Proceedings of the Twenty-Ninth International
Florida Artificial Intelligence Research Society Conference

Smart Learning: Time-Dependent Context-Aware Learning Object Recommendations

Christopher Krauss
Fraunhofer Institute FOKUS
Kaiserin-Augusta-Allee 31, Berlin, Germany
Email: christopher.krauss@fokus.fraunhofer.de

Abstract

In a digital classroom, analysis of students' interactions with the learning media provides important information about users' behavior, which can lead to a better understanding and thus optimizes teaching and learning. However, over the period of a course, students tend to forget the lessons learned in class. Learning predictions can be used to recommend learning objects users need most, as well as to give an overview of current knowledge and the learning level.

The representation of time based data in such a format is difficult since the knowledge level of a user with a learning object changes continuously depending on various factors. This paper presents work in progress for a doctoral approach to extend the traditional user-item-matrix of a recommendation engine by a third dimension – the time value. Moreover, in this approach the learning need consists of different context factors each influencing the relevance score of a learning object.

Introduction

Recommendation engines in a closed domain, like in Intelligent Learning, are following a special paradigm: At the end of a course, where only a closed user group interacts with a finite amount of items, in this case learning objects, most students provided feedback on almost all items. However, common prediction techniques do not cover that the user's need for learning particular items changes significantly over time – inversely proportional to the user's knowledge level on these learning objects. Thus, this domain needs a time-dependent context-sensitive representation of its user-model. This helps users to observe their learning level and get appropriate learning recommendations as well as teachers to get a direct feedback on the learning behavior.

The aim of the Smart Learning Recommender (SLR) is to introduce new features for Learning Management Systems (LMS) to support a blended-learning approach for universities, chamber of crafts and adult education centers. Thereby, students can keep track of their individual predicted knowledge level on different learning objects at every point in time and get personalized learning recommendations based on the determined learning need value.

The rest of the paper is organized as follows: Section 2 introduces related work and section 3 explains the general idea of the learning need functions as well as mathematical fundamentals for this approach, followed by a section on the learning recommendation algorithm. Section 5 discusses the main challenges of this approach as well as experimental design. At the end, the paper concludes with a summary and an outlook.

Related Work

Manouselis et al. argued that more than the half of all published recommender systems in the area of Intelligent Learning Technologies were still at a prototyping or concept level and only 10 have been evaluated in trials with real participants (Manouselis et al. 2011). Most of these systems are designed to predict items in a closed system using the two-dimensional Collaborative Filtering user-item-matrix, such as "CourseRank" (Koutrika et al. 2009) of the Standford University, "Altered Vista" (Recker and Wiley 2001) that uses Association Rules of frequently used learning objects in courses or "RACOFI", a rule-applying collaborative filtering system "that assists on-line users in the rating and recommendation of audio (Learning) Objects" (Boley 2003). However, these recommenders only work on a flat item hierarchy and without time or extended context data. Nevertheless, it seems to be very important to include the intrinsic and extrinsic motivation of students, in terms of "pedagogical aspects like prior knowledge, learning goals or study time" (Manouselis et al. 2011).

The "APOSDLE" Recommendation Service (Stern et al. 2010) uses an extended user profile as input for appropriate content recommendations and a web tool for ontology evaluation for identifying semantic similarities. The "Multi-Attribute Recommendation Service" (Manouselis, Vuorikari, and Van Assche 2007), in turn, uses ratings on different attributes and criteria for the same learning object in order to calculate proper recommendations. Moreover, in (Huang et al. 2009) a Markov chain model is used to calculate sequences of learning objects and recommend learning paths and the "Learning Object Sequencing" (Shen and Shen 2004) uses a novel sequencing rule algorithm by processing topical ontologies. The SLR engine follows a similar approach by taking the context, in terms of various factors, as well as item sequences and hierarchies into account.

Only a very few papers address the time-aspect in recommenders for learning objects: Drachsler et al. underline the significance of the attribute that represents time taken to complete learning objects (Drachsler et al. 2009). And Pelanek et al. evaluated the closed correlation between multidimensional student skills and the timing of problem solving that "may be useful for automatic problem selection and recommendation in intelligent tutoring systems and for providing feedback to students, teachers, or authors of educational materials" (Pelanek and Jarusek 2015).

However, most research on time-dependent recommendation engines have been done in the area of movie predictions: For instance, a time-context based Collaborative Filtering algorithm (He and Wu 2009) proposed by Liang He describes the inclusion of rating time in the computation of predictions for movie ratings. And Zhang et al. (Zhang and Liu 2010) describe an approach to consider changes in users' interests when recommending the items for the users. While a novel K-nearest neighbor algorithm (Liu et al. 2012) finds time-based neighborhoods of a user. Even though, this movie recommenders inspire the development of SLR, they mostly analyze and predict the users' interests in items instead of considering the users' knowledge. Learning environments, in turn, represent a totally different prediction paradigm: students have to learn all relevant objects in order to pass the final exam, no matter if they are interested in it or not.

