

ML4UM for Bayesian Network User Model Acquisition

Frank Wittig

Department of Computer Science, Saarland University,
P. O. Box 15 11 50, D-66041 Saarbrücken, Germany
wittig@cs.uni-sb.de

Abstract. This paper addresses the primary workshop question on “how to apply machine learning techniques to acquire and continuously adapt user models” for the particular representation of user models as Bayesian networks. On the basis of an integrative framework for learning Bayesian networks for user modeling and user-adaptive systems, respectively, we discuss some of the methods we developed along these lines, in the light of the questions that are to be discussed during the workshop. As this paper is intended to give an overview of our research, it omits some technical details, that can be found in related publications.

1 General Issues in Machine Learning for User Modeling

We begin by quoting this workshop’s “Call for Papers”:

“Machine learning is concerned with the formation of models from observations. Hence, learning algorithms *seem* to be promising candidates for user model acquisition systems.”

As this statement indicates in subtle manner, a successful application of machine learning (ML) techniques in the user modeling (UM¹) context (ML4UM) is indeed a non-trivial task that usually includes much more than a straightforward application of an ML algorithm together with collected user data. The following list summarizes some typical issues that are critical regarding a successful application of ML techniques for the acquisition and adaptation processes of UMs (see [1] for a detailed discussion of a subset of this list):

- *Limited training data*
- *Individual differences between users*
- *(Temporally) changing aspects of the domains*
- *Complexity / efficiency of the learning algorithms*
- *Interpretability of the learned UMs*
- *Characteristics of typical UM data, i.e., implicit feedback, missing data*
- *Exploitation of a priori available knowledge*

¹ We will use the abbreviation ‘UM’ for both terms ‘user modeling’ and ‘user model’. It will become clear from the context which one is meant.

One of the main workshop questions is that on “how the user fits into the picture” and “how he can query/use the UM”. A prerequisite to enable the user to query/use the UM in a reasonable manner is that the learned UM has to be understandable for him/her (or at least, that it can be transformed into an understandable representation). Such interpretable UMs enable or make it easier to (let the system) generate justifications and/or explanations of decisions. It is well-known that the provision of such information usually leads to a better acceptance of the system by its users.

Thus, the issue of ensuring the interpretability of results of the ML process is a key issue regarding explanation components or queries to the UM by users. For a wide range of formalisms for representing UMs such as rule-based representations, decision trees, fuzzy methods etc., this can be achieved quite straightforwardly, but there also exist methods that do not yield interpretable UMs in their basic form in this sense, e.g., artificial neural nets, k-nearest-neighbors methods. Bayesian networks (BNs) lay somewhere between those extremes of the spectrum of (not) interpretable UM representations, as we will discuss in more detail later.

In addition to proposing potential solutions to the primary question of how to continuously adapt a learned BN UM, in this paper we focus on results we achieved in our research efforts to ensure the interpretability of the learned BNs. Although we do not explicitly address IR issues, many of the results are relevant for IR because BNs become also applied in (user-adaptive) IR systems (e.g., [2]).

2 Learning Bayesian Networks for User Modeling

Bayesian networks have become increasingly popular as one of the inference technique of choice for user-adaptive systems. Table 1 lists some recent research of UM with BNs in a wide range of application scenarios that applies to some extent ML techniques. Note, that these systems (except our READY-system) use off-the-shelf learning methods that were not developed with the particular UM context in mind. The main goal of our research is to adapt existing BN learning algorithms for an application in user-adaptive systems and/or to develop new ones that are especially well suited for this particular context with those critical issue listed in the previous section.

