An Adaptive Normalized Gaussian Network and Its Application to Online Category Learning

Claudius Gläser, Frank Joublin

2010

Preprint:

This is an accepted article published in Proceedings of the International Joint Conference on Neural Networks (IJCNN), IEEE World Congress on Computational Intelligence (WCCI). The final authenticated version is available online at: https://doi.org/[DOI not available]

An Adaptive Normalized Gaussian Network and Its Application to Online Category Learning

Claudius Gläser and Frank Joublin

Abstract— **In online applications, where training samples sequentially arise during execution, incremental learning schemes have to be applied. In this paper we propose an adaptive Normalized Gaussian Network model (NGnet) suitable for incremental learning. Following a statistical account we present a truly sequential training procedure. Key to the learning algorithm are local unit manipulation mechanisms for network growth and pruning which continuously adapt the network's complexity according to task demands. We evaluate our model in artificial and real-world categorization tasks. Thereby, we additionally introduce a framework for the categorization on adaptive feature spaces. In the system, a simultaneous extraction of class-discriminative features facilitates the NGnet's categorization of input patterns. We present simulation results which demonstrate that the framework realizes a rapid learning from few examples, small-sized network models, and an improved generalization ability. A comparison to incremental support vector machine classification yields a favorable performance of our model.**

I. Introduction

Normalized Gaussian Networks (NGnets) are special types of Radial Basis Function (RBF) networks which have been successfully applied in the domains of function approximation [1] and classification [2], [3]. A RBF network is a feed-forward model composed of three layers, where a radial symmetric Gaussian function serves as activation function for the hidden layer units. The specification of the network complexity, i.e. the selection of the number of hidden units, is a severe problem. Once the complexity has been determined, training is usually carried out in two stages: Firstly, unsupervised clustering is used to position the hidden units in the input space and, secondly, supervised learning trains the output weights [4]. It has been further suggested to apply evolutionary algorithms to simultaneously determine network size and parameters [5], [6].

Nevertheless, such training schemes cannot be used for online learning in applications where training samples sequentially arise. Consequently, there is a need for adaptive algorithms capable of adjusting the network's complexity as well as its parameters online according to task demands. Existing approaches to do so include Platt's Resource Allocating Network (RAN) [7] as well as its extension RANEKF [8]. Both approaches allocate new hidden units based on the novelty of data. However, RANEKF makes use of extended Kalman filtering instead of the least mean squared algorithm to update the network parameters. Minimum RAN (MRAN) [9] further applies a pruning strategy for removing hidden units whose contributions to the network output are small. The GGAP-RBF network [10] introduces a measure for the significance of hidden units and uses it as a criterion for network growth and pruning. Thereby, a neuron's significance quantifies the neuron's contribution to the network output taking all previously received input data into account.

Claudius Gläser and Frank Joublin are with the Honda Research Institute Europe, Carl-Legien-Strasse 30, 63073 Offenbach, Germany (email: {claudius.glaeser, frank.joublin}@honda-ri.de).

In this paper we propose an adaptive network which follows a thorough probabilistic interpretation of NGnets [11]. Besides Expectation-Maximization (EM) training it comprises local unit manipulation mechanisms for network growth and pruning. This includes the allocation of new units, the splitting and pruning of existing ones, and, most importantly, the merging of similar units. Particularly the merging of hidden units is seldom considered in existing approaches, even though our experiments demonstrate its significance for obtaining compact network models. We evaluate the truly sequential training scheme for a benchmark function approximation problem as well as artificial and real-world categorization tasks.

We further propose a categorization framework which additionally extracts class-discriminative features in an incremental manner. These features facilitate categorization by which faster training and smaller-sized networks can be obtained. The feature extraction runs in parallel to the categorization such that our network operates on a changing feature space to which it continuously adapts. In our experiments we highlight the advantage of this framework compared to one which does not extract class-discriminative features and particularly demonstrate the suitability of the network to adapt to changing task demands.

The remainder is organized as follows. After introducing NGnets as well as their probabilistic interpretation in Section II we propose local unit manipulation mechanisms in Section III. Next, Section IV introduces the categorization framework. Simulation results are presented in Section V. Finally, Section VI concludes the paper.

II. Normalized Gaussian Network

A. Network Architecture

A Normalized Gaussian Network (NGnet) [4] can serve as an universal function approximator. For the sake of generality, in the following we consider the approximation of mappings $\Omega : \Re^N \rightarrow \Re^D$ from an N-dimensional input space to a D-dimensional output space. Given an input $\boldsymbol{y}$ an NGnet's output $\widetilde{\boldsymbol{c}}(\boldsymbol{y})$ is calculated according to

$$\widetilde{\boldsymbol{c}}(\boldsymbol{y}) = \frac{1}{\sum_{j=1}^{M} \phi_j(\boldsymbol{y})} \cdot \sum_{i=1}^{M} \boldsymbol{\alpha}_i \cdot \phi_i(\boldsymbol{y}). \qquad (1)$$

Thereby, $\phi_i(\boldsymbol{y})$ denotes the response of the i-th hidden unit to input $\boldsymbol{y}$, M is the number of hidden units, and $\boldsymbol{\alpha}_i$ the weight vector from unit i to the output neurons (see Fig. 1).

978-1-4244-8126-2/10/$26.00 ©2010 IEEE

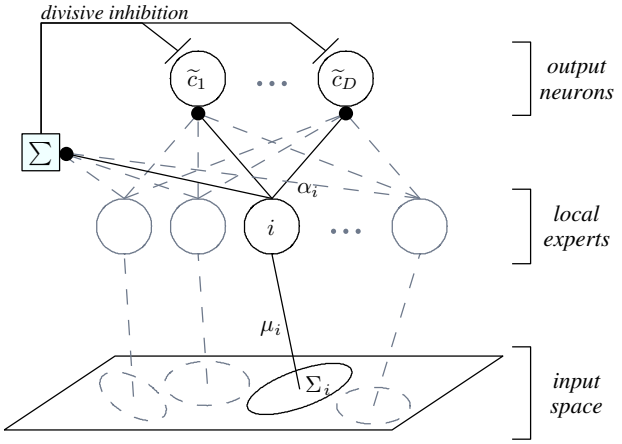


Fig. 1. The architecture of an NGnet.

The response of a hidden unit i is described by a radial basis function, i.e. a multivariate Gaussian of form

$$\phi_i(\boldsymbol{y}) = \exp\left(-\frac{1}{2}\cdot(\boldsymbol{y}-\boldsymbol{\mu_i})^T\boldsymbol{\Sigma}_i^{-1}(\boldsymbol{y}-\boldsymbol{\mu_i})\right), \quad (2)$$

where $\boldsymbol{\mu_i}$ and $\boldsymbol{\Sigma}_i$ denote the center and covariance matrix of the Gaussian. The hidden units consequently represent local models (or local experts [12]) of the mapping to be approximated. We will use these terms interchangeably.

In summary, an NGnet is similar to a standard RBF network except for the normalization of the output by the total hidden layer activity. The effect of this normalization is twofold: Firstly, it results in a competition between the units. This competition softly partitions the input space into regions, such that each local model is responsible for inputs stemming from its associated region. Secondly, as discussed in [13], the normalization results in an improved inter- and extrapolation ability compared to standard RBF networks.

