6

Ensemble Learning for Multi-source Information Fusion

Jörg Beyer, Kai Heesche, Werner Hauptmann, Clemens Otte, and Rudolf Kruse

Abstract. In this chapter, we propose a new ensemble learning method. The main objective of this approach is to jointly use data-driven and knowledge-based submodels, like mathematical equations or rules, in the modeling process. The integration of knowledge-based submodels is of particular interest, since they are able to provide with information not contained in the data. On the other hand, data-driven models can complement the knowledge-based models with respect to input space coverage. For the task of appropriately integrating the different models, a method for partitioning the input space for the given models is introduced. Using that kind of ensembles, the advantages of both models are combined, i.e., robustness and physical transparency of the knowledge-based models and approximation abilities of the data-driven learning. The benefits of this approach are demonstrated for a real-world application.

Jörg Beyer
Siemens AG - CT IC 4, Otto-Hahn-Ring 6, 80200 Munich, Germany
Otto-von-Guericke-University Magdeburg - School of Computer Science,
Universitätsplatz 2, 39106 Magdeburg, Germany
e-mail: joerg.beyer.ext@siemens.com

Kai Heesche
Siemens AG - CT IC 4, Otto-Hahn-Ring 6, 80200 Munich, Germany

Werner Hauptmann
Siemens AG - CT IC 4, Otto-Hahn-Ring 6, 80200 Munich, Germany

Clemens Otte
Siemens AG - CT IC 4, Otto-Hahn-Ring 6, 80200 Munich, Germany

Rudolf Kruse
Otto-von-Guericke-University Magdeburg - School of Computer Science,
Universitätsplatz 2, 39106 Magdeburg, Germany

D.K. Pratihar and L.C. Jain (Eds.): Intelligent Autonomous Systems, SCI 275, pp. 123–141.
springerlink.com

1 Introduction

Modern technical systems are characterized by an increasing degree of sophisticated behavior. The traditional way of modeling has been by mathematical equations representing the physical behavior. However, the identification of parameters is time-consuming and expensive. Data-driven models, like artificial neural networks, can be used to approximate physical phenomena. But, the data-driven modeling approach usually suffers from the lack of physical understanding of the model parameters. The resulting model only relies on the training data and does not use any other information source available. Thus, it is desirable to combine available information in terms of knowledge-based models, i.e., models which are based on domain or process knowledge, designed without training data and to complement this information with the data-driven approach.

The integration of knowledge-based submodels has several advantages:

- enhancing the interpretability, i.e., a domain expert can easily comprehend the decisions,
- providing information not contained in the training data and
- reducing the amount of required training data.

For these reasons, an important factor for the generation of adequate models of a technical system is the use of available information in terms of knowledge-based models and the supplementation of this information by data-driven models learnt on the training data. Since a knowledge-based model represents a particular subsystem, information with respect to its validity has to be included in the overall model.

The objective of the proposed approach is to generate an ensemble that is able to integrate the available knowledge-based submodels and to complement these submodels by data-driven ones. Using that kind of ensembles the advantages of both models are combined, i.e., the robustness and physical transparency of the knowledge-based models and the approximation abilities of data-driven learning.

The use of multiple models is also motivated by the paradigm that different partial models can complement each other by appropriate compensation of weaknesses and strengths of the individual models. Much of the work on ensemble techniques has strong parallels with the research on information fusion (IF) systems. In common with the research on IF, several architectures exist and different combination schemes have been developed. Later in this chapter, we give a review of IF.

The chapter is organized as follows: In Secton 2, an introduction of IF is given and Section 3 describes different methods for creating ensembles. In Section 4, two ensemble models for combining data-driven and knowledge-based models are proposed. In Section 5, some experiments on a real-world application are outlined. Section 6 concludes the study.

2 Multiple Source Fusion

Information fusion is an important technique in different application domains, such as sensor fusion [9], identity verification [2], signal and image processing [5], and others. Due to the heterogeneity of the applications, several definitions of the term: *information fusion* exist. In Section 2.1, some definitions of IF are stated and the adopted definition of IF is given. A classification of IF is described in Section 2.2.

2.1 Definition

There exist many definitions of IF or data fusion. In [25], IF is described as a "multilevel, multifaceted process dealing with the automatic detection, association, correlation, estimation, and combination of data and information from multiple sources." This is a general definition that suggests the combination of data or information without specifying its objective. Wald considers data fusion as formal framework that formulates means and tools for the combination of data from different sources [23]. In this definition, the focus lies on the framework used to fuse data. Wald also states that data fusion "aims at obtaining information of greater quality". The term *quality* means that the fused information is somehow more appropriate to the application than the original information. The most general definition comprising any type of source, knowledge, and resource used to fuse different pieces of information is given by Dasarathy [7], which states that IF "encompasses the theory, techniques and tools created and applied to exploit the synergy in the information acquired from multiple sources (sensors, databases, information gathered by humans beings, etc.) in such a way that the resulting decision or action is in some sense better (qualitatively or quantitatively, in terms of accuracy, robustness, etc.) than would be possible, if any of these sources were used individually without such synergy exploitation."

Fusion implies the combination of information from more than one source. There are different reasons for fusion of multiple sources:

- The combined solution is able to attain more accurate, transparent, and robust results, since the different information sources can complement each other with respect to their strengths and weaknesses.
- A model that depends on a single source is not robust in the sense that if the single source is erroneous, the whole model is affected. Models based on fused information sources are more robust, since other sources are able to compensate erroneous information.

Here, we chose to use IF as the process of merging and integrating heterogeneous information components from multiple sources, for instance, in the form of sensors, human experts, symbolic knowledge, or physical process models.