System	Domain	Batch-Learning	Adaptation
Albrecht et al. [3]	MUD Games	CPTs	–
Billsus and Pazzani [2]	personalized news	CPTs	CPTs
Lau and Horvitz [4]	WWW search	CPTs	–
Horvitz et al. [5]	Office-alarms	CPTs & structure	–
Nicholson et al. [6]	ITS	CPTs & structure	–
<i>Bohnenberger et al. [7]: READY Dialog</i>		<i>CPTs & structure</i>	<i>CPTs & structure</i>

Table 1. User-adaptive systems that use ML for BNs

A BN consists of two components: The first component is a directed acyclic graph (DAG)—the *structure* of the BN—that represents the causal independencies that hold

in the domain to be modeled. Nodes represent random variables and directed links between nodes are commonly interpreted as causal influences between these variables. The second component of a BN is a vector of *conditional probability tables (CPTs)* that quantify the (uncertain) relationships between nodes and their parents. A node's CPT consists of conditional probabilities for each state of the node conditioned on its parents' state configuration. A BN represents a joint probability distribution over the states of its variables. On the basis of this representation, inference algorithms can be applied to derive conclusions about arbitrary sets of variables conditioned on available evidences related to other variables observed in the domain.

There are several properties of BNs that make them well suited for an application in the user modeling context: First, it is common practice to interpret the networks' links in a causal manner, a fact that contributes to both a potentially simplified construction process and a more interpretable user model from the user's point of view. Second, BNs are able to handle uncertainty in the domain under consideration with regard to arbitrary subset of variables, e.g., a user's goals, interests, etc. There exist several formalisms that are strongly related to BNs and have been successfully applied in a variety of UM scenarios, e.g., influence diagrams, object-oriented BNs and dynamic BNs.

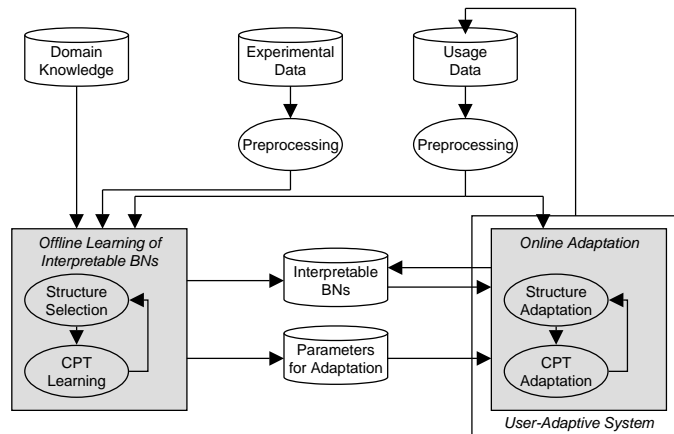


Fig. 1. Learning Bayesian networks for user modeling. (Cylinders represent repositories of data and BNs, respectively, ellipses stand for algorithmic procedures and boxes symbolize larger conceptual entities. The flow of information between these components is depicted by directed arcs.)

As a basis for our research on learning BNs for UM we used the integrated approach as shown in Figure 1. This generic framework is flexible with regard to several “dimensions” that will be discussed below. Typically, a particular user-adaptive system does not make use of all aspects of this integrated approach, e.g., many systems omit the structural learning/adaptation part.

Offline learning and online adaptation (see the grey boxes in Figure 1) During the offline phase, general UMs are learned on the basis of data from other previous system

users or data acquired by user studies. These models are in turn used as a starting point for the interaction with a particular new user. The initial general model is then adapted to the individual current user and can be saved after the interaction for future use when this particular user will interact the next time with the system. In such a situation, this individual model can be retrieved from the model base and thus, there is no further need to start with the general model, yielding probably a better adaptation right from the beginning. Note, that the offline learning procedure may also yield parametric information on how to adapt to individual users. The general idea behind this approach is that different parts of the learned UM need different ways of adaptation, i.e., some parts need faster adaptation to an individual user than others. Details on this and a comparison of alternative methods of adaptation to individual users will be discussed in a following section.