B. Probabilistic Interpretation & Online Training

In [11] a stochastic interpretation of NGnets has been proposed. Here, we will review the main aspects of this view, since it constitutes the basis of our following analysis.

Consider a sample $(\boldsymbol{y}, \boldsymbol{c})$ to be a stochastic event for which a single local model i is responsible. Furthermore, let each local model be fully described by its probability density functions (pdfs) over the input and output space [11].

$$p(\boldsymbol{y}|i,\boldsymbol{\Theta}) = G(\boldsymbol{y},\boldsymbol{\mu}_i,\boldsymbol{\Sigma}_i) \quad (3)$$

$$p(\boldsymbol{c}|\boldsymbol{y},i,\boldsymbol{\Theta}) = G(\boldsymbol{c},\boldsymbol{\alpha}_i,\boldsymbol{\Gamma}_i) \quad (4)$$

Here, $\boldsymbol{\Theta} = \{\{\boldsymbol{\mu}_i,\boldsymbol{\Sigma}_i,\boldsymbol{\alpha}_i,\boldsymbol{\Gamma}_i\}_{i=1}^M\}$ denotes the parameters of the NGnet and $G(\boldsymbol{x},\boldsymbol{m},\boldsymbol{S})$ the multivariate normal distribution with mean $\boldsymbol{m}$ and covariance matrix $\boldsymbol{S}$ evaluated at $\boldsymbol{x}\in\Re^K$:

$$G(\boldsymbol{x},\boldsymbol{m},\boldsymbol{S}) = \frac{1}{(2\pi)^{K/2}|\boldsymbol{S}|^{1/2}} \cdot\exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{m})^T\boldsymbol{S}^{-1}(\boldsymbol{x}-\boldsymbol{m})\right). \quad (5)$$

Under further assumptions on the competition between local models [11], the probability that the local model i is responsible for an input $\boldsymbol{y}$ turns out to be

$$p(i|\boldsymbol{y},\boldsymbol{\Theta}) = \frac{\phi_i(\boldsymbol{y})}{\sum_{j=1}^M\phi_j(\boldsymbol{y})}. \quad (6)$$

Consequently, we obtain

$$p(\boldsymbol{c}|\boldsymbol{y},\boldsymbol{\Theta}) = \sum_{i=1}^M\frac{\phi_i(\boldsymbol{y})}{\sum_{j=1}^M\phi_j(\boldsymbol{y})}\cdot G(\boldsymbol{c},\boldsymbol{\alpha}_i,\boldsymbol{\Gamma}_i) \quad (7)$$

whose expectation matches the definition of an NGnet (see (1)). Therefore, the parameters $\boldsymbol{\Theta}$ of the NGnet can be estimated by maximum likelihood learning on the log-likelihood of the observed data $(\{\boldsymbol{y}\},\{\boldsymbol{c}\})$. Online training, thus, can be achieved by an iterative Expectation-Maximization (EM) algorithm as proposed in [11].

III. Local Model Manipulation Mechanisms

One of the main problems when using an NGnet is the specification of the network's complexity, i.e. the selection of the number of hidden units (local experts). Solving this problem is usually done by incorporating domain knowledge. Difficult mapping tasks will obviously necessitate more hidden units than simple tasks. However, it is desirable to build general purpose network models which are able to autonomously adapt their complexity based on the problem at hand. For an NGnet, this involves mechanisms for assigning new local experts and removing, splitting, or merging existing ones. Furthermore, criteria for deciding when to execute the model manipulation mechanisms have to be defined.

In previous work several methods for an incremental build-up of an NGnet have been proposed [1], [9], [10], [14], [15]. Some of them implement in part similar algorithms to the ones we outline next. However, these methods usually assume the complexity of the mapping task to be constant over time. Consequently, they increase an NGnet's complexity until a sufficient approximation quality is achieved. In contrast, we explicitly take into account a varying task complexity and, thus, present mechanisms which continuously adapt an NGnet's complexity according to task demands. As one example, we introduce the merging of experts which turns out to be beneficial for obtaining small-sized networks and improving generalization (see results in Section V).

A. Model Removal

Local experts with little or no contribution to an NGnet's approximation are redundant and should be removed. Therefore, let $p(i|\boldsymbol{y}_t,\boldsymbol{c}_t,\boldsymbol{\Theta})$ denote the posterior probability of assigning a sample $(\boldsymbol{y}_t,\boldsymbol{c}_t)$ to the i-th local expert. It can be calculated according to Bayes rule:

$$p(i|\boldsymbol{y}_t,\boldsymbol{c}_t,\boldsymbol{\Theta}) = \frac{p(i|\boldsymbol{y}_t,\boldsymbol{\Theta})\cdot p(\boldsymbol{c}_t|\boldsymbol{y}_t,i,\boldsymbol{\Theta})}{\sum_{j=1}^M p(j|\boldsymbol{y}_t,\boldsymbol{\Theta})\cdot p(\boldsymbol{c}_t|\boldsymbol{y}_t,j,\boldsymbol{\Theta})} = \frac{\phi_i(\boldsymbol{y}_t)\cdot G(\boldsymbol{c}_t,\boldsymbol{\alpha}_i,\boldsymbol{\Gamma}_i)}{\sum_{j=1}^M\phi_j(\boldsymbol{y}_t)\cdot G(\boldsymbol{c}_t,\boldsymbol{\alpha}_j,\boldsymbol{\Gamma}_j)}. \quad (8)$$

Furthermore, let ρ_i denote the running average over this posterior, where η is a time constant:

$$\rho_i(t) = (1-\eta)\cdot\rho_i(t-1) + \eta\cdot p(i|\boldsymbol{y}_t,\boldsymbol{c}_t,\boldsymbol{\Theta}). \quad (9)$$

It can be easily seen that ρ_i is proportional to the average number of samples for which the local model i best describes the mapping task. Consequently, ρ_i serves as an importance weight for the i-th expert, such that $\rho_i < \theta_{remove}/M$ with $0 < \theta_{remove} \ll 1$ constitutes a criterion for removing the i-th local expert.

Let $\mathcal{M}_i$ denote the i-th expert and $\boldsymbol{\mathcal{M}} = \{\mathcal{M}_i\}_{i=1}^{M}$ the set of all local experts. When removing the i-th model, we also adapt the ρ_j such that $\sum_j \rho_j$ before and after the removal remains unchanged, i.e.

$$\boldsymbol{\mathcal{M}}^* = \boldsymbol{\mathcal{M}} \backslash \{\mathcal{M}_i\} \tag{10}$$

$$\rho_j^* = \frac{|\boldsymbol{\mathcal{M}}|}{|\boldsymbol{\mathcal{M}}^*|} \cdot \rho_j \qquad , \forall j \text{ with } \mathcal{M}_j \in \boldsymbol{\mathcal{M}}^*. \tag{11}$$

Here, $|\boldsymbol{\mathcal{S}}|$ denotes the cardinality of the set $\boldsymbol{\mathcal{S}}$.