2.2 Classification of Information Fusion

In this study, complementary and cooperative information sources are distinguished. They are discriminated with respect to the relationships among the information sources. In complementary fusion, each source provides with information from a different region of the input space, i.e., their responsibilities do not overlap. Locally, these sources provide with a high performance. However, outside their regions, the results are not valid. Cooperative fusion means that the information is shared among several information sources in the same region of the input space and has to be fused for a more complete modeling of the underlying process.

3 Ensemble Models

There exist many approaches, which address the issue of learning and combining local models. The resulting model, referred to as ensemble, is generally more accurate than any of the submodels generating the ensemble. Both empirical [14] and theoretical [13], [17] research has demonstrated that in a good ensemble, the submodels are accurate on different parts of the input space, so that they complement one another. Figure 1 shows a common ensemble model.

The algorithms for learning local models can be discriminated with respect to several aspects: the architecture (parallel or sequential learning of the submodels), the way they divide the training data into subsets, or how they fuse the outputs of the local models.

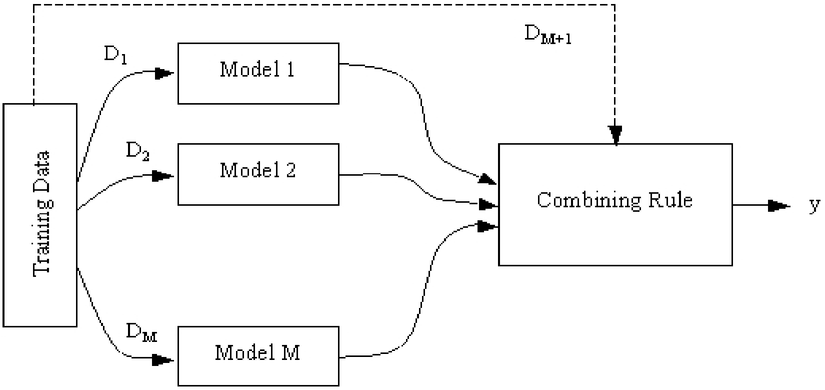

Fig. 1 A common ensemble model. The ensemble members are trained on possibly different data sets D_j. The dashed line indicates that the Combining Rule can be trainable dependent on the combining method

3.1 Stacked Generalization

Stacked generalization (or stacking) creates an ensemble of submodels, whose outputs are used as inputs to a second level combiner to learn the mapping between the ensemble member outputs and the target values [26].

The basic idea is that the outputs of the ensemble members have information that can be used to construct good combinations of the members and a procedure is sought for combining them. In the first step, a set of submodels is trained (with possibly different training data sets, submodel parameters, etc.). The outputs of these submodels and their corresponding true target values are then used as input/output training pairs for the second level combiner.

3.2 Boosting

Boosting is a method for improving the accuracy of any given learning algorithm [19]. The submodels in a boosting ensemble are trained sequentially on training data that has been filtered by the previously trained submodels in the ensemble. The boosting procedure is as follows: the first submodel is trained with a random subset of the training data. The training data set for the second submodel is chosen as the most *informative* given the first submodel. In case of classification, the submodel is trained on training data, the half of which is misclassified by the first submodel, and the other half is correctly classified. The third submodel is trained with input patterns, on which the first two submodels disagree. The submodels are combined using either averaging or a voting scheme.

A popular variation of the original boosting algorithm is AdaBoost (*adaptive boosting*) [11]. In AdaBoost, submodels can be added until some desired training error has been achieved. For that purpose, each training pattern receives a weight that determines its probability of being selected for the training of a new submodel. If a training pattern is misclassified, its probability of being selected in a subsequent submodel is increased. In this way, training data of consecutive submodels are focused on hard to separate patterns. This process can be repeated to form an ensemble, whose joint decision has arbitrarily high accuracy on the training set.

3.3 Mixture-of-Experts

The mixture-of-experts (ME) model consists of a set of models, also called experts, that perform a local function approximation [15]. ME models can be described as input-dependent mixture models, which solve problems by the divide-and-conquer strategy, i.e., they learn to decompose complex problems into simpler, easier to solve subproblems. This decomposition is learned by a gate function by partitioning the input space and assigning submodels to these regions. The output y of the ME model for an input vector $\mathbf{x}$ is computed as the combination of the weighted outputs $y\left(\mathbf{x}, \boldsymbol{\theta}_j\right)$ of the M submodels

$$y = \sum_{j=1}^{M} \pi_j y\left(\mathbf{x}, \boldsymbol{\theta}_j\right) , \tag{1}$$

where $\boldsymbol{\theta}_j$ are the parameters of model j, π_j is the j-th output of the gate model and is constrained to $\sum_{j=1}^{M} \pi_j = 1$.

There exist several variations of the ME model, which differ in the kind of training algorithm [16], [24], [4] and gate function [27].

3.4 Piecewise Linear Regression Models

Piecewise linear models assume a different linear behavior of the true function in different regions of the input space. The model described in [10] assumes that the input space can be divided into disjoint regions characterized by different (linear) behavior of the function to be approximated. The model learns the local linear models by an appropriate clustering of the input space.

A switching regression model assumes that the target values are generated by a number of distinct processes [21]. Quandt developed a method for estimating the switching point, i.e., the point where the processes switch, by searching through all possible switch points and finding the maximum of an appropriate likelihood function [18].

4 Fusion of Locally Valid Heterogeneous Models

The fusion of locally valid heterogeneous models is a crucial process during the training and affects the reliability and performance of the results of the integrated model. To assign the available knowledge-based models to the regions of input space, for which they are defined, the fusion rule has to take into account their validity ranges. For this purpose, a gate function, similar to the ME approach, is used.

