

Learning for User Adaptive Systems: Likely Pitfalls and Daring Rescue

Martin E. Müller

Institute of Computer Science

University of Augsburg, D-86199 Augsburg, Germany

Martin.E.Mueller@informatik.uni-augsburg.de

Abstract

Adaptive user interfaces adapt themselves to the user by reasoning about the user and refining their internal model of the user's needs. In machine learning, artificial systems learn how to perform better through experience. By observing examples from a sample, the learning algorithm tries to induce a hypothesis which approximates the target function. It seems obvious, that machine learning exactly offers what is desperately needed in intelligent adaptive behavior. But when trying to adapt by learning, one will sooner or later encounter one or more well-known problems, some of which have been discussed in [Webb *et al.*, 2001]. We propose a framework for describing user modeling problems, identify several reasons for inherent noise and discuss few promising approaches which tackle these problems.

Keywords. Machine Learning for User Modeling, Sample size, Noise, Interpreting User Interactions.

1 Problems in ML for UM

There are several demands on adaptive systems:¹ they need to be quick because response latency or delay in adaptation is generally unwanted. They need to be very accurate, for wrong predictions can be worse than no adaptation at all. Additionally, they need to be able to work with very few examples as the user is in general not willing to give feedback or spend extra effort in interaction. Furthermore, one requires a user model to be scrutible and, of course, non-trivial. Finally, the effort in domain modeling shall be minimized, which means, that there is only very few background knowledge available. Each single problem rules out a few method or favors a certain learning approach — but there is no single solution which satisfies all requirements. The formalization of a Machine Learning problem as defined along with the introduction of the PAC learning model in [Valiant, 1984] can be transferred and applied to user modeling problems quite well. Since the PAC model is a rather pessimistic measure, the question arises, whether user models are

learnable at all: Many systems make heavy use of domain restrictions and severe implicit requirements on the structure of the domain. This, and the nature of feedback are permanent sources of noise which might explain why learning about the user is such a hard problem.

As an example, one might consider (affective) adaptive tutoring systems: The goal is to boost a student's learning progress as it is measured by a model of a student's skill. To increase a student's motivation an affective interface should express admiration when a task was solved correctly. But as correct affective behavior depends on a student's personality, the system needs to learn correct (user dependent) emotive behavior. However, the teaching signal is based on the student's skill as measured in relation to domain knowledge, while the actions taken by the system try to motivate the student further under the assumption that increased motivation yields increases learning progress. Accordingly, it must be clearly stated, what kind of information shall be learned by which data. [André and Müller, 2003] discuss this issue in the context of affective interaction.

Figure 1 shows the parallels and differences of machine learning (lower part) and user modeling (upper part). In machine learning, a learning algorithm \mathbf{A} induces a hypothesis h , which, according to the inductive hypothesis, shall approximate the target function \mathbf{t} based upon several observations in a sample \mathbf{s} . Each observation in \mathbf{s} is assumed to be labeled by a teacher signal which is assumed to be correct with respect to the learning target.

In user modeling, the learning target is an intensional description of a user's needs or plans \mathbf{i}_u , whereas the label \mathbf{o} that is assigned to an observable event presupposes a certain planning behavior of the user and an according interpretation of the observed interaction: The intention behind performing an observable action $\mathbf{o}(x)$ on x is veiled by the fact, that the user tries to realize his interest \mathbf{i}_u based on his very own idea of the functionality of interactive elements of the system (labeled by a question mark in figure 1).

Furthermore, the label $\mathbf{o}(x)$ that is taken as a teacher signal for the learning task is based on system's built-in interpretation of the user's interactions x . As such, it is subject to noise for two reasons: First, the observation $\text{obs}(x)$ itself may be noisy, and second, the interpretation $\text{int}(\text{obs}(x))$ of the user's interactions is also prone to delivering vague or uncertain results.

The outcome of the user modeling process is a user

¹Here and in the remainder of the paper we focus on individual, content-based user modeling.