B. Model Assignment

A new local expert should be assigned, if a training sample $(\boldsymbol{y}_t, \boldsymbol{c}_t)$ is obtained which is either not sufficiently well described by any of the existing local experts or for which the NGnet's approximation yields a large error. These criteria can be expressed as follows

$$\max_i p(\boldsymbol{y}_t, \boldsymbol{c}_t | i, \boldsymbol{\Theta}) < \theta_{assign_1} \tag{12}$$

$$e_t > \theta_{assign_2} \tag{13}$$

where

$$\begin{aligned} p(\boldsymbol{y}_t, \boldsymbol{c}_t | i, \boldsymbol{\Theta}) &= p(\boldsymbol{c}_t | \boldsymbol{y}_t, i, \boldsymbol{\Theta}) \cdot p(\boldsymbol{y}_t | i, \boldsymbol{\Theta}) \\ &= G(\boldsymbol{c}_t, \boldsymbol{\alpha}_i, \boldsymbol{\Gamma}_i) * G(\boldsymbol{y}_t, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \quad (14) \\ e_t &= [\boldsymbol{c}_t - \widetilde{\boldsymbol{c}}(\boldsymbol{y}_t)]^T [\boldsymbol{c}_t - \widetilde{\boldsymbol{c}}(\boldsymbol{y}_t)] \quad (15) \end{aligned}$$

as well as θ_{assign_1} and θ_{assign_2} being thresholds.

If any of these conditions is fulfilled, we create a new local model $\mathcal{M}_{new}$, add it to the NGnet, and adapt the ρ_j such that $\sum_j \rho_j$ before and after the assignment remain unchanged:

$$\boldsymbol{\mathcal{M}}^* = \boldsymbol{\mathcal{M}} \cup \{\mathcal{M}_{new}\} \tag{16}$$

$$\rho_j^* = \frac{|\boldsymbol{\mathcal{M}}|}{|\boldsymbol{\mathcal{M}}^*|} \cdot \rho_j \qquad , \forall j \text{ with } \mathcal{M}_j \in \boldsymbol{\mathcal{M}}. \tag{17}$$

The new model can be initialized as proposed in [15]:

$$\boldsymbol{\mu}_{new} = \boldsymbol{y}_t \tag{18}$$

$$\boldsymbol{\alpha}_{new} = \boldsymbol{c}_t \tag{19}$$

$$\boldsymbol{\Sigma}_{new} = \min_i \left(\frac{[\boldsymbol{\mu}_i - \boldsymbol{\mu}_{new}]^T [\boldsymbol{\mu}_i - \boldsymbol{\mu}_{new}]}{N} \right) \cdot \boldsymbol{I} \tag{20}$$

$$\boldsymbol{\Gamma}_{new} = \max_i \boldsymbol{\Gamma}_i \tag{21}$$

$$\rho_{new} = \frac{1}{|\boldsymbol{\mathcal{M}}| + 1} \tag{22}$$

Here, $\boldsymbol{I}$ denotes the identity matrix and N the input dimensionality. In the special case of assigning the first local model, $\boldsymbol{\Sigma}_{new}$ and $\boldsymbol{\Gamma}_{new}$ become initialized to some predefined $\boldsymbol{\Sigma}_{init}$ and $\boldsymbol{\Gamma}_{init}$, respectively, as well as $\rho_{new} = 1$.

C. Model Splitting

If the i-th local model's quality of approximating the mapping task becomes insufficient, the input space region corresponding to its receptive field should be refined and covered by multiple experts. An insufficient approximation quality reflects itself in a diffuse probability distribution $p(\boldsymbol{c}|\boldsymbol{y}, i, \boldsymbol{\Theta}) = G(\boldsymbol{c}, \boldsymbol{\alpha}_i, \boldsymbol{\Gamma}_i)$ over the output space. Consequently, the size of the Gaussian (for which $|\boldsymbol{\Gamma}_i|$ is an indicator) is an appropriate criterion for splitting a model. In summary, the i-th model becomes split if

$$|\boldsymbol{\Gamma}_i| > \theta_{split}, \tag{23}$$

where θ_{split} is a threshold. If this criterion is met, $\mathcal{M}_i$ becomes adjusted to $\mathcal{M}_i^*$, a new model $\mathcal{M}_{new}$ is created, and finally added to the model pool.

$$\boldsymbol{\mathcal{M}}^* = (\boldsymbol{\mathcal{M}} \backslash \mathcal{M}_i) \cup \{\mathcal{M}_i^*, \mathcal{M}_{new}\} \tag{24}$$

The splitting is done along the prominent dimension of the receptive field, which is related to the method proposed in [1]. Therefore, let $\boldsymbol{\zeta}_n$ and κ_n denote the eigenvectors and eigenvalues of $\boldsymbol{\Sigma}_i$ sorted in descending order of the eigenvalues, i.e. $\kappa_1 \geq \kappa_2 \geq \ldots \geq \kappa_N$. The spin-off models are initialized as

$$\boldsymbol{\mu}_i^*, \boldsymbol{\mu}_{new} = \boldsymbol{\mu}_i \pm \xi_1 \cdot \sqrt{\kappa_1} \cdot \boldsymbol{\zeta}_1 \tag{25}$$

$$\boldsymbol{\alpha}_i^*, \boldsymbol{\alpha}_{new} = \boldsymbol{\alpha}_i \tag{26}$$

$$\boldsymbol{\Sigma}_i^*, \boldsymbol{\Sigma}_{new} = \frac{\xi_2}{\kappa_1} \cdot \boldsymbol{\zeta}_1 \boldsymbol{\zeta}_1^T + \sum_{n=2}^{N} \frac{1}{\kappa_n} \cdot \boldsymbol{\zeta}_n \boldsymbol{\zeta}_n^T \tag{27}$$

$$\boldsymbol{\Gamma}_i^*, \boldsymbol{\Gamma}_{new} = 0.5 \cdot \boldsymbol{\Gamma}_i \tag{28}$$

$$\rho_i^*, \rho_{new} = 0.5 \cdot \rho_i, \tag{29}$$

where ξ_1 and ξ_2 are constants controlling their overlap.

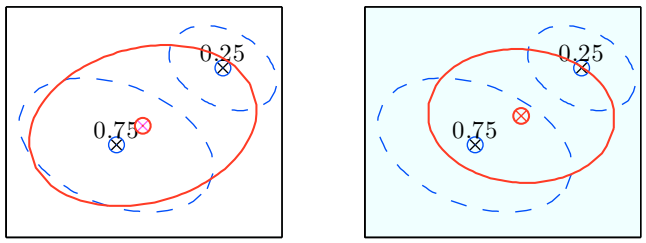
D. Model Merging

If multiple local models are sufficiently similar, they can be merged to one local expert. Thereby, the similarity depends on the overlap between the experts' pdfs over the input and output space, respectively. Let $\mathcal{U}(\mathcal{M}_i, \mathcal{M}_j)$ be a function measuring the similarity between two local models i and j with $0 \leq \mathcal{U}(\mathcal{M}_i, \mathcal{M}_j) \leq 1$, where a value of 1 corresponds to model identity and a value of 0 to total model dissimilarity. Furthermore, let $\mathcal{V}(p(\boldsymbol{a}), q(\boldsymbol{a}))$ be a function measuring the overlap between two multivariate pdfs $p(\boldsymbol{a})$ and $q(\boldsymbol{a})$ with $0 \leq \mathcal{V}(p(\boldsymbol{a}), q(\boldsymbol{a})) \leq 1$. Then we define

$$\begin{aligned} \mathcal{U}(\mathcal{M}_i, \mathcal{M}_j) &= \mathcal{V}(p(\boldsymbol{y}|i, \boldsymbol{\Theta}), p(\boldsymbol{y}|j, \boldsymbol{\Theta})) \\ &\quad \cdot \mathcal{V}(p(\boldsymbol{c}|\boldsymbol{y}, i, \boldsymbol{\Theta}), p(\boldsymbol{c}|\boldsymbol{y}, j, \boldsymbol{\Theta})) \\ &= \mathcal{V}(G(\boldsymbol{y}, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i), G(\boldsymbol{y}, \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)) \\ &\quad \cdot \mathcal{V}(G(\boldsymbol{c}, \boldsymbol{\alpha}_i, \boldsymbol{\Gamma}_i), G(\boldsymbol{c}, \boldsymbol{\alpha}_j, \boldsymbol{\Gamma}_j)) \quad (30) \end{aligned}$$