In Section 4.1, validity functions defined for knowledge-based models, are described. In Section 4.2, we propose an adaptation of the ME model, called heterogeneous mixture-of-experts (HME). The HME model uses the validity functions during the partitioning process to assign knowledge-based models to the correct regions of the input space. In Section 4.2.2, we use a clustering algorithm as gate function that considers the data density and predictive performance of the local models for separation of the input space.

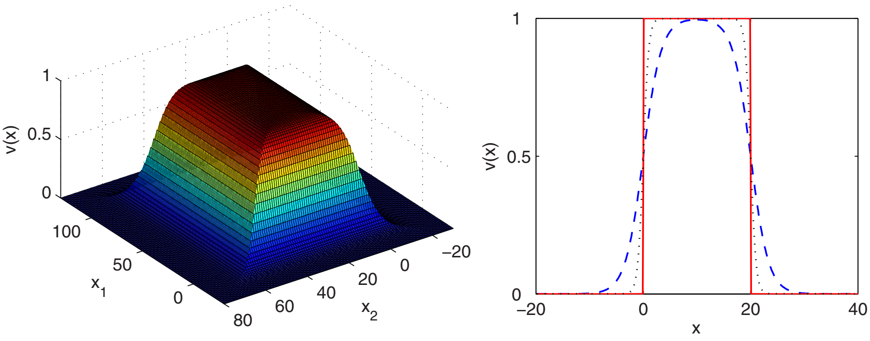

Fig. 2 Left: This figure shows an example of a validity range with the following parameter setting: $l_1 = 0$ and $u_1 = 100$ in one dimension and $l_2 = 0$ and $u_2 = 50$ in the second dimension and a slope $s = 8$. Right: The slope of the borders depends on s for $s = 1$ (*dashed line*), $s = 4$ (*dotted line*), and $s = 100$ (*solid line*)

4.1 Validity Function

The validity function of a knowledge-based model represents the region of input space the model is designed for. The validity function of model j is defined as

$$v_j\left(x^{(n)}\right) = \left(\frac{1}{1+\exp\left(s_j\left(x^{(n)}-u_j\right)\right)} - \frac{1}{1+\exp\left(s_j\left(x^{(n)}-l_j\right)\right)}\right), \quad (2)$$

where l_j and u_j determine the lower and the upper bounds of the validity range and s_j defines the slope of the border. The effect of different values of s_j is shown in Fig. 2. The larger s_j is, the steeper is the slope of the border. In this way, the transition between the local models can be controlled. These parameters are determined by domain expert.

4.2 Heterogeneous Mixture-of-Experts

This sub-section discusses about hetero-geneous mixture-of-experts.

4.2.1 Introduction

In this section, we define heterogeneous mixture-of-experts (HME) to fuse information of multiple information sources [3]. The basic idea of this approach is to additionally include information about the specific validity ranges of the predefined knowledge-based models to be used for the partitioning of the input space. Thus, it is ensured that the predefined models are assigned to those domains of the input space they are explicitly designed for. On the other hand, data-driven models are used to close the gaps between different knowledge-based models with respect to input space coverage.

The HME model can be interpreted as a generating one, i.e., the data are generated by a set of M independent processes, which are randomly selected. Fig. 3 shows an example of an HME model. The introduction of a latent variable $Z = \left\{ z_j^{(n)} : \; j = 1, \ldots, M, \; n = 1, \ldots, N \right\}$ where $z_j^{(n)}$ is 1 if input vector $\mathbf{x}^{(n)}$ was generated by model j and 0 otherwise, and the data set $D = \left\{ \mathbf{x}^{(n)} \in \Re^k, \; t^{(n)} \in \Re, \; n = 1, \ldots, N \right\}$, allows the HME model to be trained with the Expectation-Maximization (EM) algorithm [8]. The probabilistic model can be seen in Fig. 4, which shows the belief network of the HME model. This expresses the assumption that the target $t^{(n)}$ is dependent on the input $\mathbf{x}^{(n)}$ and the multinomial random variable $z^{(n)}$. We define the conditional scalar output $t^{(n)}$ given the input vector $\mathbf{x}^{(n)}$ and the parameter of the model as:

$$P\left(t^{(n)} \left| \mathbf{x}^{(n)}, \Theta \right.\right) = \sum_{j=1}^{M} P\left(z_j^{(n)} \left| \mathbf{x}^{(n)}, \boldsymbol{\theta}_g \right.\right) P\left(t^{(n)} \left| \mathbf{x}^{(n)}, \boldsymbol{\theta}_j \right.\right), \tag{3}$$

where Θ comprises the parameter of the gate $\boldsymbol{\theta}_g$, and of the models $\boldsymbol{\theta}_j$, $j = 1, \ldots, M$. The probability $P\left(t^{(n)} \left| \mathbf{x}^{(n)}, \boldsymbol{\theta}_j \right.\right)$ represents the conditional

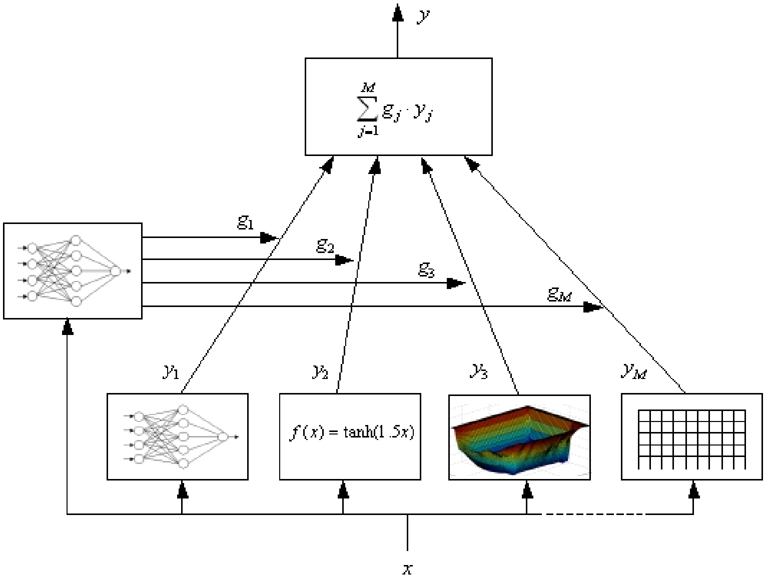

Fig. 3 Architecture of the HME model. The selection of the local models depends on **x**. The gate and the local models may use different feature subsets of the input vector

