Learner Modeling in Academic Networks

Mohamed Amine Chatti, Darko Dugošija, Hendrik Thüs, Ulrik Schroeder
Informatik 9 (Learning Technologies)
RWTH Aachen University, Germany
chatti,dugosija,thues,schroeder@cs.rwth-aachen.de

***Abstract*—Learning analytics (LA) deals with the development of methods that harness educational data sets to support the learning process. To achieve particular learner-centered LA objectives such as intelligent feedback, adaptation, personalization, or recommendation, learner modeling is a crucial task. Learner modeling enables to achieve adaptive and personalized learning environments, which are able to take into account the heterogeneous needs of learners and provide them with tailored learning experience suited for their unique needs. In this paper, we focus on learner modeling in academic networks. We present theoretical, design, implementation, and evaluation details of PALM, a service for personal academic learner modeling. The primary aim of PALM is to harness the distributed publication information to build an academic learner model.**

***Keywords*-personalization; adaptation; learner modeling; interest mining; learning analytics.**

I. Introduction

Recently, there is an increasing interest in learning analytics (LA) as an integral part of technology-enhanced learning (TEL) environments. LA is an emerging TEL research area that focuses on the development of methods for analyzing and detecting patterns within data collected from educational settings, and leverages those methods to support the learning experience. There are many objectives in LA according to the particular point of view of the different stakeholders. Possible objectives of LA include monitoring, analysis, prediction, intervention, tutoring/mentoring, assessment, feedback, adaptation, personalization, recommendation, and reflection [1].

In order to achieve particular learner-centered LA objectives such as intelligent feedback, adaptation, personalization, or recommendation, learner modeling is a crucial task. The challenge is to create a thorough learner profile that can be used to trigger effective intervention, adaptation, personalization, or recommendation actions. This is a highly challenging task since learner activities are often distributed over open and increasingly complex learning environments. The capacity to build a detailed picture of the learner activities across a broader learning context would provide a more accurate analytics results. A big challenge to tackle here is lifelong learner modeling [2]. The aim is that data gathered from different (learning) environments would be fed into a personal lifelong learner model, which would be the ”store” where the learner can archive all learning activities throughout her life.

Academic researchers are lifelong learners. And, interests that can be gathered from a researcher's publication activities are an integral part of her lifelong learner model. In this paper, we focus on learner modeling in academic networks and present the details of the personal academic learner modeling service (PALM) that enables to create an interest-based learner model from distributed publication information.

II. Learner Modeling

A learner (user) model is a representation of information about an individual learner that is essential for adaptation and personalization tasks. The five most popular and useful features in user modeling include the user's interests, knowledge, goals, background, and individual traits. In the last years, user's interests have become the most important user feature to be modeled in educational adaptive systems [3]. The user's interests are often represented in terms of a weighed vector of keywords or concepts [4].

Recently, Kay and Kummerfeld [2] stressed the need for a lifelong learner model as ”a store for the collection of learning data about an individual learner”. The authors note that to be useful, a lifelong learner model should be able to hold many forms of learning data from diverse sources and to make that information available in a suitable form to support learning. Lifelong learner modeling is thus the process of creating and modifying a model of a learner, who tends to acquire new or modify his existing knowledge, skills, or preferences continuously over a longer time span. The lifelong learning process is usually motivated by some kind of a long-term goal and may evolve by different means e.g. by education, experience, training or personal development. The continuous collection of personal data related to a learner is crucial for personalized and lifelong learning.

In this paper, we address learner modeling in academic networks. We view academic researchers as professional learners and focus on mining their interests from their publication activities in co-authorship networks. This raises the main research question:

- How can we model a professional learner in respect to his academic interests?

A learner model which incorporates the evolution of academic interests can be used to support the learner in the lifelong learning process in an academic sense by e.g. motivating the learner through visualization of his academic development. It can also be integrated into a recommender system which can automatically supply the learner with new knowledge sources. To support the learner modeling task in academic networks, we developed the personal academic learner modeling service (PALM). PALM combines web, text, and interest mining techniques in order to create a learner model based on the collected academic information. Mining the learner interests and the visualized clustering of academic publications would allow to track the evolution of the academic interests of a learner. To note, however, that a model based only on the information gathered from the authors' publications will be far from the ideal academic learner profile. Such a profile would have to keep track and contain all of the relevant academic information about a learner, including research interests in areas in which the learner never published anything.

