KEYSTONE COST ACTION IC1302, SHORT TERM SCIENTIFIC MISSIONS

SCIENTIFIC REPORT

STSM Topic: Keyword-based search and keyword interpretation in recommendations systems
STSM Applicant: María del Carmen Rodríguez Hernández
Applicant's affiliation: University of Zaragoza
Applicant's address: Department of Computer Science and Systems Engineering. Building Ada Byron. Maria de Luna, 1, 50018, Zaragoza, Spain.
E-mail: 692383@unizar.es, mary0485@gmail.com

Host institute: Department of Engineering "Enzo Ferrari". University of Modena and Reggio Emilia. Avenue Vivarelli 10, 41125, Modena, Italy.
Host: Francesco Guerra, francesco.guerra@unimore.it

1. Purpose of the STSM

The purpose of this research stay has been to contribute in the field of context-based recommendations and keyword-based search in mobile environments. The main goals of research were the following:

1. Suitable keyword-based mechanisms to express the necessity of information of the user (types of items he/she is interested in).
2. Suitable techniques to enrich the information about items recommended by using external data sources (i.e., add extra information about the recommended items by querying other sources).
3. Suitable strategies to obtain quantitative ratings from textual opinions (i.e., sentiment analysis through keyword search and the interpretation of the semantics of keywords) and to find relevant opinions for a potential item to recommend.

2. Description of the work carried out during the STSM

During this STSM we work on the first two tasks of those mentioned above. We study keyword-based search mechanisms in order to express the necessity of information of the users in the query of pull recommendations, by using keywords. For this, we adapted the "Pull-Based Recommendation" module implemented in the architecture proposed in [1], with the Hidden Markov Model (HMM) and Information Retrieval (IR) techniques, by using structured external data sources. We adjusted (by using the Entity-Relation Model of the Figure 1) the datasets LDOS-CoMoDa [2] and InCarMusic [3] to two different databases.

2.1 For the application of the **HMM** [4], we planted the following problem to resolve: $P(Q|\, O, \lambda) = ?$

Given a observation sequence $\mathrm{O} = \mathrm{O}_1, \mathrm{O}_2, \ldots, \mathrm{O}_\mathrm{T}$, and the model λ, how to choose a corresponding state sequence $\mathrm{Q} = \{\mathrm{q}_1, \mathrm{q}_2, \ldots, \mathrm{q}_\mathrm{T}\}$ with the highest probability (which best explains the observations)?

For solving the above problem, we use the Viterbi algorithm [5, 6] that finds the most likely explanation for an observation sequence. For that, we consider the states Q as the

feature names that characterize to the items (e.g., artist and category, for the music domain) and the item type (e.g., film, music, restaurant, etc.). Whereas observations O are considered the values of the item types and the model $\lambda = (A, B, \Pi)$ is composed of A, B and Π:

$A = \{a_{ij}\}$, are the state transition probabilities

$B = [b_j(T)]$, are the observation probabilities

$\Pi = [\pi_i]$, are the initial state probabilities

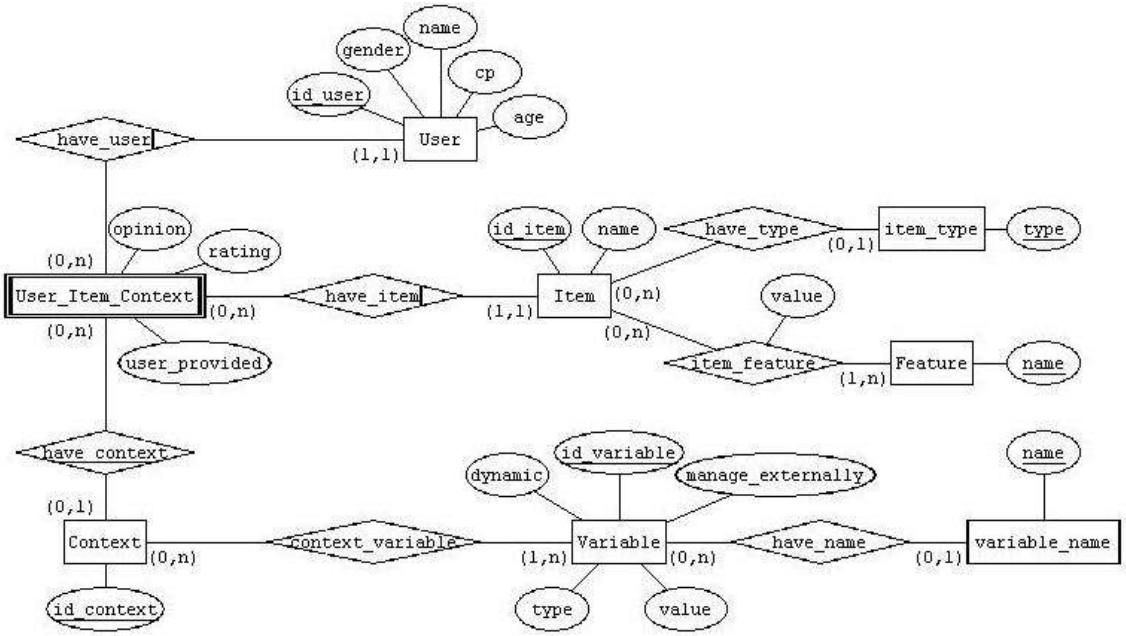

Figure 1: Entity-Relation Model for Context-Aware Recommendation System.