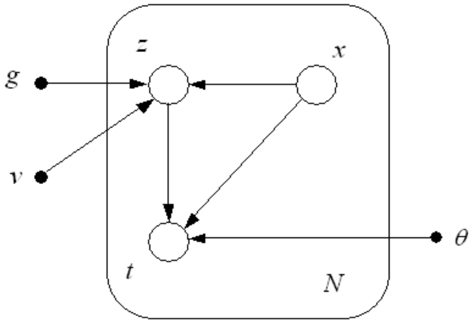

Fig. 4 Graphical representation of the HME model. Random variables will be represented by open circles and deterministic parameters will be denoted by smaller solid circles. The box contains N copies of the nodes shown inside it

densities of target $t^{(n)}$ for model j and $P\left(z_j^{(n)}\left|\mathbf{x}^{(n)},\boldsymbol{\theta}_g\right.\right)$ is the weighting coefficient of model j. The negative log likelihood function is used as error function:

$$\begin{aligned}\log L = &-\sum_{n=1}^{N}\sum_{j=1}^{M} h_j^{(n)} \log P\left(z_j^{(n)}\left|\mathbf{x}^{(n)},\boldsymbol{\theta}_g\right.\right)\\ &-\sum_{n=1}^{N}\sum_{j=1}^{M} h_j^{(n)} \log P\left(t^{(n)}\left|\mathbf{x}^{(n)},\boldsymbol{\theta}_j,z_j^{(n)}\right.\right),\end{aligned} \tag{4}$$

where $h_j^{(n)}$ represents the posterior probability of selecting model j for input vector $\mathbf{x}^{(n)}$. The first term of the right-hand side of equation (4) is the part of the error that the gate contributes to the overall error. It can be interpreted as the entropy of the distribution of the input vectors among the models. The second term of equation (4) presents the error component, which the individual models contribute. This is the cross-entropy among the posterior probability $h_j^{(n)}$ and the probability that model j has generated the target value $t^{(n)}$.

To guide the partitioning of the gate, the posterior probability $h_j^{(n)}$ is computed in the E step:

$$h_j^{(n)} = \frac{v_j P\left(z_j^{(n)}\left|\mathbf{x}^{(n)},\boldsymbol{\theta}_g\right.\right) P\left(t^{(n)}\left|\mathbf{x}^{(n)},\boldsymbol{\theta}_j,z_j^{(n)}\right.\right)}{\sum_{l=1}^{M} v_l P\left(z_l^{(n)}\left|\mathbf{x}^{(n)},\boldsymbol{\theta}_g\right.\right) P\left(t^{(n)}\left|\mathbf{x}^{(n)},\boldsymbol{\theta}_l,z_l^{(n)}\right.\right)}\,. \tag{5}$$

The role of the gate's output and the mapping v_j can be interpreted as follows: Based on the performance the gate's task is to assign a selection probability to the submodel, whereas the mapping v_j evaluates the validity of the model for an input vector. By doing so in (5), the gate will be enforced to decrease the weights of model outputs, if the input vectors are located outside their domains. The particular amount of decrease in weight is dependent on v_j.

The M step then involves finding the optimal set of parameters of the gate and the local models. It decomposes into the following separate optimization problems:

- For the parameter of the gate:

$$L_g = -\sum_{n=1}^{N}\sum_{j=1}^{M} h_j^{(n)} \log P\left(z_j^{(n)} \middle| \mathbf{x}^{(n)}, \boldsymbol{\theta}_g\right) \tag{6}$$

- For the parameter of the data-driven local models:

$$L_j = -\sum_{n=1}^{N}\sum_{j=1}^{M} h_j^{(n)} \log P\left(t^{(n)} \middle| \mathbf{x}^{(n)}, \boldsymbol{\theta}_j, z_j^{(n)}\right) \tag{7}$$

The parameters found on the M step are then used to begin another expectation step. These two steps are repeated until an appropriate convergence criterion, e.g., previously determined number of training iterations, is fulfilled.

4.2.2 Clustering Gating Function

In this section, we use a clustering algorithm to partition the input space and to assign local models to these partitions. It corresponds to the gate function in the HME model. To generate an appropriate partitioning of the input space, not only the data density in the input space is considered but also the performance of local models in the output space and the validity ranges of knowledge-based models.