III. Related Work

This section gives an overview of related work in this field of research. Chen et al. [5] use relevant journal articles and conference papers in TEL as data source and apply text mining techniques to automatically construct TEL concept maps. The system is based on four main steps: information retrieval from academic articles, concept item extraction, research keyword indexing, and calculation of "relation strength". Rather than extracting concept items, PALM leverages text mining and information retrieval techniques to extract academic interests.

Another way to deal with detection and extraction of user interests is presented by Gu et al. [6]. Unlike explicit interest mining from predefined sources as used in PALM, it mainly deals with implicit interest mining through behavior analysis.

FreeSearch [7] offers new methods for simple literature search that works on any catalogs, without requiring in-depth knowledge of the metadata schema. Practically, the system is a joint publication search engine that combines the data from DBLP, TIBKat, CiteSeer and BibSonomy. It features spelling suggestions, field suggestions, basic visualisation, query translation, and duplicate detection. If all resulting publications have the same author, then also some advanced visualizations are provided. This includes a tag cloud generated from the publications titles and the number of publications per year. While FreeSearch just focuses on the aggregation of publication information from various sources, the primary aim of PALM is to use the distributed publication information to extract academic interests.

IV. PALM

In the ensuing sections, we will describe PALM[1] with an eye on the architectural, implementation, and evaluation details. The system design will be followed by a detailed description of the different modules and their underlying functionalities. PALM encompasses three main modules: (1) data retrieval, (2) interest mining, and (3) visualization modules. PALM was designed to be able to:

- have access to a vast amount of online publication data,
- gather, aggregate, and mine this data to create a learner model,
- visualize the learner model, and
- share the learner model (for reuse by different adaptive and personalized applications)

A. *Data retrieval*

The first task in PALM is data retrieval. We store the author name in the local database and start the crawling process. This is done by creating a custom service that crawls through the most commonly used publication services and collects all the relevant academic publication information such as authors, titles, abstracts, keywords, and dates. The publication services used include Google Scholar Citations, CiteSeer, BibSonomy, Mendeley, dblp, and Microsoft Academic Search.

B. *Interest mining*

The second task in PALM is interest mining. Now that the publication text information is available, it needs to be normalized, combined and prepared for text mining before any further interest mining steps are taken. The underlying infrastructure must be able to support this variety and amount of data.

1) Data collection: As explained in the previous section, the publication data is retrieved from different sources. Each source has its own API and provides data in its own format. Thus, the first challenge is to normalize the data to a common format in order to make a later aggregation of the data possible. The result format contains the publications' source, title, and authors, and if available year, abstract, keywords, number of citations, and tags. As soon as the data from a certain source becomes available, PALM checks for duplicates (simple check using name and date of the publication) and stores the publication data in the database (one table per source). Once the publications are successfully stored, the second step of the process, cleaning the data, begins.

2) Data cleaning: To be able to combine the data, we must first process and prepare the input data. This includes:

- stripping the strings (removing whitespaces and tabs from start and end of the string)

[1] subprogra.informatik.rwth-aachen.de/~ddugosija/llmian/

- removing the new lines and special or undetected (non UTF-8) characters
- converting the input to lowercase
- checking if the date (year) is in correct format by using regular expression

When the data is cleaned, we are ready to extract unique publications using the combined data from all sources.

3) Data aggregation: To combine the publication data from different sources into a single unique-publication data table, we defined a *priority list* for the different used sources. Our experiments have shown that the information extracted from the Microsoft Academic Search (MAS) is of highest quality. It features least formatting errors, and the data is almost complete. The publications returned from MAS also usually contain the crucial ”keyword” elements and the abstracts are in full length. This takes MAS to the top of our *priority list* for publication information. For example, if there are two or more abstract elements for a single publication, the abstract extracted from MAS will be favored and the others will be ignored. The second place in the *priority list* is CiteSeerX, followed by BibSonomy, DBLP and finally Google Scholar Citations. The reason that Google Scholar is last on the list is that the information from Google Scholar is extracted with a HTML crawler piecewise per publication, which makes the data extracted prone to formatting errors whenever the HTML structure of the respected publication if faulty. Google Scholar also tends to shorten the publications' abstract and concatenate it with ”...” once a certain abstract length is reached.