Experimental data and usage data Two further dimensions concern the kind or type of data that is available. In principle, we distinguish between (a) experimental data and (b) usage data (see upper part of Figure 1). Experimental data is collected in controlled environments just as done in psychological experiments. Usage data is collected during the real interaction between users and the system. Obviously, these two types differ characteristically: Usage data often includes more missing data and rare situations are underrepresented in such data sets, while experimental data mostly does not represent the “reality”. Often, a combination of both types occur. Because of our offline/online approach we can handle this problem for example by learning a general model on the basis of experimental data and then adapting it using usage data of the individual user.

Learning the BNs’ conditional probabilities and structures Since BNs consist of two components, the learning and adaptation tasks are also two-dimensional: (a) learning the conditional probabilities (CPTs) and (b) learning the BNs’ structures. For both partial tasks exist a number of standard algorithms (see [8] for an overview). In the UM context, we often have to deal with sparse data but on the other side in most cases we have additional domain knowledge available. This is reflected in our approach by introducing such knowledge into the learning procedures to improve the results, especially when the data is indeed sparse (see upper left part of Figure 1). For the learning of the conditional probabilities we developed a new method that we will briefly describe in the following section. When learning structures, background knowledge can be incorporated by specifying “starting” structures manually that reflect the basic assumptions that one makes in the domain. This bipartite character of the learning task is reflected in our new techniques we developed for the adaptation of initially learned BN UMs.

Degree of interpretability As already discussed, an important point made by many researchers in the UM community is the interpretability and transparency of the models. The issue of interpretability is therefore also an integral part of all aspects of our approach. We try to ensure or at least improve the interpretability of the final learned models, e.g. by respecting whatever background information that is a priori available and that can be introduced into and exploited within the learning process.

3 Improving the Degree of Interpretability

An important issue related to the interpretability of BNs are so-called *hidden variables* for the learning task. Variables are called hidden, if there are no values available for them in the training data—e.g. due to missing sensors or because their values cannot be observed directly at all. Consider for example a user-adaptive system (like our READY prototype) that aims to adapt to the level of cognitive load its user is currently suffering from, e.g., by interacting with him or her in a less error-prone way (for example by providing very short and clear instructions) in order to make it less likely that the user will make a mistake using the system. Such an entity like “cognitive load”—or more general “stress”—cannot be measured directly. The only chance is to draw inferences based on observable values for symptoms like heart-rate, pupil-dilatation and so on. For such hidden variables, sophisticated learning methods have to be applied. In the case of BNs, the well-known EM procedure (see, e.g., [8]) as well as the gradient-based APN method [9] are commonly used to learn the CPTs of hidden variables.

Figure 2 shows an example BN UM that includes two hidden variables “Actual Cognitive Load” and “Relative Speed of Speech Generation”. We used this particular BN to model the consequences of time pressure in addition to the need to perform an additional task on the speech of experimental subjects of an empirical study that we performed (see [10] for a detailed description). The underlying assumption that we used to construct the BN’s structure is that a person is able to reduce its actual cognitive load by slowing down its speed of speech generation. A wide range of speech symptoms has been observed that can be used to make inferences about the values of the two hidden variables. As this short

description indicates, such a structure is interpretable in that it represents the underlying (theoretical) assumption. Beside an interpretation of the structure itself by the experienced user, an explanation component for BNs (see, e.g., [11]) can generate verbal statements to justify its decisions, like “An increase in time pressure leads with high probability to an increase in the actual cognitive load which in turn leads to a smaller number of syllables produced by the experimental subjects.”. A structure without these hidden variables may yield something less elaborate like “A high time pressure yields a small number of syllables.”, that may leave the user asking why this should be this way.