Consequently, measuring the pair-wise similarity between local experts reduces to calculating the overlap between multivariate Gaussian pdfs. Fortunately, the *Bhattacharyya Coefficient (BC)* provides an approximation for this. It is defined as

$$BC(p, q) = \int_{\boldsymbol{a}} \sqrt{p(\boldsymbol{a}) \cdot q(\boldsymbol{a})} d\boldsymbol{a} \tag{31}$$



(a) (b)

Fig. 2. The merging of two multivariate Gaussian distributions (dashed lines) with importance weights $\omega_1 = 0.75$ and $\omega_2 = 0.25$ is illustrated for the algorithms using (a) the Kullback-Leibler divergence and (b) the Jenson-Shannon divergence.

for which a closed form solution exists in case of multivariate Gaussians $p(\boldsymbol{a}) = G(\boldsymbol{a}, \boldsymbol{\mu}_p, \boldsymbol{\Sigma}_p)$ and $q(\boldsymbol{a}) = G(\boldsymbol{a}, \boldsymbol{\mu}_q, \boldsymbol{\Sigma}_q)$:

$$D_B(p,q) = \frac{1}{8} \cdot (\boldsymbol{\mu}_p - \boldsymbol{\mu}_q)^T * \boldsymbol{\Sigma}^{-1} * (\boldsymbol{\mu}_p - \boldsymbol{\mu}_q) + \frac{1}{2} \cdot \log \frac{|\boldsymbol{\Sigma}|}{\sqrt{|\boldsymbol{\Sigma}_p| \cdot |\boldsymbol{\Sigma}_q|}} \quad (32)$$

$$BC(p,q) = \exp(-D_B(p,q)) \quad (33)$$

Here, D_B is called the *Bhattacharyya distance* with $\boldsymbol{\Sigma} = (\boldsymbol{\Sigma}_p + \boldsymbol{\Sigma}_q)/2$. In summary, we calculate the similarity $\mathcal{U}(\mathcal{M}_i, \mathcal{M}_j)$ between two local models according to (30) where we set $\mathcal{V}(p,q) = BC(p,q)$. If

$$\mathcal{U}(\mathcal{M}_i, \mathcal{M}_j) > \theta_{merge}, \quad (34)$$

we merge $\mathcal{M}_i$ and $\mathcal{M}_j$ to a new expert $\mathcal{M}_{new}$. Thereby, θ_{merge} is some predefined threshold denoting the maximum overlap between local experts. The NGnet is finally adapted such that

$$\boldsymbol{\mathcal{M}}^* = (\boldsymbol{\mathcal{M}} \backslash \{\mathcal{M}_i, \mathcal{M}_j\}) \cup \{\mathcal{M}_{new}\}. \quad (35)$$

The creation of the new model $\mathcal{M}_{new}$ involves the merging of multivariate Gaussian pdfs ($G(\boldsymbol{y}, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$ and $G(\boldsymbol{y}, \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)$ to $G(\boldsymbol{y}, \boldsymbol{\mu}_{new}, \boldsymbol{\Sigma}_{new})$ as well as $G(\boldsymbol{c}, \boldsymbol{\alpha}_i, \boldsymbol{\Gamma}_i)$ and $G(\boldsymbol{c}, \boldsymbol{\alpha}_j, \boldsymbol{\Gamma}_j)$ to $G(\boldsymbol{c}, \boldsymbol{\alpha}_{new}, \boldsymbol{\Gamma}_{new})$) and the determination of ρ_{new}. The latter is set to $\rho_{new} = \rho_i + \rho_j$, whereas for the former problem multiple approaches exist.

In general, we would like to cluster K Gaussian pdfs $G(\boldsymbol{a}, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i), i = 1, \ldots, K$ (in our case K=2) and represent the cluster by a Gaussian pdf $G(\boldsymbol{a}, \boldsymbol{m}, \boldsymbol{S})$ such that

$$\sum_{i=1}^{K} \omega_i \cdot D(G(\boldsymbol{a}, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) || G(\boldsymbol{a}, \boldsymbol{m}, \boldsymbol{S})) \quad (36)$$

is minimal. Thereby, D denotes a measure for the divergence between the pdfs and $0 \leq \omega_i \leq 1$ weights with $\sum_i \omega_i = 1$. In our scenario we set $\omega_r = \rho_r / \sum_{k \in \{i,j\}} \rho_k$ with $r \in \{i,j\}$ such that important local experts dominate the merging over unimportant ones.

Existing approaches for solving such a minimization problem mainly differ in the used divergence measure D. Here, two of them are of particular interest: Firstly, *Kullback-Leibler divergence*-based clustering [16] and, secondly, *Jenson-Shannon divergence*-based clustering [17]. The results of both approaches are exemplarily depicted in Fig. 2.

Algorithm 1 Merge Local Models

```
Calculate the similarity U(M_i, M_j),   ∀{M_i, M_j}
{a, b} ← arg max_{i,j} U(M_i, M_j)
while U(M_a, M_b) > θ_merge do
    Merge M_a and M_b to M_new
    M ← (M \ {M_a, M_b}) ∪ {M_new}
    Update the similarity U(M_i, M_j),   ∀{M_i, M_j}
    {a, b} ← arg max_{i,j} U(M_i, M_j)
end while
```

As can be seen, the divergence measures result in different clusters. More precisely, the Gaussian obtained via Kullback-Leiber divergence-based clustering is larger than the one obtained by Jenson-Shannon divergence-based clustering. It is nearly the union of the individual Gaussian's receptive fields. Thus, Kullback-Leibler divergence-based clustering seems to be the appropriate technique when the receptive fields of Gaussians should be joined. However, it is inappropriate for joining (normalized) probability distributions, for which Jenson-Shannon divergence-based clustering yields better results. Since the competition between the local experts of an NGnet overwrites the normalization of $p(\boldsymbol{y}|i, \boldsymbol{\Theta})$ (see (8)) we used Kullback-Leibler divergence-based clustering to construct $G(\boldsymbol{y}, \boldsymbol{\mu_{new}}, \boldsymbol{\Sigma}_{new})$ of the new local expert $\mathcal{M}_{new}$. In contrast, Jenson-Shannon divergence-based clustering is used for calculating $G(\boldsymbol{c}, \boldsymbol{\alpha_{new}}, \boldsymbol{\Gamma}_{new})$.

In summary, the merging of local models can be done using the greedy strategy depicted in Algorithm 1.

