A Case-Based Recommendation Approach for Market Basket Data

Anna Gatzioura and Miquel Sànchez-Marrè, *Universitat Politècnica de Catalunya · BarcelonaTech*

Recently, recommender systems have become an important part of various applications. A novel case-based recommendation approach aims to overcome the current limitations while providing more insight into user preferences and item selection patterns.

Recommender systems (RSs) have been used in numerous domains to support both users in handling information overload and finding adequate items to cover their needs and providers in identifying user needs and increasing the amount and diversity of the items they sell.[1]

The most widely used recommendation techniques tend to recommend similar items to those liked by a user in the past or items that similar users liked based on the hypothesis that users have a stable behavior over time and that similar users share similar tastes. However, these methodologies show limited performance when new users or less popular items appear—often called a *cold start*—while others can lead to overspecialization of the recommended items. In addition, because they tend to ignore the underlying structure of user preferences and don't evaluate the hidden concepts under which items are selected, their accuracy decreases when trading items with unequal probability distributions.

Previous research, mainly in the fields of market research and customer behavior analysis, has revealed the existence of patterns that determine the structure of users' market baskets, which is defined as the set of items bought by one customer in a single visit to a store. Market basket analysis (MBA) refers to the search for meaningful associations in customer purchase data, and it aims to discover the patterns that define the composition of those baskets.[2] The market basket domain is characterized by the existence of a large number of items and transactions that consist of co-occurring items. More than simply predicting whether a single item will be liked by a user, the intention is to capture the presence or absence of an item within a concrete buying concept and to become able to recommend complementary items. Most of the current recommendation methodologies don't take into account these co-occurrences, and the association rules (ARs) methodology that has been used as a basis for recommendations in such cases has several limitations—such as computational and evaluation difficulties—when applied to large datasets.[3] There's a need for intelligent recommendation methodologies that can generate valuable recommendations based on user

1541-1672/15/$31.00 © 2015 IEEE
Published by the IEEE Computer Society

habits and purchase patterns, yet that overcome the drawbacks of the current recommendation techniques.

Here, we present an analysis and recommendation approach for situations in which a user's experience depends on the set of items selected together more than on each item's stand-alone attributes. Case-based reasoning (CBR) is particularly appropriate because the sets of items selected together can be adequately modeled and compared. In addition, the performance of a case-based reasoner benefits from the existence of a large amount of data, such as in the MBA domain. To determine the structure of user preferences and generate valuable recommendations, the implemented methodology uses a hierarchical categorization of items in transactions. By evaluating sets of selected items, the system can identify those items selected within concrete concepts and recommend them in similar situations to new or existing users.