Learning Need & Relevance Function

Learning recommendation is all about identifying the learning need of a user u for an item i. The user-item-pair is presented by a relevance score $rscore_{u,i}$ having the value from 0 to 1, where 0 indicates the lowest relevance and 1 indicates the highest possible relevance. The relevance score defines a time and context dependent value and is expressed as a time dependent function:

$$rscore_{u,i,t} = rf_{u,i}(t)$$

The relevance function $rf_{u,i}(t)$ of user u for item i is derived from several sub-functions $rf_{u,i,x}(t)$ of individual factors $x_1, ..., x_n$, as a function of time t, each representing another context. The factor itself is based on real user-item value pairs $rscore_{u,i,t,x}$. Since the real learning need changes continuously over time, the factor can be abstracted as continuous function, as well. The different factor types and considered formulas per user, item and time are:

- Interaction with a learning object: This factor indicates how much of the available material for a learning object was accessed by a student at a specific moment in time:

$$rscore_{u,i,t,x1} = 1 - \frac{accessedContent}{availableContent}$$

- Processing time of a learning object: This factor indicates how long the student learned a learning object. It is 0 when the student needed exactly the intended time and between 0 and 1 if he needs more or less time as defined in the metadata.

$$rscore_{u,i,t,x2} = |1 - \sqrt{\frac{timeNeededForLearning}{timeIntendedForLearning}}|$$

  If the user needed more than 4 times of the actual intended time, the learning need is 1.

- Self-assessments for this learning object: A student can explicitly define his knowledge level in particular points in time on a 1 to 5 stars scale:

$$rscore_{u,i,t,x3} = 1 - \frac{currentKnowledgeLevel}{highestKnowledgeLevel}$$

- Performance in exercises: The percentage of wrong answered questions represents the relevance of the exercise factor:

$$rscore_{u,i,t,x4} = \frac{wrongAnswers}{allAnswers}$$

- Fulfilled pre-requisites: The more a student learned the underlying learning objects, the higher the relevance score of the subsequent items:

$$rscore_{u,i,t,x5} = \frac{fulfilledPreRequisites}{allPreRequisites}$$

- The lecture times factor indicates the timely relevance of a learning object for face-to-face lectures, in terms of current time stamp t divided by the lectures time stamp:

$$rf_{u,i,x6}(t) = 1 + |1 - \frac{t}{timeOfLecture}|$$

- Exam relevance: Learning objects that are more relevant for exams, show a higher relevance score than optional contents – in terms of a constant value defined in the learning object metadata.

- Forgetting effect: After learning an object, the gained knowledge will slowly decrease over time. After each learning iteration, the forgetting factor was set to 0 and had a less effect on the future learning need. This factor has been analyzed with real students in order to model an appropriate forgetting factor. Therefore, students were asked in regular intervals to remember specific details of a before learned topic. This preliminary study resulted in a first approximation of the forgetting factor and will be published in future work.

- Collaborative learning needs: The relevance functions of similar users on this learning object are taken into account in order to offset underestimations and bad learning plannings for the current user. In an initial phase, the mean average learning need of all other students for this learning object represent this factor. The plan is to replace this mean learning need by a weighted average factor of nearest-neighbors, who show the most similar learning curves.

Each factor's relevance score represents an aspect of the learning need, is restricted to the range of [0,1] and must be evaluated in the future. At the end, all single-factor functions are weighted. The weighted average of all factors describes the total learning need of the learning object for that user and is calculated as

$$rf_{u,i}(t) = \frac{\sum_{x=0}^{n}(w_x * rf_{u,i,x}(t))}{\sum_{x=0}^{n} w_x}$$

Here w_x is the weight of a single factor x in $\{x_1, ..., x_n\}$ and n is the number of factors. At the beginning, the weights are predefined by experts, such as teachers.

Multi-level Learning Recommendations

The overall recommendation engine analyses the current learning need values of the requesting student for all contents in the course. Thereby, the model of the SLR can be created offline: The relevance functions are computed in regular intervals by processing all existing user-item-time-triplets. When a user requests recommendations, the relevance scores for all items are calculated on demand by considering the current time value for t. Afterwards, the items are sorted by their learning need value. The result is a Top-N list of learning objects beginning with the highest relevance scores that represents the most important topics for the student that need to be learned at that time.