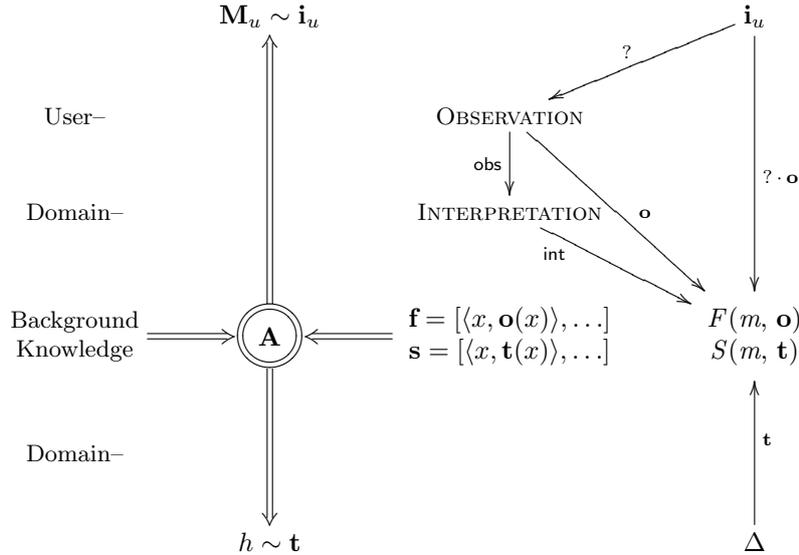


Figure 1: Parallels and differences in ML and UM

model M_u which shall approximate i_u — but actually, M_u rather learns to predict the most probable interaction based on earlier feedback.

Concluding, it seems that a closer investigation on how interactions pertain to user interests and feedback labels elucidates problems one encounters when trying to quickly induce user models from recorded data (see also [Kobsa, 2001]).

2 Examples for Learning User Models

Machine Learning plays an increasingly important role in user adaptive systems. Accordingly, many different techniques have been successfully employed in various approaches. We give a brief, non-exhaustive, overview of techniques by examples and point out some hidden disadvantages.

2.1 Naïve Bayesian Classifiers

NBCs, [Cowell *et al.*, 1999], are very popular for learning classifiers and thus, adaptive recommender systems, c.f. [Billsus and Pazzani, 1999]. They deliver fast and precise results which withstand most comparative evaluations. They are easy to implement for they do not need any background knowledge. However, the requirement that observables are pairwise conditionally independent is usually violated. The MANIC system, [Stern and Woolf, 2000], tries to learn, whether a student would 'want' to read a certain paragraph within an instructional text. If the student unfolds an offered node, this is taken as positive feedback. To predict likelihood of some topic to be viewed, the prior probabilities of *InstructionalType*, *MediaType*, and *Abstractness* are assumed to be independent. Even though NBCs require severe domain restrictions that in general do not hold, it has been shown, that NBCs perform well enough.²

²In addition, there remains the problem of interpreting 'not clicking' as 'not wanting': For an object x of matching difficulty $o(x) = 0$ implies $i_u(x) = 0$ which is not necessarily true, but this problem is independent of the learning

2.2 Bayesian Networks

Bayesian networks are a very popular approach in user adaptive interfaces, because they inherently provide us with a very elaborate theory on how to deal with uncertainty. However, with growing networks, computational effort increases rather quickly. The crucial question concerning BNs addresses the origin of topology and prior probability distribution. Both BNs and NBCs already work with 'random' or Laplacian priors, or relative frequencies from (few) observations. Nevertheless, initial probabilities need to be justified somehow. The READY system tries to recognize, reason about and adapt to a user's cognitive capacity which is, e.e. determined by time pressure, lack of knowledge or cognitive load. The structure of the underlying BN is motivated by a causal network whose architecture and initial probability distribution have been empirically validated, [Jameson *et al.*, 2000]. In this system, the user model M_u shall not primarily describe a user's interest but his cognitive state. As the interpretation of observed symptoms is empirically validated, noise caused by $\text{int}(\text{obs}(x))$ can be assumed to be marginal. Accordingly, this approach can be regarded to an approach where o delivers samples of high quality.