In detail, the process of aggregating the data includes:

- preparing an empty array of combined publications (containing ”title”, ”authors”, ”year”, ”abstract”, ”keywords”, ”tags” and ”citations”);
- if the MAS information exists, fill the array with MAS data (”title”,”authors”,”year”,”abstract”, ”keywords”, ”tags”);
- for each existing publication from other sources and for each existing entry in combined publications array calculate the similarity between the titles;
- if similarity between the titles is greater then 90% assume that the publication is the same and fill in the missing publication information elements according to the *priority list*;
- otherwise assume that publications are different and add the new publication along with the respected data to the combined publication array according to the *priority list*;
- repeat until there are no new publications.

4) Term extraction: Having the combined data ready, we can start with the term extraction task. Knowing the nature and features of the data at hand always helps to choose the right method. We exploit the fact that our data is of academic origin which is much less prone to typos and other types of errors that we usually have to take into account while doing text processing. This simplifies the text mining task significantly. For the purposes of this work, we leveraged three linguistic/statistic methods that proved to produce good results. These include Yahoo Content Analysis, Topia Term Extractor, and the Topia-based service FiveFilters. In our work, combining Yahoo Content Analysis (with a high precision) and Topia Term Extractor (with a high recall) has improved the term extraction results significantly.

5) Term weighting: Combined together, Yahoo Content Analyser and Topia-based analysers deal with keyword extraction exceptionally good but with the exception of Yahoo Content Analysis, they give insufficient clues to the relevance strength of the extracted terms. To address this issue, we implemented in PALM a new custom term weighting function - $WordFreq$ that provides basic statistical information about the analysed data and calculates score for each word and each multi-word term based on normalized term frequency and different 'importance factors', such as the context of the word and the time factor.

The idea behind WordFreq is to apply a fair term weighting algorithm. The rationale of weight normalization is that the titles usually contain more important and more compact information then e.g. abstracts. And, keywords and tags are discutably closer to being identified as interests than the information from titles or abstracts. The second important factor is time, or year in which the word or multi-word term occurs. The time factor helps to keep track of the evolution of academic interests. It will also favor the more recent interests more then the interests from the past. Further, due to high precision, terms extracted by Yahoo Content Analyser are favored a bit more then the terms extracted by the Topia-based analysers. All these factors have played an important role in the overall ranking of the term extraction results in PALM.

The first step of the term weighting process with WordFreq is to split the combined publication data in year clusters. As a result, we get a set of publications published by an author for each publication year. The publications where the year information is missing are separately clustered together. After the clustering has been done, we separately concatenate the titles, abstracts, keywords and tags for each year cluster with a whitespace in between. The result is a two-dimensional array that looks as follows:

```
  publication[year0][abstract0.abstract1.abstract2
publication[year0][title0.title1.title2...]
publication[year0][keyword0.keyword1.keyword2...]
publication[year0][tag0.tag1.tag2...]
publication[year1][abstract0...]
publication[year1][title0...]
...
```

Similar to the process of combining the data to form a unique list of publications, we need to clean the input.

However, to produce one large string that consists of words only, the input cleaning is now done in a more rough way and includes:

- stripping the strings of whitespaces and tabs at the start and the end of the string;
- removing special or non-detected (non UTF-8) characters;
- removing brackets, slashes, dashes and terminating characters and replacing them with whitespaces where needed;
- lowercasing the string;
- removing stopwords defined in the *stopword list* (e.g. ”the”, ”an”, ”we”, ”this”, ”that” etc.).

As a result, we get a clean string consisting only of words separated by whitespaces. For example, a sentence like: Once the input is clean, we apply the standard Porter stemmer to return the words to their ”root” form. This is also done separately for each year cluster and separately for abstracts, titles, keywords, and tags.