For each local model one cluster is used. The training process comprises two main steps. First, the assignment of data to cluster prototypes is dependent on the distance between the data and the prototypes, the validity ranges and the predictive performance of the corresponding local models:

$$r_j^{(n)} = \frac{\exp\left(\left\|\mathbf{x}^{(n)} - \boldsymbol{\mu}_j\right\|^2\right)\exp\left(-1/v_j\left\|t^{(n)} - y\left(\mathbf{x}^{(n)}, \boldsymbol{\theta}_j\right)\right\|^2\right)}{\sum_{l=1}^{M}\exp\left(\left\|\mathbf{x}^{(n)} - \boldsymbol{\mu}_l\right\|^2\right)\exp\left(-1/v_l\left\|t^{(n)} - y\left(\mathbf{x}^{(n)}, \boldsymbol{\theta}_l\right)\right\|^2\right)}\ , \tag{8}$$

Second, both the cluster prototypes and the data-driven submodels are trained according to their weighted error for the training data:

$$\boldsymbol{\mu}_j = \frac{\sum_{n=1}^{N} r_j^{(n)}\mathbf{x}^{(n)}}{\sum_{n=1}^{N} r_j^{(n)}}\ , \tag{9}$$

and

$$\mathit{\Delta}\boldsymbol{\theta}_j = \sum_{n=1}^{N} r_j^{(n)}\left(t^{(n)} - y\left(\mathbf{x}^{(n)}, \boldsymbol{\theta}_j\right)\right)\frac{\partial y\left(\mathbf{x}^{(n)}, \boldsymbol{\theta}_j\right)}{\partial\boldsymbol{\theta}_j}\ . \tag{10}$$

At the end of the training process, the clustering algorithm has partitioned the input space among the submodels. While the knowledge-based models are only active inside their validity ranges, the data-driven models are responsible for the remaining input space.

5 Applications of Information Fusion

In Section 5.1, some applications for fusion of analytical and data-driven models are described. In Section 5.2, we describe the deployment of the HME model in a real-world application.

5.1 Combinations of Analytical and Data-Driven Models

In the field of machine learning, several approaches address the combination of analytical and data-driven models. Data-driven models can be either used to approximate nonlinear parts of the process to model parts of the process that are not observable, or as a state or disturbance estimator.

In [20], an RBF-network and an analytical model of the rolling mill process control system are combined. For unknown inputs, the RBF-network produces a correction factor close to one, thus, in these cases, the output of the overall model is determined by the analytical model alone. The advantage of this approach is that a baseline performance can be guaranteed by the analytical model.

Abonyi et al. describe an approach for using first principles models and data-driven ones, e.g., artificial neural networks (ANNs), for Generic Model Control [1]. The first principle model determines the dominant structure of a controller while data-driven models are used as a state or disturbance estimator. Van Lith et al. combine a physical framework, which builds the basis structure and complement it with fuzzy models derived from data [22].

In [12], a partial analytical model is combined with an ANN for dynamic modeling of an industrial fed-batch crystallization process. Since the target outputs of the ANN are not measured, the network outputs are fed to the analytical part of the hybrid model and the hybrid model's output are compared with available data. The network parameters are updated depending on the observed error.

The objective of our approach is to use of a combining rule that is data-generated and does not need manual adaption. The combining rule decides which submodel or submodels is/are responsible for generating the output depending on the particular input vector. Furthermore, it must be able to train data-driven submodels for parts of the input space not covered by knowledge-based models.

5.2 Modeling of Energy Flow in a Hybrid Electric Vehicle

The application addresses the simulation of electrical energy flow in the electrical system of a hybrid electric vehicle. Four distinct driving modes can be defined and represented by the available expert knowledge: a pure electric drive mode, a hybrid drive mode, a brake mode, and a drag mode. Depending on the current drive mode, electric energy is either used to drive the electric motor or produced by the generator. In pure electric drive mode and hybrid drive mode, energy is provided by the battery to drive the electric motor. In brake mode and drag mode, the electric motor is operating as a generator to regenerate the kinetic energy used for charging the battery. Domain experts designed specific models for each mode. These models represent complementary information sources because they are defined for different regions of the input space and each model provides with information for different mutually exclusive driving modes. Furthermore, the battery must maintain certain chemical limits. These limits determine the maximum charge and discharge capabilities of the battery depending on its state of charge and temperature.

The training data set in this example consists of about 10.00 input patterns, where each input pattern is 5-dimensional. The validation data set comprises approximately 100.00 input patterns. The target is one-dimensional and represents the electrical energy in kW.

Both ensemble methods: HME with a multi-layer perceptron (MLP) gate and HME with a cluster gate, are compared with a standard ME, an ensemble of MLPs, a single MLP, and a radial basis function (RBF) network. Two HME models use four local models each. Two characteristic maps and a mathematical model represent the pure electric drive mode, brake, and drag mode. However, since the hybrid drive mode is too complex to provide with a simple mathematical model, for this mode a two-layer MLP with 5 input units, 6 hidden units and one output unit were trained. Each mode uses different input features for the modeling. As gate, an MLP with 4 hidden units was applied.

The ME consists of 4 MLPs with 8 hidden units and as gate an MLP with 6 hidden units were used. The single MLP comprises 12 hidden units. The RBF network comprises 14 Gaussian basis functions. In the ensemble, 10 members are combined. All members have the same architecture, i.e., MLPs with a single hidden layer of 6 units. The output of the ensemble is computed as follows:

$$y_{ens} = \frac{1}{K} \sum_{j=1}^{K} y_j \left(\mathbf{x}^{(n)} \right) , \tag{11}$$

where $y_j \left(\mathbf{x}^{(n)} \right)$ is the output of the j member and K is the number of ensemble members.

As predictive measure, the means absolute error is used:

$$e = \frac{1}{N} \sum_{n=1}^{N} \left| t^{(n)} - f\left(\mathbf{x}^{(n)}\right) \right| . \tag{12}$$

Table 1 summarizes 10-fold-crossvalidation runs that are performed to estimate the predictive error of the regression models on previously unseen data. The HMEs with both MLP gate and clustering gate have achieved superior performance due to the incorporation of available information sources. Figs. 5 and 6 show the outputs of the gate (the activation of the submodels) of the two HME models. In most of the cases, the MLP gate selects only one submodel for each input vector. In case of clustering gate, the activations of different submodels are distributed slightly more than the MLP gate. This behavior is consistent with the knowledge of the domain expert that the submodels are defined for different slightly overlapping modes. Against the background of domain knowledge, the ME model is not able to identify the driving modes and has divided the input space in a technically non-plausible way. This is illustrated in Fig. 7. The overall output is composed of the outputs of the submodels.

