Compact Modeling of Data Using Independent Variable Group Analysis

Esa Alhoniemi, Antti Honkela, Krista Lagus, Jeremias Seppä, Paul Wagner, and Harri Valpola

***Abstract*—We introduce a modeling approach called independent variable group analysis (IVGA) which can be used for finding an efficient structural representation for a given data set. The basic idea is to determine such a grouping for the variables of the data set that mutually dependent variables are grouped together whereas mutually independent or weakly dependent variables end up in separate groups.**

Computation of an IVGA model requires a combinatorial algorithm for grouping of the variables and a modeling algorithm for the groups. In order to be able to compare different groupings, a cost function which reflects the quality of a grouping is also required. Such a cost function can be derived, for example, using the variational Bayesian approach, which is employed in our study. This approach is also shown to be approximately equivalent to minimizing the mutual information between the groups.

The modeling task is computationally demanding. We describe an efficient heuristic grouping algorithm for the variables and derive a computationally light nonlinear mixture model for modeling of the dependencies within the groups. Finally, we carry out a set of experiments which indicate that IVGA may turn out to be beneficial in many different applications.

***Index Terms*—compact modeling, independent variable group analysis, mutual information, variable grouping, variational Bayesian learning**

I. Introduction

The study of effective ways of finding compact representations for data is important for the automatic analysis and exploration of complex data sets and natural phenomena. Finding properties of the data that are not related can help in discovering compact representations as it saves from having to model the mutual interactions of the unrelated properties.

It seems evident that humans group related properties as a means for understanding complex phenomena. An expert of a complicated industrial process such as a paper machine may describe the relations between different control parameters and measured variables by groups: A affects B and C, and so on. This grouping is of course not strictly valid as all the variables eventually depend on each other, but it helps in describing the most important relations, and thus makes it possible for the human to understand the system. Such groupings also significantly help the interaction with the process. Automatic discovery of such groupings would help in designing visualizations and control interfaces that reduce the cognitive load of the user by allowing her to concentrate on the essential details.

Analyzing and modeling intricate and possibly nonlinear dependencies between a very large number of variables (features) is a hard problem. Learning such models from data generally requires very much computational power and memory. If one does not limit the problem by assuming only linear or other restricted dependencies between the variables, essentially the only possibility is to try to model the data set using different model structures. One then needs a principled way to score the structures, such as a cost function that accounts for the model complexity as well as the accuracy of the model.

As far as we know, there does not exist a computationally feasible algorithm for grouping of variables that is based on *dependencies* between the variables. The main contribution of this article is derivation and detailed description of all the elements that are required for construction of such a model. We also experimentally show that the model can indeed be used to obtain useful results in various applications.

The remainder of the article is organized as follows. In Section II we describe a computational modeling approach called Independent Variable Group Analysis (IVGA) by which one can learn a structuring of a problem from data. In short, IVGA does this by finding a partition of the set of input variables that minimizes the mutual information between the groups, or equivalently the cost of the overall model including the cost of the model structure and the representation accuracy of the model. Its connections to related methods are discussed in Section II-B.

The problem of modeling-based estimation of mutual information is discussed in Section III. The approximation turns out to be equivalent to variational Bayesian learning. Section III also describes one possible computational model for representing a group of variables as well as the cost function for that model. The algorithm that we use for finding a good grouping is outlined in Section IV along with a number of speedup techniques.

In Section V we examine how well the IVGA model works both on an artificial toy problem and two real data sets: printed circuit board assembly component database and ionosphere radar measurements.

Initially, IVGA was introduced in [1], and some further experiments were presented in [2]. In the current article we derive the connection between mutual information and variational Bayesian learning and describe the current, improved computational method in detail. The mixture model for mixed

E. Alhoniemi is with the Department of Information Technology, University of Turku, FI-20014 University of Turku, Finland. (e-mail: esa.alhoniemi@utu.fi)

A. Honkela, K. Lagus, J. Seppä, and P. Wagner are with the Adaptive Informatics Research Centre, Helsinki University of Technology, P.O. Box 5400, FI-02015 TKK, Finland. (e-mail: antti.honkela@tkk.fi, krista.lagus@tkk.fi)

H. Valpola is with the Laboratory of Computational Engineering, Helsinki University of Technology, P.O. Box 9203, FI-02015 TKK, Finland. (e-mail: harri.valpola@tkk.fi)

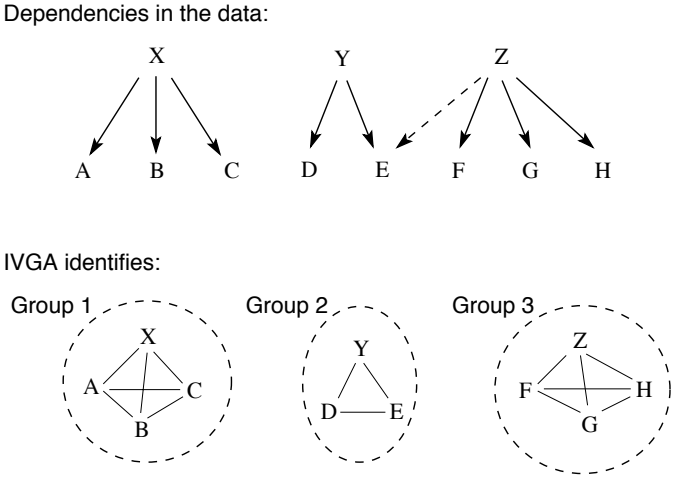

Fig. 1. An illustration of the IVGA modeling approach. The upper part of the figure shows the actual dependencies between the observed variables. The arrows that connect variables indicate causal dependencies. The lower part depicts the variable groups that IVGA might find here. One actual dependency is left unmodeled, namely the one between Z and E. Note that IVGA does not reveal causalities, but dependencies between the variables only.

real and nominal data is presented along with derivation of the cost function. Details of the grouping algorithm and necessary speedups are also presented. Completely new experiments include an application of IVGA to supervised learning.

II. Modeling using Independent Variable Group Analysis

The ultimate goal of independent variable group analysis (IVGA) is to partition a set of variables (also known as attributes or features) into separate groups so that the statistical dependencies of the variables within each group are strong. These dependencies are modeled, whereas the weaker dependencies between variables in different groups are disregarded. The modeling approach is depicted in Fig. 1.

In order to determine a grouping for observed data, a combinatorial grouping algorithm for the variables is required. Usually this algorithm is heuristic since an exhaustive search over all possible variable groupings is computationally infeasible.

The combinatorial optimization algorithm needs to be complemented by a method to score different groupings or a cost function for the groups. Suitable cost functions can be derived in a number of ways, such as using the mutual information between different groups or as the cost of an associated model under a suitable framework such as minimum description length (MDL) or variational Bayes. All of these alternatives are actually approximately equivalent, as presented in Sec. III.

It is vital that the models of the groups are fast to compute and that the grouping algorithm is efficient, too. In Sec. IV-A, such a heuristic grouping algorithm is presented. Each variable group is modeled by using a computationally relatively light mixture model which is able to model nonlinear dependencies between both nominal and real valued variables at the same time. Variational Bayesian modeling is considered in Sec. III, which also contains derivation of the mixture model.

It should be noted that the models used in the model-based approaches need not be of any particular type. As a matter of fact, all the models of a particular modeling problem do not necessarily need to be of same type, but each variable group could even be modeled using a different type of model. Therefore, IVGA could potentially be seen as a *concept*, not just as an algorithm. However, we have neither derived nor tried any other models than the mixture model reported in this article. Without experimental evaluation it is not possible to comment on the general applicability of the approach using arbitrary models.

A. Motivation for Using IVGA

The computational usefulness of IVGA relies on the fact that if two variables are dependent on each other, representing them together is efficient, since redundant information needs to be stored only once. Conversely, a joint representation of variables that do not depend on each other is inefficient. Mathematically speaking, this means that the representation of a joint probability distribution that can be factorized is more compact than the representation of a full joint distribution. In terms of a problem expressed using association rules of the form ($A = 0.3$, $B = 0.9 \Rightarrow F = 0.5$, $G = 0.1$): The shorter the rules that represent the regularities within a phenomenon, the more compact the representation is and the fewer association rules are needed. IVGA can also be given a biologically inspired motivation. With regard to the structure of the cortex, the difference between a large monolithic model and a set of models produced by IVGA roughly corresponds to the contrast between full connectivity (all cortical areas receive inputs from all other areas) and more limited, structured connectivity.

The IVGA modeling approach has shown to be sound. A very simple initial method described in [1] found appropriate variable groups from data where the features were various real-valued properties of natural images. Recently, we have extended the model to handle also nominal (categorical) variables, improved the variable grouping algorithm, and carried out experiments on various different data sets.

IVGA can be viewed in many different ways. First, it can be seen as a method for finding a compact representation for data using multiple independent models. Secondly, IVGA can be seen as a method of clustering variables. Note, however, that this is not equivalent to taking the transpose of the data matrix and performing ordinary clustering, since dependent variables need not be close to each other in the Euclidean or any other common metric. Thirdly, IVGA can also be considered as a variable or feature selection method.

B. Related Work

One of the basic goals of the unsupervised learning is to obtain compact representations for observed data. The methods reviewed in this section are related to IVGA in the sense that they aim at finding a compact representation for a data set using multiple independent models. Such methods include multidimensional independent component analysis (MICA,

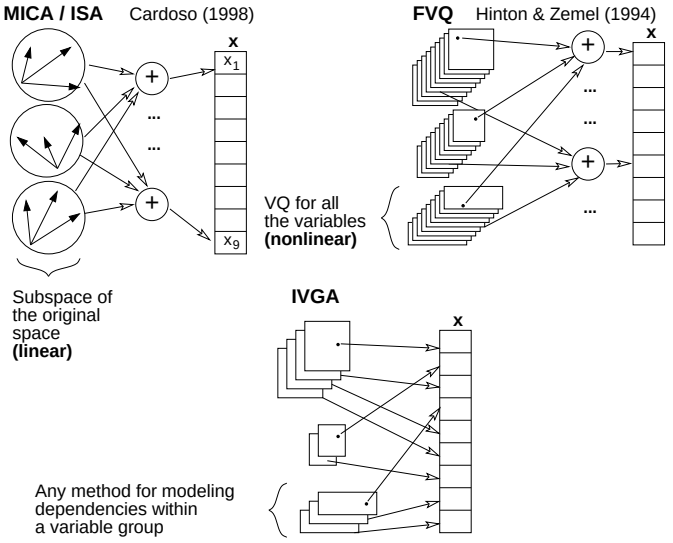

Fig. 2. Schematic illustrations of IVGA and related algorithms, namely MICA/ISA and FVQ that look for multi-dimensional feature subspaces in effect by maximizing a statistical independence criterion. The input $\mathbf{x}$ is here 9-dimensional. The number of squares in FVQ and IVGA denote the number of variables modeled in each sub-model, and the number of black arrows in MICA is equal to the dimensionality of the subspaces. Note that with IVGA the arrows depict all the required connections, whereas with FVQ and MICA only a subset of the actual connections have been drawn (6 out of 27).

also known as independent subspace analysis, ISA) [3] and factorial vector quantization (FVQ) [4], [5].