The next step is to calculate distinct weight factors for abstracts, titles, keywords, and tags for each year cluster. Let

- N be the total number of publications in cluster,
- K be the number of publications in cluster that contain at least some keywords,
- T be the number of publications in cluster that contain at least some tags, and
- $maxlength = \max(abstractLength, titlesLength)$ be the length normalization variable.

The distinct factors (pro year cluster) are then calculated as:

- $abstractWordFactor = \frac{maxlength}{abstractLength}$
- $titlesWordFactor = \frac{maxlength}{titlesLength}$
- $tagsWordFactor = \frac{maxlength}{tagsLength} \cdot \frac{T}{N}$
- $keywordsWordFactor = \frac{maxlength}{keywordsLength} \cdot \frac{K}{N}$

After computing the distinct factors for abstracts, titles, keywords, and tags, we calculate weights for single-word (e.g. learning) and multi-word terms (e.g. social network analysis).

To calculate weights for single-word terms, first, we generate single-word frequency arrays for each year cluster, separately for abstracts, titles, keywords, and tags. This results in four single-word frequency arrays as follows:

```
  abstractWordFrequency[(abstractWord1,freq1),
(abstractWord2=freq2), ...]
titleWordFrequency[(titleWord1,freq3),
(titleWord2=freq4), ...]
keywordWordFrequency[(keywordWord1,freq5),
(keywordWord2=freq6), ...]
tagWordFrequency[(tagWord1,freq7),
(tagWord2=freq8), ...]
```

Then we prepare the $wordWeight$ arrays that should contain unique word/weight pairs from the four arrays for each year cluster. To do this, for each year we initiate the array with the words from the $abstractWordFrequency$ array and calculate their respective weights by multiplying the respective frequencies with the $abstractWordfactor$. Then, for each word from the other three frequency arrays:

- if the word is already in the $wordWeight$ array, we calculate the new weight of the word as: $newWeight = oldWeight + respectiveFrequency \times respectiveFactor$
- otherwise we calculate the weight of the word as: $weight = respectiveFrequency \times respectiveFactor$ and add the word/weight pair to the $wordWeight$ array.

Since the resulting $wordWeight$ arrays (one array for each year) are usually quite large, for each year cluster we only take the top 10% of the word/weight pairs having the highest weights and save them in the database.

To calculate weights for multi-word terms, we apply a simple weighting scheme to the multi-word terms extracted by the three analysers. For each $multiWordTerm$ and for each $word$ from the top 10% of the $wordWeight$ array:

- if $multiWordTerm$ contains the $word$, add the weight of the word to the $multiWordTermWeight$
- repeat until all extracted multi-word terms are weighted

Now we need to bring the time factor into consideration. For each year cluster, we compute
$timeFactor = totalYears/(currentYear + 1)$
and simply multiply the weight of the extracted terms with $timeFactor$.

For example, if the terms ”social”, ”network”, ”analysis”, ”social network”, ”network analysis” and ”social network analysis” were extracted, the algorithm would favor the multi word term ”social network analysis” rather then ”network analysis” or ”social network” since it would carry the added weight of three words instead of only two. Single-word terms are disfavored by the algorithm but they could still show up in the top results, if they are deemed significant enough. Similarly, if e.g. a term ”social network” occured more recently then a term ”technology enchanced learning”, it may weight more even though it is only a two-word term, due to the time factor.

C. Visualization

The visualization of the academic interests is done via a simple user interface. We generate a general interest cloud by taking and combining the top few (currently seven) interests per year, or display a temporal evolution of interests with multiple clouds where a cloud represents the interests in a specific year.

D. API

It is important to make it possible to reuse a learner model by different adaptive and personalized applications. PALM, thus, provides an open API to get access to the generated learner model. The response is in JSON format. There are currently functions to list all the queried authors, get their interests, list publications as extracted from the sources, as well as unique publications.

E. Evaluation

This section describes the experiments conducted with PALM. The following questions are addressed:

- How consistent are the extracted academic interests compared to those defined by authors themselves?
- How well does the service perform?

The experiment was performed through personal interviews and empirical comparison of manually and automatically extracted interests.