The problem of learning BNs with hidden variables is that the learning procedures typically yield resulting CPTs related to these variables that do not reflect the intended relationships. For example, an application of the EM algorithm with the structure of Figure 2 learned a CPT for “Actual Cognitive Load” that reflected that an increase of time pressure would result in a decrease of actual cognitive load—a relationship between

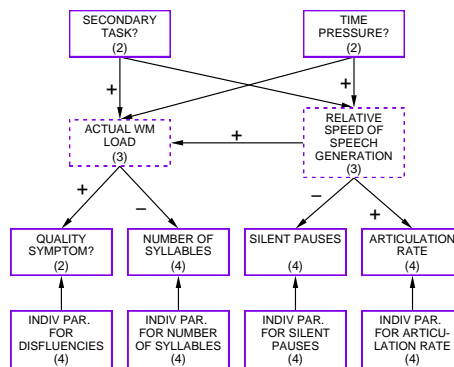


Fig. 2. Example BN UM with hidden variables. (The numbers in parenthesis represent the number of possible discrete values)

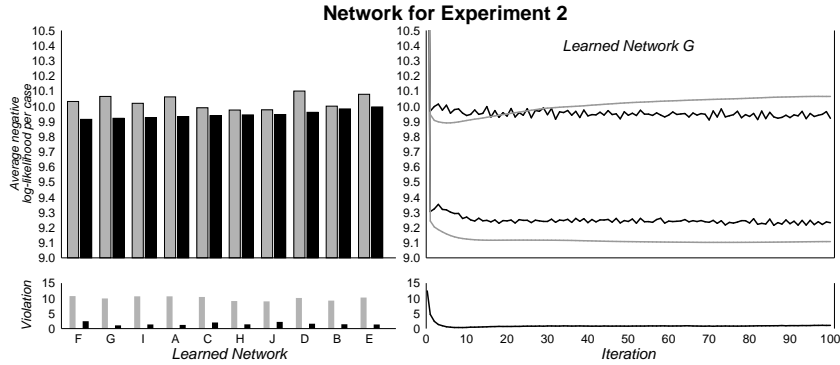


Fig. 3. Improving the BNs’ Interpretability. (Explanations in text)

these two variables that is not conform to common knowledge. Since a similar observation is made with regard to the children of the hidden variable, the overall behavior of the BN (e.g., influences of the time pressure variable on the speech symptoms) remains as expected. Such effects are largely due to the high dimensionality of the search space. Typically, the learning algorithm becomes stuck in one of the large number of local optima that numerically model the underlying joint probability distribution reasonably well, but fail to represent the intended semantics regarding these hidden variables.

In order to address this problem, we developed a method to introduce available *qualitative* background knowledge into the learning process that we called “learning with qualitative constraints”. A detailed description of the method is presented by [12]. The basic idea is the extension of the standard scoring function for learning the CPTs θ of a BN—the (log-)likelihood of the data $(\log)P(\mathbf{D}|\theta)$ —with a “penalty term” $violation(\theta, \mathbf{C})$ that measures the amount of violated *qualitative constraints* and thus the degree of the BN’s interpretability. These qualitative constraints \mathbf{C} are specified before learning takes place and represent known qualitative (monotonic) relations between two variables, i.e., statements such as “An increase of time pressure leads to a higher actual cognitive load” (cf. the ‘+’ and ‘-’ signs Figure 2). In this way, (intermediate) solutions that “violate” the qualitative knowledge—and are therefore less interpretable—are scored worse than the more interpretable ones. Thus, the learning with qualitative constraints contributes to finding *interpretable* local optima. Such learning results are significantly better suited for an application within explanation components besides the side-effect that it becomes easier to locate potential shortcomings in the UM.

Figure 3 shows the results of a 32-fold cross-validation of this method with empirical data of 32 subjects of our experimental study. Note, that the results presented in [12] were based on synthetic data. The new analysis presented in this paper shows that the method is able to cope with real-world data and the results from earlier artificial analyses also hold for this more challenging case. On the left-hand side, we present the ten different results for ten different randomly initialized starting BNs for the learning task. The upper part shows the average negative log-likelihood values of the final learned BNs. We observe an improvement in the quality of the learning results

by the introduction of qualitative constraints (black bars vs. gray ones).² That shows, that the introduction of available prior knowledge in order to avoid some of the “uninterpretable” local optima helps to learn better models. More interesting regarding the focus of the workshop are the results in the lower part that show the values of the penalty term that measures the degree of interpretability. We see, that our method indeed yields (more) interpretable BNs with penalty values near 0. The right-hand part of Figure 3 contains the time course of learning of one particular of the ten learning tasks. The two upper curves represent the results when scoring against the test data (within the cross-validation methodology) whereas the lower ones are those when scoring against the data used for learning. We see that our method limits overfitting, i.e., it ensures that while learning continues, the quality of the intermediate results does not degrade. The lower graph shows that the penalties are eliminated early in the learning procedure.

