

Learning about User in the Presence of Hidden Context

Ivan Koychev

GMD FIT.MMK

D-53754 Sankt Augustin, Germany

phone: +49 2241 14 2194 fax: +49 2241 14 2146

Ivan.Koychev@gmd.de

Abstract. This paper presents an algorithm for learning drifting and recurring user interests. The algorithm uses a meta-learning level for learning the current context, searches into the past observations for episodes that are relevant to the current context, "remembers" them and "forgets" the irrelevant ones. Finally the algorithm learns only from the selected relevant observation. The experiments conducted with a data set about calendar scheduling recommendations show that the presented algorithm improves the predictive accuracy significantly.

Keywords:

Interest Drift, User Profiling, User-Adaptivity, Recommender Systems

Introduction

Many approaches and systems for recommending information, products and other items have been developed (e.g. [1], [2], [13], [15], etc.). These systems try to help users in finding pieces of information or other objects in which the users will presumably be interested [6]. There have been mainly two different approaches so far. Content-based filtering systems take individual preferences for certain objects features of into account. Collaborative filtering systems on the other hand, typically build on similarities between users with respect to the objects in which users implicitly or explicitly express an interest. The combination between these approaches appears to have synergetic effect (e.g. [1], [13] etc.)

Many of those systems use machine learning methods for acquiring interest profiles. However, user interests can change over time. Some of the systems are provided with mechanisms that are able to track drifting user interests [10], [4], [2] and [8]. The problem about learning drifting user interests is relevant to the problem known as concept drift in the area of machine learning. Hence, some important works about learning drifting concepts are also discussed in the next sections.

In this paper it is assumed that the user interests are not only changing, but also possibly recur. In particular the user interests can be quite wide and the user can currently focus his attention on a small subset of his broad interests. For example, the whole set of user interests in case of WWW browsing can include interests that are

relevant to his job, as well his hobbies etc. Even the user's job related interests could be quite extensive and interdisciplinary. A system that assists the user in Internet browsing should be flexible enough to recognize what are his current interests and provide him with relevant recommendation. When the current user interests often change, a precise user profile cannot be learned from a small set of relevant recent observations only. Hence, the system can search for past episodes where the user has demonstrated a similar set of interests and try to learn a more precise description of the current user interests, "remembering" relevant and "forgetting" irrelevant observation.

This paper presents an approach of learning about user interests and preferences in the presence of changing and recurring hidden context. The main idea is to use a meta-learning level for learning current context. The algorithm searches in the past observations for episodes that are relevant to this context and finally the algorithm learns from selected observations (i.e. "remembers" relevant and "forgets" irrelevant old observations). This approach will be introduced in Section 3 together with experiments, which provide evidence of its impact on the predictive accuracy for a real life data set, presented in Section 4.

Tracking Drifting Interest

It is assumed that a sequence of selected objects is given. Based on these observations about a user, a recommender system aims at recommending objects, which are relevant to his current interests. As mentioned before, often the current user interests change according to up-to-the-minute user goals. In this section we will briefly discuss different approaches developed in the areas of machine learning and user modeling devoted to this problem.

In the area of machine learning there is plenty of works dedicated to learning changing (aka shifting, drifting or evolving) concepts (e.g. [9], [16], [17], etc.) The systems use different forgetting mechanisms to cope with this problem. Typically, those time-based forgetting mechanisms delete examples that are not up to date. Consequently they use partial memory learning in case of time-based forgetting [9]. Those time-based forgetting mechanisms often use a so-called time window. The concept descriptions are learned from the newest observations only (e.g. only last l examples are used for training), assuming that, if concept changes, then the old examples will become irrelevant to the current period. An improvement of this approach is the use of heuristics to adjust the size of the window according to the current predictive accuracy of the algorithm [17]. Maloof and Michalski [9] employ a similar approach, which uses a time-based function to provide each instance with an age. Examples that are older than a certain age are forgotten. These approaches totally forget the observations that are outside the given window or older than a certain age. The examples, which remain in the partial memory, are equally important for the learning algorithms. This is abrupt and total forgetting of old information, which in some cases can be valuable. To avoid loss of useful knowledge learned from old examples some systems keep old rules till they are competitive with the new ones [10]. Another approach learns a hybrid model consisting of both a short-term and a

long-term model of user interests [2]. The short-term model is learned from the most recent observations. This hybrid user model is flexible enough to consider changes in user interests and keeps track of long-term user interests as well.

