Agglomerative Independent Variable Group Analysis

Antti Honkela [a,*] Jeremias Seppä [a] Esa Alhoniemi [b]

[a] *Adaptive Informatics Research Centre, Helsinki University of Technology*
P.O. Box 5400, FI-02015 TKK, Finland

[b] *University of Turku, Department of Information Technology*
FI-20014 University of Turku, Finland

Abstract

Independent Variable Group Analysis (IVGA) is a method for grouping dependent variables together while keeping mutually independent or weakly dependent variables in separate groups. In this paper two variants of an agglomerative method for learning a hierarchy of IVGA groupings are presented. The method resembles hierarchical clustering, but the choice of clusters to merge is based on variational Bayesian model comparison. This is approximately equivalent to using a distance measure based on a model-based approximation of mutual information between groups of variables. The approach also allows determining optimal cutoff points for the hierarchy. The method is demonstrated to find sensible groupings of variables that can be used for feature selection and ease construction of a predictive model.

Key words: hierarchical clustering, independent variable group analysis, mutual information, variable grouping, variational Bayesian learning

1 Introduction

Large data sets often contain subsets of variables that are only weakly dependent or independent. Simple examples of this include, for example, pixels related to different objects in an image or genes related to different pathways in a gene expression data set. Automatically discovering such groups can help

* Corresponding author.
Email address: antti.honkela@tkk.fi (Antti Honkela).
URL: http://www.cis.hut.fi/projects/ivga/ (Antti Honkela).

in understanding the structure of the data set as well as focusing further modelling efforts to smaller and more meaningful subproblems.

Grouping or clustering variables based on their mutual dependence was the objective of the Independent Variable Group Analysis [13,1] (IVGA) method. In this paper we extend this approach by considering hierarchical grouping or clustering of variables. The resulting hierarchy provides even more insight into the dependence structure of the variables in a data set. Alternatively, the method can be seen as a systematic search procedure for exploring regular flat clusterings or groupings.

The method presented in this paper is called *Agglomerative Independent Variable Group Analysis* (AIVGA). AIVGA is based on the idea of agglomerative hierarchical clustering of variables [7]. Initially, each variable is placed in a group of its own. The groups are then combined by greedily selecting the operation that decreases the cost most. The result is a hierarchy of groupings of different sizes.

The general problem of variable grouping or clustering and some related methods are studied in Sec. 2. The computational methodologies behind AIVGA and the algorithm itself are presented in Sec. 3. Sec. 4 presents experimental results on applying the algorithm to a synthetic data set with known structure and using it as a preprocessing step in classification and prediction tasks. The paper concludes with discussion and conclusions in Secs. 5 and 6.

2 Variable Grouping

IVGA is a method of grouping mutually dependent variables together while keeping mutually independent variables in separate groups [13,1]. This is done by maximising a bound on the marginal likelihood of a set of models enforcing the grouping. This is in turn approximately equal to minimising an approximation of the mutual information between the groups. The original IVGA algorithm [13,1] uses a heuristic combinatorial hill climbing search to find the optimal grouping [13]. In this method, the size of the resulting grouping is not predetermined but optimised during the learning process.

Finding the optimal grouping with IVGA can be seen as clustering the variables based on a distance measure of mutual dependence. A somewhat similar approach based on direct estimation of mutual information and its multivariate generalisation of multi-information [20] has been proposed in [19].

In the original IVGA [13,1] as well as the AIVGA method presented in this paper, the mutual information of the data is not estimated directly. Instead,

a model-based approach is used: probabilistic models are inferred for the variables separately and grouped together, and the configuration yielding the highest estimated marginal likelihood is taken as the best grouping. As shown in [1], this method is approximately equivalent to minimising the mutual information between the groups, except that it takes into account model complexity as well as finite sample size and thus often prefers a larger number of groups. In [1], the groups are modelled using finite mixture models, but in principle other probabilistic models could be used as well.

The result of the AIVGA algorithm can be seen as a hierarchical clustering of variables, similarly as the solution of an IVGA problem can be seen as a regular clustering of the variables. For each level in the clustering, there is a probabilistic model for the data consisting of a varying number of independent parts, but there is no single generative model for the hierarchy. Graphical models corresponding to two consecutive stages of the algorithm are illustrated in Fig. 1. The Bayesian marginal likelihood objective used in AIVGA allows determining the optimal points to cut the tree similarly as in the Bayesian hierarchical clustering method [9].

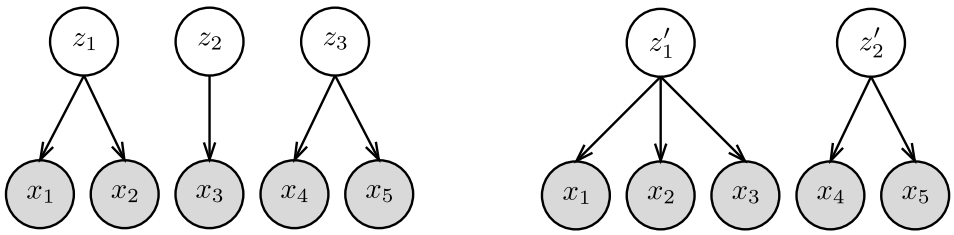

Fig. 1. Two IVGA models from consecutive stages of the AIVGA algorithm as graphical models. The shaded nodes represent observed variables whereas the unshaded nodes are latent. Each group of observed variables has a separate latent variable, the mixture index variable z_i, explaining the variables within the group.