IV. Categorization on Adaptive Feature Spaces

To incrementally learn categories during online operation, the learning system has to continuously build up associations between sequentially arising stimuli and category labels. The adaptive NGnet we presented is suited to cope with this task. However, it is desirable to additionally build feature representations which facilitate the categorization task. Such a feature extraction should improve over time as more information is accumulated and, thus, has to run in parallel to categorization. Most present approaches to online category learning disregard this fact, insofar as they rely on a fixed set of (predefined) features (see [3] for an exception of this). In contrast, our system incorporates both feature extraction and categorization (see Fig. 3 (a)) such that an input pattern $\boldsymbol{x}$ is first transformed into its feature representation $\boldsymbol{y}$ which is subsequently used for categorization. Both parts are subject to simultaneous incremental learning. Consequently, the categorization is carried out using a continuously changing feature space.

A. Category-Discriminative Feature Extraction

Our feature extraction layer is composed of two stages. In the first stage we extract category-discriminative features by means of Maximizing Renyi's Mutual Information (MRMI) [18]. Secondly, we apply Principle Component Analysis (PCA) to obtain principle feature dimensions and perform dimensionality reduction.

The aim of the MRMI algorithm is to learn a transformation matrix $\boldsymbol{R}$ which extracts category-discriminative features $\boldsymbol{y} = \boldsymbol{R} * \boldsymbol{x}$. It makes use of the information-theoretic principle

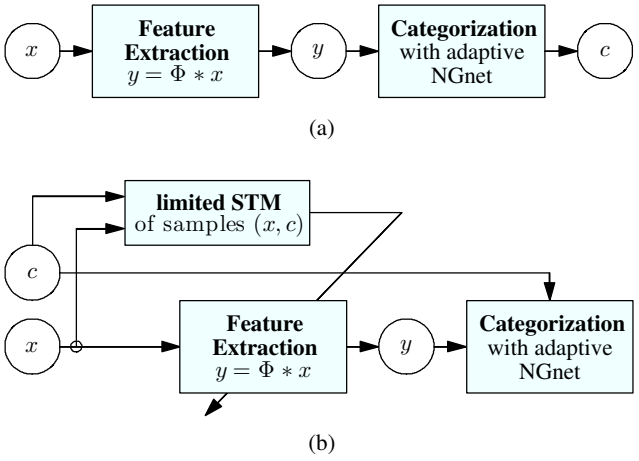


Fig. 3. The framework for the categorization using an adaptive feature space: (a) Input samples $\boldsymbol{x}$ become transformed into feature patterns $\boldsymbol{y}$ which are subsequently categorized. (b) The feature extraction layer is simultaneously learned using a limited STM buffer (see text for details).

of maximizing the mutual information between the features and category labels. Therefore, let $\boldsymbol{X}$ and C be a stochastic input stimulus vector and its associated category label with $\boldsymbol{x}(k)$ and $c(k)$ being random realizations of them. The mutual information $I(\boldsymbol{Y}; C) = H(\boldsymbol{Y}) - H(\boldsymbol{Y}|C)$ with $\boldsymbol{Y} = \boldsymbol{R} * \boldsymbol{X}$ then describes the amount of information the random feature vector $\boldsymbol{Y}$ carries about the category C. In [18] it has been proposed to replace Shannon's definition of entropy $H(\boldsymbol{Y})$ by Renyi's quadratic entropy $H_2(\boldsymbol{Y})$ which can be efficiently calculated using Parzen windows:

$$I(\boldsymbol{Y}; C) \cong H_2(\boldsymbol{Y}) - H_2(\boldsymbol{Y}|C) \tag{37}$$

$$H_2(\boldsymbol{Y}) \cong -\log \frac{1}{K} \sum_{k=1}^{K} G(\boldsymbol{y}(k) - \boldsymbol{y}(k-1), 2\sigma^2 \boldsymbol{I}) \tag{38}$$

Here, $G(\boldsymbol{z}, \sigma^2 \boldsymbol{I}) = \exp(-\frac{1}{2} \frac{\boldsymbol{z}^T \boldsymbol{z})}{2\sigma^2})$ is a Gaussian kernel evaluated at $\boldsymbol{z}$, where the kernel is centered at the origin and has a diagonal isotropic covariance matrix.

Consider a training set composed of samples $\boldsymbol{x}_j(k)$ as representatives of category j where $\boldsymbol{y}_j(k) = \boldsymbol{R} * \boldsymbol{x}_j(k)$. Furthermore, let K_j denote the number of samples belonging to category j, K_c the number of categories, and $K_T = \sum_{j=1}^{K_c} K_j$ the length of the training set. Then the information-theoretic criterion can be formulated as follows:

$$I(\boldsymbol{Y}; C) = -\log \frac{1}{K_T} \sum_{k=1}^{K_T} G(\boldsymbol{y}(k) - \boldsymbol{y}(k-1), 2\sigma^2) + \sum_{j=1}^{K_c} \left(\frac{K_j}{K_T} \log \frac{1}{K_j} \sum_{k=1}^{K_j} G(\boldsymbol{y}_j(k) - \boldsymbol{y}_j(k-1), 2\sigma^2) \right) \tag{39}$$

Consequently, $\boldsymbol{R}$ can be learned via stochastic gradient ascent on $I(\boldsymbol{Y}; C)$.

To perform dimensionality reduction we apply PCA on the feature space learned via MRMI. Without loss of generality we assume the stimuli $\boldsymbol{X}$ to be white with zero mean and unit variance. Then the covariance of the feature vectors $\boldsymbol{Y}$ is $cov(\boldsymbol{Y}, \boldsymbol{Y}) = \boldsymbol{R} * \boldsymbol{R}^T$. Let $\boldsymbol{\Psi} = [\boldsymbol{\psi}_1, \boldsymbol{\psi}_2, \ldots, \boldsymbol{\psi}_N]$ be the eigenvectors of $\boldsymbol{R} * \boldsymbol{R}^T$ and $\boldsymbol{\Lambda} = [\lambda_1, \lambda_2, \ldots, \lambda_N]$ the associated eigenvalues. Then the principle feature component space can be obtained by

$$\boldsymbol{Y} = \boldsymbol{\Phi} * \boldsymbol{X} = (\boldsymbol{\Psi}^T * \boldsymbol{R}) * \boldsymbol{X}. \tag{40}$$

The eigenvalues $\boldsymbol{\Lambda}$ represent the distribution of the features' energy among each of the principle components. Consequently, one can restrict feature extraction to the principle feature components whose cumulative energy content exceeds a threshold θ_{PCA} with $0 \leq \theta_{PCA} \leq 1$. Therefore, let the columns of $\boldsymbol{\Psi}$ be arranged such that their associated eigenvalues are sorted in descending order and let $\mathcal{E}(k)$ be the cumulative energy content among the first k principle components, i.e. $\mathcal{E}(k) = \sum_{i=1}^{k} \lambda_i / \sum_{j=1}^{N} \lambda_j$. Then we choose $\boldsymbol{\Psi} = [\boldsymbol{\psi}_1, \boldsymbol{\psi}_2, \ldots, \boldsymbol{\psi}_k]$ with $\mathcal{E}(k-1) < \theta_{PCA} \leq \mathcal{E}(k)$.