Table 1 Error of the models on the hybrid vehicle data set

model	predictive error	
	training	testing
HME with MLP gate	2.10	2.14
HME with cluster gate	2.25	2.31
ME	2.78	2.88
ensemble	2.39	2.47
RBF	3.85	3.96
MLP	2.51	2.61

Table 2 shows the distribution of the responsibilities of the mode models for data of the corresponding driving mode. The values indicate that the mode models are correctly assigned to the partitions of the driving modes. These responsibilities are depicted as shaded background in Figs. 5 and 6 and confirm the results in Table 2.

Further, the incorporation of available knowledge requires fewer training data. The smaller the size of the training data set, the less robust are the results of data-driven models. In Table 3 and Fig. 8, the results for different sizes of the training data sets are shown. The results indicate that the proposed models require fewer training data compared to other regression methods to yield good predictive performance. For training data set sizes of $D/2$, $D/4$, and $D/8$ (where D indicates the original training data set), the predictive

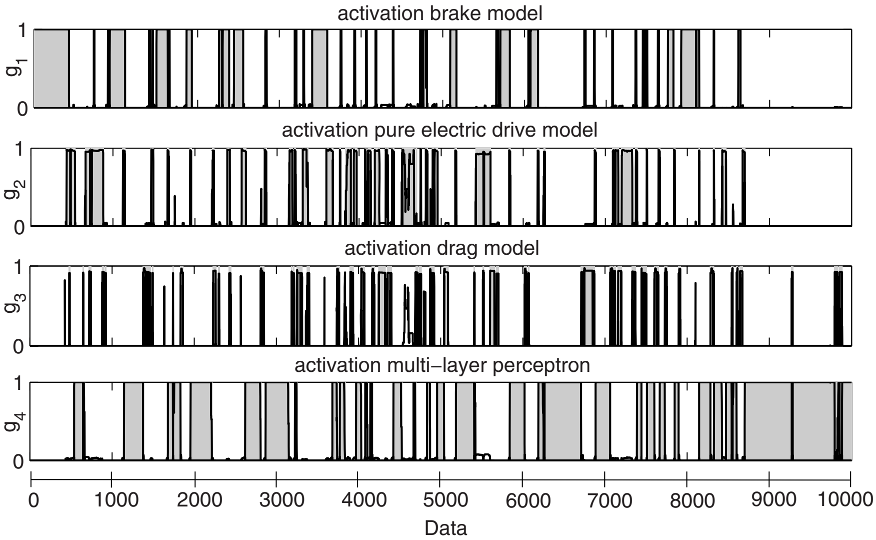

Fig. 5 The figure shows the activations of four submodels by the MLP gate of the HME model. The shaded background indicates data that correspond to the driving mode, which is represented by the mode model. The HME model has correctly identified the different driving modes and the mode models are responsible for data of the corresponding driving mode

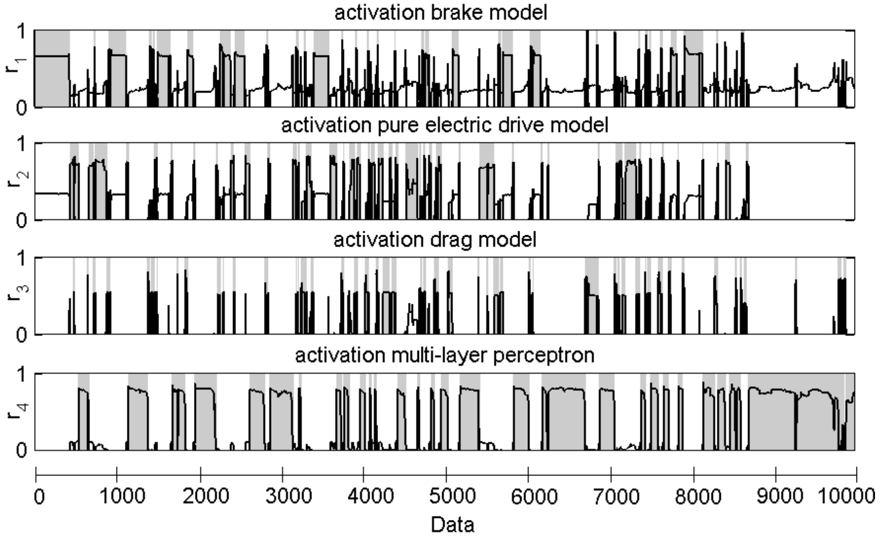

Fig. 6 The figure shows the activations of four submodels by the cluster gate of the HME model. The shaded background indicates data that correspond to the driving mode, which is represented by the mode model. The HME has correctly identified the different driving modes. The activations of the different submodels are distributed slightly more than the MLP gate