The time-based forgetting mechanism presented in [8] and [7], suggests the usage of gradual forgetting functions, which provide each example with a weight according to its appearance over time. The importance of an example diminishes slowly with time. This forgetting mechanism can be fruitfully combined with other partial memory learning approaches (e.g. time window) mentioned above.

Another approach to learning and tracking changing concepts employs two-level learning algorithms to adjust to changing contexts by trying to detect (via meta-learning) contextual clues and using this information to focus the learning process [5] and [17]. The approach presented in this paper also employs a meta-learning level, however it doesn't assume that the attributes represent current context explicitly. It assumes that the recent observations are able to provide information about current context. The recent relevant observations cannot be sufficient to learn an accurate description of the current user interests. Though, it is assumed that the learned description is accurate enough to be able to distinguish the past episodes, which are relevant to the current context. Moreover, the approach assumes the possibility of recurring user interests and aims at benefiting from this by "remembering" relevant past observation and by "forgetting" irrelevant ones.

Learning User Interests in the Presence of Recurring Context

The recurrence of drifting interests can be guided by different conditions. For example, the interests' drift can be driven by a periodical time-function. In this case, we can use forgetting mechanisms that forget out-of-date examples (i.e. using a time window). However, the examples that represent the current user's interests can be insufficient for learning accurate descriptions. Therefore remembering the old examples, which are relevant to the current period, is expected to improve the predictive accuracy. Hence, in case of periodical recurrence, the observations that have occurred in the neighborhood of the following times are also important:

$$t_{remember} = t_{current} - s \times p \quad | s = 1, 2, \dots, \lfloor n/p \rfloor \quad (1)$$

where: p is the length of the period, n is the length of the observed sequence and t represents the time.

Quite often recurrence of the context cannot be defined a priori and it appears more randomly. Additionally, often the context is hidden and explicit indicators about its changes and recurrences cannot be discovered easily. Hence, in such cases the aim should be to learn more about the current context and then to search for old observations, which were done in a similar context, aiming at selecting a bigger training data set and in this way to learn a more accurate description of the current user interests. An algorithm that makes use of this idea consists of the following three steps:

- 1) *Learning about current context.* A relative small time window is used to learn a description of the current context (i.e. learning a description of the user interests based on the recent observations about the user).
- 2) *Remembering relevant past episodes.* The learned description in step 1) is tested against the rest of the training set. The episodes that show a predictive accuracy that is greater than a predefined threshold are selected (i.e. selecting the episodes that are relevant to the current context).
- 3) *Learning from context-related examples.* The new data set, selected in step 2), is used for learning a new description of the current user interests, which is expected to be more accurate.

The above algorithm requires a predefinition of the following settings:

- *The size of the time window* used in step 1). This time window should be big enough to allow a sufficiently accurate description of the current context to be learned, as well as short enough to be able to track fast changing user interests. Some enhancements like adaptive time window [17] and gradual forgetting [8] can be employed aiming at improving predictive accuracy.
- *The episode selection criterion* for step 2). This criterion should be able to distinguish the examples/episodes that are relevant to the learned context in step 1). It is desirable that criterion be resistant to noise in the observations.
- *The threshold for the episode-selecting criterion* in step 2). After the episode selection criterion has been established, a suitable threshold should be defined, which should secure as far as possible, that only relevant old observations be selected.
- *The learning algorithms* used in steps 1) and 3). The same or different learning algorithms can be used in those steps.

Those settings should be defined empirically based on preliminary investigation of the application domain. The implementation of the algorithm described in the next section gives example of such definitions.

Experiments with Recommendations for Calendar Users

Mitchell et al [10] developed a software assistant that helps schedule a particular user's calendar: a calendar manager, called CAP (Calendar APprentice). CAP learns the users' scheduling preferences through routine use, enabling it to give customized scheduling advice to each user. It can be considered as an analogy to a human secretary who might assist someone in managing a calendar.

The user's scheduling preferences depend very much from a hidden context. Some of this context can be assumed and explicitly presented and used for improving predictive accuracy (e.g. academic semesters etc.). However, there are many other events and conditions that can influence the meeting schedule and which cannot be explicitly represented by an attribute space (e.g. room availability, the schedule preferences of other participants of a meeting and many others). Under this condition the predictive accuracy of the system can oscillate with very high amplitude. A more

comprehensive investigation and analysis of the specifics of the domain can be found in [10].