Figure 1 shows, how the learning need of an item is presented graphically to a learner. The personal knowledge level defines how successful a student is in learning an item and is inversely proportional to the learning need of the student towards the learning object. The knowledge level $kl(t)$ can be computed from the relevance function $rf(t)$ as

$$kl(t) = 1 - rf(t)$$

Using this visualization, students have a chance to understand how the system calculates the prediction and might change factor weights to adjust new recommendations.

Moreover, Learning Objects are stored in a multi-level hierarchy in order to topically structure a course – top-level items represent a container with a set of sub-level items. A leaf item (a learning object without children) contains the minimum information set that is at least required to understand the given (sub-)topic. However, the user may provide item feedback, in terms of self-assessments, exercises, interactions and processing time, on all hierarchy levels – even for the same topic. This differentiation allows a representation of diverging knowledge levels for top- and sub-level items – e.g. a student might have a good high-level understanding of a topic, but misses some details on specific sub-level contents or vice versa. In case the user has not provided the same type of feedback on the item's parent before, it is implicitly transferred from the child to the parent object. So, the parent may implicitly represent the average of all child learning objects.

Hence, the engine needs to avoid recommending the same

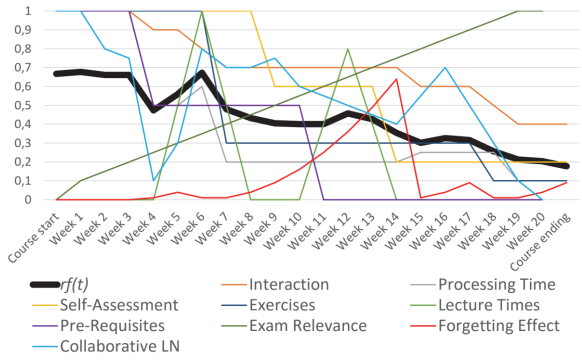

Figure 1: Example of a learning need function with individual factors

topic with different detail levels within the predicted list. An algorithm iterates over the generated Top-N list, beginning with the most relevant learning object. An item will be eliminated from the list, in case a related child or parent learning object that describes the same topic, was recommended before and thus, shows a higher score. As a result, students will get recommendations for all topics of a course in a predicted order, but only on an appropriate detail level.

Challenges and Experimental Design

The introduced three-dimensional user-item-matrix (containing user-item-time-triplets) also leverages common collaborative filtering approaches. The calculation and weighting of nearest neighbors will be done by also considering the time aspect. Therefore, the deviation of two user-item-pairs will not only be based on the subtraction of two constants any more – as for common collaborative filtering approaches. It will be based on the correlation of the corresponding relevance functions of two learners. The assumption: the higher the correlation coefficient of two learning need functions of different students on the same item, the more similar their knowledge and their learning behavior. Taking this information into account, the system shall recommend similar learners, experts in specific areas as well as classify and cluster general learning types. Algorithms covering time-dependent item-based as well as neighborhood-based classification and rating prediction are going to be evaluated during this project. Another big challenge of this approach comes from predictions of the future learning need by extrapolating specific factor functions, for instance the forgetting effect. In the planned experiments, different algorithms, weights and settings are going to be further analyzed.

Since this work shows a novel approach for time-dependent learning object recommendations, the need for an evaluation based on an academic data set is very high. Unfortunately, only a few data sets are published (e.g. (Cortez and Silva 2008) or (Wang et al. 2014)) and no data set matches all requirements of this approach. At least, the challenge data set from KDD Cup 2010 on Educational Data Mining (Stamper et al. 2010) matches some of the requirements. It is divided into 5 different packages (e.g. "Algebra I" and "Bridge to Algebra" from 2005 and 2008) with between 575 and 6,043 students per package. It contains a detailed description of the students' performances when solving mathematical problems and thus, represents typical learning behavior. One evaluation approach would be to subdivide the KDD item data into different context factors – each influencing the total learning need. However, the KDD data set contains a lot information on the interaction with learning objects as well as the processing time and results in exercises, but data on other essential factors as well as structured metadata on the hierarchy and topical sequences of learning objects are missing.

That is why new studies with learners are going to be conducted in three consecutive 5 month courses – each with the same new generated learning objects and metadata, but different students in each course. At the beginning of the each course, the participants are asked to answer surveys with demographic information and their motivation. During the

course, participants will get access to the learning objects exclusively via a provided Learning Companion Application in order to keep track of their learning behavior. Moreover, they can give feedback at any time, how helpful a specific recommendation was and whether the learning need shows a proper presentation of their actual knowledge. This data is important for accuracy measurements as well as the analysis and adjustment of factor weights. At the end, the participants are asked to answer a second survey summarizing their overall perception of the system. The final exam grade will be related with the tracked learning behavior data, but due to local data protection regulations, it is optional only. The studies help to increase the performance of the system with each experimental iteration. It is planned to publish the mined information as anonymized open source data set.