The AIVGA approach can be contrasted with models such as the hierarchical latent class model [22] in which the latent variables associated with different groups are members of higher level groups. This leads to a rooted tree graphical model with all leaf nodes observed and all other nodes representing latent variables. The hierarchical latent class model presented in [22] is, however, limited to categorical data whereas AIVGA can also be applied to continuous and mixed data. The simpler structure of the separate IVGA models makes the method computationally more efficient.

AIVGA is closely related to the hierarchical clustering algorithm using mutual information presented in [11]. The main difference is that AIVGA provides a generative model for the data at each level of the hierarchy. Also, the Bayesian cost function allows determining the optimal cutoff point or points for the hierarchy.

In the context of clustering, the IVGA and AIVGA methods are especially

interesting as the mixture models used to model the groups yield a secondary soft clustering of the samples in addition to the hard clustering of the variables. This relates IVGA to biclustering and makes AIVGA in a sense a hierarchical alternative of biclustering. However, both IVGA and AIVGA always cluster all the variables and samples, while standard biclustering methods concentrate only on interesting subsets of variables and samples.

The IVGA models formed in different stages of the AIVGA algorithm have several other possible interpretations and therefore many connections to other related methods as well. These are reviewed in detail in [1].

3 Algorithm

Let us assume that the data set $\boldsymbol{X}$ consists of vectors $\mathbf{x}(t),\ t = 1, \ldots, T$. The vectors are N-dimensional with the individual components denoted by $x_j,\ j = 1, \ldots, N$, and let $\boldsymbol{X}_j = (x_j(1), \ldots, x_j(T))$ and $\boldsymbol{X}_{\mathcal{G}_i} = \{\boldsymbol{X}_j | j \in \mathcal{G}_i\}$. The objective of IVGA and AIVGA 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 sum of the marginal log-likelihoods of the models $\mathcal{H}_i$ for the different groups is maximised. As shown in [1], this is approximately equivalent to minimising the mutual information (or multi-information [20] when $M > 2$) between the groups. This equivalence is only approximate as the Bayesian approach takes into account model complexity as well as finite sample size and often prefers forming multiple groups, whereas mutual information always decreases as the number of groups is increased. In some cases, such as when the number of samples is small compared to the dimensionality, the effect of these factors may be very significant, making the correspondence between our cost and the mutual information across groupings of different sizes very weak.

To approximate the marginal log-likelihoods, variational Bayes (VB) approximations $q_i(\boldsymbol{\theta}_i)$ are fitted to the posteriors $p(\boldsymbol{\theta}_i | \boldsymbol{X}_{\mathcal{G}_i}, \mathcal{H}_i))$ of the models of the different groups. The models are fitted to minimise the cost function or free energy $\mathcal{C}$

$$\begin{aligned} \mathcal{C}(\mathcal{G}) &= \sum_i \mathcal{C}(\boldsymbol{X}_{\mathcal{G}_i} | \mathcal{H}_i) = \sum_i \int \log \frac{q_i(\boldsymbol{\theta}_i)}{p(\boldsymbol{X}_{\mathcal{G}_i}, \boldsymbol{\theta}_i | \mathcal{H}_i)} q_i(\boldsymbol{\theta}_i) d\boldsymbol{\theta}_i \\ &= \sum_i \left[D_{\mathrm{KL}}(q_i(\boldsymbol{\theta}_i) || p(\boldsymbol{\theta}_i | \boldsymbol{X}_{\mathcal{G}_i}, \mathcal{H}_i)) - \log p(\boldsymbol{X}_{\mathcal{G}_i} | \mathcal{H}_i) \right] \\ &\geq -\sum_i \log p(\boldsymbol{X}_{\mathcal{G}_i} | \mathcal{H}_i), \end{aligned} \tag{1}$$

where $D_{\mathrm{KL}}(q||p)$ is the Kullback–Leibler divergence between q and p.

The cost is approximately related to the mutual information as

$$\mathcal{C}(\mathcal{G}) \geq -\sum_i \log p(\boldsymbol{X}_{\mathcal{G}_i}|\mathcal{H}_i) \approx TI_{\mathcal{G}}(\mathbf{x}) + TH(\mathbf{x}), \tag{2}$$

where $H(\mathbf{x})$ is the entropy of the random vector $\mathbf{x}$ and $I_{\mathcal{G}}(\mathbf{x}) = \sum_i H(\{x_j|j \in \mathcal{G}_i\}) - H(\mathbf{x})$ is the mutual or multi-information related to the grouping $\mathcal{G}$ [1].

To summarise, our aim is to minimise the sum of free energies of a number of models spanning all the variables. As a byproduct, approximations $q_i(\boldsymbol{\theta}_i)$ of the posterior distributions $p(\boldsymbol{\theta}_i|\boldsymbol{X}_{\mathcal{G}_i}, \mathcal{H}_i)$ of the parameters $\boldsymbol{\theta}_i$ of the models $\mathcal{H}_i$ for each group are also obtained, such that the approximations minimise the Kullback–Leibler divergence between the approximation and the exact posterior.

3.1 Agglomerative Grouping Algorithm

An outline of the AIVGA algorithm is presented as Algorithm 1. The algorithm is a classical agglomerative algorithm for hierarchical clustering [7]. It is initialised by placing each observed variable in a group of its own. After that two groups are merged so that the reduction in the cost is as large as possible. As the cost function of Eq. (1) is additive over the groups, changes can be evaluated locally by only recomputing models involving variables in the changed groups.