This section presents the results from experiments conducted with the CAP data set¹. The attributes used for describing the calendar events in the current experiments are listed in Table 1. The task is to predict the meeting *Duration*, *Day-of-week*, *Location* and *Start-time*.

third-most-common-time-last-60-days-this-meeting-type
third-most-common-time-last-60-days
second-most-common-time-last-60-days-this-meeting-type
second-most-common-time-last-60-days
most-common-time-these-attendees-last-60-days
most-common-time-these-attendees
most-common-time-last-60-days-this-meeting-type
most-common-time-last-60-days
most-common-day-these-attendees-last-60-days
most-common-day-these-attendees
duration-of-next-meeting-with-these-attendees
duration-of-last-meeting-with-these-attendees
day-of-week-of-next-meeting-with-these-attendees
day-of-week-of-last-meeting-with-these-attendees
req-seminar-type
req-course-name
req-speakers
single-person?
action
cmu-attendees?
group-attendees?
position-attendees
department-attendees
sponsor-attendees
known-attendees?
<i>Duration</i>
<i>day-of-week</i>
<i>Location</i>
<i>start-time</i>

Table 1. The list of features that are used for describing calendar events.

The settings of the algorithm listed in the previous section are defined for the conducted experiments as follows:

- *The size of the time window:* Preliminary experiments show that for different prediction tasks the size of the window that produces best predictive accuracy can be quite different. For the given data set the best accuracy is reached for the window of the following size: *Location* - 200; *Duration* - 350; *Start-time* - 350; *Day-of-week* - 400.
- *The episode selection criterion* for step 2). The criterion used in this implementation selects the examples e_j for the new data set taking into account the average predictive accuracy in its neighborhood. In particular a small episode

¹ <http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-5/www/cap-data.html>

around the example, which includes the previous two and next two examples, is used. An event will be selected for the new training set $e_j \in S_{new}$ if the average predictive accuracy for this episode is greater or equal than a predefined threshold τ .

- *The threshold for the episode-selecting criterion* in step 2) is setup to $\tau = 0.6$ in all experiments.
- *The learning algorithm* used in steps 1) and 3) is Induction on Decision Tree (aka ID3) [14]. The reason for selecting this algorithm was that this algorithm generates explicit user profiles, which is an important aspect for user modeling. This was the reason for this algorithm to be used in CAP as well.

<i>Prediction task</i>	<i>CAP</i>	<i>ID3-FM</i>	<i>Context Learning</i>
Location	64	58	67
Duration	63	71	79
Start-time	34	39	48
Day-of-week	50	52	66
Average	53	55	65

Table 2. Comparison of predictive accuracy for the User

Table 2 presents the results from experiments with data for User 1. In this experiment a new description of user preferences is learned after each 10 meetings. The learned description on each step is tested on the next 10 meetings. The results are compared with the CAP. The average predictive accuracy of the ID3 with full memory (ID3-FM) to some extent outperforms the CAP. This is slightly surpassing, because CAP is designed to track changing user preferences better than a simple learning algorithm. It is well known that some implementation detail like attribute selection criteria and used pruning method, can change significantly the outcome of the algorithm. However, the comparison between full-memory learning algorithm (ID3-FM) and the presented two-level learning algorithm is more important in the context of this paper. The same implementation of the basic learning algorithm is used, which make the results comparable. The results from the experiments show that the context-learning algorithm is able to improve the average predictive accuracy for each feature. All those improvements are significant (using *t-test* with $\alpha = 0.01$).

Figure 1 shows the results from experiments for the predicted features. Figure 2 enlarges one of them (Day-of-Week) for a better visualization. It can be seen that the user's preferences can change abruptly, which leads to a dramatic decreasing of the predictive accuracy. The presented two-level algorithm performs better than the basic algorithm.

Experiments with this data set, which use the Winnow and Weighted-Majority algorithms, were reported in [3]. The Winnow with a large feature set reaches the best average accuracy, which is equal to that reached by presented algorithm in the presented experiments. The results reported in [3] should be compared with the one level learning algorithm, and it can be concluded that on this data set, the Winnow algorithm performs better than ID3. Using such algorithms in the presented two level learning is expected to improve the predictive accuracy additionally. However these

algorithms are not suitable for producing explicit user profiles, which is considered to be important in the area of user modeling.