2.3 Finite State Machines and Hidden Markov Models

In general, a user's personality is said to be a set of user specific characteristics which persist through time, while emotions are rather short termed. Being impatient is a feature of a person's personality and won't change rapidly; but an impatient will sooner react angrily than a patient person when repeatedly confronted with unwanted data. Accordingly, [Gmytrasiewicz and Lisetti, 2001] describe an agent's personality by a finite state machine. The FSM consists of nodes describing emotional states and links describing transitions between emotional states. For example, if the user is bothered, a cooperative system behavior algorithm.

makes the user feel relaxed again, while an uncooperative system response results in an angry user. In context of learning emotive behavior, the problem of a proper theory of emotions arises. In general, the OCC model, [Ortony *et al.*, 1988], is widely accepted, but both number and types of emotions vary. [Murray and Arnott, 1993] describe discriminative features in emotion recognition from spoken language. But assuming a restriction to reliable observables and a commensurate discretization, one ends up with only four features and three values each. Using the attribute–value pairs described in [Murray and Arnott, 1993], one can not learn to distinguish *fear* from *anger* any more. Much more alarming is the fact, that *happiness* cannot be discriminated from *anger* either. Among others, this is a very important example for the need for a modeling language with sufficient expressive power.³

2.4 Artificial Neural Networks

ANNs offer methods for huge data sets, weak background knowledge and sometimes even unlabeled data. [Sklar, 2002] uses a multilayer perceptron to train agents which copy or teach a user’s online Tron game playing behavior (called *clones*, *peers*, and *instructors*) using backpropagation. The input layer consists of eight surrounding obstacle sensors and the output describes the next action to take (turns or forward). The goal was to approximate a preset winning rate for which this model already delivered quite impressive results. For a more sophisticated model describing a more general game playing strategy, the system would have to take into account knowledge of more complex wall configurations or even ‘understanding’ of an aerial view of the scene.⁴ The training set consisted of one half of 500 games played by 58 different humans. This example shows, that ANNs are capable of simulating rather complex behavior by approximating much simpler functions with literally no required background knowledge on even very noisy data⁵. On the other hand, training takes a lot of observed instances, which in context of adaptive systems means, that for personalization many, many interactions are needed before the system can adapt to the user. Similar considerations apply to self organizing maps which are also capable of unsupervised learning. Accordingly, they can be applied to clustering problems similar to those often solved by NBCs, see [Eilert *et al.*, 2001]. But even though self-organizing maps are able to cluster large sets of data, it is not guaranteed, that ‘similar’ concepts will be located next to each other on the map. Finally (and quite obviously), it has to be noted, that implicit knowledge as modeled by a trained ANN can only partially be explained and therefore remains rather incomprehensible to the user.

³Just as conditionally independence in NBCs is usually not guaranteed, HMMs are rather unspecific models insofar as the current state is only determined by the previous state but not by a larger history.

⁴Similarly, the network would require knowledge about the game history in terms of past game states which need to be incorporated into the ANN architecture by windowing or a recurrent topology.

⁵In this example, the sample can be assumed to be rather noise-free: There are exactly three observables for each keystroke (left, right, or straight), and they can hardly be misinterpreted.

2.5 Relational Learning

Complex domain descriptions involve huge background knowledge and require a rich description language. Sometimes, a user model consists of a ‘program’ describing procedural behavior. Trying to satisfy the demand for scrutable user models, it seems reasonable to employ machine learning techniques which deliver intensional descriptions instead of ‘extensional knowledge’ (as, e.g. in many collaborative approaches). One machine learning technique which is able to satisfy these needs is the underestimated approach of relational learning by inductive logic programming (ILP).⁶ However, there are few recent approaches in UM which make use of ILP because of its ability to learn relational knowledge, explicate implicit knowledge and its weak restrictions on the domain. [Kay and McCreath, 2001] describe an ILP approach to the problem of personalized mail filtering called MUMILP. Learning how to filter mail is supported by clearly labeled examples: Moving mail into folders delivers reliable classification data as a teacher signal **t**. Filtering spam is a binary decision, while filing mails into several folders is a more sophisticated learning task. In terms of ILP (or relational learning in general), the task is to learn a relation between messages and folders. Accordingly, the domain model requires e-mails to be represented in a way which includes predicates which identify the *sender* and other meta data as well as string occurrence etc. Resulting rules then may identify **spam** by a key word ‘**cash**’ as a mail **subject**, or classifies mails from ‘**jack**’ or ‘**jill**’ as mails from **friends**.