In MICA, the goal is to find independent linear feature subspaces that can efficiently be used to reconstruct the data. Thus, each subspace is able to model the linear dependencies in terms of the latent directions defining the subspace. FVQ can be seen as a nonlinear version of MICA, where the component models are vector quantizers over all the variables. The main difference between these and IVGA is that in IVGA, only one model affects a given observed variable. In all the others, all the models contribute to modeling every observed variable. This difference, visualized in Fig. 2, makes the computation of IVGA significantly more efficient.

There are also a few other methods for grouping the variables based on different criteria. A graph-theoretic partitioning of a graph induced by a thresholded association matrix between variables was used in [6]. The method requires choosing an arbitrary threshold for the associations, but the groupings could nevertheless be used to produce smaller decision trees with equal or better predictive performance than using the full dataset.

A framework for grouping variables of a multivariate time series based on possibly lagged correlations was presented in [7]. The correlations are evaluated using Spearman's rank correlation that can find both linear and monotonic nonlinear dependencies. The grouping method is based on a genetic algorithm, although other possibilities are presented as well. The method seems to be able to find reasonable groupings, but it is restricted to time series data and certain types of dependencies only.

Modular mixture model [8] is a hierarchical model with separate mixture models for different groups of variables and additional higher level mixtures to model residual dependencies between the groups. While the model itself is an interesting generalization of IVGA, the learning method described in [8] considers a fixed model structure only and cannot be used to infer a good grouping.

Module networks [9] are a very specific class of models that is based on grouping of similar variables together. They are used for discrete data only and all the variables of a group are restricted to have exactly the same distribution. The dependencies between different groups are modeled as a Bayesian network. Sharing the same model within a group makes the model easier to learn from scarce data, but severely restricts its possible uses.

For certain applications, it may be beneficial to view IVGA as a method for clustering variables. In this respect it is related to methods such as double clustering, co-clustering, and biclustering which also form a clustering not only for the samples, but for the variables, too [10], [11]. The differences between these clustering methods are illustrated in Fig. 3.

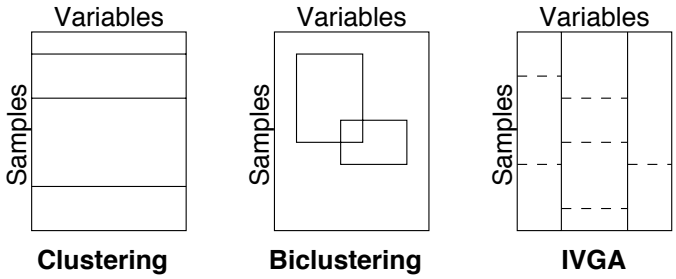

Fig. 3. Schematic illustrations of IVGA together with regular clustering and biclustering. In biclustering, homogeneous regions of the data matrix are sought for. The regions usually consist of a part of the variables and a part of the samples only. In IVGA, the variables are clustered based on their mutual dependencies. If the individual groups are modeled using mixture models, a secondary clustering of each group is also obtained, as marked by the dashed lines in the rightmost subfigure.

IVGA can also be used for feature, or input variable, selection for supervised learning as demonstrated in Sec. V-C. In that case one needs to consider which input variable(s) are dependent with – that is, grouped in the same group with – the output variable(s) of interest. For extensive presentations on variable selection with many references, see [12], [13].

III. A Modeling-Based Approach to Estimating Mutual Information

Learning a good grouping of variables can be seen either as a model comparison problem or a problem of estimating the mutual information for the groupings. Estimating mutual information of high dimensional data is very difficult as it requires an estimate of the probability density. If a model-based density estimate is used, the problem of minimizing the mutual information becomes approximately equivalent to a problem of maximizing the marginal likelihood $p(\mathcal{D}|\mathcal{H})$ of the model. Thus minimization of mutual information is equivalent to finding the best model for the data. This model comparison task can be performed efficiently using variational Bayesian techniques.

A. Approximating the Mutual Information

Let us assume that the data set $\mathcal{D}$ consists of vectors $\mathbf{x}(t),\ t = 1, \ldots, T$. The vectors are N-dimensional with the individual components denoted by $x_j,\ j = 1, \ldots, N$. Our aim is to find a partition of $\{1, \ldots, N\}$ to M disjoint sets $\mathcal{G} = \{\mathcal{G}_i | i = 1, \ldots, M\}$ such that the mutual information

$$I_{\mathcal{G}}(\mathbf{x}) = \sum_i H(\{x_j | j \in \mathcal{G}_i\}) - H(\mathbf{x}) \tag{1}$$

between the sets is minimized. When $M > 2$, this is actually a generalization of mutual information commonly known as multi-information [14]. As the entropy $H(\mathbf{x})$ of the whole data is constant, this can be achieved by minimizing the first sum. The component entropies in that sum can be approximated by using the distribution $p(\mathbf{y})$ of the data in the given group $\mathbf{y} = (x_j)_{j \in \mathcal{G}_i}$ as

$$\begin{aligned} H(\mathbf{y}) &= -\int p(\mathbf{y}) \log p(\mathbf{y})\, d\mathbf{y} \approx -\frac{1}{T} \sum_{t=1}^{T} \log p(\mathbf{y}(t)) \\ &\approx -\frac{1}{T} \sum_{t=1}^{T} \log p(\mathbf{y}(t) | \mathbf{y}(1), \ldots, \mathbf{y}(t-1), \mathcal{H}) \\ &= -\frac{1}{T} \log p(\{\mathcal{D}_j | j \in \mathcal{G}_i\} | \mathcal{H}_i), \end{aligned} \tag{2}$$

where $\mathcal{D}_j$ denotes the observations for x_j and $\mathcal{H}_i$ is the model for group $\mathcal{G}_i$. Two approximations were made in this derivation. First, the expectation over the data distribution was replaced by a discrete sum using the data set as a sample of points from the distribution. Next, the data distribution was replaced by the posterior predictive distribution of the data sample given the past observations. The sequential approximation is necessary to avoid the bias caused by using the same data twice, both for sampling and for fitting the model for the same point. A somewhat similar approximation based on using the probability density estimate implied by a model has been applied for evaluating mutual information also in [15]. The relation between mutual information and Bayesian measures of independence was noted in [16] in a discrete setting and the corresponding relation between entropy and marginal likelihood in the discrete case was presented in [17].

Using the result of Eq. (2), minimizing the criterion of Eq. (1) is equivalent to maximizing

$$\mathcal{L} = \sum_i \log p(\{\mathcal{D}_j | j \in \mathcal{G}_i\} | \mathcal{H}_i). \tag{3}$$

This reduces the mutual information minimization to a standard Bayesian model selection problem.

When varying the number of groups M, the two problems are not exactly equivalent. The mutual information cost (1) is always minimized when all the variables are in a single group, or multiple statistically independent groups. In the case of the Bayesian formulation (3), the global minimum may actually be reached for a nontrivial grouping even if the variables are not exactly independent. This allows determining a suitable number of groups even in realistic situations when there are weak residual dependencies between the groups. The main insight provided by the relation between the methods is that for a fixed number of groups, the best model in the probabilistic sense is also the one with the smallest mutual information for the corresponding grouping.

B. Variational Bayesian Learning

Unfortunately evaluating the exact marginal likelihood is intractable for most practical models as it requires evaluating an integral over a potentially high dimensional space of all the model parameters $\boldsymbol{\theta}$. This can be avoided by using a variational method to derive a lower bound of the marginal log-likelihood using Jensen's inequality [18]

$$\begin{aligned} &\log p(\mathcal{D} | \mathcal{H}) = \log \int_{\boldsymbol{\theta}} p(\mathcal{D}, \boldsymbol{\theta} | \mathcal{H})\, d\boldsymbol{\theta} \\ &= \log \int_{\boldsymbol{\theta}} \frac{p(\mathcal{D}, \boldsymbol{\theta} | \mathcal{H})}{q(\boldsymbol{\theta})} q(\boldsymbol{\theta})\, d\boldsymbol{\theta} \geq \int_{\boldsymbol{\theta}} \log \frac{p(\mathcal{D}, \boldsymbol{\theta} | \mathcal{H})}{q(\boldsymbol{\theta})} q(\boldsymbol{\theta})\, d\boldsymbol{\theta}, \end{aligned} \tag{4}$$

where $q(\boldsymbol{\theta})$ is an arbitrary distribution over the parameters. If $q(\boldsymbol{\theta})$ is chosen to be of a suitable simple factorial form, the bound becomes tractable.

Closer inspection of the right hand side of Eq. (4) shows that it is of the form

$$\begin{aligned} \mathcal{B} &= \int_{\boldsymbol{\theta}} \log \frac{p(\mathcal{D}, \boldsymbol{\theta} | \mathcal{H})}{q(\boldsymbol{\theta})} q(\boldsymbol{\theta})\, d\boldsymbol{\theta} \\ &= \log p(\mathcal{D} | \mathcal{H}) - D_{\mathrm{KL}}(q(\boldsymbol{\theta}) || p(\boldsymbol{\theta} | \mathcal{H}, \mathcal{D})), \end{aligned} \tag{5}$$

where $D_{\mathrm{KL}}(q||p)$ is the Kullback–Leibler divergence between distributions q and p. The Kullback–Leibler divergence $D_{\mathrm{KL}}(q||p)$ is non-negative and zero only when $q = p$. Thus it is commonly used as a distance measure between probability distributions although it is not a proper metric [19]. For a more through introduction to variational methods, see for example [18].

