which, however, does not have high statistical significance. We note that the number of genotype parameters was reduced from 1806 to 7 due to the usage of local learning. The better result of the direct coding in the first order generalization seems to imply, that the system is more adapted – due to the higher

degrees of freedom – to the database A than in the indirect case. This advantage obviously vanishes for the second order generalization. Here the indirect coding scheme performs at least equally good. Due to the immense amount of possible connections in the brain it is not possible, that precise connection patterns are coded in the genome. Biologically much more plausible seems therefore the direct coding of only a few fundamental structure properties, like in our model the layer sensitivity, the lateral competition strength, and the degree of anatomic convergence. The determination of a large number of local cell connectivities on the other hand could be indirectly coded by a kind of unsupervised learning method embedded in the ontogenetic building process.

Another interesting question is the degree of dependence of the found combination features, and whether in the high-dimensional cases for $L = 50$ actually all features contribute to the classification. To analyze the combination feature output of the best results we use a principal component analysis (PCA) on the C2 layer activations. Since we are interested in the coupling of the feature output at a common position (x, y) in all feature planes, we generate an ensemble of 72000 vectors $\mathbf{x}_i = (c_2^1(x, y), c_2^2(x, y), \dots, c_2^K(x, y))^T \in \mathbb{R}^K$ at 10 random positions (x, y) from all 7200 COIL100 images. Here K is the number of planes of the layer C2, which is equal to L , the number of combination features. We performed this test on the best vision systems (for $L = 9$ and $L = 50$) according to the second order generalization for the direct coding and the indirect coding. The results are displayed in Figure 8. The normalized principal components (PC) are coded as gray values in the figure using black for negative and white for positive values. Columns in the display correspond to single principal component vectors. Under each illustration of the eigenvectors we display the corresponding eigenvalues determining the variance along the corresponding vector. In the case of $L = 9$ we get similar results for the direct and the indirect coding: All K planes – all principal components – are reasonably activated. This changes in the case of $L = 50$. Here we can observe horizontal gaps in the PC columns for the direct coding. In addition, the eigenvalues are more concentrated on the first components than in the indirect case. Although some activations

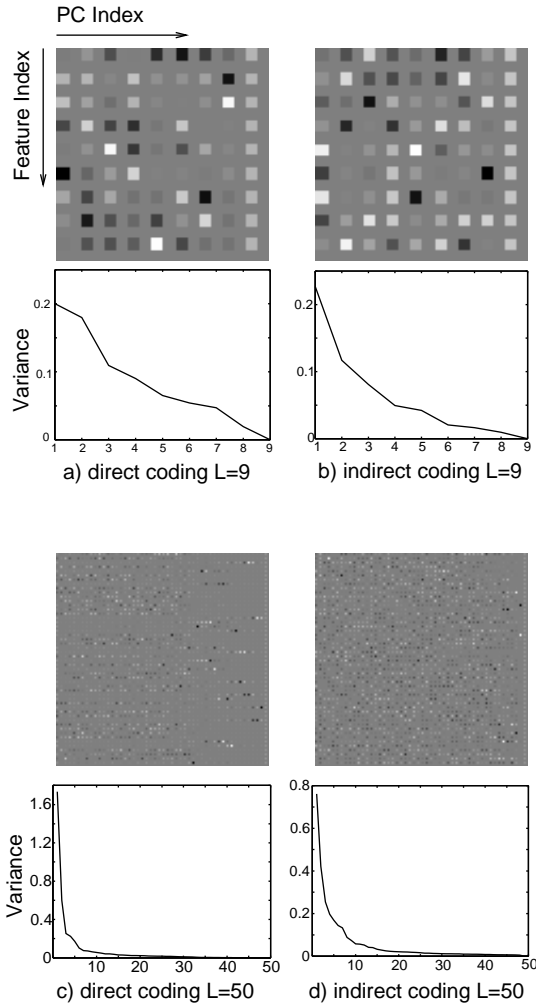


Fig. 8. Principal component analysis of the feature detector output on common positions in the C2 layer. Note the difference between the y-axis in c) and d), which illustrates the more peaked distribution of variance in the direct coding case.

are detectable at the right end of the gaps, these activations correspond to very small eigenvalues. Therefore the overall effect of these components in the nearest neighbor matching is very limited. The fact, that the indirect coding scheme uses a larger number of principal components is caused by the applied local learning, which optimizes also the sparse use of the different features. Therefore, the spectrum of the indirect coding is more uniformly distributed. Note also that the DC component with all features equally activated (the rightmost PC column in the graph)

is not present in the data, due to a zero eigenvalue. This is caused by the WTM nonlinearity, which suppresses a common activation of all features at a common position.