Conclusion

The novel Smart Learning Recommender aims at assisting students during blended-learning courses – in lectures as well as during the preparation of these lectures, the wrap-up and exam learning phase. Thereby, the engine shows an extended user model: the item feedback of each user is subdivided into different context factors. In contrast to rule-based recommendation engines and classification machine learning algorithms, it also respects the changing knowledge level on specific learning objects in a continuous time interval. Moreover, the system respects the overall course structure, in terms of the best topical sequence and thematic hierarchies consisting of topics and sub-topics. Due to the lack of an appropriate academic data set, studies with real learners shall evaluate typical learning behaviors, how SLR performs with different settings and how users accept learning recommendations. An analysis of the students' knowledge level at several points in time shall result in an accurate representation of the different factors and their weights.

Acknowledgments

The author would like to especially thank Agathe Merceron, Manfred Hauswirth and Stefan Arbanowski for their supervision, as well as Rakesh Chandru and Kim Mensing for their assistance. This work is sponsored by the German Federal Ministry of Education and Research.

References

Boley, H. 2003. Racofi: A rule-applying collaborative filtering system. in 2003 ieee. In *WIC International Conference on Web Intelligence/Intelligent Agent Technology*, 430–434.

Cortez, P., and Silva, A. M. G. 2008. Using data mining to predict secondary school student performance. *In Proceedings of 5th Future Business Technology Conference (pp. 5-12), Porto, Portugal, April, 2008, EUROSIS*.

Drachsler, H.; Hummel, H.; Van den Berg, B.; Eshuis, J.; Waterink, W.; Nadolski, R.; Berlanga, A.; Boers, N.; and Koper, R. 2009. Effects of the isis recommender system for navigation support in self-organised learning networks. *Journal of Educational Technology & Society* 12(3):115–126.

He, L., and Wu, F. 2009. A time-context-based collaborative filtering algorithm. In *IEEE International Conference on Granular Computing, 2009, GRC'09.*, 209–213. IEEE.

Huang, Y.-M.; Huang, T.-C.; Wang, K.-T.; and Hwang, W.-Y. 2009. A markov-based recommendation model for exploring the transfer of learning on the web. *Journal of Educational Technology & Society* 12(2):144–162.

Koutrika, G.; Bercovitz, B.; Kaliszan, F.; Liou, H.; and Garcia-Molina, H. 2009. Courserank: A closed-community social system through the magnifying glass. In *ICWSM*.

Liu, Y.; Xu, Z.; Shi, B.; and Zhang, B. 2012. Time-based k-nearest neighbor collaborative filtering. In *International Conference on Computer and Information Technology, 2012 IEEE 12th*, 1061–1065. IEEE.

Manouselis, N.; Drachsler, H.; Vuorikari, R.; Hummel, H.; and Koper, R. 2011. Recommender systems in technology enhanced learning. In *Recommender systems handbook*. Springer. 387–415.

Manouselis, N.; Vuorikari, R.; and Van Assche, F. 2007. Simulated analysis of maut collaborative filtering for learning object recommendation. In *Proceedings of the 1st Workshop on Social Information Retrieval for Technology Enhanced Learning*, 27–35.

Pelanek, R., and Jarusek, P. 2015. Student modeling based on problem solving times. *International Journal of Artificial Intelligence in Education* 25(4):493–519.

Recker, M. M., and Wiley, D. A. 2001. A non-authoritative educational metadata ontology for filtering and recommending learning objects. *Interactive learning environments* 9(3):255–271.

Shen, L.-p., and Shen, R.-m. 2004. Learning content recommendation service based-on simple sequencing specification. In *Advances in Web-Based Learning–ICWL 2004*. Springer. 363–370.

Stamper, J.; Niculescu-Mizil, A.; Ritter, S.; Gordon, G.; and Koedinger, K. 2010. Algebra i 2008-2009. *Challenge data set from KDD Cup 2010 Educational Data Mining Challenge. Find it at http://pslcdatashop.web.cmu.edu/KDDCup/downloads.jsp*.

Stern, H.; Kaiser, R.; Hofmair, P.; Kraker, P.; Lindstaedt, S. N.; and Scheir, P. 2010. Content recommendation in aposdle using the associative network. *J. UCS* 16(16):2214–2231.

Wang, R.; Chen, F.; Chen, Z.; Li, T.; Harari, G.; Tignor, S.; Zhou, X.; Ben-Zeev, D.; and Campbell, A. T. 2014. Studentlife: assessing mental health, academic performance and behavioral trends of college students using smartphones. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, 3–14. ACM.

Zhang, Y., and Liu, Y. 2010. A collaborative filtering algorithm based on time period partition. In *Third International Symposium on Intelligent Information Technology and Security Informatics (IITSI), 2010*, 777–780. IEEE.