In addition to the interpretation as a lower bound of the marginal log-likelihood, the quantity $-\mathcal{B}$ may also be interpreted as a code length required for describing the data using a suitable code [20]. The code lengths can then be used to compare different models, as suggested by the minimum description length (MDL) principle [21]. This provides an alternative justification for the variational method. Additionally, the alternative interpretation can provide more intuitive explanations on why some models provide higher marginal likelihoods than others [22]. For the remainder of the paper, the optimization criterion will be the cost function

$$\begin{aligned} \mathcal{C} &= -\mathcal{B} = \int_{\boldsymbol{\theta}} \log \frac{q(\boldsymbol{\theta})}{p(\mathcal{D}, \boldsymbol{\theta} | \mathcal{H})} q(\boldsymbol{\theta})\, d\boldsymbol{\theta} \\ &= D_{\mathrm{KL}}(q(\boldsymbol{\theta}) || p(\boldsymbol{\theta} | \mathcal{H}, \mathcal{D})) - \log p(\mathcal{D} | \mathcal{H}) \end{aligned} \tag{6}$$

that is to be minimized.

Using this cost as an approximation of the negative marginal log-likelihood yields an estimate of the mutual information in Eq. (1) as

$$I_{\mathcal{G}}(\mathbf{x}) \approx \frac{1}{T} \sum_i \mathcal{C}(\{x_j | j \in \mathcal{G}_i\}) - H(\mathbf{x}), \tag{7}$$

where $\mathcal{C}(\{x_j | j \in \mathcal{G}_i\})$ is the cost of the model for group $\mathcal{G}_i$. In order to make sure that all the estimates are non-negative, the entropy of the full data $H(\mathbf{x})$ may be approximated by

the scaled minimal value of the cost over different models, including the model with all the variables in a single group. The accuracy of this approximation is studied empirically using a toy example in Sec. V-A. The attained results are mostly qualitatively correct between different groupings, although the numerical values are not especially accurate.

C. Mixture Model for the Groups

In order to apply the variational Bayesian method described above to solve the IVGA problem, a class of models for the groups needs to be specified. This class of models may vary depending on the goal of modeling, but it naturally needs to be such that one can derive an appropriate cost function and update equations for the parameters of the model.

In this work mixture models have been used for modeling of the groups. Mixture models are a good choice because they are simple while being able to model also nonlinear dependencies. The resulting IVGA model is illustrated as a graphical model in Fig. 4.

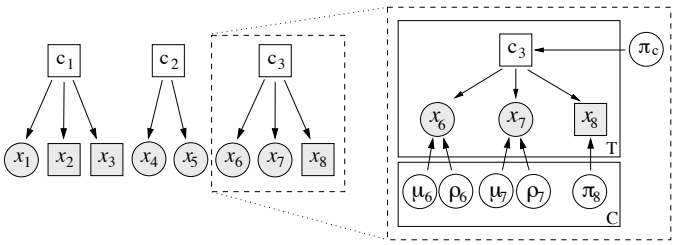

Fig. 4. The IVGA model as a graphical model. The nodes represent variables of the model with the shaded ones being observed. The left-hand side shows the overall structure of the model with independent groups. The right-hand side shows a more detailed representation of the mixture model of a single group of three variables. Variable c indicates the generating mixture component for each data point. The boxes in the detailed representation indicate that there are T data points and in the rightmost model there are C mixture components representing the data distribution. Rectangular and circular nodes denote discrete and continuous variables, respectively.

As shown in Fig. 4, different variables are assumed to be independent within a mixture component and the dependencies only arise from the mixture. For continuous variables, the mixture components are Gaussian and the assumed independence implies a diagonal covariance matrix. Different mixture components can still have different covariances [23]. The applied mixture model closely resembles other well-known models such as soft c-means clustering and soft vector quantization [24].

For nominal variables, the mixture components are multinomial distributions. All parameters of the model have standard conjugate priors. The exact definition of the model and the approximation used for the variational Bayesian approach are presented in Appendix A and the derivation of the cost function can be found in Appendix B.

IV. A Variable Grouping Algorithm for IVGA

The number of possible groupings of n variables is called the nth Bell number B_n. The values of B_n grow with n faster than exponentially, making an exhaustive search of all groupings infeasible. For example, $B_{100} \approx 4.8 \cdot 10^{115}$. Hence, some computationally feasible heuristic — which can naturally be any standard combinatorial optimization algorithm — for finding a good grouping has to be deployed.

To further complicate things, the objective function for the combinatorial optimization, the sum of marginal log-likelihoods of the component models, can only be evaluated approximately and there is another intertwined algorithm to optimize these.

In this section, we describe an adaptive heuristic grouping algorithm which is currently used in our IVGA model. The algorithm tries to simultaneously determine the best grouping for the variables and compute the models for the groups. After that, we also present three special techniques which are used to speed up the computation.

A. The Algorithm

The goal of the algorithm is to find such a variable grouping and such models for the groups that the total cost over all the models is minimized. Both of these tasks are carried out at the same time, which may seem somewhat confusing at the first glance.

The algorithm has an initialization phase and a main loop during which five different operations are consecutively applied to the current models of the variable groups and/or to the grouping until the end condition is met. A flow-chart illustration of the algorithm is shown in Fig. 5 and the phases of the algorithm are explained in more detail below.

Initialization. Each variable is assigned into a group of its own and an initial model for each group is computed.

Main loop. The following five operations are consecutively used to alter the current grouping and to improve the models of the groups. Each operation of the algorithm is assigned a probability which is adaptively tuned during the main loop: If an operation is efficient in minimizing the total cost of the model, its probability is increased and vice versa.

Model recomputation. The purpose of this operation is twofold: (1) It tries to find an appropriate complexity for the model for a group of variables—which is the number of mixture components in the mixture model; (2) It tests different model initializations in order to avoid local minima of the cost function of the model. As the operation is performed multiple times for a group, an appropriate complexity and good initialization is found for the model of the group.

A mixture model for a group is recomputed so that the number of mixture components may decrease, remain the same, or increase. It is slightly more probable that the number of components grows, that is, a more complex model is computed. Next, the model is initialized. For a Gaussian mixture model this means randomly selecting the centroids among the training data, and rough training of the model for some iterations. If a model for the group had been computed earlier, the new model is compared to the old model. Of these, the model with the smallest cost is selected as the current model for the group.

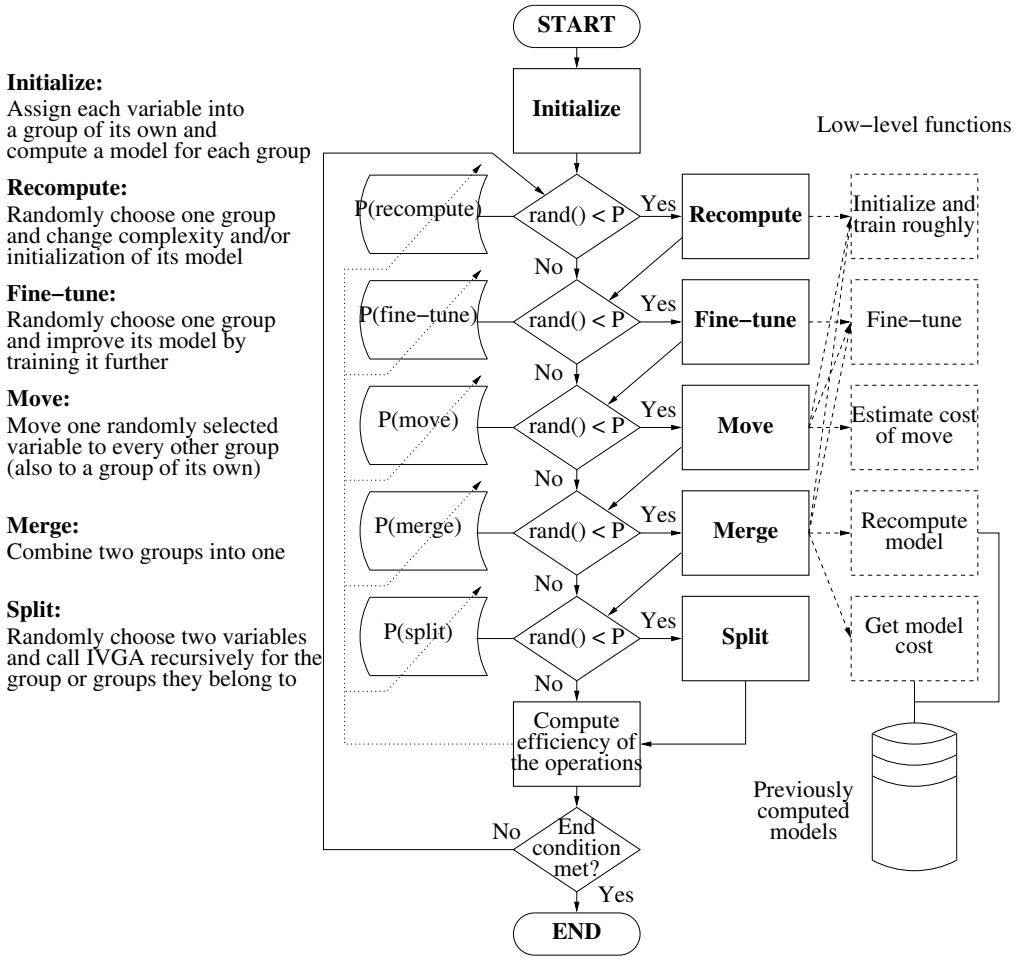

Fig. 5. An illustration of the variable grouping algorithm for IVGA. The solid line describes control flow, the dashed lines denote low-level subroutines and their calls so that the arrow points to the called routine. The dotted line indicates adaptation of the probabilities of the five operations. Function rand() produces a random number on the interval [0,1]. Adaptation of the probabilities shown in the left hand side of diagram is described in Sec. IV-B1. The low-level functions in the right hand side of the diagram are as follows: (1) Initialization and rough training as well as fine tuning and model recomputation operations all use the the iteration formulae described in Appendix B-C; (2) Estimate for the cost of a move – which is explained in Sec. IV-B3 – uses both the iteration algorithm (Appendix B-C) and computation of the cost (Appendix B-A); (3) the model cost of a previously computed model can be retrieved from a data structure which is kept in main memory during the IVGA run. If a previously computed model needs to be reconstructed (see Sec. IV-B2), it is carried out by retrieving the model parameters from the data structure and using the iteration formulae of Appendix B-C.