Concerning the spam filtering application, **f** can be assumed to be rather noise-free. Learning several classes simultaneously is more prone to noise as a single mail might belong to several classes at the same time. On the other hand, the fact that a certain mail belongs to a folder **recreation**, may support the proposition, that it does not belong to **conferences**.⁷ But if we are concerned with folders for several conferences with overlapping committees, this discrimination is not as easy as in the first case — but it is still a problem that can be solved using ILP methods.

ILP still suffers from its bad reputation as being vulnerable to noise, rather slow and heavily dependent on modeled background knowledge. Most disadvantages can be circumvented by the choice of proper bias as it also done in other approaches.

3 Hidden Pitfalls and How to Avoid Them

What are the most popular machine learning methods in user modeling? So far, we have described a few working systems which make use of different approaches and illustrated what kind of knowledge can be learned by which means. Table 1 tries to summarize advantages and problems that are likely to arise. It is clear, that a qualitative comparison always heavily depends on the actual application domain.

⁶It is interesting that user models now favor probabilistic or statistical approaches, while early UM approaches were based on logical descriptions, see [Kobsa, 1988].

⁷This has been discussed in detail in [Müller, 2002].

Algorithm	Sample Size	Backg. Knowledge	Scrutability	Training Effort	Run Time	Comment
NBC	Few/Avg ^a	Observables.	Average	Small	Fast	Cond. Independence
BN	Few/Avg ^a	Causal Model. Observables.	Good	Avg/High	Fast	Prior prob. Dist.; Validation
ANN	Many	None	Poor/None	(Very) High	Fast	Implicit knowledge, universal approx.
ILP	Average	Avg/High	Very Good	(Very) High	Average	Discovery of implicit knowledge ^b
GA/GP	Avg/Many ^c	None	Code-dep. ^e	(Very) High ^d	Code-dep. ^e	Representation ^e

a works with Laplacian distribution. *b* given appropriate background knowledge. *c* works with random initialization. *d* can be parallelized. *e* determined by representation of data as genome.

Table 1: A comparison of different ML techniques

4 Conclusion

Throughout this article, the user modeling problem was put into close relation to the machine learning problem. The aim in machine learning is to satisfy $h \sim \mathbf{t}$ by learning from \mathbf{t} -labeled examples in a sample \mathbf{s} . In user modeling, one tries to learn $\mathbf{M}_u \sim \mathbf{i}_u$, but the examples are labeled by $\mathbf{o} \neq \mathbf{i}_u$. Accordingly, the error measured when comparing \mathbf{M}_u with \mathbf{o} is the predictive error of the inferred user model on observed interactions—but it is not the ‘descriptive’ error in the user model itself. Similar considerations apply to the evaluation of hypotheses: To check, whether h is ε -good, the error estimation is based on the probability distribution Δ on the domain. In user modeling, ε needs gives an estimate using a testing subset of the feedback sample \mathbf{f} —while a proper evaluation would require a comparison of \mathbf{M}_u and \mathbf{i}_u .

This article presented a more formal description of the machine learning and user modeling problem. This framework was used to describe several ML4UM approaches. Several reasons for well-known problems one might encounter when applying machine learning ‘out of the box’ were shown.

It seems that—with reasonable restrictions and biases—user models can be learned in nearly all application domains. It has to be emphasized though, that learning is not an all-purpose tool: Sometimes it takes too long, sometimes there is too few data, sometimes there is too few background knowledge and sometimes, the tool simply does not fit the problem.

Keeping this in mind, one can circumvent several obstacles, if:

1. The content of the user model is self-contained and clearly restricted; and so is the domain
2. The domain is sufficiently well described
3. There is a legitimate trade-off between scrutability, adaptivity and domain modeling effort
4. One can live with assumptions on the domain and biases in learning
5. There is enough noise-free, reliable data available

Concluding, it should be noted, that the user model learning domain has one big advantage: In learning theory, Δ is assumed to be unknown. But in user adaptive system, we have the chance to guess, what a user wants based on earlier observations, on empirical evaluations, on different models of personalities and rich contextual knowledge. If we are able to achieve

more knowledge about the process that makes a user interact in a certain way, we might be able to denoise \mathbf{f} by understanding the intended actions. As a consequence, an intelligent, adaptive user interface will be able to ‘understand’ observed interactions $\mathbf{o}(x)$ and thus yield a better approximation of the teaching signal \mathbf{t} .