Algorithm 1 Outline of the AIVGA algorithm.

```
c ← N, G_i ← {x_i}, i = 1, ..., N
while c > 1 do
    c ← c − 1
    Find groups G_i and G_j such that C({G_i ∪ G_j}) − C({G_i, G_j}) is minimal
    Merge groups G_i and G_j
    Save grouping of size c
end while
```

The AIVGA algorithm requires fitting $\mathcal{O}(N^2)$ models for the groups. This is necessary as for the first step all pairs of variables have to be considered for possible merge. After this, only further merges of the previously merged group with all the other groups have to be considered again.

The expected results of merging two groups are estimated by learning a new model for the variables in the union of the two groups. The resulting cost of the combined model is then compared to the sum of the costs of the two independent models for the groups.

3.2 Asymmetric Agglomerative Grouping Algorithm

One potential application of IVGA and consequently of AIVGA is feature selection for classification or regression [1]. Following the procedure outlined in [1], AIVGA could be applied to this task by applying it to the full data set consisting of both the features and classification or regression target or targets, and noting which features ended up in the same group with the target or targets.

This approach has, however, several potential drawbacks. First and most importantly, it is possible that the target(s) and features are only merged to the same group at the last step, in which case no information for the solution of the feature selection problem is gained. Even if this does not happen, a lot of effort is wasted in grouping the features before merging them in a large chunk with the target(s).

In order to avoid these problems, a simple modification of the basic agglomerative grouping algorithm, the *asymmetric AIVGA* is introduced. The algorithm is otherwise the same as regular AIVGA, except only merge operations involving a group with a target variable are allowed. In the basic setting with one target variable, the features are effectively merged to the group containing the target one at a time thus producing a ranking of best features.

The computational complexity of asymmetric AIVGA is comparable to that of regular AIVGA. While the more restricted set of allowed moves reduces the initialisation step complexity to $\mathcal{O}(N)$, the theoretical complexity of the merge phase remains the same. Additionally, the fact that the change in the merge phase always involves the largest possible model means that the newly considered models are all as large as possible, hence inducing the worst-case complexity for the model fitting in the merge phase.

3.3 Models of the Groups

In [1], the groups were modelled using finite mixture models. These require significant computational effort to determine a suitable number of mixture components needed for good results. To avoid this, the individual groups in AIVGA are modelled with infinite Dirichlet process (DP) mixture models [3], that avoid having to explicitly determine the number of mixture components. As a result, the algorithm using DP mixtures is slightly faster than the approach based on finite mixtures and restarts with different numbers of components used earlier.

Using the stick-breaking representation of the Dirichlet process [8,17], the

mixing proportions π_k can be represented as

$$\pi_k(V) = v_k \prod_{j=1}^{k-1}(1 - v_j), \quad k = 1, 2, \ldots, \tag{3}$$

where $v_j \sim \text{Beta}(1, \alpha_v)$. Thus the model for the vector $\mathbf{y} = (y_1, \ldots, y_{N_i})^T$ of variables in the group $\mathcal{G}_i$ is

$$p(\mathbf{y}|V, \eta) = \sum_{k=1}^{\infty} \pi_k(V) p(\mathbf{y}|\eta_k), \tag{4}$$

where $\eta = \{\eta_k\}_{k=1}^{\infty}$ are the parameters of the mixture components.

The mixture components $p(\mathbf{y}|\eta_k)$ are products of individual distributions for every variable. These distributions are Gaussian for continuous variables and multinomial for categorical variables. For purely continuous data, the resulting model is thus a mixture of diagonal covariance Gaussians. Variances of the Gaussians can be different for different variables and for different mixture components. The component model is the same that was used in [1], except that hyperparameter adaptation is not used.

The mixture is learned using the variational Dirichlet process (VDP) mixture algorithm [12]. In order to perform inference over DP mixtures in practice, the model has to be reduced to one with a finite number of distinct parameters. In the VDP method the infinite mixture is approximated with an infinite variational approximation which is such that only finitely many mixture components K actually differ from their prior values. As the approximations are nested over K, learning can be performed efficiently by increasing K and thus releasing new components from the prior until there are no more significant reductions in the free energy. The new components are constructed by splitting the highest responsibility component randomly, not using the bisector of its principal component as in [12]. This is because the definition of principal components is more difficult for mixed continuous and categorical data. Detailed learning rules used for updating the mixture are presented in Appendix A.

4 Experiments

In this section we present the results of three experiments with AIVGA. In the first experiment, AIVGA was applied to a synthetic data set with somewhat complicated dependence structure. Then, AIVGA and asymmetric AIVGA were applied to feature selection for classification using four example data sets. Finally, an example of splitting a large association rule based predictive model into a set of smaller and lighter models is considered.

4.1 Synthetic Toy Example

In the first experiment, a set of 200 data points generated according to the graphical model in Fig. 2 was used to test the AIVGA method. All the discrete variables had 8 states and their conditional probabilities were drawn from Dirichlet$(2, \ldots, 2)$. The continuous variables were Gaussian with their conditional means drawn from N$(0, 1)$ and their conditional precisions from Gamma$(10, \frac{1}{10})$, where Gamma(α, β) has shape α and inverse scale β.

AIVGA was run 10 times on this data. Each run took around 18 seconds on a single core AMD Opteron processor. The results of the run yielding the lowest average cost over the whole tree of groupings are shown in Fig. 3. The best grouping found was $\{\{1, 2\}, \{3, 4\}, \{5, 6\}, \{7, 8\}, \{9, 10, 11, 12\}\}$, this was the same for all 10 simulations and this was also the configuration confirming with the tree structure in Fig. 2, that yielded the lowest cost. Placing some dependent variables into separate groups is understandable given the small sample and the weakness of the dependence between different groups. For instance the mutual-information estimation technique presented in [18] gave negative or essentially zero estimates for mutual information between dependent variables such as x_1 and x_8. In the results of the mutual-information based hierarchical clustering method [11] shown in Fig. 4, the groups are also more confused.