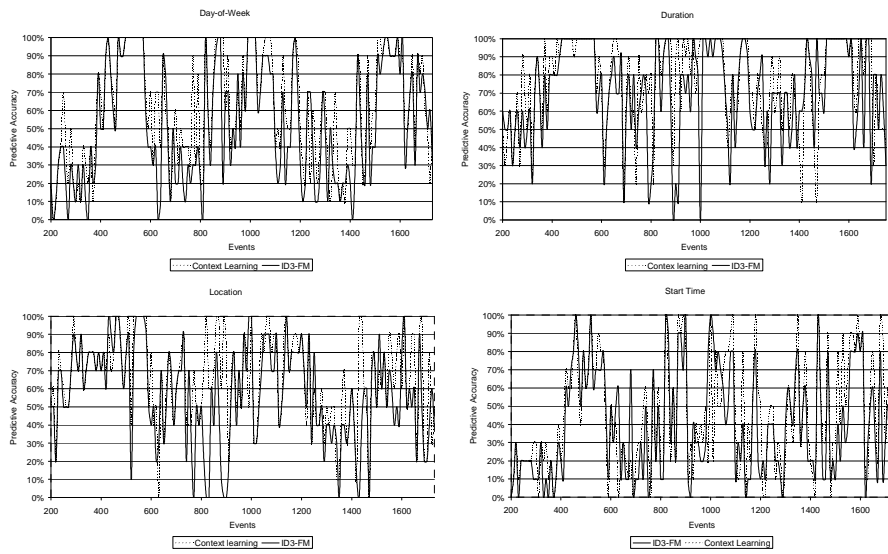


Fig. 1. . The improvement in predictive accuracy for the predicted features.

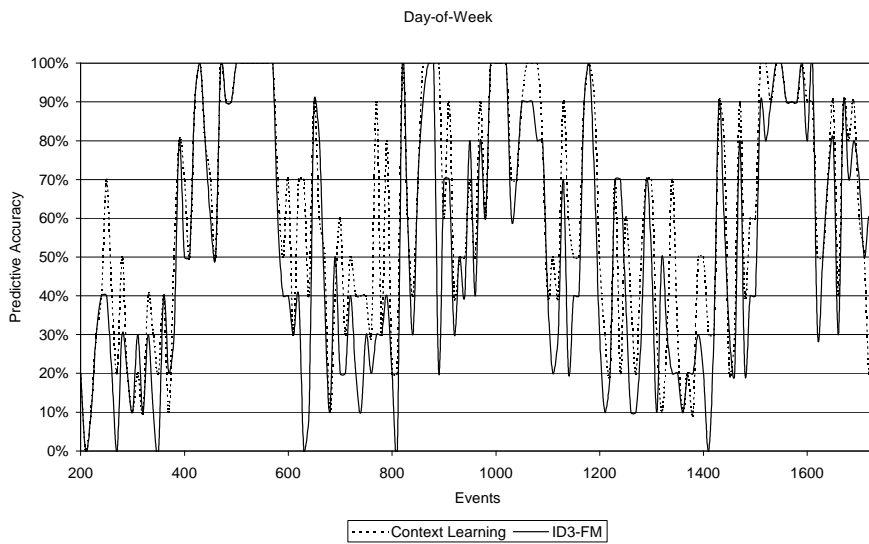


Fig. 2. The improvement in predictive accuracy for feature Day-of-Week.

Related Works

A software assistant for scheduling meetings is described in [10]. It employs induction on decision tree to acquire assumptions about individual habits of arranging meetings. The learning method uses a time window to adapt faster to the changing preferences of the user. The newly generated rules are merged with old ones. The rules that perform poorly on the test set drop down the list.

The FLORA systems [17] are also systems for coping with concept drift. They use a time window with flexible size, which is adapted dynamically. The window size and thus the rate of forgetting is supervised and dynamically adjusted by heuristics that monitor the learning process. Additionally the described system architecture assumes that the learner maintains a store of concept descriptions relevant to previous contexts. When the learner suspects a context change it will examine the potential of previous stored descriptions to provide better classification.

A method that can learn and track changing context by using meta-learning is presented in [16]. The assumption is that the domain provides explicit clues as to the current context (e.g., attributes with characteristic values). A two-level learning algorithm is presented that effectively adjusts to changing contexts by trying to detect (via meta-learning) contextual clues and using this information to focus the learning process. Two systems based on this model are presented. They differ in the underlying learning algorithm and in the way they use contextual information: METAL(B) combines meta-learning with a Bayesian classifier, while METAL(IB) is based on an instance-based learning algorithm.

A system, which learns user's interest profiles by monitoring web and e-mail habits, is presented in [4]. A clustering algorithm is used to detect user interests, which are then clustered to form interest themes. User profiles must also adapt to changing interests of the users over time. This research shows that user's interests can be tracked over time by measuring the similarity of interests in a time period.

An offline meta-learning algorithm for identification of hidden context is presented in [5]. The approach assumes that concepts are likely to be stable for some period of time. It uses batch learning and contextual clustering to detect stable concepts and to extract hidden context.