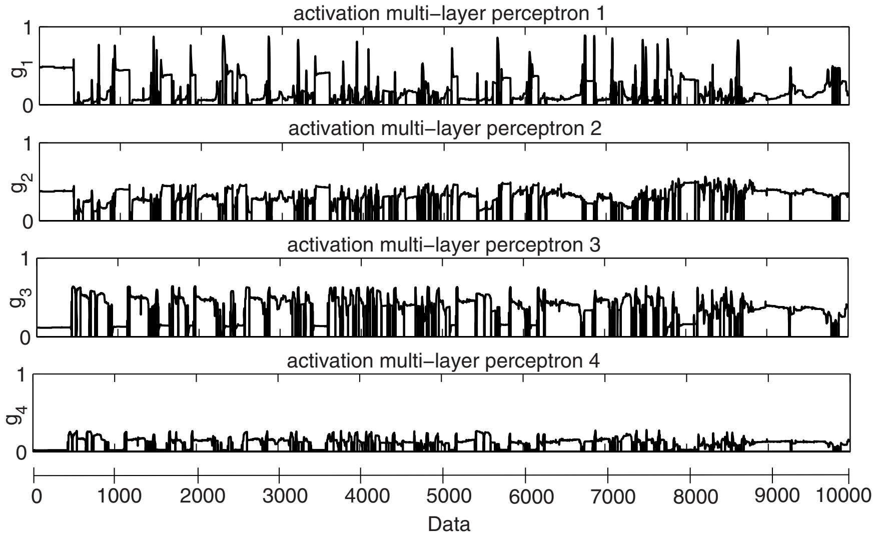

Fig. 7 The figure shows the activations of four data-driven submodels by the gate of the ME model. The ME model is not able to identify the driving modes and has divided the input space in a technically non-plausible way

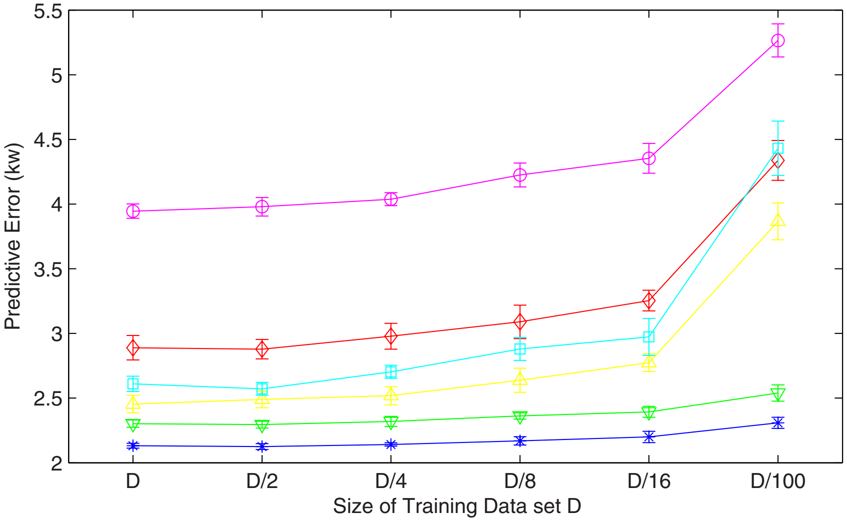

Fig. 8 The plot shows the predictive error of the models for different sizes of the training data set. The HME models with both MLP gate (*asterisk*) and clustering gate (*downward-pointing triangle*) have a slightly increasing error for small sizes of training data set. For small sizes of the training data set, the error increases of the ME model (*diamond*), the ensemble (*upward-pointing triangle*), the RBF (*circle*), and the MLP (*square*)

Table 2 Responsibilities of the mode models for data of the corresponding driving mode

HME model	driving mode (in %)			
	brake	pure electric drive	drag	hybrid
MLP gate	98	94	93	98
cluster gate	89	92	85	87

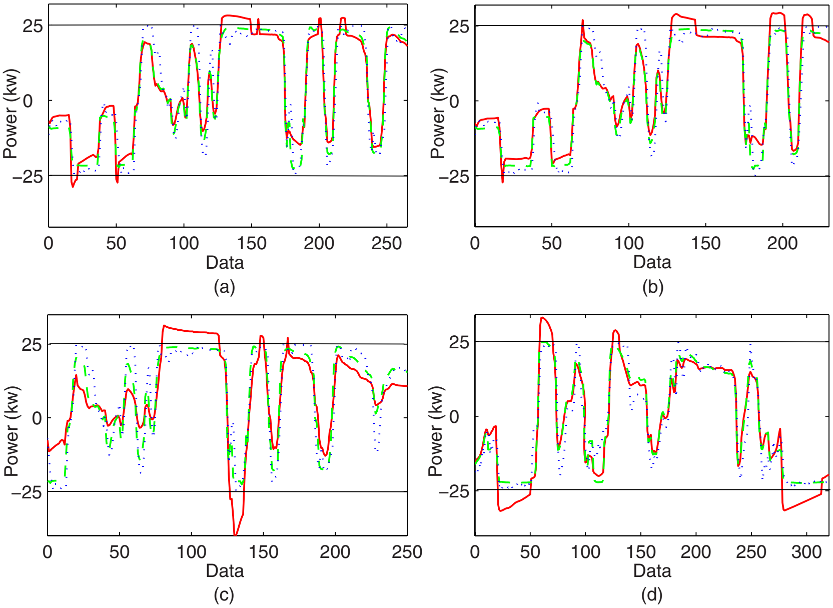

Fig. 9 The plots show examples of violations of the chemical battery limits (depicted as *horizontal lines*) of (**a**) the ME (*solid line*), (**b**) the ensemble (*solid line*), (**c**) the RBF (*solid line*), and (**d**) the MLP (*solid line*). The corresponding target values and outputs of the HME with MLP gate are depicted as *dotted* and *dashed lines*

performance of the models is approximately equal. However, for smaller sizes of the training data set, the error increases for purely data-driven models.

The chemical battery limits are violated by all models, except the HME models, because they predict energy flows that cannot be provided by the battery as shown in Fig. 9. The necessary information about these limits is not contained in the training data, but they are implicitly contained in the given knowledge-based models. Thus, purely data-driven models are not capable to maintain these limits.