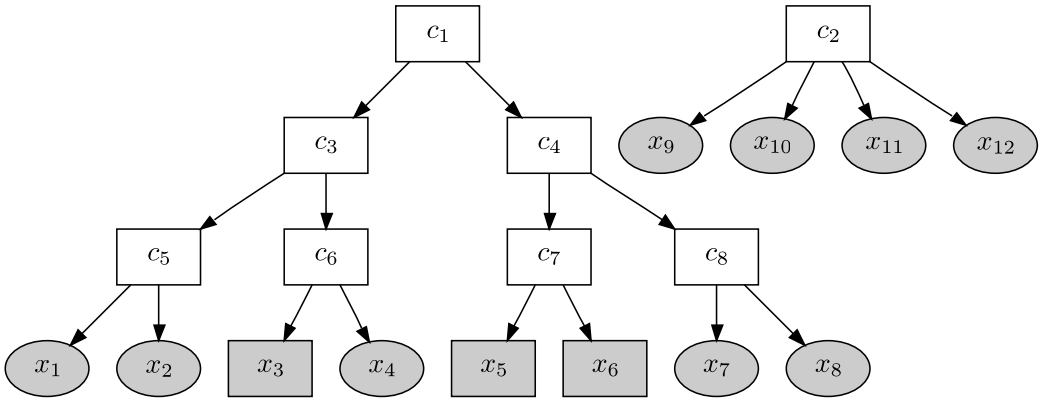

Fig. 2. A graphical model representation of the model used in the toy example. Square nodes represent discrete variables and round nodes continuous variables. The latent variables c_i are not observed, only the variables x_i (shaded nodes).

4.2 Feature Selection for Classification

In this experiment, AIVGA and asymmetric AIVGA were applied to feature selection for classification using nearest-neighbour (1NN) classifier and several data sets from the UCI repository [5]. The tested data sets were 'image', 'ionosphere', 'sonar', and 'waveform'. The properties of these data sets are

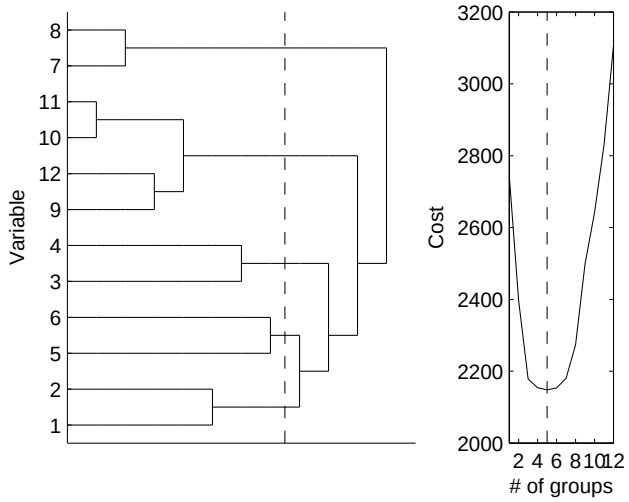

Fig. 3. Results of the toy experiment. Left: Dendrogram of the groupings with the optimal cutting point marked by dashed line. Right: Cost function values at different stages.

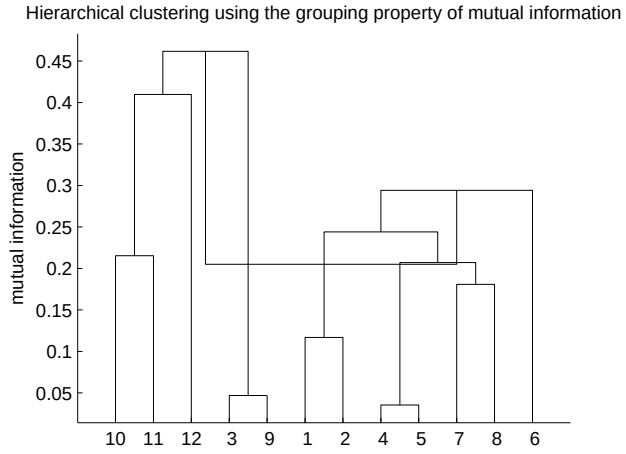

Fig. 4. Results of the toy experiment using the mutual-information based hierarchical clustering method [11].

presented in Table 1. All the features in all the data sets were continuous and they were normalised to zero mean and unit variance.

The AIVGA algorithms were used for feature selection by running them on the training part of the data set consisting of both the features and the target labels and noting which features were in the same group as the target in the grouping with the lowest cost. The features were treated as continuous variables with a Gaussian model in the mixture while the class was considered discrete with a multinomial model in the mixture. A k-nearest neighbour (k-NN) classifier with $k = 1$ was trained using the selected features of the same training set and it was tested on a separate test set. The whole procedure was repeated over 100 different splits to training and test sets.

Data set	Features	Samples	Training samples	Validation samples	Notes
image	18	2310	310	100	[1]
ionosphere	33	351	251	100	
sonar	60	208	150	50	[2]
waveform	21	5000	400	100	

Table 1
Properties of the data sets used in the experiment. The number of features is reported after removing ones that were constant in the whole data set. The number of samples is the total number of samples in the data set with the numbers used for training in all the methods and validation in sequential floating forward selection (SFFS) and Mann–Whitney (M–W) test based methods indicated separately. The validation set in these methods is split out of the training set.