An intelligent agent called NewsDude that is able to adapt to changing user interests is presented in [2]. It learns two separate user models: one represents the user's short-term interests and the other one represents the user's long-term interests. The short-term model is learned from the most recent observations only. It represents user models that can adjust more rapidly to the user's changing interests. If the short-term model cannot classify the story at all, it is passed on to the long-term model. The purpose of the long-term user model is to model the user's general preferences for news stories and compute predictions for stories that could not be classified by the short-term model.

In [9] a method for selecting training examples for a partial memory learning system is described. The forgetting mechanism of the method selects extreme examples that lie at the boundaries of concept descriptions and removes examples that are irrelevant or outdated for the learning task from the partial memory. The method uses a time-based forgetting function to remove examples, which are older than a certain age from the partial memory.

Conclusion

The paper describes a two-level learning algorithm that is able to learn user interests and preferences in the presence of hidden context. It is assumed that the user interests recur over time. The algorithm tries to benefit from this by remembering relevant past observations and forgetting irrelevant ones. Conducted experiments with recommendation about calendar scheduling demonstrate that the approach is able to improve the predictive accuracy significantly. Further experiments and investigations of the predefined settings listed in section 3 are expected to improve the predictive accuracy of the algorithm additionally. The development of approaches that can dynamically adapt one or more of this parameter is expected to be a valuable contribution.

References

1. Balabanovic, M. and Shoham, Y. Fab: Content-Based, Collaborative Recommendation. In *Communications of the ACM*, March 1997/Vol.40, No. 3, p. 77-87.
2. Billsus, D., and Pazzani, M. J. A Hybrid User Model for News Classification. In *Kay J. (ed.), UM99 User Modeling Proceedings of the Seventh International Conference*, Springer-Verlag, Wien, New York, (1999), pp. 99-108.
3. Blum A.: Empirical Support of Winnow and Weighted-Majority Algorithms: Results on a Calendar Scheduling Domain. In *Machine Learning 26*, Kluwer Academic Publishers (1997), p. 5-23.
4. Grabtree, I. Soltysiak, S.: Identifying and Tracking Changing Interests. *International Journal of Digital Libraries* vol. 2, Springer Verlag, (1998), 38-53.
5. Harries, M. and Sammut, C.: Extracting Hidden Context. *Machine Learning 32*, Kluwer Academic Publishers (1998), 101-126.
6. Kobsa, A., Koenemann, J. and Pohl, W. (2001). Personalized Hypermedia Presentation Techniques for Improving Online Customer Relationships. *The Knowledge Engineering Review*, forthcoming.
7. Koychev, I. Gradual Forgetting for Adaptation to Concept Drift. In Proceedings of ECAI 2000 Workshop "Current Issues in Spatio-Temporal Reasoning". Berlin, Germany (2000), pp. 101-106.
8. Koychev, I. and Schwab, I.: Adaptation to Drifting User's Interests - *Proceedings ECML2000/MLnet workshop "ML in the New Information Age"*, Barcelona, Spain, 2000, pp. 39-45.
9. Maloof, M. and Michalski, R.: Selecting examples for partial memory learning. *Machine Learning 41*, Kluwer Academic Publishers (2000), 27-52.
10. Mitchell, T., Caruana, R., Freitag, D., McDermott, J. and Zabowski, D.: Experience with a Learning Personal Assistant. *Communications of the ACM 37.7* (1994), 81-91.
11. Mladenic D. Personal WebWatcher: Implementation and Design. Technical Report IJS-DP-7472, Department of Intelligent Systems, J. Stefan Institute, Slovenia, (1996).
12. Pazzani M. and Billsus D. Learning and Revising User Profiles: The Identification of Interesting Web Sites. *Machine Learning, 27*, Kluwer Academic Publishers (1997), p. 313-331.
13. Pazzani, M. (1999): A Framework for Collaborative, Content-Based and Demographic Filtering. In *Artificial Intelligence Review 13*(5), 393-408.

14. Quinlan, R.: Induction of Decision Trees. *Machine Learning* 1, Kluwer Academic Publishers (1986), 81-106.
15. Schwab, I., Pohl, W. and Koychev, I.: Learning to Recommend from Positive Evidence, Proceedings of Intelligent User Interfaces, ACM Press (2000), pp.241-247.
16. Widmer, G.: Tracking Changes through Meta-Learning. *Machine Learning* 27: Kluwer Academic Publishers (1997), 256-286.
17. Widmer, G. and Kubat, M.: Learning in the presence of concept drift and hidden contexts: *Machine Learning* 23, Kluwer Academic Publishers (1996), 69-101.