References

- [André and Müller, 2003] Elisabeth André and Martin E. Müller. Learning affective behavior. In Constantine Stephanidis, editor, *Proc. 10th Intl. Conf. Human-Computer Interaction HCII 2003, Universal Access in Human-Computer Interaction*, 2003.
- [Billisus and Pazzani, 1999] D. Billisus and M. Pazzani. A hybrid user model for news story classification. In Judy Kay, editor, *User Modeling: Proceedings of the 7th International Conference, UM-1999*, pages 99–108. Springer, 1999.
- [Cowell *et al.*, 1999] Robert G. Cowell, A. Philip Dawid, Steffen L. Lauritzen, and David J. Spiegelhalter. *Probabilistic Networks and Expert Systems*. Springer, 1999.
- [Eilert *et al.*, 2001] S. Eilert, A. Mentrup, M. E. Müller, R. Rolf, C.-R. Rollinger, F. Sievertsen, and F. Trenkamp. Bikini: User adaptive news classification in the world wide web. In *Workshop "Machine Learning for User Modeling"*, 8th Intl. Conf. on User Modeling, 2001.
- [Gmytrasiewicz and Lisetti, 2001] Piotr J. Gmytrasiewicz and Christine L. Lisetti. Emotions and personality in agent design and modeling. In Mathias Bauer, Piotr Gmytrasiewicz, and Julita Vassileva, editors, *User Modeling: Proceedings of the 8th International Conference, UM-2001*. Springer, 2001.
- [Jameson *et al.*, 2000] Anthony Jameson, Barbara Gromann-Hutter, Leonie March, and Ralf Rummer. Creating an empirical basis for adaptation decisions. In Henry Lieberman, editor, *IUI2000: International Conference on Intelligent User Interfaces*. ACM, 2000.
- [Kay and McCreath, 2001] J. Kay and E. McCreath. Automatic induction of rules for e-mail classification. In R. Schäfer, M. E. Müller, and S. A. Macskassy, editors, *Proceedings of the UM2001 Workshop on Machine Learning for User Modeling*, pages 59–66, July 2001.
- [Kobsa, 1988] Alfred Kobsa. User models in dialog systems. In Alfred Kobsa and Wolfgang Wahlster, editors, *User Models in Dialog Systems*. Springer, 1988.
- [Kobsa, 2001] Alfred Kobsa. Generic user modeling systems. *User Modeling and User-Adapted Interaction*, 11:49–63, 2001.
- [Müller, 2002] M. E. Müller. *Inducing Conceptual User Models*. PhD thesis, Institute of Cognitive Science, University of Osnabrück, 2002. Electronic publication.
- [Murray and Arnott, 1993] I. R. Murray and J. L. Arnott. Toward the simulation of emotion in synthetic speech: A review of literature on human vocal emotion. *Journal of Acoustical Society of America*, 93(2), 1993.
- [Ortony *et al.*, 1988] Andrew Ortony, Gerald L. Clore, and Allan Collins. *The Cognitive Structure of Emotions*. Cambridge University Press, 1988.
- [Sklar, 2002] Elizabeth Sklar. It takes a virtual village: Towards an automated interactive agency. In *Proc. AAAI Fall Symposium on Personalized Agents*, 2002.
- [Stern and Woolf, 2000] Mia K. Stern and Beverly Park Woolf. Adaptive content in an online lecture system. In Peter Brusilovsky, Oliviero Stock, and Carlo Strappavara, editors, *Adaptive Hypermedia and Adaptive Web-Based Systems: AH-2000*, number 1892 in LNCS. Springer, 2000.
- [Valiant, 1984] L. G. Valiant. A theory of the learnable. *Communications of the ACM*, 27:1134–1142, 1984.
- [Webb *et al.*, 2001] Geoffrey I. Webb, Michael J. Pazzani, and Daniel Billisus. Machine learning for user modeling. *User Modeling and User-Adapted Interaction*, (11):19–29, 2001.