Model fine-tuning. When a good model for a group of variables has been found, it is sensible to fine-tune it further so that its cost approaches a local minimum of the cost function. During training, the model cost is never increased due to characteristics of the training algorithm.

However, tuning a model of a group takes many iterations of the learning algorithm and it is not sensible to do that for all the models that are used.

Moving a variable. This operation improves an existing grouping so that a single variable is moved from its original group to a more appropriate group. First, one variable is randomly selected among all the variables of all groups. The variable is removed from its original group and moving it to every other group (also to a group of its own) is tried. For each new group candidate, the cost of the model is roughly estimated. If the move reduces the total cost compared to the original one, the variable is moved to a group which yields the highest decrease in the total cost.

Merge. The goal of the merge operation is to combine two groups in which the variables are mutually dependent. In the operation, two groups are selected randomly among the current groups. A model for the variables of their union is computed. If the cost of the model of the joint group is smaller than the sum of the costs of the two original groups, the two groups are merged. Otherwise, the two original groups are retained.

Split. The split operation breaks down one or two existing groups. The group(s) are chosen so that two variables are randomly selected among all the variables. The group(s) corresponding to the variables are then taken for the operation. Hence, the probability of a group to be selected is proportional to the size of the group. As a result, more likely heterogeneous large groups are chosen more frequently than smaller ones. The operation recursively calls the algorithm for the

union of the selected groups. If the total cost of the resulting models is less than the sum of the costs of the original group(s), the original group(s) are replaced by the new grouping. Otherwise, the original group(s) are retained.

End condition. Iteration is stopped if the decrease of the total cost is very small in several successive iterations.

B. Speedup Techniques Used in Computation of the Models

Computation of an IVGA model for a large set of variables requires computation of a huge number of models (say, thousands), because in order to determine the cost of an arbitrary variable group, a unique model for it needs to be computed (or, at least, an approximation of the cost of the model). Therefore, fast and efficient computation of the models is crucial. We use the following three special techniques to speed up the computation of the models. Note that the effectiveness of the speedups depends on the problem at hand. For each technique, also this aspect has briefly been commented below.

1) Adaptive Tuning of Operation Probabilities: During the main loop of the algorithm described above, five operations are used to improve the grouping and the models. Each operation has a probability which dictates how often the corresponding operation is performed (see Fig. 5). As the grouping algorithm is run for many iterations, the probabilities are slowly adapted instead of keeping them fixed because

- it is difficult to determine probabilities which are appropriate for an arbitrary data set; and
- during a run of the algorithm, the efficiency of different operations varies—for example, the split operation is seldom beneficial in the beginning of the iteration (when the groups are small), but it becomes more useful when the sizes of the groups tend to grow.

The adaptation is carried out by measuring the efficiency (in terms of reduction of the total cost of all the models) of each operation. The probabilities of the operations are gradually adapted so that the probability of an operation is proportional to the efficiency of the operation. The adaptation is based on low-pass filtered efficiency, which is defined by

$$\text{efficiency} = -\frac{\Delta \mathcal{C}}{\Delta t}, \tag{8}$$

where $\Delta\mathcal{C}$ is the change in the total cost and Δt is the amount of CPU time used for the operation.

Based on multiple tests (not shown here) using various data sets, it has turned out that adaptation of the operation probabilities instead of keeping them fixed significantly often speeds up the convergence of the algorithm into a final grouping. However, there is a risk of emphasizing exploitation of the current grouping by fine tuning the mixture models at the expense of ignoring exploration of new groupings and this may lead to suboptimal results.

2) "Compression" of the Models: Once a model for a certain variable group is computed, it is sensible to store it, because a previously computed good model for the group may be needed later.

Computation of many models—for example, a mixture model—is stochastic, because often a model is initialized randomly and trained for a number of iterations. However, computation of such a model is actually *deterministic* provided that the state of the (deterministic) pseudorandom number generator just before initialization is known. Thus, in order to reconstruct a model after it has been computed once, we need to store (i) the random seed, (ii) the number of iterations that were used to train the model, and (iii) the model structure. Additionally, it is also sensible to store (iv) the cost of the model. So, a mixture model can be compressed into two floating point numbers (the random seed and the cost of the model) and two integers (the number of training iterations and the number of mixture components).

Note that this model compression principle is completely general: it can be applied in any algorithm in which compression of multiple models is required.

Compression of models is clearly a trade-off between memory usage and computation time. The technique enables learning in large data sets with reasonable memory requirements, and it can be easily ignored if memory consumption is not an issue.

3) Fast Estimation of Model Costs When Moving a Variable: When the move of a variable from one group to all the other groups is attempted, computationally expensive evaluation of the costs of multiple models is required. We use a specialized speedup technique for fast approximation of the costs of the groups: Before moving a variable to another group for real, a quick pessimistic estimate of the total cost change caused by the move is calculated, and only those new models that look appealing are tested further.

A quick estimate for the change of cost when a variable is moved from one group to another is computed as follows. The posterior probabilities of the mixture components are fixed and only the parameters of the components related to the moved variable are changed. The total cost of these two groups is then calculated for comparison with their previous cost. The approximation can be justified by the fact that if a variable is highly dependent on the variables in a group, then the same mixture model should fit it as well.

Fast estimation of variable moves is algorithmically the most controversial speedup, because the steps in the combinatorial optimization are selected based on incomplete information. To study the effects of the speedup, a set of experiments was performed using different sized subsets of a data set of features extracted from a large collection of images. The results of selected runs using methods with and without the speedup are illustrated in Fig. 6. The results of these and other runs show that both methods are equally likely to yield good results, but the fast estimates significantly speed up learning for large problems. However, for small problems, it may in some cases even be better not to use the fast estimates.

V. Applications, Experiments

The problems in which IVGA can be found to be useful can be divided into the following categories. First, IVGA can be used for *confirmatory* purposes in order to verify human intuition of an existing grouping of variables. The first synthetic problem presented in Sec. V-A can be seen as an

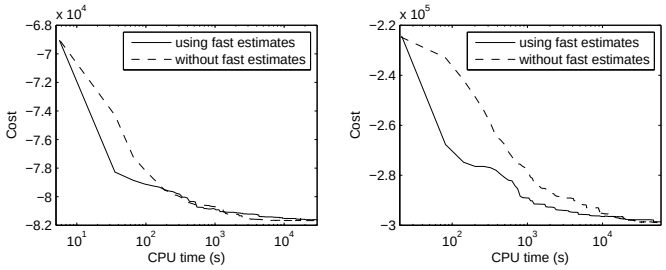

Fig. 6. Convergence of the IVGA model with and without fast cost estimation heuristic for a data set with 60 variables (left) and 120 variables (right).

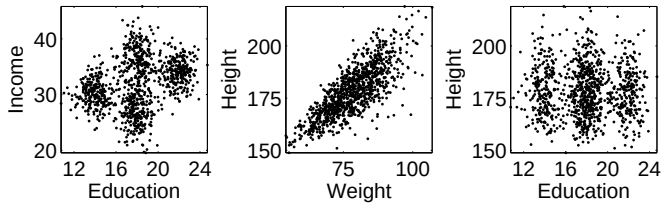

Fig. 7. Comparison of different two-dimensional subspaces of the data. Due to the dependencies between the variables shown in the two leftmost figures it is useful to model those variables together. In contrast, in the rightmost figure no such dependency is observed and therefore no benefit is obtained from modeling the variables together.

example of this type. Second, IVGA can be used to *explore* observed data, that is, to make hypotheses or learn the structure of the data. The discovered structure can then be used to divide a complex modeling problem into a set of simpler ones as illustrated in Sec. V-B. Third, we can use IVGA to reveal the variables that are dependent with the class variable in a classification problem. In other words, we can use IVGA also for *variable selection* in supervised learning problems. This is illustrated in Sec. V-C.

A. Toy Example

In order to illustrate the IVGA algorithm using a simple and easily understandable example, a data set consisting of one thousand points in a four-dimensional space was synthesized. The dimensions of the data are called "education", "income", "height", and "weight". All the variables are real-valued and the units are arbitrary. The data was generated from a distribution in which both education and income are statistically independent of height and weight.

Fig. 7 shows plots of education versus income, height vs. weight, and for comparison a plot of education vs. height. One may observe that in the subspaces of the first two plots of Fig. 7 the data points lie in few, more concentrated clusters and thus can generally be described (modeled) with a lower cost in comparison to the third plot. As expected, when the data was modeled, the resulting grouping was

$$\{\{\text{education}, \text{income}\}, \{\text{height}, \text{weight}\}\}\,.$$

Tab. I compares the costs of some possible groupings. It also shows the corresponding estimates of the mutual information of Eq. (7) together with the true mutual information of the generative model, both evaluated in nats. While the numerical estimates are not especially accurate, the ordering is mostly correct and the ratios of the values are mostly very close to the true ratios.

Grouping	Total Cost	Parameters	MI Estimate	True MI
{e,i,h,w}	12233.4	288	0.15	0.00
{e,i}{h,w}	**12081.0**	80	0.00	0.00
{e}{i}{h}{w}	12736.7	24	0.66	0.86
{e,h}{i}{w}	12739.9	24	0.66	0.86
{e,i}{h}{w}	12523.9	40	0.44	0.55
{e}{i}{h,w}	12304.0	56	0.22	0.31

TABLE I
A COMPARISON OF THE TOTAL COSTS OF SOME VARIABLE GROUPINGS OF THE SYNTHETIC DATA. THE VARIABLES EDUCATION, INCOME, HEIGHT, AND WEIGHT ARE DENOTED HERE BY THEIR INITIAL LETTERS. ALSO THE NUMBER OF PARAMETERS OF THE LEARNED OPTIMAL GAUSSIAN MIXTURE COMPONENT DISTRIBUTIONS ARE SHOWN. THE TWO LAST COLUMNS INDICATE THE VALUE OF THE ESTIMATED AND TRUE MUTUAL INFORMATION (MI), RESPECTIVELY. THE TOTAL COSTS ARE FOR MIXTURE MODELS OPTIMIZED CAREFULLY USING OUR IVGA ALGORITHM. THE MODEL SEARCH OF OUR IVGA ALGORITHM WAS ABLE TO DISCOVER THE BEST GROUPING, THAT IS, THE ONE WITH THE SMALLEST COST.