Overall, our method yields two contributions: (a) *a reduction of overfitting* and (b) *an improvement of the degree of interpretability* of the learned BNs—which has been the main motivation for the development of the learning with qualitative constraints.

4 Alternative CPT Adaptation Methods

In this section, we discuss the workshop’s primary question of how to learn and continuously adapt UMs for the particular representation as a BN. We present and discuss a comparative study of alternative methods. A more elaborate discussion of the algorithms and their evaluation is presented by [13]. Here, we focus on those aspects that are relevant to the present workshop.

When learning UMs, two common alternatives exist: (a) learning *general UMs* on the basis of observations acquired from a sample of users and (b) learning *individual UMs* on the basis of data from a particular user. Each of this approaches has its own typical benefits. A natural strategy is to combine the advantages of general and individual UMs: Learn a general UM which can be applied to a new user; adapt the model to each user during the interaction. We discuss two such adaptive approaches: (i) the *parameterized UM* that makes use of *individual parameter variables* to model individual differences between users within a general UM (see the bottom line variables of Figure 2) and (ii) the *differential adaptive BN UM*, that we introduced in [13]. Essentially, the parameterized UM is a dynamic BN in which the individual parameters represent variables whose “real” value do not change over time, e.g., in the example BN of Figure 2 the user’s average articulation rate. The differential adaptive UM essentially learns a general UM together with adaptation rates for the different parts of the model based on the variances of the characteristics of the different users. These local adaptation rates are used within the standard Bayesian adaptation procedure for conditional probabilities [8] to revise the different parts of the UM with different speeds in the light of new interaction data from a new user. The underlying idea is that those parts where large differences between individual users exist become adapted faster to the individual user than those on which most users agree.

Figure 4 shows prototypical results of a comparison of the four alternatives using our experimental data for the prediction of a particular variable of the learned BN UM

² Note, that a lower value represents a better result.

when individual differences between users are indeed present. The evaluation consisted of a 32-fold cross-validation using a BN structure similar to Figure 2 but without hidden variables, i.e., the independent variables were connected directly to the speech symptom variables. After the value has been predicted, these observations are used as evidences to adapt the UM according to the proposed method. The graph presents the averaged results.

The results for the general model (i.e. the offline learning of a BN UM without any online adaptation to the individual user) are shown by the solid thick curve. First, note that the only reason why this curve is not a straight horizontal line is that there is considerable random fluctuation in the quadratic loss variable. Looking at the general and the individual UM (starting from scratch, using only the adaptation mechanism to incrementally acquire the UM), we see that there are important individual differences between the users: The individual UM catches up to the general UM and significantly outperforms it after it has processed enough interaction data. Moreover, the differential adaptive model outperforms the parameterized model as well as the others significantly.

Our experience with a number of similar analyses with different datasets in different domains can be summarized as follows: Although the parameterized and differential adaptive models perform best overall, the general and individual models each show competitive performance under certain conditions. Consequently, one of these models may be turn out to be best if these conditions are met and the practical considerations speak for the model in question, e.g., no need for an online adaptation procedure or an offline learning phase.