B. Adaptation to Changing Feature Space

In an incremental learning system the feature space $\mathcal{S}_y$ (on which categorization is based on) continuously changes. Therefore, the NGnet has to continuously adapt its local experts, i.e. their input probability distributions $p(\boldsymbol{y}|i, \boldsymbol{\Theta}) = G(\boldsymbol{y}, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$. Since we apply a linear feature extraction, a step-wise feature space adaptation $\mathcal{S}_y \rightarrow \tilde{\mathcal{S}}_y$ can be described by an affine transformation

$$\tilde{\boldsymbol{y}} = \boldsymbol{A} * \boldsymbol{y} + \boldsymbol{b} \tag{41}$$

with $\boldsymbol{y} \in \mathcal{S}_y$ and $\tilde{\boldsymbol{y}} \in \tilde{\mathcal{S}}_y$. Thereby, it is

$$\boldsymbol{A} = \tilde{\boldsymbol{\Phi}} * \boldsymbol{\Phi}^{-1} = \tilde{\boldsymbol{\Psi}}^T * \tilde{\boldsymbol{R}} * \boldsymbol{R}^{-1} * \boldsymbol{\Psi} \tag{42}$$

$$\boldsymbol{b} = \boldsymbol{0} \tag{43}$$

where $\boldsymbol{\Psi}, \boldsymbol{R}, \boldsymbol{\Phi}$ as well as $\tilde{\boldsymbol{\Psi}}, \tilde{\boldsymbol{R}}, \tilde{\boldsymbol{\Phi}}$ denote the feature extraction matrices before and after the training of the feature extraction layer. Consequently, the new Gaussian pdfs under this transformation are

$$G(\tilde{\boldsymbol{y}}, \boldsymbol{A} * \boldsymbol{\mu}_i + \boldsymbol{b}, \boldsymbol{A} * \boldsymbol{\Sigma}_i * \boldsymbol{A}^T). \tag{44}$$

C. Incremental Learning Framework

Our framework for the simultaneous learning of categories and feature representations is illustrated in Fig. 3 (b). It makes use of a limited short-term memory buffer (STM) of samples $(\boldsymbol{x}, c)$ which are used to train the feature extraction layer. In summary, the incremental learning is carried out according to the following algorithm:

Algorithm 2 Incremental Learning

Initialize the feature extraction to $\boldsymbol{R} = \boldsymbol{I}$, $\boldsymbol{\Psi} = \boldsymbol{I}$
Initialize an empty NGnet
for all training samples $(\boldsymbol{x}_t, c_t)$ **do**
  Update the STM with $(\boldsymbol{x}_t, c_t)$
  Train the feature extraction on the entries of the STM
  Adapt the NGnet to the new feature space
  Calculate the sample $(\boldsymbol{y}_t, c_t)$
  Update the NGnet according to $(\boldsymbol{y}_t, c_t)$
end for

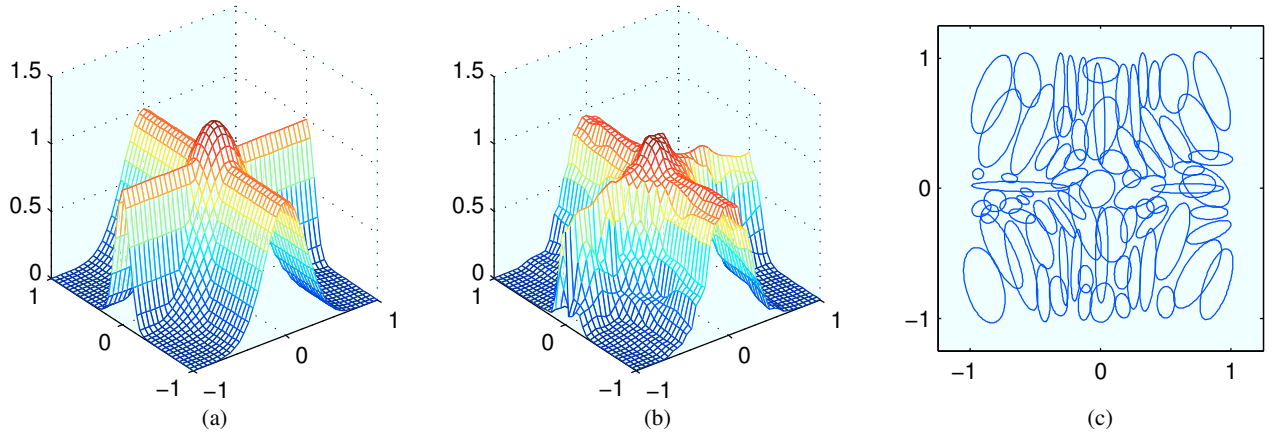


Fig. 4. The NGnet applied to the cross function: (a) the target function, (b) the network's approximation, and (c) the receptive fields of the local experts.

V. Simulation Results

To test our framework we performed simulations in the domains of function approximation and categorization. For the latter we further applied fixed and adaptive feature spaces. The used parameter settings are shown in Table I.

A. Function Approximation

To illustrate the NGnets ability to incrementally acquire local experts, we initially applied it for the approximation of the cross function which is defined as

$$c = \max\left\{\exp(-10y_1^2),\ \exp(-50y_2^2),\ 1.25 \cdot \exp(-5(y_1^2 + y_2^2))\right\}. \quad (45)$$

Fig. 4 shows the results of this simulation. As can be seen, the adaptive NGnet adequately approximates the cross function. The receptive fields of the local experts become oriented along the dimension of minimal gradient. Furthermore, the partitioning of the input space becomes fine-grained in regions where the cross function's gradient is high, whereas just few experts are needed in regions where the cross function does not change a lot. Fig. 5 plots the evolution of the root mean squared error (RMSE) as well as the number of local expert. The plots illustrate that the first training samples yield a sharp increase in the number of local models whereas the error significantly decreases. After that, the network allocates new experts until convergence is reached. However, even after saturation in the number of experts, the error further decreases since experts continue to specialize to certain input regions.

B. Categorization - Artificial Example

In a second experiment we applied the adaptive NGnet to an artificial binary categorization problem. Therefore, we

TABLE I
Parameter settings used in the experiments

Exp.	η	θ_{remove}	θ_{assign_1}	θ_{assign_2}	θ_{split}	θ_{merge}	ξ_1	ξ_2	θ_{PCA}
A	0.01	0.01	0.1	0.2	0.1	0.7	1	4	-
B&C	0.01	0.01	1	1	0.1	0.8	1	4	0.95

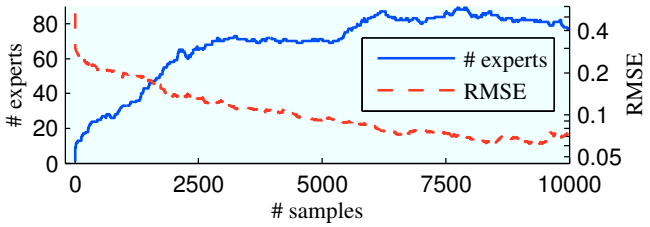


Fig. 5. The evolution of the number of experts and the root mean squared error (RMSE) for the approximation of the cross function.

randomly created input patterns $\boldsymbol{x} \in \Re^4$ with $x_i \sim \mathcal{N}(5, 1)$. The target categories of the input patterns were defined as

$$c = \text{sign}\left[x_1 \cdot x_2 - x_3 \cdot x_4\right] \in \{-1, 1\}. \quad (46)$$

A test set composed of 2500 input samples was used to calculate the categorization error. Thereby, we applied the sign function to the continuous outputs of the NGnet to yield a category decision. We further evaluated our frameworks ability to simultaneously extract category-discriminative features. Therefore, we compare the results of two simulation runs: with and without using feature extraction.