[1] Obtained by concatenating the training and testing sets available in the UCI repository.

[2] Found in the UCI repository under `undocumented/connectionist-bench`.

The AIVGA algorithms were compared to using all features without any selection procedure, the Mann–Whitney (M–W) test (which is equivalent to Wilcoxon rank sum test) [14], and the sequential floating forward selection (SFFS) method [16].

There are many feature selection methods based on mutual information, starting from [4], that were not included in the comparison. The methods were omitted because the great multitude of alternatives would have made a meaningful comparison difficult, so some of the most relevant general methods were selected for the comparison instead. Original IVGA [1] was not included in the comparison because it would have required a much more sophisticated experimental setting combining the results of multiple runs as in [1] to yield good results.

In the M–W method, the features were ranked according to the p-value given by the test for the class-conditional distributions of the features being different. The optimal number of features was then found by further splitting the training set 10 times to a new training set and a separate validation set and finding which number gave the best classifier performance in this context. As the description shows, the method is very fast and easy to implement.

The SFFS method is a wrapper method that uses the employed classifier internally on a number of splits of the training set to new training and validation sets to find the optimal set of features. As the original specification in [16] specifies no end condition for the algorithm, it was run until it ended into a loop or a fixed timer expired. The timer was set to allow a slightly longer

runtime than AIVGA used.

Given this setting, both M–W and SFFS methods used the 1NN classifier internally to select the optimal set of features whereas the AIVGA methods were completely agnostic of the classifier to be employed in testing.

The average classification accuracies attained in 100 repetitions are presented in Table 2. The associated average runtimes needed by the methods are listed in Table 3. Overall none of the methods is clearly superior to others in the attained accuracy with each being at least indistinguishable from the best method for at least one data set. Asymmetric AIVGA is the most accurate on average over all data sets, but the margin over M–W is tiny. Symmetric AIVGA is clearly worse than the rest on average because of the catastrophic failure on the 'sonar' data set, even though it yields the best results on 'image' and 'waveform' data sets. The running times of the AIVGA algorithms are comparable to those of SFFS, while M–W is significantly faster. The experiment was also repeated using k-NN classifier with $k = 5$ (results not shown). The rankings of the methods for different data sets were in this case practically always the same as with $k = 1$.

	AIVGA	Asymm. AIVGA	M–W	SFFS	No selection
Image	**93.1 %**	92.7 %	92.5 %	92.6 %	90.0 %
Ionosphere	84.0 %	86.4 %	**88.3 %**	**88.5 %**	86.9 %
Sonar	68.1 %	**83.8 %**	83.1 %	79.5 %	**84.8 %**
Waveform	**76.4 %**	76.1 %	74.9 %	73.2 %	74.8 %

Table 2
Average classification accuracies achieved in the feature selection experiment. The best result on each data set along with those that do not statistically significantly deviate from it have been emphasised, as determined by the Mann–Whitney test using 0.05 significance level.

	AIVGA	Asymm. AIVGA	M–W	SFFS	No selection
Image	103.6 s	71.0 s	1.2 s	361.7 s	0.2 s
Ionosphere	878.8 s	613.9 s	1.8 s	404.5 s	0.0 s
Sonar	3265.0 s	1669.7 s	1.6 s	1706.0 s	0.0 s
Waveform	123.9 s	92.6 s	1.9 s	441.7 s	0.5 s

Table 3
Average runtimes of different methods in the feature selection experiment, including the time needed for performing the final classification.

4.3 Simplifying a Predictive Model

In this experiment, we considered a system to support and speed up user input of component data of a printed circuit board assembly robot. The system is based on a predictive model which models the redundancy of the existing data records using association rules. The application is described in detail in [2].

Our goal was to find out if a set of small models would work better than a large monolithic model – and if so, how we could determine a set of such models. We divided the 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 remaining 20 %). AIVGA was run seven times for the training data set. The results of all runs were quite similar to each other. In Fig. 5, the grouping tree (dendrogram) that contains the grouping with the lowest cost is shown.

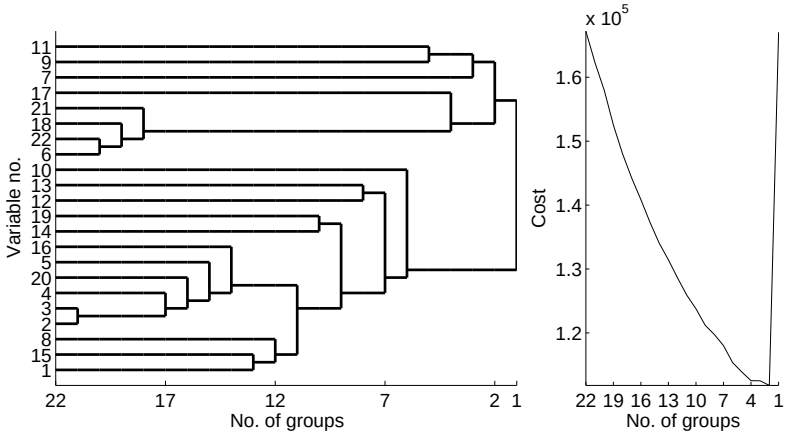

Fig. 5. A grouping (left) and cost history graph (right) for the component data. The model with the lowest cost consists of two groups.

After the groupings were found, association rules were used for modelling of the dependencies of (i) the whole data and (ii) variable groups of the 21 different variable groupings. The association rules are based on so-called frequent sets, which in our case contain the attribute value combinations that are common in the data. Support of a set is defined as the proportion of entries of the whole data in which the attribute values of the set are present.