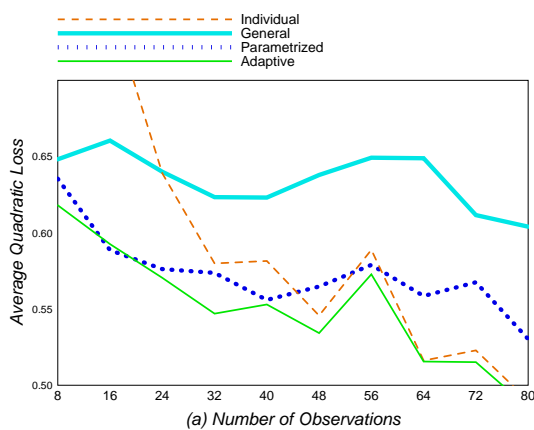


Fig. 4. Comparison of adaptation methods

5 Structural Learning and Adaptation

The preceding sections have dealt with the learning and/or adaptation of the BNs' CPTs. In the following, we will discuss the structural learning part.

Structural learning algorithms are seldomly applied in the UM context. There are several reasons for this observation: (a) the structure of BNs can often be specified adequately by experts on the basis of a causal interpretation of the links, (b) the complexity of structure learning algorithms make an application in many scenarios difficult, and—a more practical reason—(c) there exist few BN software packages that currently include such methods (although this situation seems to be changing now). Thus, the question arises, whether it is worth to apply structural learning of BNs in UM.

Since the structure encodes the causal relationships between the variables, it represents an important contribution to the BN's overall interpretability. To exploit available

causal background knowledge for the learning task, it is possible to specify (a) a starting structure for the search procedure that is conform with the prior knowledge and is expected to be modified only regarding minor parts, and (b) so-called *structural constraints* that limit the search space. Such structural constraints may be the presence or absence of a particular link, or the fact that a variable needs to remain parentless during learning. The latter applies for example to independent variables (of a user study).

We performed an initial study using our empirical dataset to compare learning with against without the structural part (see Figure 5). Again, a 32-fold cross-validation was conducted using the SEM method of [14] for structural learning. The EM algorithm has been used to learn the CPTs for the fixed structure of Figure 2 that has also been chosen to be the starting point for the structure search with SEM. The results show that in our scenario it is indeed worth to apply structural learning to increase the learned BNs' quality.

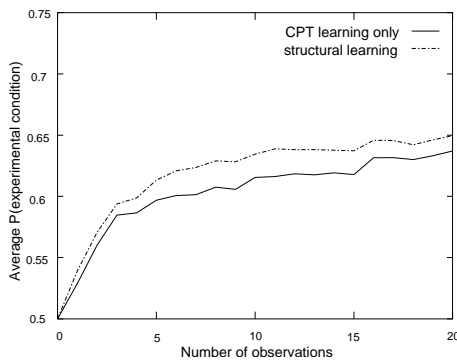


Fig. 6. Recognition results

the analogous results for the CPT learning case with a fixed BN structure. Figure 6 shows that the recognition accuracy also slightly improves with the modeling quality (represented by the likelihood measure) of the structurally learned time slice.

6 Conclusion With Regard to Workshop Questions

We addressed the primary workshop question for a particular UM representation by presenting an overview of an integrative conceptualization for learning BNs for user-adaptive systems. Some parts of this framework have been discussed in more detail by reporting results that we achieved in our research efforts along these lines. The overall goal of this research is the adaptation of existing methods and/or the development of

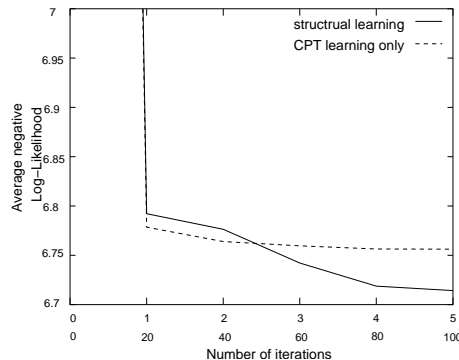


Fig. 5. Structural vs. CPT learning

By a second analysis, we evaluated whether this better quality with regard to representing the empirical data does also hold when the structurally learned BNs were used in one of our application scenarios: The recognition of time pressure and the presence/absence of a secondary task on the basis of observed speech symptoms. Therefore, the learned BNs were used as time slices of dynamic BNs to infer the values of these two variables on the basis of several utterances. We followed the same evaluation procedure as described by [10] where we presented

new ones that are specialized to the particular issues that play an important role when applying ML techniques in the UM context.