VI. CONCLUSION

The work presented here is the first study of the optimization of a biologically inspired vision network, which is capable of performing a complex 3D real world object classification task, using evolution strategies. We compared the optimization using a direct and an indirect coding of the combination feature bank. In both settings we included the optimization of the system processing nonlinearities with a direct coding. We showed that the used biologically inspired hierarchical architecture has a very robust behavior. Therefore, the directly coded evolutionary optimization is capable of performing well in a very high dimensional search space over 1800 dimensions. We demonstrated that an indirectly coded optimization using local unsupervised learning based on a sparse representation can reduce this search space to 7 dimensions and performs approximately equally well with regard to second order generalization.

An important result is that we were able to show a consistent improvement over earlier manually tuned versions of the used neural architecture. This highlights the superiority of the systematic evolutionary approach. We could also show that the optimized architecture is highly competitive with other current high-performing recognition methods like support vector machines. Compared to the complexity of the human visual system, our architecture may represent only a tiny fraction of its functionalities and abilities. We believe, however, that our methodology of combining biologically motivated architectures with evolution will be highly beneficial for developing even more complex architectures for more difficult visual processing problems. We consider especially the incorporation of local learning as a promising component for this approach.

ACKNOWLEDGMENT

This work was supported by the German Ministry of research (BMBF) under grant LOKI 01IB001E. Additionally, we would like to thank Martina Hasenjäger and Alexandra Mark for critical reading of the manuscript.