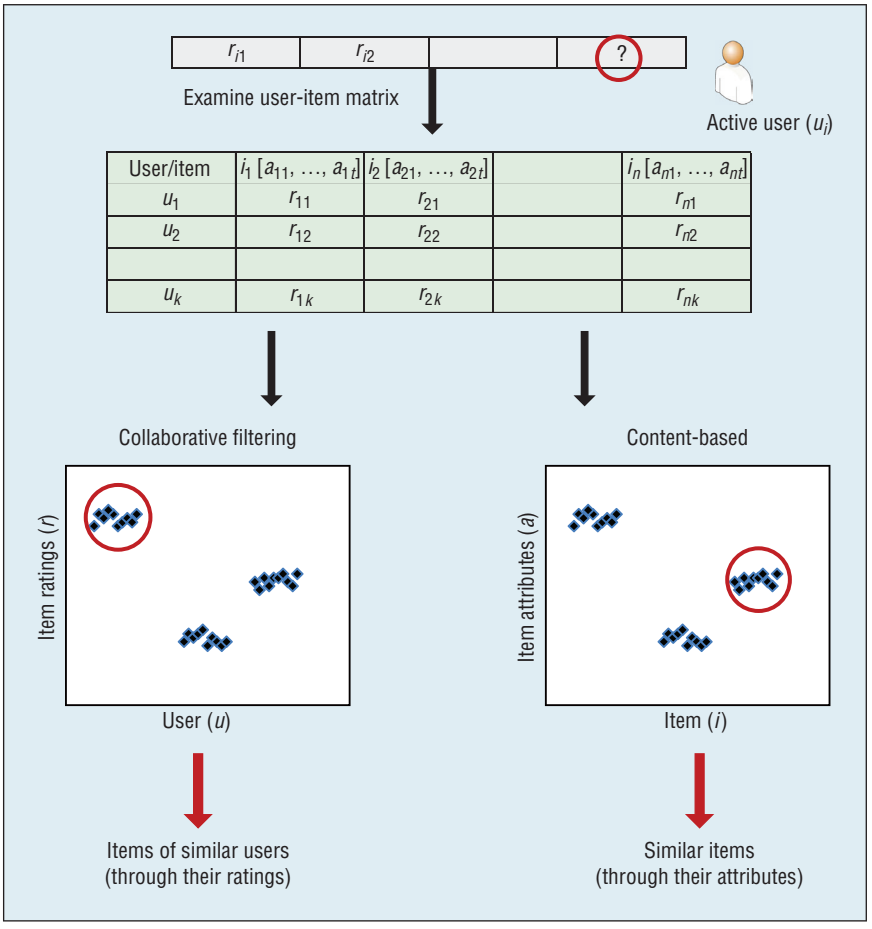

Figure 1. Collaborative filtering and content-based recommendations. The main differences are in the way that these techniques explore the user preference matrix to extract information about user selections and generate recommendations. CF groups users based on the similarity of their ratings while CB groups items based on the similarity of their characteristics.

Recommender Systems and Recommendation Techniques

RSs are software tools and techniques for information retrieval and filtering that aim to provide meaningful and effective item recommendations to the active user.[4] The term *item* refers to the type of entity (product, service, information, and so on) being recommended; it depends on the application area and on the specific system's objectives. RSs usually generate a set of (top-*N*) items expected to be liked by the user or intend to predict whether a specific item will be of interest.

The widely used recommendation methodologies in commercial applications can be mainly divided into collaborative filtering (CF) and content-based (CB). Figure 1 illustrates their main differences.

In CF recommendation techniques, items among those liked by similar users ("neighbors") are recommended to the active user. A user profile is built of the items that the user has rated highly, thus similarities in user tastes are deduced from previous ratings. Although widely used in commercial applications, collaborative RSs still have to overcome scalability and cold-start problems that limit their performance.[5]

On the other hand, in CB techniques, user profiles are built from the characteristics of the items that a user has rated highly, and the items that he or she hasn't yet tried are compared against them. The items with the higher estimated possibility of being liked are then recommended.[4,6] Because CB techniques rely on more specific information about users and items, they're able to recommend new items. However, they must overcome the recommendations' limited diversity and possible overspecialization.

Various hybrid approaches have been proposed to leverage the strengths of both techniques, overcome their current limitations, and improve their recommendation accuracy.[1,6] The following are the basic recommendation techniques identified in the literature[1]:

- CF (memory-based, model-based),
- CB filtering (neural networks, probabilistic models, naïve Bayes classifier),

Related Work in Case-Based Reasoning and Recommenders

Case-based reasoning (CBR) is a problem-solving paradigm closely related to the human way of reasoning and acting in everyday situations when facing new problems. CBR uses old experiences to solve new problems, based on the following sentence, known as the CBR assumption: "Similar problems have similar solutions."

A situation experienced in the way that it has been captured and learned is referred to as *past/previous case* and is stored in the case base. A new situation asking for a solution forms the description of a *new/target case*. An important part of the CBR methodology is its learning ability, which comes as a natural result of its problem-solving process: the case base is updated each time a new experience is obtained. This knowledge can be reused when needed without implementing the whole process from scratch, or to highlight a methodology that should be avoided in a similar situation. Therefore, case-based reasoners are able to improve their problem-solving performance over time.[1]

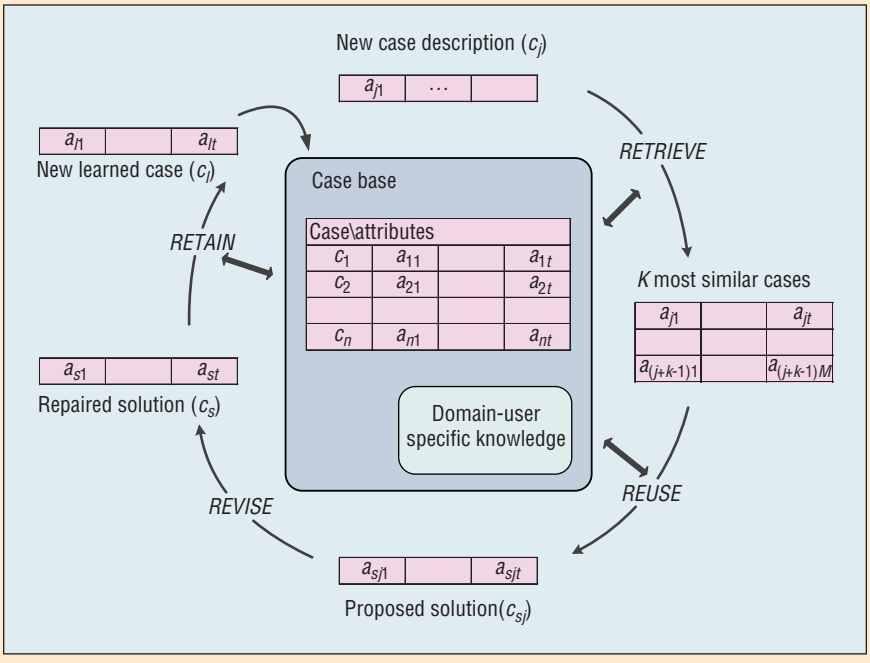

Figure A. Case-based reasoning (CBR) cycle. A cyclical process comprising of four processes (retrieve, reuse, revise, retain) that retrieves from the case base the most similar cases to the new case and adapts their solutions to the needs of the current problem to find an adequate solution.