B. Printed Circuit Board Assembly

In the second experiment, we constructed predictive models to aid user input of component data of a printed circuit board assembly robot. When a robot is used in the assembly of a new product which contains components that have not been previously used, the data of the new components need to be manually determined and added to the existing component database of the robot by a human operator. The component data can be seen as a matrix. Each row of the matrix contains attribute values of one component and the columns of the matrix depict component attributes, which are not mutually independent. Building an input support system by modeling of the dependencies of the existing data using association rules has been considered in [25]. A major problem of the approach is that extraction of the rules is computationally heavy and memory consumption of the predictive model which contains the rules (in our case, a trie) is very high.

We divided the component data of an operational assembly robot (5 016 components, 22 nominal attributes) into a training set (80 % of the whole data) and a testing set (the rest 20 %). The IVGA algorithm was run 200 times for the training set. In the first 100 runs (avg. cost 113 003), all the attributes were always assigned into one group. During the last 100 runs (avg. cost 113 138) we disabled the adaptation of the probabilities (see Sec. IV-A) to see if this would have an effect on the resulting groupings. In these runs, we obtained 75 groupings with 1 group and 25 groupings with 2–4 groups. Because we were looking for a good grouping with more than one group, we chose a grouping with 2 groups (7 and 15 attributes). The cost of this grouping was 112 387 which was not the best among all the results over 200 runs (111 791), but not very far from it.

Next, the dependencies of (1) the whole data and (2) the 2 variable groups were modeled using association rules. The large sets required for computation of the rules were

computed using a freely available software implementation[1] of the Eclat algorithm [26]. Computation of the rules requires two parameters: minimum support ("generality" of the large sets that the rules are based on) and minimum confidence ("accuracy" of the rules). The minimum support dictates the number of large sets, which is in our case equal to the size of the model. For the whole data set, the minimum support was 5 %, which was the smallest computationally feasible value in terms of memory consumption. For the models of the two groups it was set to 0.1 %, which was the smallest possible value so that the combined size of the two models did not exceed the size of the model for the whole data. The minimum confidence was set to 90 %, which is a typical value for the parameter in many applications.

The rules were used for one-step prediction of the attribute values of the testing data. The data consisted of values selected and verified by human operators, but it is possible that these are not the only valid values. Nevertheless, predictions were ruled incorrect if they differed from these values. Computation times, memory consumption, and prediction accuracy for the whole data and the grouped data are shown in Tab. II. Grouping of the data both accelerated computation of the rules and improved the prediction accuracy. Also note that the combined size of the models of the two groups is only about 1/4 of the corresponding model for the whole data.

	Whole data	**Grouped data**
Computation time (s)	48	9.1
Size of trie (nodes)	9 863 698	2 707 168
Correct predictions (%)	57.5	63.8
Incorrect predictions (%)	3.7	2.9
Missing predictions (%)	38.8	33.3

TABLE II
SUMMARY OF THE RESULTS OF THE COMPONENT DATA EXPERIMENT. ALL THE QUANTITIES OF THE GROUPED DATA ARE SUMS OVER THE TWO GROUPS. ALSO NOTE THAT IN THIS PARTICULAR APPLICATION THE SIZE OF TRIE IS EQUAL TO THE NUMBER OF ASSOCIATION RULES.

The potential benefits of IVGA in an application of this type are as follows. (1) It is possible to compute rules which yield better prediction results, because the rules are based on small amounts of data, i.e, it is possible to use smaller minimum support for the grouped data. (2) Discretization of continuous variables—which is often a problem in applications of association rules—is automatically carried out by the mixture model. (3) Computation of the association rules may even be completely ignored by using the mixture models of the groups as a basis for the predictions. Of these, (1) was demonstrated in the experiment whereas (2) and (3) remain a topic for future research.

C. Feature Selection for Supervised Learning: Ionosphere Data

In this experiment, we investigated whether the variable grouping ability of IVGA could be used for feature selection in classification. An obvious way to apply IVGA in this manner is to find out which variables are grouped in the same group together with the class variable and to use only these in the actual classifier.

In the experiment, we used the Ionosphere data set [27] which contains 351 instances of radar measurements consisting of 34 attributes and a binary class variable. One attribute was constant in the data, so it did not have any effect on the classification result and was removed from the data.

In all tests, we used k-nearest neighbor (k-NN) classifier (see for example [28]). Unless stated otherwise, one k-NN run was always carried out in the following manner. The data set was randomly divided into three nonoverlapping parts: a training set with 250 samples, a validation set with 50 samples, and a test set with 51 samples. The training data was normalized to zero mean and unit variance and the same normalization was applied to the validation and test sets. The validation set was always used to select the optimal value for k. The feature vectors of the test set were classified by using the samples of the training set and the number of correct classifications was counted and stored. This procedure was repeated 100 times.

We tried four different approaches to feature selection: no selection at all, IVGA, Mann-Whitney (M-W) test (which is equivalent to Wilcoxon rank sum test) [29], and sequential floating forward selection (SFFS) method [30]. IVGA and M-W test are so-called filtering selection methods, since they both are a kind of preprocessing step before – and totally independent of – the classifier whereas SFFS was used as a so-called wrapper method which used the classifier itself for the selection of the features. For more information on the filtering and wrapper approaches, see for example [12]. Use of the three selection methods is described in detail below; note that of these, we used IVGA and M-W test in an identical way after ranking of the variables using the methods.

a) IVGA: For each partition of data to training, validation, and test sets, we repeated the following: (1) We ran the IVGA algorithm 100 times for the training data set; (2) The variables were ranked in descending order so that the variables which were most frequently in the same group with the class variable became first; (3) We used $d = 1, \ldots, 33$ first variables of the ranked variables. The optimal number of variables and the optimal value for k were selected jointly using the validation data set and the classification error was computed using the test set.

b) M-W test: The M-W test is a statistical test which can be used to accept or reject the hypothesis that the medians of two samples are equal. For each partition of the data to training, validation, and test sets, we repeated the following: (1) We compared the distributions of the two classes for each variable in the test set and obtained the corresponding p-value for the variable; (2) The variables were sorted in ascending order according to the p-values; (3) We determined the optimal set of variables, the optimal value for k as well as the test error in a way that was identical to variable selection using IVGA.

c) SFFS: SFFS is principally a different approach from the two previous ones, because it requires a multivariate criterion for assessing the performance of a feature set; we measured the accuracy using a k-NN classifier. For each

[1]See http://www.adrem.ua.ac.be/~goethals/software/index.html

partition of the data set to training, validation, and test sets, the algorithm was run in the following way: (1) The training data was given to the SFFS algorithm. Because SFFS used the k-NN classifier, it always internally divided the 250 training vectors to a training data set of 170 samples and two validation data sets of 40 samples each. At each step of the SFFS algorithm, this division was performed 10 times, the first validation set was used to optimize k, and the second validation set for evaluation of the classification error. (2) Using the variables determined by the SFFS algorithm, the optimal value for k was determined using the validation set of 50 samples and the classification error was computed using the test set.

The results of our experiments are shown in Tab. III. Note that the test was run 100 times so that on each run randomly selected training, validation, and data sets were used in order to benchmark the feature selection methods. Therefore, it is not possible to indicate the variables selected by the methods, because the set of variables varied between different runs.

The performance of the three feature selection methods were quite similar using the k-NN classifier. Of these, IVGA was somewhat more accurate than M-W and SFFS – which both gave weaker results than using no variable selection at all! In addition, we also tried both linear and nonlinear SVMs[2]. Without feature selection and selection using M-W, the classification accuracy of the linear SVM was better than k-NN. The nonlinear SVMs could clearly improve the results except for the case when no feature selection was used. The best result (90.4 %) of the whole test was obtained by using nonlinear SVM with features selected using SFFS. In IVGA, the class variable was handled simply as a part of the data, whose joint distribution was modeled using IVGA in an unsupervised manner. Based on this, it seems that IVGA is indeed able to reveal useful dependencies and structure in the data. On the other hand, the feature selection using IVGA is in a sense in harmony with the k-NN classifier, because mixture models used by IVGA are mostly local models of data, and also k-NN is totally dependent on local characteristics of the data.

Selection method	**Avg. no. of variables**	**Avg. time (s)**	**k-NN (%)**	**Linear SVM (%)**	**Nonlinear SVM (%)**
None	33	0.1	86.3	**87.1**	65.4
IVGA	5.7	906.4	**87.1**	81.0	87.0
M-W	5.5	1.9	85.4	86.4	87.6
SFFS	6.8	2568.2	85.4	84.6	**90.4**

TABLE III
COMPARISON OF THE FEATURE SELECTION METHODS: CLASSIFICATION ACCURACIES (IN PERCENT) USING k-NN AND SVM CLASSIFIERS. FOR EACH METHOD, ALSO THE AVERAGE NUMBER OF SELECTED VARIABLES IN ONE SELECTION ROUND AND THE CORRESPONDING AVERAGE COMPUTATION TIME (IN CPU SECONDS) IS REPORTED. NOTE THAT IN ORDER TO GUARANTEE A FAIR COMPARISON BETWEEN THE SVM AND THE k-NN CLASSIFIERS IN THE CASE WHERE SFFS WAS USED FOR VARIABLE SELECTION, THE SFFS SHOULD HAVE BEEN RUN USING SVM FOR THE FEATURE SELECTION. HOWEVER, SUCH A TEST WAS NOT CARRIED OUT, BECAUSE IT WOULD HAVE BEEN COMPUTATIONALLY VERY DEMANDING AND, MORE IMPORTANTLY, OUR PRIMARY GOAL WAS NOT TO COMPARE DIFFERENT CLASSIFIERS BUT FEATURE SELECTION METHODS.

In [32], dimension reduction by random projection and principal component analysis were extensively tested by using different classifiers using the ionosphere data set. However, in that study a separate validation set for determination of the best k or the number of features d was not used. Instead, they used a training set with 300 samples and a testing set of 51 samples so that the samples of the test set were always classified using the samples of the training set. The classification error was computed using values $k = 1$ and $k = 5$ and the results were averaged over 100 runs for each k. We also carried out an additional test in an identical manner using IVGA by running IVGA on the training set to obtain a ranking of the features and classifying the test set using the two values of k and different values of d. The results of that experiment are shown in Tab. IV, where the classification accuracies using PCA and RP are adopted from [32]. Using IVGA, a slightly better classification accuracy was obtained – which may be due to the fact that in our experiment the class information was utilized whereas in [32] it was not.