Table 3 Predictive error of the models for different sizes of the training data set on the hybrid vehicle data

model	size of training data set D					
	D	$D/2$	$D/4$	$D/8$	$D/16$	$D/100$
HME with MLP	2.14	2.13	2.14	2.16	2.20	2.30
HME with cluster gate	2.31	2.31	2.33	2.35	2.39	2.51
ME	2.88	2.91	2.96	3.08	3.22	4.33
ensemble	2.47	2.49	2.54	2.64	2.82	3.81
RBF	3.96	4.00	4.08	4.21	4.37	5.26
MLP	2.61	2.62	2.69	2.82	2.99	4.35

6 Conclusions

By applying the proposed ensemble learning model, it is possible to fuse information from multiple sources, represented by knowledge-based models. In this way, information can be incorporated in the modeling process that is not contained in the training data. For example constraints can be implicitly contained in knowledge-based models but domain experts may not be able to describe them, since the domain experts do not explicitly perceive it or they cannot define such constraints. Data-driven submodels are used to complement knowledge-based ones with respect to the coverage of the input space. To be able to integrate given knowledge-based models into the process of simultaneously training the data-driven submodels and a gate model, it is crucial to incorporate the validity ranges of the knowledge-based models. A further advantage is the need of fewer training data, which is beneficial if a few training data are available or if the acquisition of data is expensive.

We have tested the HME models successfully for the simulation of electrical energy flow in the electrical system of a hybrid electric vehicle. They have achieved a superior performance compared to previous approaches.

References

1. Abonyi, J., Madar, J., Szeifert, F.: Combining first principles models and neural Networks for generic model control. Springer Engineering Series (2002)
2. Bengio, S., Marcel, C., Marcel, S., Mariéthoz, J.: Confidence measures for multimodal identity verification. Information Fusion 3, 267–276 (2002)
3. Beyer, J., Heesche, K., Hauptmann, W., Otte, C.: Combined knowledge-based and data-driven modeling by heterogeneous mixture-of-experts. In: Mikut, R., Reischl, M. (Hrsg.) Workshop Computational Intelligence, vol. 18, pp. 225–236. Universitätsverlag Karlsruhe (2008)
4. Bishop, C.M., Svensén, M.: Bayesian hierarchical mixtures of experts. In: Proceedings of the 19th Conference on Uncertainty in Artificial Intelligence, pp. 57–64 (2003)

5. Bloch, I.: Information Fusion in Signal and Image Processing: Major Probabilistic and Non-Probabilistic Numerical Approaches. John Wiley & Sons Inc., Chichester (2008)
6. Breiman, L.: Bagging predictors. Machine Learning 24, 123–140 (1996)
7. Dasarathy, B.V.: Information fusion - what, where, why, when, and how? Information Fusion 2, 75–76 (2001)
8. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society 39, 1–38 (1977)
9. Durrant-Whyte, H.F.: Sensor models and multisensor integration. International Journal of Robotics Research 7, 97–113 (1988)
10. Ferrari-Trecate, G., Muselli, M.: A new learning method for piecewise linear regression. In: Proceedings of International Conference on Artificial Neural Networks, pp. 444–449 (2002)
11. Freund, Y., Schapire, R.E.: Decision–theoretic generalization of on–line learning and an application to boosting. Journal of Computer and System Sciences 55, 119–139 (1997)
12. Georgieva, P., de Azevedo, S.F.: Neural network-based control strategies applied to a fed-batch crystallization process. International Journal of Computational Intelligence 3, 224–233 (2006)
13. Hansen, L., Salamon, P.: Neural network ensembles. IEEE Transactions on Pattern Analysis and Machine Intelligence 12, 993–1001 (1990)
14. Hashem, S.: Optimal linear combinations of neural networks. Neural Networks 10, 599–614 (1997)
15. Jacobs, R.A., Jordan, M.I., Nowlan, S.J., Hinton, G.E.: Adaptive mixtures of local experts. Neural Computation 3, 79–87 (1991)
16. Jordan, M.I., Jacobs, R.A.: Hierarchical mixtures of experts and the EM algorithm. Neural Computation 6, 181–214 (1994)
17. Krogh, A., Vedelsby, J.: Neural networks ensembles, cross validation, and active learning. In: Tesauro, G., Touretzky, D., Leen, T. (eds.) Advances in Neural Information Processing Systems, vol. 7, pp. 231–238. MIT Press, Cambridge (1995)
18. Quandt, R.E.: The estimation of the parameters of a linear regression system obeying two separate regimes. Journal of the American Statistical Association 53, 873–880 (1958)
19. Schapire, R.E.: The strength of weak learnability. IEEE Transactions on Pattern Analysis and Machine Intelligence 5, 197–227 (1990)
20. Schlang, M., Feldkeller, B., Lang, B., Poppe, T., Runkler, T.: Neural computation in steel industry. In: Proceedings European Control Conf. ECC 1999, VDI-Verlag (1999)
21. Titterington, D.M., Smith, A.F.M., Makov, U.E.: Statistical Analysis of Finite Mixture Distributions. John Wiley, New York (1985)
22. van Lith, P.F., Betlem, B.H.L., Roffel, B.: Combining prior knowledge with data driven modeling of a batch distillation column including start-up. Computers and Chemical Engineering 27, 1021–1030 (2003)
23. Wald, L.: Some terms of reference in data fusion. IEEE Transactions on Geoscience and Remote Sensing 37, 1190–1193 (1999)
24. Waterhouse, S., MacKay, D., Robinson, T.: Bayesian methods for mixture of experts. In: Advances of Neural Information Processing Systems, pp. 351–357 (1996)

25. White, F.E.: Data fusion lexicon. Joint Directors of Laboratories, Technical Panel of C3, Data Fusion Sub-Panel, Naval Ocean Systems Center, San Diego (1987)
26. Wolpert, D.H.: Stacked generalization. Neural Networks 5, 241–259 (1992)
27. Xu, L.: RBF nets, mixture experts, and bayesian ying–yang learning. Neurocomputing 19, 223–257 (1998)