Below is shown a fragment of the HMM representation. Specifically, is focused in the database InCarMusic but the idea is the same for the LDOS-CoMoDa database.

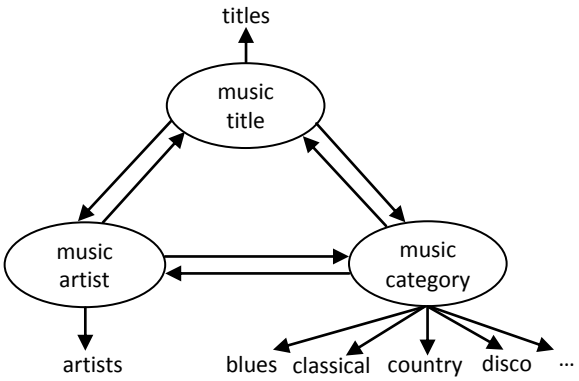

Figure 2: Representation of HMM for InCarMusic database.

The Viterbi algorithm needs as input the observations and the HMM, which are obtained from the files “hmm_model.dat” and “observations.txt”, respectively. These files are generated automatically from the database.

Specifically the “observations.txt” contains the values of the item features (e.g., artists, blues, classical, country disco, etc., for the Figure 2) and the name of the items (e.g., titles, for the Figure 2).

The “hmm_model.dat” file has a specific structure. An example of the structure of the “hmm_model.dat” file for 2 states (e.g., rainy, sunny) and 3 observations (e.g., walk, buy, clean) is displayed in the Figure 3.

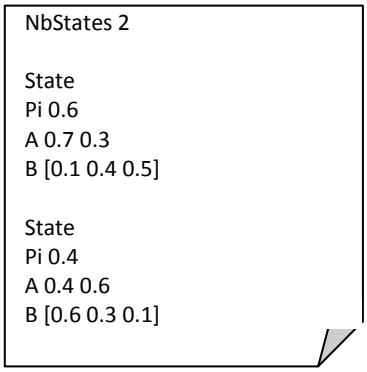

Figure 3: Example of the structure of the “hmm_model.dat” file.

In our example, the HMM is composite of 12 states (8 related to the LDOS-CoMoDa database, 3 related to the InCarMusic database, and the state “other”). The states are composite with the item type (e.g., music, for the Figure 2) and the feature names (e.g., title, artist and category, for the Figure 2). In the model λ, for each state are represented the state transition probabilities A, the observation probabilities B and y7the initial state probabilities Π.

In general, the keyword-based pull recommendation process with the current solution is represented in the Figure 4.

1. **Query:** the user introduces in the GUI the keywords as the query.
2. **Query pre-processing:** the keywords are preprocessed by using the standard analyzer of Lucene 2.4.0[1], which apply the filters “standard token” (normalizes tokens extracted), “lower case token” (normalizes token text to lower case) and “stop words” (removes stop words from token streams by using a file that contain the list of stop words). The file “observations.txt” is preprocessed too.
3. **Viterbi algorithm:** given the keywords (as the observation sequence O) and the HMM λ, allows to determine the item type (with the highest probability) that needs the user, which should be considered during the recommendation process.
4. **Filtering database:** the database data is filtered considering the item type identified in the before step. The data filtered will be used by the pull recommendation algorithm.
5. **Pull recommendation algorithm:** algorithm that allows obtaining items recommended as an answer to a query submitted by the user.
6. **Items recommended:** a list of items recommended are provided to the user.

[1] https://lucene.apache.org/

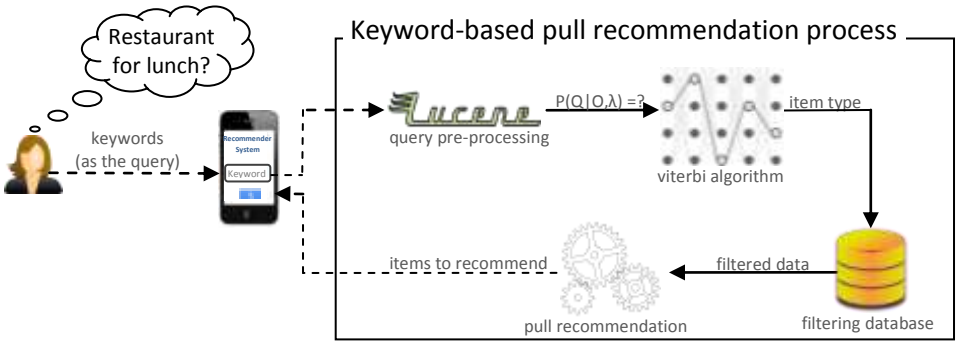

Figure 4: Keyword-based pull recommendation process by using HMM.