1) Personal interviews: Personal interviews were conducted with 7 computer science researchers working at the RWTH Aachen University. The interests extracted by PALM were presented to the authors who were asked to gauge the consistency of the results. It is important to note, that the authors were asked to approximate the overall consistency of their academic interests, including the topics in which they never published anything. The result varied between 50% and 100%, with the average of around 70%. The reported high consistency originates from authors that either just started their research career or did not change their area of interest in a longer time span. On the other hand, the low consistency can be explained by the fact that some authors have a broad spectrum of academic interests, but they did not publish much (or anything at all) in these areas.

2) Empirical comparison of results: The second method of evaluation included 22 authors of full papers at the LAK 2012 Conference [8], and consisted of an empirical comparison of the interests automatically extracted by PALM with the authors' given interests collected manually from different sources on the Web (e.g. personal webpages, blogs, LinkedIn). Thereby, the performance of PALM is measured by computing:

- *Recall*, the percentage of the extracted academic interests that are indeed correct.
- *Precision*, the percentage of the correct extracted academic interests out of all extracted academic interests.

Combining the results gathered from the set of the 22 authors, PALM achieved an average precision of 85.2% and an average recall of 78.8%.

V. Conclusion and Future Work

In learner-centered learning analytics initiatives that support e.g. intelligent feedback, adaptation, personalization, or recommendation, learner modeling is a crucial task. In this paper, we focused on learner modeling in academic networks. We presented the details of the personal academic learner modeling service (PALM) that enables to create an academic learner model from distributed publication information. The evaluation showed that PALM provides good results. Future work in PLEM includes the improvement of the weighting algorithm by taking other factors into consideration (e.g. citations count), a dashboard for the visual analytics of the generated learner model that would support awareness and self-reflection, and a recommendation module that leverages the learner model to suggest publications, researchers, or networks of interest. Future work will also include a lifelong learner modeling framework that enables to mine, aggregate, manage, and visualize learner interests from different sources, such as academic networks, social networks (e.g. facebook, Twitter, LinkedIn), learning management systems (LMS), and personal learning environments (PLE) in order to create a more complete lifelong learner model.

References

[1] M. A. Chatti, A. L. Dyckhoff, H. Ths, and U. Schroeder, "A reference model for learning analytics," *International Journal of Technology Enhanced Learning*, vol. 4, no. 5/6, pp. 318–331, 2012.

[2] J. Kay and B. Kummerfeld, "Lifelong learner modeling," in *Adaptive Technologies for Training and Education*, P. J. Durlach and A. M. Lesgold, Eds. Cambridge University Press, 2011, pp. 140–164.

[3] P. Brusilovsky and E. Milln, "User models for adaptive hypermedia and adaptive educational systems," in *The Adaptive Web, LNCS 4321*, P. Brusilovsky, A. Kobsa, and W. Nejdl, Eds. Springer-Verlag Berlin Heidelberg, 2007, ch. 1, pp. 3–53.

[4] S. Gauch, M. Speretta, A. Chandramouli, and A. Micarelli, "User profiles for personalized information access," in *The Adaptive Web, LNCS 4321*, P. Brusilovsky, A. Kobsa, and W. Nejdl, Eds. Springer-Verlag Berlin Heidelberg, 2007, ch. 2, pp. 54–89.

[5] N.-s. Chen, C.-w. Wei, and H.-j. Chen, "Mining e-learning domain concept map from academic articles," *Computers & Education*, vol. 50, no. 3, pp. 1009–1021, 2008.

[6] R. Gu and M. Zhu, "Interest mining in virtual learning environments." *Online Information Review, 32(2), 133-146*, 2008. [Online]. Available: http://www.vldb.org/pvldb/2/vldb09-98.pdf

[7] C. S. Firan, W. Nejdl, M. Georgescu, and X. Sun, "Freesearch - literature search in a natural way," in *Proceedings of the Fifth Workshop on Human-Computer Interaction and Information Retrieval*, 2011.

[8] S. Buckingham Shum, D. Gasevic, and R. Ferguson, Eds., *LAK '12: Proceedings of the 2nd International Conference on Learning Analytics and Knowledge*. New York, NY, USA: ACM, 2012.