Within this framework, we emphasized the need for learning interpretable UMs, which is in our opinion strongly related to one of the questions listed in the workshop's CFP as a prerequisite to be able to provide a functionality for inspecting/querying the UM.

References

1. Webb, G., Pazzani, M.J., Billsus, D.: Machine learning for user modeling. *User Modeling and User-Adapted Interaction* **11** (2001) 19–29
2. Billsus, D., Pazzani, M.J.: A hybrid user model for news story classification. In Kay, J., ed.: *UM99, User Modeling: Proceedings of the Seventh International Conference*. Springer, Wien (1999) 99–108
3. Albrecht, D.W., Zukerman, I., Nicholson, A.E.: Bayesian Models for Keyhole Plan Recognition in an Adventure Game. *User Modeling and User-Adapted Interaction* **8** (1998) 5–47
4. Lau, T., Horvitz, E.: Patterns of search: Analyzing and modeling Web query refinement. In Kay, J., ed.: *User Modeling: Proceedings of the Seventh International Conference, UM99*, Vienna, New York, Springer Wien New York (1999) 119–128
5. Horvitz, E., Jacobs, A., Hovel, D.: Attention-sensitive alerting. In Laskey, K.B., Prade, H., eds.: *Uncertainty in Artificial Intelligence: Proceedings of the Fifteenth Conference*. Morgan Kaufmann, San Francisco (1999) 305–313
6. Nicholson, A., Boneh, T., Wilkin, T., Stacey, K., Sonenberg, L., Steinle, V.: A case study in knowledge discovery and elicitation in an intelligent tutoring application. In Breese, J., Koller, D., eds.: *Uncertainty in Artificial Intelligence: Proceedings of the Seventeenth Conference*, San Francisco, Morgan Kaufmann (2001) 386–394
7. Bohnenberger, T., Brandherm, B., Großmann-Hutter, B., Heckmann, D., Wittig, F.: Empirically grounded decision-theoretic adaptation to situation-dependent resource limitations. *Künstliche Intelligenz* **16** (2002) 10–16
8. Heckerman, D.: A tutorial on learning with Bayesian networks. In Jordan, M.I., ed.: *Learning in graphical models*. MIT Press, Cambridge, MA (1998)
9. Binder, J., Koller, D., Russell, S., Kanazawa, K.: Adaptive probabilistic networks with hidden variables. *Machine Learning* **29** (1997) 213–244
10. Müller, C., Großmann-Hutter, B., Jameson, A., Rummer, R., Wittig, F.: Recognizing time pressure and cognitive load on the basis of speech: An experimental study. In Vassileva, J., Gmytrasiewicz, P., Bauer, M., eds.: *UM2001, User Modeling: Proceedings of the Eighth International Conference*, Berlin, Springer (2001)
11. Druzdzel, M.J.: Qualitative verbal explanations in Bayesian belief networks. *Artificial Intelligence and Simulation of Behaviour Quarterly* **94** (1996) 43–54
12. Wittig, F., Jameson, A.: Exploiting qualitative background knowledge in Bayesian network learning algorithms. In Boutilier, C., Goldszmidt, M., eds.: *Uncertainty in Artificial Intelligence: Proceedings of the Sixteenth Conference*, San Francisco, Morgan Kaufmann (2000) 644–652
13. Jameson, A., Wittig, F.: Leveraging data about users in general in the learning of individual user models. In Nebel, B., ed.: *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence*, San Francisco, CA, Morgan Kaufmann (2001) 1185–1192
14. Friedman, N.: Learning belief networks in the presence of missing values and hidden variables. In: *Proceedings of the Fourteenth International Conference on Machine Learning*. (1997)