The results are shown in Fig. 6. In (a) we can identify three major training phases for both simulations: (1) Initially, the number of experts increases due to the (one-shot) memorization of prototypical category members. (2) After that, the number of experts saturates since new input samples can be sufficiently described by already memorized prototypes. (3) Finally, the number of experts decreases to a minimum which is maintained afterwards. In this phase, the NGnet is able to find commonalities among the prototypes (overlap between the experts is large) and merges them accordingly. This results in a categorizer of minimum complexity as well as an improved generalization ability.

The plots also show that the incorporation of feature extraction enables our framework to find a less complex categorizer at a much faster time scale than if we do not extract category-discriminative features (see inset for a zoomed plot). This is due to the fact that the extracted

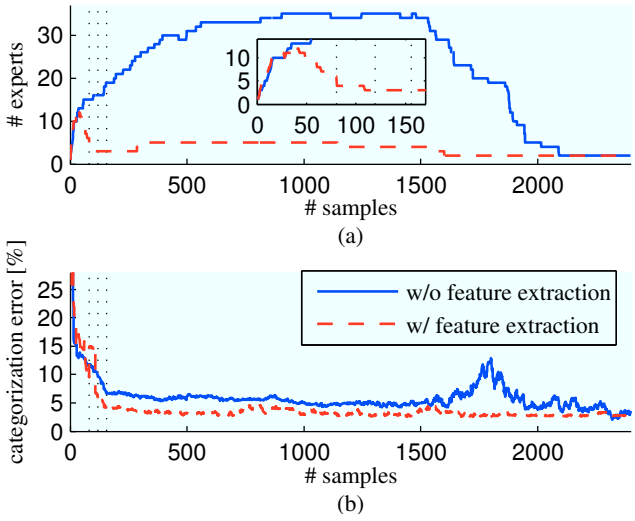


Fig. 6. The evolution of (a) the number of experts and (b) the categorization error for the artificial categorization problem. The plots correspond to simulation runs with and without simultaneous learning of the feature extraction. For the test incorporating feature extraction the vertical dotted lines indicate instances in time where feature dimensions have been pruned. The inset in (a) shows a zoomed plot for the number of experts.

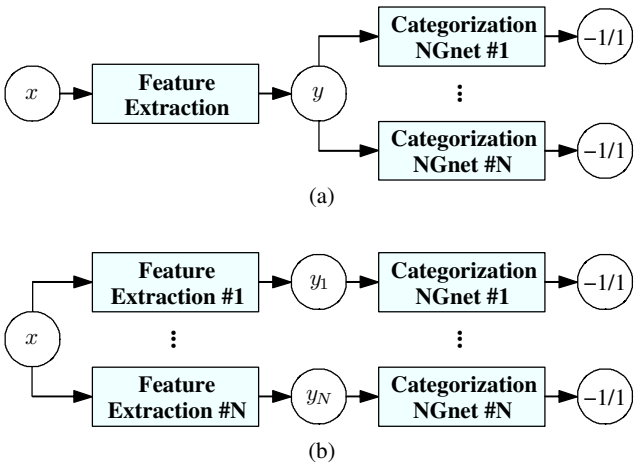


Fig. 7. System architectures used for the categorization of the Letter Recognition dataset: Either (a) a common feature space on which all NGnets operate or (b) individual category-specific feature spaces are used.

features facilitate the generalization ability of the NGnet, since more category prototypes coincide at similar locations of the feature space (which results in an increased overlap between experts). The finally extracted feature f is given by $f \approx (x_1 + x_2) - (x_3 + x_4)$ which is a suitable linear approximation of the target function. Even though a less complex categorizer is learned using feature extraction, it yields a better categorization performance compared to not using feature extraction (see (b)). Overall, we achieve $\approx 3\%$ categorization error.

C. Categorization - Letter Recognition

Next, we evaluated the categorization framework on the Letter Recognition Data Set of the UCI Machine Learning Repository [19]. This database contains 16-dimensional integer-type feature vectors of 20000 pixel images of the 26 English capital letters. For the following test we restricted the set to the letters A, E, I, M, Q, U, Y which results in training and test sets composed of 4114 and 1317 samples, respectively. We sequentially presented training samples to the categorization system depicted in Fig. 7 (a). We used 7 NGnets, each of them acting as a binary categorization module for one of the letters. We further used one feature extraction module which produces the class-discriminative feature space the NGnets operate on.

The results of this simulation are shown in Fig. 8 where we plot the number of experts as well as the categorization error for each of the NGnets. As can be seen, curves similar to the previous example are obtained: the categorization error quickly decreases whereas the number of experts for each module initially increases, then saturates, and finally decreases to a minimum as the networks generalize among their experts. However, it can be observed that some letters are easier to categorize than others. This is exemplarily shown for the letters I and Q. The NGnet corresponding to I remains less complex than the one corresponding to Q throughout the training. It necessitates $\approx$ 500 samples less to generalize, which is indicated by the drops in the number of experts. Even though a less complex NGnet is used for the recognition of an I, the performance is better than for the recognition of a Q. These results demonstrate the ability of our network to autonomously adapt its internal complexity according to the difficulty of the task. It is further shown, that the categorization task becomes easier over time as robust class-discriminative features become extracted. Consequently, at the end of the training we observe networks of smaller size compared to the beginning.

Next, we applied the framework depicted in Fig. 7 (b) to the same task. In this test each NGnet comprises an individual feature space which discriminates the corresponding letter from all other letters. Consequently, much of the knowledge about the categories shifts into the extracted features by which less complex categorization modules are needed. This is confirmed by the results shown in Fig. 9 where we compare the average number of experts per NGnet as well as the overall categorization error to the framework of Fig. 7 (a). We observe a significant decrease in the size of the NGnets whereas the error just slightly increases. Overall, we obtain a miscategorization rate of $\approx$ 2-3 %.

To allow a comparison to a state of the art approach, we further applied incremental support vector machines (iSVMs) [20] to the letter recognition task. More precisely, 7 iSVMs have been used as binary categorizers for the single letters. Fig. 9 plots the performance curves for two different parameter settings of the iSVMs. The first setting was chosen such that the categorization errors for the iSVMs and our framework are similar. In this case, the iSVMs employ significantly more complex classifiers (in terms of the mean number of support vectors) than our framework. The second parameter setting was chosen to obtain comparably complex iSVMs. In this case, our framework shows much better performance than the iSVMs. From this we conclude that the use of individual feature spaces within our framework yields a better complexity-to-error balance.