No. of features d	**PCA** $k=1$	**PCA** $k=5$	**RP** $k=1$	**RP** $k=5$	**IVGA** $k=1$	**IVGA** $k=5$
5	87.6	88.7	85.9	84.5	**89.4**	88.8
10	88.5	86.5	86.4	83.7	**89.4**	88.1
15	88.7	84.5	86.5	83.9	**89.1**	86.4
20	88.4	84.2	86.7	83.8	**88.7**	85.8
25	**87.9**	84.3	87.1	83.3	87.2	85.1
30	**87.2**	84.2	86.4	84.1	86.1	84.6
34 (all)	**86.6**	84.7				

TABLE IV
COMPARISON OF THE ACCURACY OF THE k-NN CLASSIFIER USING VALUES $k = 1$ AND $k = 5$ WHEN DIFFERENT NUMBER d OF FEATURES ARE USED. THE FEATURES ARE COMPUTED EITHER USING PRINCIPAL COMPONENT ANALYSIS (PCA), RANDOM PROJECTION (RP), OR IVGA. THE ACCURACIES OF PCA AND RP ARE ADOPTED FROM [32]; THE RESULTS OF IVGA ARE COMPUTED USING AN IDENTICAL PROCEDURE THAT WAS USED TO PRODUCE THEM.

VI. DISCUSSION

Many real-world problems and data sets can be divided into smaller relatively independent subproblems or subsets. Automatic discovery of such divisions can significantly help in applying different machine learning techniques to the data by reducing computational and memory requirements of processing. Modeling using IVGA calls for finding the divisions by partitioning the observed variables into separate groups so that the mutual dependencies between variables within a group are strong whereas mutual dependencies between variables in different groups are weaker.

In this paper, IVGA has been used to find a single grouping of the variables of the given data set to supposedly independent groups. In many cases, there may still exist interesting weak residual dependencies between the different variable groups. One avenue for future research is to extend the grouping model into a hierarchical IVGA that is able to model the residual dependencies between the groups of variables as in modular mixture models [8].

[2]We used a freely available software package [31] with default settings; see also `http://svmlight.joachims.org/`. The package was used in a similar manner in [32] for training of SVMs.

Biclustering – clustering of both variables and samples – is very popular technique for solving certain problems in bioinformatics. In such applications it could be useful to ease the strict grouping of the variables using IVGA. This could be accomplished by allowing different partitions in different parts of the data set using, for instance, a mixture-of-IVGAs type of model.

From the perspective of considering IVGA as a general concept it might be useful to implement models of different type including also linear models. This would allow modeling of each variable group with the best model type for that particular subproblem, and depending on the types of dependencies within the problem. Such extensions naturally require the derivation of a cost function for each additional model family, but there are simple tools for automating this process [33], [34].

The stochastic nature of the grouping algorithm makes its computational complexity difficult to analyze. Empirically the time required for convergence to a neighborhood of a locally optimal grouping seems to have a roughly quadratic dependence on both the number of variables and the number of data samples. The latter characteristics is due to the fact that in practise the data does not exactly follow the mixture model and thus multiple mixture components are used when there are many samples. Convergence to the exact local optimum typically takes significantly longer, but it is usually not necessary as even nearly optimal results are often good enough in practice.

Although the presented IVGA model appears quite simple, several computational speedup techniques are needed for it to work efficiently enough. Some of these may be of interest by themselves, irrespective of the context of this work. In particular worth mentioning are the adaptive tuning of operation probabilities in the grouping algorithm (Sec. IV-B1) as well as the model compression principle (Sec. IV-B2).

By providing the source code of the method for public use we invite others both to use IVGA and to contribute to extending it. A MATLAB package of our IVGA implementation is available at http://www.cis.hut.fi/projects/ivga/.[3]

VII. Conclusion

In this paper, we have presented independent variable group analysis (IVGA), which is a method for modeling data through mutually independent groups of variables. The approach has been shown to be useful in real-world problems: It decreases computational burden of other machine learning methods and also increases their accuracy by letting them concentrate on the essential dependencies of the data. The general nature of IVGA allows many potential practical applications. It can be viewed as a tool for compact modeling of data, an algorithm for clustering variables, or as a means for feature selection.

Appendix A
Specification of the Mixture Model

A mixture model for the random variable $\mathbf{x}(t)$ can be written with the help of an auxiliary variable $c(t)$ denoting the index of the active mixture component as illustrated in the right part of Fig. 4 (in Sec. III-C). In our IVGA model, the mixture model for the variable groups is chosen to be as simple as possible for computational reasons. This is done by restricting the components $p(\mathbf{x}(t)|\theta_i, \mathcal{H})$ of the mixture to be such that different variables are assumed independent. This yields

$$\begin{aligned} p(\mathbf{x}(t)|\mathcal{H}) &= \sum_i p(\mathbf{x}(t)|\theta_i, \mathcal{H}) p(c(t) = i) \\ &= \sum_i p(c(t) = i) \prod_j p(x_j(t)|\theta_{i,j}, \mathcal{H}), \end{aligned} \tag{9}$$

where $\theta_{i,j}$ are the parameters of the ith mixture component for the jth variable. Dependencies between the variables are modeled only through the mixture. The variable c has a multinomial distribution with parameters $\boldsymbol{\pi}_c$ that have a Dirichlet prior with parameters $\boldsymbol{u}_c$:

$$p(c(t)|\boldsymbol{\pi}_c, \mathcal{H}) = \text{Multinom}(c(t);\ \boldsymbol{\pi}_c) \tag{10}$$

$$p(\boldsymbol{\pi}_c|\mathbf{u}_c, \mathcal{H}) = \text{Dirichlet}(\boldsymbol{\pi}_c;\ \mathbf{u}_c). \tag{11}$$

The use of a mixture model allows for both categorical and continuous variables. For continuous variables the mixture is a heteroscedastic Gaussian mixture, that is, all mixture components have their own precisions. Thus

$$p(x_j(t)|\theta_{i,j}, \mathcal{H}) = N(x_j(t);\ \mu_{i,j}, \rho_{i,j}), \tag{12}$$

where $\mu_{i,j}$ is the mean and $\rho_{i,j}$ is the precision of the Gaussian. The parameters $\mu_{i,j}$ and $\rho_{i,j}$ have hierarchical priors

$$p(\mu_{i,j}|\mu_{\mu_j}, \rho_{\mu_j}, \mathcal{H}) = N(\mu_{i,j};\ \mu_{\mu_j}, \rho_{\mu_j}) \tag{13}$$

$$p(\rho_{i,j}|\alpha_{\rho_j}, \beta_{\rho_j}, \mathcal{H}) = \text{Gamma}(\rho_{i,j};\ \alpha_{\rho_j}, \beta_{\rho_j}). \tag{14}$$

For categorical variables, the mixture is a simple mixture of multinomial distributions so that

$$p(x_j(t)|\theta_{i,j}, \mathcal{H}) = \text{Multinom}(x_j(t);\ \boldsymbol{\pi}_{i,j}). \tag{15}$$

The probabilities $\boldsymbol{\pi}_{i,j}$ have a Dirichlet prior

$$p(\boldsymbol{\pi}_{i,j}|\mathbf{u}_j, \mathcal{H}) = \text{Dirichlet}(\boldsymbol{\pi}_{i,j};\ \mathbf{u}_j). \tag{16}$$

Combining these yields the joint probability of all parameters (here $\mathbf{c} = [c(1), \ldots, c(T)]^T$):

$$\begin{aligned} p(\mathcal{D}, \mathbf{c}, \boldsymbol{\pi}_c, \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\rho}) = \\ \prod_t \Big[p(c(t)|\boldsymbol{\pi}_{\boldsymbol{c}})\Big] p(\boldsymbol{\pi}_c|\mathbf{u}_c) \\ \prod_i \Bigg[\prod_{j:x_j \text{ categorical}} \Big[p(\boldsymbol{\pi}_{i,j}|\mathbf{u}_j)\Big] \\ \prod_{j:x_j \text{ continuous}} \Big[p(\mu_{i,j}|\mu_{\mu_j}, \rho_{\mu_j}) p(\rho_{i,j}|\alpha_{\rho_j}, \beta_{\rho_j})\Big] \Bigg] \\ \prod_t \Bigg[\prod_{j:x_j \text{ categorical}} p(x_j(t)|c(t), \boldsymbol{\pi}_{\cdot,j}) \\ \prod_{j:x_j \text{ continuous}} p(x_j(t)|c(t), \mu_{\cdot,j}, \rho_{\cdot,j}) \Bigg]. \end{aligned} \tag{17}$$

All the component distributions of this expression have been introduced above in Eqs. (12)-(16).

[3] Also the MATLAB scripts for the synthetic data and the ionosphere data experiments can be found in the same location.

The corresponding variational approximation is

$$q(\mathbf{c},\boldsymbol{\pi}_c,\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\rho}) = q(\mathbf{c})q(\boldsymbol{\pi}_c)q(\boldsymbol{\pi})q(\boldsymbol{\mu})q(\boldsymbol{\rho}) = \prod_t \Big[q(c(t)|\mathbf{w}(t))\Big] q(\boldsymbol{\pi}_c|\hat{\mathbf{u}}_c) \prod_i \Bigg[\prod_{j:x_j \text{ categorical}} \Big[q(\boldsymbol{\pi}_{i,j}|\hat{\mathbf{u}}_{i,j})\Big] \prod_{j:x_j \text{ continuous}} \Big[q(\mu_{i,j}|\hat{\mu}_{\mu_{i,j}},\hat{\rho}_{\mu_{i,j}}) q(\rho_{i,j}|\hat{\alpha}_{\rho_{i,j}},\hat{\beta}_{\rho_{i,j}})\Big]\Bigg], \quad (18)$$

where the newly introduced parameters are the variational parameters of the factors

$$q(c(t)) = \text{Multinom}(c(t);\ \mathbf{w}(t)) \quad (19)$$
$$q(\boldsymbol{\pi}_c) = \text{Dirichlet}(\boldsymbol{\pi}_c;\ \hat{\mathbf{u}}_c) \quad (20)$$
$$q(\boldsymbol{\pi}_{i,j}) = \text{Dirichlet}(\boldsymbol{\pi}_{i,j};\ \hat{\mathbf{u}}_{i,j}) \quad (21)$$
$$q(\mu_{i,j}) = N(\mu_{i,j};\ \hat{\mu}_{\mu_{i,j}},\hat{\rho}_{\mu_{i,j}}) \quad (22)$$
$$q(\rho_{i,j}) = \text{Gamma}(\rho_{i,j};\ \hat{\alpha}_{\rho_{i,j}},\hat{\beta}_{\rho_{i,j}}). \quad (23)$$

Because of the conjugacy of the model, these are optimal forms for the components of the approximation, given the factorization. Specification of the approximation allows the evaluation of the cost of Eq. (6) and the derivation of update rules for the parameters as shown below in Appendix B. The hyperparameters $\mu_{\mu_j}, \rho_{\mu_j}, \alpha_{\rho_j}, \beta_{\rho_j}$ are updated using type II maximum likelihood estimation. The parameters of the fixed Dirichlet priors are set to values corresponding to the Jeffreys prior.