REFERENCES

- [1] M. Egmont-Petersen, D. de Ridder, and H. Handels, "Image processing with neural networks – a review," *Pattern Recognition*, vol. 35, no. 10, pp. 2279–2301, 2002.
- [2] X. Yao, "Evolving artificial neural networks," *Proceedings of the IEEE*, vol. 87, no. 9, pp. 1423–1447, 1999.
- [3] F. Rothlauf, *Representations for genetic and evolutionary algorithms*. Springer, 2002.
- [4] B. Sendhoff, M. Kreutz, and W. von Seelen, "A condition for the genotype–phenotype mapping: Causality," in *Genetic Algorithms: Proceedings of the 7th International Conferences (ICGA)*, T. Bäck, Ed. Morgan Kaufmann, 1997, pp. 73–80.
- [5] P. Schuster and P. Stadler, "Sequence redundancy in biopolymers: A study on RNA and protein structures," in *Viral Regulatory Structures*, G. Myers, Ed., vol. XXVIII of Santa Fe Institute Studies in the Sciences of Complexity. Reading MA: Addison-Wesley, 1997.
- [6] E. T. Rolls and S. M. Stringer, "On the design of neural networks in the brain by genetic evolution," *Progress in Neurobiology*, vol. 6, no. 61, pp. 557–579, 2000.
- [7] B. Sendhoff and M. Kreutz, "A model for the dynamic interaction between evolution and learning," *Neural Processing Letters*, vol. 10, no. 3, pp. 181–193, 1999.
- [8] S. Quartz and T. Sejnowski, "The neural basis of cognitive development: A constructivist manifesto," *Behavioral and Brain Sciences*, vol. 9, pp. 537–596, 1997.
- [9] A. G. Rust, R. Adams, S. George, and H. Bolouri, "Towards computational neural systems through developmental evolution," in *Lecture Notes in Computer Science*, S. Wermter, J. Austin, and D. Willshaw, Eds., vol. 2036, 2001, pp. 188–202.
- [10] K. Fukushima, "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biological Cybernetics*, vol. 39, pp. 139–202, 1980.
- [11] G. Wallis and E. T. Rolls, "A model of invariant object recognition in the visual system," *Progress in Neurobiology*, vol. 51, pp. 167–194, 1997.
- [12] M. Riesenhuber and T. Poggio, "Hierarchical models of object recognition in cortex," *Nature Neuroscience*, vol. 2, no. 11, pp. 1019–1025, 1999.
- [13] H. Wersing and E. Körner, "Learning optimized features for hierarchical models of invariant recognition," *Neural Computation*, vol. 15, no. 7, pp. 1559–1588, 2003.
- [14] K. Tanaka, "Inferotemporal cortex and object vision: stimulus selectivity and columnar organization," *Annual Review of Neuroscience*, vol. 19, pp. 109–139, 1996.
- [15] N. K. Logothetis and J. Pauls, "Psychophysical and physiological evidence for viewer-centered object representations in the primate," *Cerebral Cortex*, vol. 5, pp. 270–288, 1995.
- [16] S. Thorpe, D. Fize, and C. Marlot, "Speed of processing in the visual system," *Nature*, vol. 381, pp. 520–522, 1996.
- [17] E. Körner, M.-O. Gewaltig, U. Körner, A. Richter, and T. Rodemann, "A model of computation in neocortical architecture," *Neural Networks*, vol. 12, no. 7-8, pp. 989–1005, 1999.
- [18] H. B. Barlow, "The twelfth Bartlett memorial lecture: The role of single neurons in the psychology of perception," *Quarterly Journal of Experimental Psychology*, vol. 37, pp. 121–145, 1985.
- [19] B. A. Olshausen and D. J. Field, "Sparse coding with an overcomplete basis set: A strategy employed by V1?" *Vision Research*, vol. 37, pp. 3311–3325, 1997.
- [20] A. J. Bell and T. J. Sejnowski, "The 'independent components' of natural scenes are edge filters," *Vision Research*, vol. 37, pp. 3327–3338, 1997.
- [21] P. O. Hoyer and A. Hyvärinen, "A multi-layer sparse coding network learns contour coding from natural images," *Vision Research*, vol. 42, no. 12, pp. 1593–1605, 2002.
- [22] H. Wersing and E. Körner, "Unsupervised learning of combination features for hierarchical recognition models," in *International Conference of Artificial Neural Networks ICANN*, J. R. Dorronsoro, Ed. Springer, 2002, pp. 1225–1230.
- [23] T. Serre, M. Riesenhuber, J. Louie, and T. Poggio, "On the role of object-specific features for real world object recognition in biological vision," in *Lecture Notes in Computer Science. Proceedings BMCV Conf.*, H. H. Bülthoff, S. W. Lee, T. A. Poggio, and C. Wallraven, Eds., vol. 2525, 2002, pp. 387–397.
- [24] Z. Pan, T. Sabisch, R. Adams, and H. Bolouri, "Staged training of neocognitron by evolutionary algorithms," in *Proceedings of the Congress on Evolutionary Computation*, P. J. Angelino, Z. Michalewicz, M. Schoenauer, X. Yao, and A. Zalzala, Eds., vol. 3. IEEE Press, 1999, pp. 1965–1972.
- [25] D. Shi, D. Chunlei, and Y. Daniel S., "Neocognitron's parameter tuning by genetic algorithms," *International Journal of Neural Systems*, vol. 9, pp. 497–509, 1999.
- [26] V. Honavar and L. Uhr, "Brain-structured connectionist networks that perceive and learn," *Connection Science*, vol. 1, pp. 139–160, 1989.
- [27] M.-Y. Teo and S.-K. Sim, "Training the neocognitron network using design of experiments," *Artificial Intelligence in Engineering*, vol. 9, no. 2, pp. 85–94, 1995.
- [28] M.-Y. Teo, L.-P. Khoo, and S.-K. Sim, "Application of genetic algorithms to optimise neocognitron network parameters," *Neural Network World*, vol. 7, no. 3, pp. 293–304, 1997.
- [29] H. Kitano, "Designing neural networks using genetic algorithms with graph generation system," *Complex Systems*, vol. 4, pp. 461–476, 1990.
- [30] A. Topchy and W. F. Punch, "Faster genetic programming based on local gradient search of numeric leaf values," in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, L. Spector, E. D. Goodman, A. Wu, W. B. Langdon, H.-M. Voigt, M. Gen, S. Sen, M. Dorigo, S. Pezeshek, M. H. Garzon, and E. Burke, Eds. San Francisco, California, USA: Morgan Kaufmann, 7-11 2001, pp. 155–162.
- [31] K. Krawiec and B. Bhanu, "Coevolutionary and linear genetic programming for visual learning," in *GECCO 2003: Proceedings of Genetic and Evolutionary Computation Conference*, E. C.-P. et al., Ed. Berlin Heidelberg: Springer, 11 July 2003, pp. 332–343.
- [32] H.-P. Schwefel and G. Rudolph, "Contemporary evolution strategies," in *Proceedings of the Third European Conference on Artificial Life : Advances in Artificial Life*, ser. LNAI, F. Morán, A. Moreno, J. J. Merelo, and P. Chacón, Eds., vol. 929. Berlin: Springer Verlag, June 1995, pp. 893–907.
- [33] T. Bäck, D. B. Fogel, and Z. Michalewicz, Eds., *Evolutionary Computation 1: Basic Algorithms and Operators*. Bristol: Institute of Physics Publishing, 2000.
- [34] M. Hüsken, J. Gayko, and B. Sendhoff, "Optimization for problem classes - neural networks that learn to learn,"

- in *IEEE Symposium on Combinations of Evolutionary Computation and Neural Networks*, X. Yao, Ed. IEEE Press, 2000, 98-109.
- [35] S. K. Nayar, S. A. Nene, and H. Murase, "Real-time 100 object recognition system," in *Proceedings of ARPA Image Understanding Workshop*, Palm Springs, 1996.
- [36] G. Schneider, H. Wersing, B. Sendhoff, and E. Körner, "Evolutionary feature design for object recognition with hierarchical networks," in *Proceedings of the 9th International Conference on Neural Information Processing - ICONIP*, L. Wang, J. Rajapakse, K. Fukushima, S.-Y. Lee, and X. Yao, Eds., vol. 4, 2002, pp. 1936-1940.
- [37] G. Hinton and S. Nowlan, "How learning can guide evolution," *Complex Systems*, vol. 1, pp. 495-502, 1987.
- [38] D. Roobaert and M. V. Hulle, "View-based 3d object recognition with support vector machines," in *Proceedings of IEEE International Workshop on Neural Networks for Signal Processing, Madison, USA*. New York, USA: IEEE, 1999, pp. 77-84.