There exists various algorithms for computation of the frequent sets which differ in computational aspects but which all give identical results. We computed the sets using a freely available implementation of the Eclat algorithm [21] [1] . For the whole data, the minimum support dictating the size of the model was set to 5 %, which was the smallest computationally feasible value in terms of

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

memory consumption. For the group models the minimum support was set to 0.1 %, which always gave clearly smaller models than the one for the whole data. Minimum confidence (the "accuracy" of the rules) was set to 90 %, which is a typical value in many applictions.

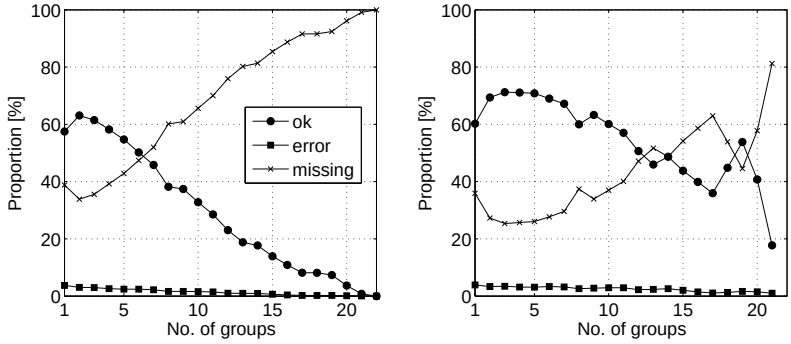

Fig. 6. Prediction results when the data have been divided into 1 ... 22 group(s). For each grouping, the results are sums of results of all the models corresponding to the grouping. The right panel indicates the prediction results when the missing predictions of the first variable of each group are ignored. In the left panel, these are taken into account. The legend shown in the left panel is the same for both illustrations.

The rules were used for one-step prediction of the attribute values of the testing data. In Fig. 6, the proportion of the correct, incorrect, and missing predictions (that is, the cases when the confidence of the best rule was below 90 %) are shown for the whole data and grouped data. For the first variable of any group, previous input does not exist, so the prediction is always missing.

Fig. 6 reveals two different aspects of the results. The left panel shows the total prediction results. The right panel shows the performance of the predictive scheme for the values for which it was even possible to try to compute a prediction. In terms of correct predictions, the total results are best using 2 groups, but the predictive scheme performs slightly better when there are 3 groups. However, the left panel indicates that if the data are grouped in 2–4 groups, the number of the correct predictions is higher than using the monolithic model. The models are also clearly lighter to compute and consume less memory. For a detailed comparison between the monolithic model and two separate models for the two optimal groups determined by AIVGA, see Table 4. For comparison, the table also includes results for non-agglomerative IVGA.

The two groups and prediction results were almost identical to the ones reported in our earlier study [1] with the same data using non-agglomerative IVGA. However, in that work we had to run IVGA 100 times and manually combine the results in order to make sure that an appropriate and useful grouping was found: There were significant variations in the final costs—as well as in the number of groups and groupings. A clear advantage of AIVGA

	1 group	2 groups (AIVGA)	2 groups (IVGA)
Correct (%)	57.5	63.1	63.8
Missing (%)	3.7	3.0	2.9
Incorrect (%)	38.8	33.9	33.3
Model size (nodes)	9 863 698	1 364 358	2 707 168
Model computation (s)	48.2	4.5	9.1
Grouping time (1 run / total) (h)	0 / 0	3.06 / 21.4	0.521 / 52.1

Table 4
Comparison between one large monolithic model, two separate models for the two groups which were discovered by non-agglomerative IVGA (the results are adopted from [1]), and AIVGA. The quantities for the 2 groups are sums over the quantities of the two models. Also note that the variables were grouped 7 times using AIVGA and 100 times using IVGA.

over non-agglomerative IVGA was that it both gave more stable results and provided a systematic way for determination of the two groups by simply using the configuration yielding the lowest cost.

5 Discussion

To complement the variable grouping or clustering method of IVGA [1], we have presented a corresponding agglomerative algorithm AIVGA for learning a hierarchical grouping of variables. Like with any hierarchical clustering method, this approach has two potential major benefits depending on the application: the returned hierarchical grouping can be used as such, or the method can be seen as a systematic search algorithm for finding a single grouping.

This systematic search can in some cases be very useful as there is often a significant variation in the results of regular IVGA caused by strong local optima. This is especially clear when contrasting the results of the experiments in Sec. 4.2 and Sec. 4.3 with those reported in [1]: AIVGA yields good results already when using the minimal cost configuration from a single run whereas with IVGA one needs to somehow combine the results of multiple independent runs, and furthermore the Bayesian averaging using the posterior probabilities evaluated from the VB costs as $p_i \propto \exp(-\mathcal{C}_i)$ is not sufficient getting good results.

The AIVGA algorithm is well-suited for parallel implementation. Parallelising

testing of different candidates for next merge is trivial, and with this optimisation the time complexity in terms of mixture model fittings can be reduced to $\mathcal{O}(N)$ using $\mathcal{O}(N)$ processors.

In general, the AIVGA method seems best suited for problems with at most a moderate number of variables, typically less than a hundred. In larger problems, the computation time grows rapidly while the choices made in the early stages of the greedy algorithm may focus the effort to a highly suboptimal part of the grouping space. In this case even the best grouping in the tree may yield significantly higher cost than the grouping found e.g. by regular IVGA in equal time.