APPENDIX B
DERIVATION OF THE COST FUNCTION AND THE UPDATE RULES

The cost function of Eq. (6) can be expressed, using $\langle\cdot\rangle$ to denote expectation over q, as

$$\Big\langle \log \frac{q(\boldsymbol{\theta})}{p(\mathcal{D},\boldsymbol{\theta}|\mathcal{H})} \Big\rangle = \big\langle \log q(\boldsymbol{\theta}) - \log p(\boldsymbol{\theta}) \big\rangle - \big\langle \log p(\mathcal{D}|\boldsymbol{\theta}) \big\rangle \quad (24)$$

Now, being expected logarithms of products of probability distributions over the factorial posterior approximation q, the terms easily split further. The terms of the cost function are presented as the costs of the different parameters and the likelihood term. Some of the notation used in the formulae is introduced in Tab. V.

Symbol	Explanation
C	Number of mixture components
T	Number of data points
D_{cont}	Number of continuous dimensions
S_j	The number of categories in nominal dimension j
$I_k(x)$	An indicator for x being of category k
Γ	The gamma function (not the distribution pdf)
Ψ	The digamma function, that is $\Psi(x) = \frac{d}{dx}\ln(\Gamma(x))$
$w_i(t)$	The multinomial probability/weight of the ith mixture component in the $\mathbf{w}(t)$ of data point t

TABLE V
NOTATION

A. Terms of the Cost Function

$$\big\langle \log q(\mathbf{c}|\mathbf{w}) - \log p(\mathbf{c}|\boldsymbol{\pi}_\mathbf{c}) \big\rangle = \sum_{t=1}^{T}\sum_{i=1}^{C} w_i(t)\big[\log w_i(t) - [\Psi(\hat{u}_{c_i}) - \Psi(\sum_{i'=1}^{C}\hat{u}_{c_{i'}})]\big] \quad (25)$$

$$\big\langle \log q(\boldsymbol{\pi}_\mathbf{c}|\hat{\mathbf{u}}_\mathbf{c}) - \log p(\boldsymbol{\pi}_\mathbf{c}|\mathbf{u}_\mathbf{c}) \big\rangle = \sum_{i=1}^{C}\Big[(\hat{u}_{c_i} - u_{c_i})[\Psi(\hat{u}_{c_i}) - \Psi(\sum_{i'=1}^{C}\hat{u}_{c_{i'}})] - \log\Gamma(\hat{u}_{c_i}) + \log\Gamma(u_{c_i})\Big] + \log\Gamma(\sum_{i'=1}^{C}\hat{u}_{c_{i'}}) - \log\Gamma(\sum_{i'=1}^{C}u_{c_{i'}}) \quad (26)$$

$$\big\langle \log q(\boldsymbol{\pi}|\hat{\mathbf{u}}) - \log p(\boldsymbol{\pi}|\mathbf{u}) \big\rangle = \sum_{j:x_j \text{ categorical}} \Bigg[\sum_{i=1}^{C}\sum_{k=1}^{S_j}\Big[(\hat{u}_{i,j,k} - u_{j,k}) [\Psi(\hat{u}_{i,j,k}) - \Psi(\sum_{k'=1}^{S_j}\hat{u}_{i,j,k'})]\Big] + \sum_{i=1}^{C}\Big[\log\Gamma(\sum_{k'=1}^{S_j}\hat{u}_{i,j,k'}) - \sum_{k=1}^{S_j}\log\Gamma\big(\hat{u}_{i,j,k}\big)\Big] + C\Big[-\log\Gamma(\sum_{k'=1}^{S_j}u_{j,k'}) + \sum_{k=1}^{S_j}\log\Gamma(u_{j,k})\Big]\Bigg] \quad (27)$$

$$\big\langle \log q(\boldsymbol{\mu}|\hat{\boldsymbol{\mu}}_\boldsymbol{\mu},\hat{\boldsymbol{\rho}}_\boldsymbol{\mu}) - \log p(\boldsymbol{\mu}|\boldsymbol{\mu}_\boldsymbol{\mu},\boldsymbol{\rho}_\boldsymbol{\mu}) \big\rangle = -\frac{CD_{\text{cont}}}{2} + \sum_{j:x_j \text{ continuous}}\sum_{i=1}^{C}\Big[\log\frac{\hat{\rho}_{\mu_{i,j}}}{2\rho_{\mu_j}} + \frac{\rho_{\mu_j}}{2}\big[\hat{\rho}_{\mu_{i,j}}^{-1} + (\hat{\mu}_{\mu_{i,j}} - \mu_{\mu_j})^2\big]\Big] \quad (28)$$

$$\big\langle \log q(\boldsymbol{\rho}|\hat{\boldsymbol{\alpha}}_\boldsymbol{\rho},\hat{\boldsymbol{\beta}}_\boldsymbol{\rho}) - \log p(\boldsymbol{\rho}|\boldsymbol{\alpha}_\boldsymbol{\rho},\boldsymbol{\beta}_\boldsymbol{\rho}) \big\rangle = \sum_{j:x_j \text{ continuous}}\sum_{i=1}^{C}\Big[\log\Gamma(\alpha_{\rho_j}) - \log\Gamma(\hat{\alpha}_{\rho_{i,j}}) + \hat{\alpha}_{\rho_{i,j}}\log\hat{\beta}_{\rho_{i,j}} - \alpha_{\rho_j}\log\beta_{\rho_j} + (\hat{\alpha}_{\rho_{i,j}} - \alpha_{\rho_j})\big(\Psi(\hat{\alpha}_{\rho_{i,j}}) - \log\hat{\beta}_{\rho_{i,j}}\big) + \frac{\hat{\alpha}_{\rho_{i,j}}}{\hat{\beta}_{\rho_{i,j}}}(\beta_{\rho_j} - \hat{\beta}_{\rho_{i,j}})\Big] \quad (29)$$

$$\Big\langle -\log p(\mathcal{D}|\mathbf{c},\boldsymbol{\pi}_{\mathbf{c}},\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\rho})\Big\rangle = \frac{T\log(2\pi)D_{\text{cont}}}{2} + \sum_{t=1}^{T}\sum_{i=1}^{C}\Bigg\{ w_i(t)\Bigg[-\sum_{j:x_j \text{ categorical}}\Big[\Psi(\hat{u}_{i,j,x_j(t)}) - \Psi(\sum_{k'=1}^{S_j}\hat{u}_{i,j,k'})\Big] + \frac{1}{2}\sum_{j:x_j \text{ continuous}}\Big[\frac{\hat{\alpha}_{\rho_{i,j}}}{\hat{\beta}_{\rho_{i,j}}}\big(\hat{\rho}_{\mu_{i,j}}^{-1} + (x_j(t)-\hat{\mu}_{\mu_{i,j}})^2\big) - \big(\Psi(\hat{\alpha}_{\rho_{i,j}}) - \log\hat{\beta}_{\rho_{i,j}}\big)\Big]\Bigg]\Bigg\} \quad (30)$$

B. On the Iteration Formulae and Initialization

The iteration formulae for one full iteration of mixture model adaptation consist of simple coordinate-wise re-estimations of the parameters. This is like variational expectation-maximization (EM) iteration. The update rules of the hyperparameters μ_{μ_j}, ρ_{μ_j}, α_{ρ_j} and β_{ρ_j} are based on type II maximum likelihood estimation, that is, maximizing $p(\mathcal{D}|\mathcal{H},\mu_{\mu_j},\rho_{\mu_j},\alpha_{\rho_j},\beta_{\rho_j})$.

Before the iteration the mixture components are initialized using the dataset and a pseudorandom seed number that is used to make the initialization stochastic but reproducible using the same random seed. The mixture components are initialized as equiprobable.

C. The Iteration Formulae

One full iteration cycle:

1) Update $\mathbf{w}$

$$w_i^*(t) \leftarrow \exp\Bigg(\Psi(\hat{u}_{c_i}) + \sum_{j:x_j \text{ categorical}}\Big[\Psi(\hat{u}_{i,j,x_j(t)}) - \Psi(\sum_{k'=1}^{S_j}\hat{u}_{i,j,k'})\Big] - \frac{1}{2}\sum_{j:x_j \text{ continuous}}\Big[\frac{\hat{\alpha}_{\rho_{i,j}}}{\hat{\beta}_{\rho_{i,j}}}\big(\hat{\rho}_{\mu_{i,j}}^{-1} + (x_j(t)-\hat{\mu}_{\mu_{i,j}})^2\big) - \big(\Psi(\hat{\alpha}_{\rho_{i,j}}) - \log\hat{\beta}_{\rho_{i,j}}\big)\Big]\Bigg)$$
$$w_i(t) \leftarrow \frac{w_i^*(t)}{\sum_{i'=1}^{C} w_{i'}^*(t)} \quad (31)$$

2) Update $\hat{\mathbf{u}}_{\mathbf{c}}$

$$\hat{u}_{c_i} \leftarrow u_{c_i} + \sum_{t=1}^{T} w_i(t) \quad (32)$$

3) Update categorical dimensions of the mixture components

$$\hat{u}_{i,j,k} \leftarrow u_{j,k} + \sum_{t=1}^{T} w_i(t) I_k(x_j(t)), \quad (33)$$

where $I_k(x_j(t)) = 1$ if $x_j(t) = k$ and zero otherwise.