2.2 Another proposes of solution was the application of **traditional Information Retrieval techniques** [7, 8]. For that, we create an index of documents, where its content can be obtained from databases automatically. Considering the ER model of the Figure 1, each document is named with the item type (e.g., music) and the feature names (artist and category) or the item name (e.g., title). For example, for the database InCarMusic the documents names are "music_title", "music_artist", "music_category". The content of each document is composed of the values of the item features (e.g., the artist names and the music categories) and the item names (e.g., titles of the music). In the Figure 5, is displayed the structure of the documents to index.

Figure 5: the structure of the documents to index.

In general, the keyword-based pull recommendation process with current solution is represented in the Figure 6.

1. **Query:** the user introduces in the GUI the keywords as the query.
2. **Query pre-processing:** the keywords are preprocessed by using the standard analyzer of Lucene 2.4.0[2], which apply the filters "standard token" (normalizes tokens extracted), "lower case token" (normalizes token text to lower case) and "stop words" (removes stop words from token streams by using a file that contain the list of stop words). The file "observations.txt" is preprocessed too.
3. **Information Retrieval algorithm:** given the keywords the system search in the index the document most relevant to the query. Thus, the system knows the item type (that is the document name of the top list) that needs the user, which should be considered during the recommendation process.
4. **Filtering database:** the database data is filtered considering the item type identified in the before step. The data filtered will be used by the pull recommendation algorithm.
5. **Pull recommendation algorithm:** algorithm that allows obtaining items recommended as an answer to a query submitted by the user.
6. **Items recommended:** a list of items recommended are provided to the user.

[2] https://lucene.apache.org/

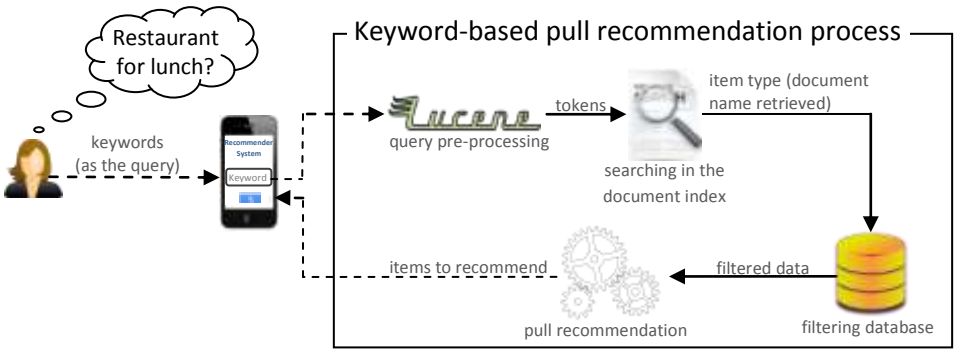

Figure 6: Keyword-based pull recommendation process by using techniques of Information Retrieval.

3. Main results obtained

The main results obtained were the implementation of two techniques for keyword-based searching in order to improve the pull-based recommendation process, implemented in the Context-Aware Mobile Recommendation System framework [1]. Moreover, the main ideas realized in the research stay were exposed in the article submitted to JISBD 2015.

4. Future collaboration with the host institution

The period of the visit was not sufficient to realize all the tasks proposed in the work plan. We will continue the researching from the University of Zaragoza with the collaboration and help of the Dr. Francesco Guerra. We will work in direction of the third task:

- Suitable strategies to obtain quantitative ratings from textual opinions (i.e., sentiment analysis through keyword search and the interpretation of the semantics of keywords) and to find relevant opinions for a potential item to recommend.

5. Publications/articles resulting from the STSM

We submitted the paper:

- "A First Step Towards Keyword-Based Searching for Recommendation Systems" to the "XX Jornadas de Ingeniería del Software y Bases de Datos" (JISBD 2015).

6. References

[1] del Carmen Rodríguez-Hernández, M., & Ilarri, S. (2014). Towards a Context-Aware Mobile Recommendation Architecture. In *Mobile Web Information Systems* (pp. 56-70). Springer International Publishing.

[2] Košir, A., Odic, A., Kunaver, M., Tkalcic, M., & Tasic, J. F. (2011). Database for contextual personalization. *Elektrotehniški vestnik*, *78*(5), 270-274.

[3] Baltrunas, L., Kaminskas, M., Ludwig, B., Moling, O., Ricci, F., Aydin, A., & Schwaiger, R. (2011). Incarmusic: Context-aware music recommendations in a car. In *E-Commerce and Web Technologies* (pp. 89-100). Springer Berlin Heidelberg.

[4] Rabiner, L. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, *77*(2), 257-286.

[5] Forney Jr, G. D. (1973). The viterbi algorithm. *Proceedings of the IEEE*, *61*(3), 268-278.

[6] Lou, H. L. (1995). Implementing the Viterbi algorithm. *Signal Processing Magazine, IEEE*, *12*(5), 42-52.

[7] Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to information retrieval* (Vol. 1, p. 496). Cambridge: Cambridge university press.

[8] Baeza-Yates, R., & Ribeiro-Neto, B. (1999). *Modern information retrieval* (Vol. 463). New York: ACM press.