In addition to the agglomerative grouping method, the presented algorithm differs from the regular IVGA [1] in that DP mixture models are used instead of finite mixtures. In our earlier work [10] we used finite mixtures with the agglomerative grouping algorithm. Based on these experiences, it seems that the difference between these different mixture modelling approaches is mainly in convenience of determining the suitable number of mixture components, which may in some cases translate to a modest speedup in computation time when using DP mixtures. Otherwise the grouping results are usually very similar with either finite mixtures or DP mixtures.

The number of samples is not that critical for the performance of AIVGA, especially if the accelerated VDP mixture algorithm [12] is applied. Still, AIVGA will actually most likely perform better relative to other mutual-information based clustering methods on small samples. Taking the example of Sec. 4.1 as an example, the results of AIVGA would not change much even if the number of samples were increased to 2000 while the methods based on classical mutual-information estimation [18,11] perform a lot better. Clearly the classical methods based on binning the data require quite a lot of samples to get a reliable estimate of the density while the model-based AIVGA works well even with relatively small data sets.

While AIVGA can be used for feature selection with reasonably good results as outlined in Sec. 4.2, the results and the exact procedure are only meant for illustration. For real applications it is most likely better to combine the pure filter approach used here with a wrapper like was done in [6] in order to take into account the specific characteristics of the applied classifier or regression model.

When applying regular AIVGA to problems like feature selection, full modelling of the group densities can be overly expensive especially when there are a lot of samples. The potentially poor performance of the symmetric AIVGA in problems with many variables is, however, nicely offset by the asymmetric algorithm focusing on the problem of interest, as illustrated by the results on

the 'sonar' data set in the experiment.

6 Conclusion

In this paper, we presented AIVGA, an agglomerative algorithm for learning hierarchical groupings of variables according to the IVGA principle. AIVGA helps the use of IVGA when further use of the results makes the optimal number of groups difficult to determine. The computation time of a single AIVGA run is usually comparable to a single run of the regular IVGA algorithm [1], so the additional information in the tree of groupings comes essentially for free. If the single best grouping is sought for, the output of AIVGA can be used to initialise the regular IVGA algorithm [1], which can then relatively quickly find if there are better solutions that are nearly but not exactly tree conforming.

In order to help others try the method out, a free MATLAB package implementing the presented AIVGA algorithms is available at `http://www.cis.hut.fi/projects/ivga/`.

The cost function of IVGA and the probabilistic modelling framework can in some cases provide more meaningful clustering results than existing algorithms, as illustrated, for instance, by the clustering of gene expression profiles in [15]. AIVGA effectively complements this by providing a structured solution mechanism as well as a richer set of potential solutions.

Acknowledgements

We thank Zoubin Ghahramani for interesting discussions and Valor Computerized Systems (Finland) Oy for providing us with the data used in the printed circuit board experiment. This work was supported in part by the IST Programme of the European Community under the PASCAL Network of Excellence, IST-2002-506778. This publication only reflects the authors' views.

A Learning of the Mixture Models

The learning of the mixture models mostly follows the algorithm VDP presented in [12]. The only differences arise from the specific form of the mixture components $p(x|\eta_i)$ employed in our model, which are handled mostly as in [1].

The correspondence of relevant parts of the notation between these two papers is presented in Table A.1.

Notation in [12]	**Notation in [1]**	**Explanation**
$\{\pi_i\}_{i=1}^{\infty}$	$\boldsymbol{\pi}_c$	Mixing proportions
$q_{z_t}(z_t = i)$	$w_i(t)$	Posterior responsibilities
η_i	$\{\mu_{i,j}, \rho_{i,j}, \boldsymbol{\pi}_{i,j}\}$	Mixture component parameters
λ	$\{\mu_{\mu_j}, \rho_{\mu_j}, \alpha_{\rho_j}, \beta_{\rho_j}, \mathbf{u}_j\}$	Mixture component hyperparameters
ϕ_i^{η}	$\{\hat{\mu}_{\mu_{i,j}}, \hat{\rho}_{\mu_{i,j}}, \hat{\alpha}_{\rho_{i,j}}, \hat{\beta}_{\rho_{i,j}}, \hat{\mathbf{u}}_{i,j}\}$	Mixture component posterior parameters

Table A.1
Correspondence of notation between [12] and [1].

Expressions for the free energy terms $\langle \log p_x(\mathbf{x}(t)|\eta_i) \rangle$ and $\left\langle \log \frac{q(\eta_i)}{p(\eta_i|\lambda)} \right\rangle$ are identical to the ones given in Appendix A of [1] (Eqs. (27)–(30) in the paper).

Contrary to [1], the mixture component hyperparameters λ were fixed and not updated. Their values were $\mu_{\mu_j} = 0; \rho_{\mu_j} = 1; \alpha_{\rho_j} = 0.01; \beta_{\rho_j} = 0.01; \mathbf{u}_j = (0.5, \ldots, 0.5)$. The meaningfulness of the hyperparameters for continuous dimensions was ensured by always pre-normalising the data to zero mean and unit variance. The parameter α_v in the DP prior was fixed to $\alpha_v = 1$.