4) Update continuous dimensions of the mixture components

$$\hat{\mu}_{\mu_{i,j}} \leftarrow \frac{\rho_{\mu_j}\mu_{\mu_j} + \frac{\hat{\alpha}_{\rho_{i,j}}}{\hat{\beta}_{\rho_{i,j}}}\sum_{t=1}^{T} w_i(t)x_j(t)}{\rho_{\mu_j} + \frac{\hat{\alpha}_{\rho_{i,j}}}{\hat{\beta}_{\rho_{i,j}}}\sum_{t=1}^{T} w_i(t)} \quad (34)$$

$$\hat{\rho}_{\mu_{i,j}} \leftarrow \rho_{\mu_j} + \frac{\hat{\alpha}_{\rho_{i,j}}}{\hat{\beta}_{\rho_{i,j}}}\sum_{t=1}^{T} w_i(t) \quad (35)$$

$$\hat{\alpha}_{\rho_{i,j}} \leftarrow \alpha_{\rho_j} + \frac{1}{2}\sum_{t=1}^{T} w_i(t) \quad (36)$$

$$\hat{\beta}_{\rho_{i,j}} \leftarrow \beta_{\rho_j} + \frac{1}{2}\sum_{t=1}^{T} w_i(t)\big[\hat{\rho}_{\mu_{i,j}}^{-1} + (\hat{\mu}_{\mu_{i,j}} - x_j(t))^2\big] \quad (37)$$

5) Update the hyperparameters

$$\mu_{\mu_j} \leftarrow \frac{1}{C}\sum_{i=1}^{C}\hat{\mu}_{\mu_{i,j}} \quad (38)$$

$$\rho_{\mu_j} \leftarrow C\left[\sum_{i=1}^{C}\Big(\hat{\rho}_{\mu_{i,j}}^{-1} + (\hat{\mu}_{\mu_{i,j}} - \mu_{\mu_j})^2\Big)\right]^{-1} \quad (39)$$

$$\alpha_{\rho_j} \leftarrow \alpha_{\rho_j} + \frac{1}{2}\big(\Psi(\alpha_{\rho_j}) - \log(\alpha_{\rho_j})\big)^{-1} + \frac{1}{2}\left(\frac{1}{C}\sum_{i=1}^{C}\Big(\log(\hat{\beta}_{\rho_{i,j}}) - \Psi(\hat{\alpha}_{\rho_{i,j}})\Big) - \log\left(\frac{C}{\sum_{i=1}^{C}\frac{\hat{\alpha}_{\rho_{i,j}}}{\hat{\beta}_{\rho_{i,j}}}}\right)\right)^{-1} \quad (40)$$

$$\beta_{\rho_j} \leftarrow \alpha_{\rho_j}\frac{C}{\sum_{i=1}^{C}\frac{\hat{\alpha}_{\rho_{i,j}}}{\hat{\beta}_{\rho_{i,j}}}} \quad (41)$$

The derivation of the fixed point update for the Gamma hyperparameters is presented in detail in [35].

ACKNOWLEDGMENT

We would like to thank Zoubin Ghahramani for interesting discussions. We also wish to thank Valor Computerized Systems (Finland) Oy for providing us with the data used in the printed circuit board assembly experiment. This work was supported in part by the Finnish Centre of Excellence Programme (2000–2005) under the project New Information Processing Principles, and by the IST Programme of the European Community, under the PASCAL Network of Excellence, IST-2002-506778. This publication only reflects the authors' views.

REFERENCES

[1] K. Lagus, E. Alhoniemi, and H. Valpola, "Independent variable group analysis," in *International Conference on Artificial Neural Networks - ICANN 2001*, ser. LNCS, G. Dorffner, H. Bischof, and K. Hornik, Eds., vol. 2130. Vienna, Austria: Springer, August 2001, pp. 203–210.

[2] K. Lagus, E. Alhoniemi, J. Seppä, A. Honkela, and P. Wagner, "Independent variable group analysis in learning compact representations for data," in *Proceedings of the International and Interdisciplinary Conference on Adaptive Knowledge Representation and Reasoning (AKRR'05)*, T. Honkela, V. Könönen, M. Pöllä, and O. Simula, Eds., Espoo, Finland, June 2005, pp. 49–56.

[3] J.-F. Cardoso, "Multidimensional independent component analysis," in *Proceedings of ICASSP'98*, Seattle, 1998.

[4] G. E. Hinton and R. S. Zemel, "Autoencoders, minimum description length and Helmholtz free energy," in *Neural Information Processing Systems 6*, J. et al, Ed. San Mateo, CA: Morgan Kaufmann, 1994.

[5] R. S. Zemel, "A minimum description length framework for unsupervised learning," Ph.D. dissertation, University of Toronto, 1993.

[6] K. Viikki, E. Kentala, M. Juhola, I. Pyykkö, and P. Honkavaara, "Generating decision trees from otoneurological data with a variable grouping method," *Journal of Medical Systems*, vol. 26, no. 5, pp. 415–425, 2002.

[7] A. Tucker, S. Swift, and X. Liu, "Variable grouping in multivariate time series via correlation," *IEEE Transactions on Systems, Man and Cybernetics, Part B*, vol. 31, no. 2, pp. 235–245, 2001.

[8] H. Attias, "Learning in high dimensions: modular mixture models," in *Proceedings of the 8th International Conference on Artificial Intelligence and Statistics*, Key West, FL, USA, 2001, pp. 144–148.

[9] E. Segal, D. Pe'er, A. Regev, D. Koller, and N. Friedman, "Learning module networks," *Journal of Machine Learning Research*, vol. 6, pp. 557–588, April 2005.

[10] Y. Cheng and G. M. Church, "Biclustering of expression data," in *Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology (ISMB)*, 2000, pp. 93–103.

[11] S. C. Madeira and A. L. Oliveira, "Biclustering algorithms for biological data analysis: A survey," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 1, no. 1, pp. 24–45, 2004.

[12] R. Kohavi and G. H. John, "Wrappers for feature subset selection," *Artificial Intelligence*, vol. 97, no. 1–2, pp. 273–324, December 1997.

[13] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *Journal on Machine Learning Research*, pp. 1157–1182, March 2003.

[14] M. Studený and J. Vejnarová, "The multiinformation function as a tool for measuring stochastic dependence," in *Learning in Graphical Models*, M. Jordan, Ed. Cambridge, MA, USA: The MIT Press, 1999, pp. 261–297.

[15] M. Nilsson, H. Gustafsson, S. V. Andersen, and W. B. Kleijn, "Gaussian mixture model based mutual information estimation between frequency bands in speech," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing 2002 (ICASSP '02)*, vol. 1, 2002, pp. I–525–I–528.

[16] D. R. Wolf, "Mutual information as a Bayesian measure of independence," 1994, uRL: http://arxiv.org/abs/comp-gas/9511002.

[17] T. Minka, "Bayesian inference, entropy, and the multinomial distribution," 2003, online tutorial, URL: http://research.microsoft.com/~minka/papers/multinomial.html.

[18] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul, "An introduction to variational methods for graphical models," in *Learning in Graphical Models*, M. Jordan, Ed. Cambridge, MA, USA: The MIT Press, 1999, pp. 105–161.

[19] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. New York: Wiley, 1991.

[20] B. J. Frey and G. E. Hinton, "Efficient stochastic source coding and an application to a Bayesian network source model," *The Computer Journal*, vol. 40, no. 2/3, pp. 157–165, 1997.

[21] J. Rissanen, "Modeling by shortest data description," *Automatica*, vol. 14, no. 5, pp. 465–471, 1978.

[22] A. Honkela and H. Valpola, "Variational learning and bits-back coding: an information-theoretic view to Bayesian learning," *IEEE Transactions on Neural Networks*, vol. 15, no. 4, pp. 800–810, 2004.

[23] G. McLachlan and D. Peel, *Finite Mixture Models*. New York: Wiley, 2000.

[24] D. J. C. MacKay, *Information Theory, Inference, and Learning Algorithms*. Cambridge: Cambridge University Press, 2003.

[25] E. Alhoniemi, T. Knuutila, M. Johnsson, J. Röyhkiö, and O. S. Nevalainen, "Data mining in maintenance of electronic component libraries," in *Proceedings of the IEEE 4th International Conference on Intelligent Systems Design and Applications*, vol. 1, 2004, pp. 403–408.

[26] M. J. Zaki, S. Parthasarathy, M. O. ara, and W. Li, "New algorithms for fast discovery of association rules," in *Proceedings of the 3rd International Conference on Knowledge Disc overy and Data Mining (KDD)*, 1997, pp. 283–286.

[27] C. L. Blake and C. J. Merz, "UCI repository of machine learning databases," 1998, URL: http://www.ics.uci.edu/~mlearn/MLRepository.html.

[28] S. Theodoridis and K. Koutroumbas, *Pattern recognition*, 2nd ed. Academic Press, 2003.

[29] H. B. Mann and D. R. Whitney, "On a test of whether one of two random variables is stochastically larger than the other," *The Annals of Mathematical Statistics*, vol. 18, no. 1, pp. 50–60, March 1947.

[30] P. Pudil, J. Novovičová, and J. Kittler, "Floating search methods in feature selection," *Pattern Recognition Letters*, vol. 15, no. 11, pp. 1119–1125, November 1994.

[31] T. Joachims, "Making large-scale SVM learning practical," in *Advances in Kernel Methods - Support Vector Learning*, B. Schölkopf, C. Burges, and A. Smola, Eds. MIT-Press, 1999.

[32] D. Fradkin and D. Madigan, "Experiments with random projections for machine learning," in *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*. New York, NY, USA: ACM Press, August 24-27 2003, pp. 517–522.

[33] J. Winn and C. M. Bishop, "Variational message passing," *Journal of Machine Learning Research*, vol. 6, pp. 661–694, April 2005.

[34] M. Harva, T. Raiko, A. Honkela, H. Valpola, and J. Karhunen, "Bayes Blocks: An implementation of the variational Bayesian building blocks framework," in *Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence (UAI 2005)*, Edinburgh, Scotland, 2005, pp. 259–266.

[35] H. Valpola and A. Honkela, "Hyperparameter adaptation in variational Bayes for the gamma distribution," Helsinki University of Technology, Publications in Computer and Information Science, Espoo, Finland, Tech. Rep. E6, 2006, available at http://www.cis.hut.fi/Publications/.