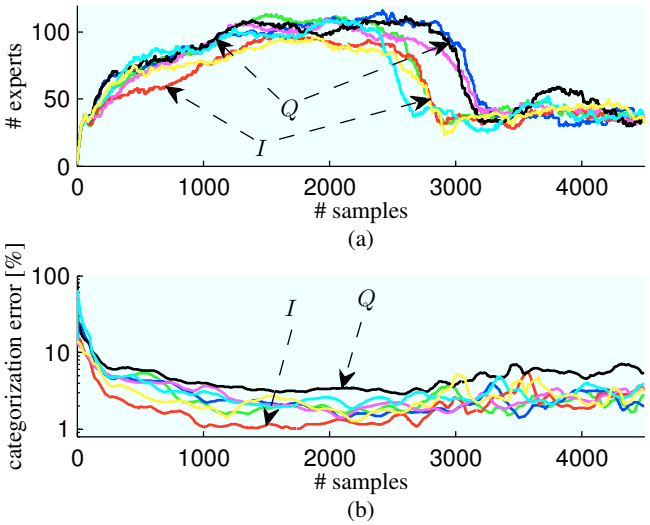


Fig. 8. In (a) the number of experts and (b) the categorization error of the NGnets, which carry out the recognition of single letters, are shown.

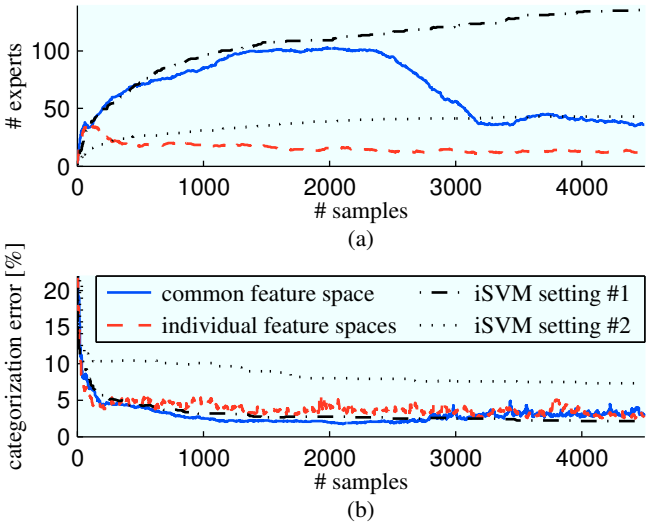


Fig. 9. The mean number of experts per NGnet and the overall categorization error are compared for the categorization systems using either a common feature space or individual category-specific feature spaces. The plots further show the mean number of support vectors and the categorization error for the use of iSVMs with two different parameter settings.

VI. Conclusion

Adaptive learning algorithms are essential for online applications where training samples are obtained one by one. Here, we presented a truly sequential training procedure for an NGnet. Key to the network model are local model manipulation mechanisms for network growth and pruning. Following a probabilistic interpretation of NGnets, we presented criteria and mechanisms for the allocation, pruning, splitting, and merging of hidden units. Our simulation results confirmed the network's ability to autonomously adapt its size and its parameters according to task requirements.

We further introduced a framework in which categorization is carried out using simultaneously extracted category-discriminative features. Our simulations showed that these features facilitate the categorization carried out by the NGnet, by which a faster training, significant less complex networks, and an improved generalization could be achieved. A comparison to the use of incremental SVMs showed a favorable performance of our framework.

We additionally illustrated the use of different system configurations: categorization using a common feature space or using category-specific feature spaces. The automatic construction of suitable system configurations according to the requirements of the task as well as a thorough performance comparison to other existing approaches will be part of our future work.

References

[1] K. Samejima and T. Omori, "Adaptive internal state space construction method for reinforcement learning of a real-world agent.," *Neural Networks*, vol. 12, no. 7-8, pp. 1143–1155, Oct 1999.

[2] S. Suresh, N. Sundararajan, and P. Saratchandran, "A sequential multi-category classifier using radial basis function networks," *Neurocomputing*, vol. 71, pp. 1345–1358, 2008.

[3] S. Ozawa, S.L. Toh, S. Abe, S. Pang, and N. Kasabov, "Incremental learning of feature space and classifier for face recognition.," *Neural Networks*, vol. 18, no. 5-6, pp. 575–584, 2005.

[4] J. Moody and C. J. Darken, "Fast learning in networks of locally-tuned processing units," *Neural Computation*, vol. 1, pp. 281–294, 1989.

[5] V.M. Rivas, M.G. Arenas, J.J. Merelo, and A. Prieto, "EvRBF: evolving RBF neural networks for classification problems," in *Proc. Int. Conf. on Appl. Informatics and Comm.*, 2007, pp. 98–103.

[6] M.D. Pérez-Godoy, A.J. Rivera, M. José del Jesus, and I. Rojas, "CoEvRBFN: An approach to solving the classification problem with a hybrid cooperative-coevolutive algorithm," in *Computational and Ambient Intelligence*, pp. 324–332. Springer, 2007.

[7] J. Platt, "A resource-allocating network for function interpolation," *Neural Computation*, vol. 3, no. 2, pp. 213–225, 1991.

[8] V. Kadirkamanathan and M. Niranjan, "A function estimation approach to sequential learning with neural networks," *Neural Computation*, vol. 5, no. 6, pp. 954–975, 1993.

[9] Y. Lu, N. Sundararajan, and P. Saratchandran, "A sequential learning scheme for function approximation using minimal radial basis function neural networks," *Neural Comp.*, vol. 9, no. 2, pp. 461–478, 1997.

[10] G.-B. Huang, P. Saratchandran, and N. Sundararajan, "A generalized growing and pruning RBF (GGAP-RBF) neural network for function approximation," *IEEE Transactions on Neural Networks*, vol. 16, no. 1, pp. 57–67, 2005.

[11] L. Xu, "RBF nets, mixture experts, and Bayesian Ying-Yang learning," *Neurocomputing*, vol. 19, pp. 223–257, 1998.

[12] R.A. Jacobs, M.I. Jordan, S.J. Nowlan, and G.E. Hinton, "Adaptive mixtures of local experts," *Neural Comp.*, vol. 3, pp. 79–87, 1991.

[13] G. Bugmann, "Normalized Gaussian radial basis function networks," *Neurocomputing*, vol. 20, no. 1-3, pp. 97–110, 1998.

[14] M. Sekino, D. Katagami, and K. Nitta, "State space self-organization based on the interaction between basis functions," in *Proc. IEEE/RSJ IROS*, 2005, pp. 2929–2934.

[15] M. Sato and S. Ishii, "On-line EM algorithm for the normalized Gaussian network.," *Neural Comp.*, vol. 12, no. 2, pp. 407–432, 2000.

[16] J.V. Davis and I. Dhillon, "Differential entropic clustering of multivariate Gaussians," in *Advances in NIPS 19*. MIT Press, 2006.

[17] T.A. Myrvoll and F.K. Soong, "On divergence based clustering of normal distributions and its application to HMM adaptation," in *Proc. EUROSPEECH*, 2003.

[18] K.E. Hild, D. Erdogmus, K. Torkkola, and J.C. Principe, "Feature extraction using information-theoretic learning," *IEEE Trans. on Pattern Anal. and Machine Intell.*, vol. 29, no. 9, pp. 1385–1392, 2006.

[19] P.W. Frey and D.J. Slate, "Letter recognition using holland-style adaptive classifiers," *Machine Learning*, vol. 6, pp. 161–182, 1991.

[20] C.P. Diehl and G. Cauwenberghs, "SVM incremental learning, adaptation and optimization," in *Proc. IJCNN*, 2003.