The only difference in the update rules to [1] is for the update of the continuous dimensions of the mixture components (Eqs. (34)–(37) in [1]), which have been made independent of previous values of the parameters by repeating the update for the mean first using the prior of the variance and then the newly update posterior. The new updates are thus

$$\hat{\alpha}_{\rho_{i,j}} \leftarrow \alpha_{\rho_j}, \quad \hat{\beta}_{\rho_{i,j}} \leftarrow \beta_{\rho_j} \tag{A.1}$$

$$\hat{\rho}_{\mu_{i,j}} \leftarrow \rho_{\mu_j} + \frac{\hat{\alpha}_{\rho_{i,j}}}{\hat{\beta}_{\rho_{i,j}}} \sum_{t=1}^{T} w_i(t) \tag{A.2}$$

$$\hat{\mu}_{\mu_{i,j}} \leftarrow \hat{\rho}_{\mu_{i,j}}^{-1} \left(\rho_{\mu_j} \mu_{\mu_j} + \frac{\hat{\alpha}_{\rho_{i,j}}}{\hat{\beta}_{\rho_{i,j}}} \sum_{t=1}^{T} w_i(t) x_j(t) \right) \tag{A.3}$$

$$\hat{\alpha}_{\rho_{i,j}} \leftarrow \alpha_{\rho_j} + \frac{1}{2} \sum_{t=1}^{T} w_i(t) \tag{A.4}$$

$$\hat{\beta}_{\rho_{i,j}} \leftarrow \beta_{\rho_j} + \frac{1}{2} \sum_{t=1}^{T} w_i(t) \left[\hat{\rho}_{\mu_{i,j}}^{-1} + (\hat{\mu}_{\mu_{i,j}} - x_j(t))^2 \right] \tag{A.5}$$

Repeat update (A.2) (A.6)

Repeat update (A.3). (A.7)

All the other updates except for hyperparameter adaptation are performed as

presented in [1].

References

[1] E. Alhoniemi, A. Honkela, K. Lagus, J. Seppä, P. Wagner, H. Valpola, Compact modeling of data using independent variable group analysis, IEEE Transactions on Neural Networks 18 (6) (2007) 1762–1776.

[2] E. Alhoniemi, T. Knuutila, M. Johnsson, J. Röyhkiö, O. S. Nevalainen, Data mining in maintenance of electronic component libraries, in: Proc. IEEE 4th Int. Conf. on Intelligent Systems Design and Applications, vol. 1, 2004.

[3] C. E. Antoniak, Mixtures of Dirichlet processes with applications to Bayesian nonparametric problems, The Annals of Statistics 2 (6) (1974) 1152–1174.

[4] R. Battiti, Using mutual information for selecting features in supervised neural net learning, IEEE Transactions on Neural Networks 5 (4) (1994) 537–550.

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

[6] G. V. Dijck, M. M. V. Hulle, Speeding up the wrapper feature subset selection in regression by mutual information relevance and redundancy analysis, in: Proceedings of the International Conference on Artificial Neural Networks (ICANN 2006), vol. 4131 of LNCS, Springer, Berlin, 2006, pp. 31–40.

[7] R. O. Duda, P. E. Hart, D. G. Stork, Pattern Classification, 2nd ed., Wiley, 2001.

[8] T. S. Ferguson, A Bayesian analysis of some nonparametric problems, The Annals of Statistics 1 (2) (1973) 209–230.

[9] K. A. Heller, Z. Ghahramani, Bayesian hierarchical clustering, in: Proc. 22nd Int. Conf. on Machine Learning (ICML 2005), 2005.

[10] A. Honkela, J. Seppä, E. Alhoniemi, Agglomerative independent variable group analysis, in: Proc. 15th European Symposium on Artificial Neural Networks (ESANN 2007), Bruges, Belgium, 2007.

[11] A. Kraskov, H. Stögbauer, R. G. Andrzejak, P. Grassberger, Hierarchical clustering using mutual information, Europhysics Letters 70 (2) (2005) 278–284.

[12] K. Kurihara, M. Welling, N. Vlassis, Accelerated variational Dirichlet process mixtures, in: B. Schölkopf, J. Platt, T. Hoffman (eds.), Advances in Neural Information Processing Systems 19, MIT Press, Cambridge, MA, 2007, pp. 761–768.

[13] K. Lagus, E. Alhoniemi, H. Valpola, Independent variable group analysis, in: Proc. Int. Conf. on Artificial Neural Networks - ICANN 2001, vol. 2130 of LNCS, Springer, Vienna, Austria, 2001.

[14] H. B. Mann, D. R. Whitney, On a test of whether one of two random variables is stochastically larger than the other, The Annals of Mathematical Statistics 18 (1) (1947) 50–60.

[15] J. Nikkilä, A. Honkela, S. Kaski, Exploring the independence of gene regulatory modules, in: J. Rousu, S. Kaski, E. Ukkonen (eds.), Proc. Workshop on Probabilistic Modeling and Machine Learning in Structural and Systems Biology, Tuusula, Finland, 2006.

[16] P. Pudil, J. Novovičová, J. Kittler, Floating search methods in feature selection, Pattern Recognition Letters 15 (11) (1994) 1119–1125.

[17] J. Sethuram, A constructive definition of Dirichlet priors, Statistica Sinica 4 (1994) 639–650.

[18] N. Slonim, G. S. Atwal, G. Tkačik, W. Bialek, Estimating mutual information and multi–information in large networks, `http://arxiv.org/abs/cs/0502017` (2005).

[19] N. Slonim, G. S. Atwal, G. Tkačik, W. Bialek, Information-based clustering, Proceedings of the National Academy of Sciences 102 (51) (2005) 18297–18302.

[20] M. Studený, J. Vejnarová, The multiinformation function as a tool for measuring stochastic dependence, in: M. Jordan (ed.), Learning in Graphical Models, The MIT Press, Cambridge, MA, USA, 1999, pp. 261–297.

[21] M. J. Zaki, S. Parthasarathy, M. Ogihara, W. Li, New algorithms for fast discovery of association rules, in: Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining (KDD), 1997.

[22] N. L. Zhang, Hierarchical latent class models for cluster analysis, Journal of Machine Learning Research 5 (2004) 697–723.