The CBR solving and learning process can be described as a cyclical process comprising four processes, known as the CBR cycle (shown in Figure A), or "the four Rs":[2]

- *retrieve* the most relevant cases,
- *reuse* the knowledge provided to the new problem,
- *revise* the solution obtained, and
- *retain* the parts of the new solution that are likely to be used for future purposes.

In addition to the knowledge obtained from previous cases, there's also domain-dependent knowledge supporting the CBR process.

Cases can be viewed as composed of two parts: the problem description and the problem solution. Thus, cases with similar descriptions are retrieved and their solutions

- knowledge-based (case-based, constraint-based),
- utility-based,
- rule-based,
- demographic, and
- other (context-aware, semantically enhanced, hybrid).

Knowledge-based and especially case-based recommenders have emerged as the primary alternative to CF recommenders, intending to overcome their shortcomings while efficiently handling the existing information overload. Case-based recommenders implement a type of CB recommendation that relies on a structured representation of cases, usually as sets of well-defined characteristics with their values.[7] These systems generally recommend items similar to those that the active user has described in his or her request (see the "Related Work in Case-Based Reasoning and Recommenders" sidebar).

Rule-based techniques generate item recommendations based on a set of rules extracted from a data corpus. ARs mining refers to transaction analysis aiming to discover interesting hidden patterns and frequent associations among existing items, usually expressed in the form of "if-then" statements.[3]

Recently, semantic analysis, latent factors, and probabilistic topic models arising from natural language processing have been successfully applied to information retrieval and RSs, especially for tag recommendations. The basic idea is that topics are sets of words from a given vocabulary, and documents are formed as probability distributions over topics. These techniques show higher

are adapted to the needs of the target problem.[1] The existence of a common and structured representation of the treated items enables case-based recommenders in calculating the similarities and generating meaningful recommendations of high quality—especially for new items or to new users. Let P, Q be the subsets of problem descriptions and solutions, respectively, then cases can be denoted as ordered pairs $c = (p, q)$, where $p \in P$ and $q \in Q$, while the case base in a CBR system can be defined as the set of the known cases, $C = (P, Q)$.

The key idea behind CBR is that similar problems have similar solutions. Therefore, a core concept of the CBR methodology—and a key factor of its successful application—is the similarity measure used to identify similar cases. These cases can have the same solution, or their solution can be easily adapted to match the current problem's characteristics. Case-based recommenders follow the general CBR cycle and rely on the CBR core concepts of similarity and retrieval. For a case-based recommender, the user query serves as a new problem specification, while the available items and their descriptions form the cases in the case base. The items to be recommended are retrieved, based on their similarity to the user's request.[3]

Let the description of a case with t attributes be $c[a_1, \ldots, a_t]$. To calculate the global similarity between two cases, first the local similarities of the different attributes have to be specified. Usually, the global similarity can be calculated as the aggregation of the local similarities from the following equation,[4]

$$Sim\left(c^I, c^R\right) = \frac{\sum_{i=1}^{n} w_i \times sim\left(a_i^I, a_i^R\right)}{\sum_{i=1}^{n} w_i},$$

where a_i^I, a_i^R are the values of the ith attribute in the input c^I and the retrieved c^R cases, respectively, $sim\left(a_i^I, a_i^R\right)$ is the local similarity function used for their comparison, and $w_i \in [0,1]$ is the corresponding weighting factor. Depending on the system's purpose and the type of recommended items, different local similarity functions are used, while the weighting factors may be specified by feature weighting algorithms or by users as expression of their preferences.

CBR recommenders have been mainly applied in product recommendations, especially in electronic stores, to support intelligent product selection by identifying the products that best match a user's request.[3] These systems focus on the processes of mapping user requirements into a proper problem specification, selecting and retrieving items based on their similarity to the user's request, and their presentation, as there are few stores that enable product customization. CBR recommenders have also been applied in travel recommendations for both travel services (hotels, museums, and so on) and complete travel plans.[5] In addition, due to their case modeling and the higher flexibility offered to decision-making processes, CBR systems can be used in applications where contextual and other user-specific information related to the time of the selection must be incorporated. Another interesting application of CBR is related to music playlists generation. A playlist is a coherent collection of music items of specific characteristics presented in a meaningful sequence, thus modeling entire playlists as cases enables identifying the relevance of songs based on their co-occurrences.[6]

References

1. R. Bergmann, *Introduction to Case-Based Reasoning*, Univ. Kaiserlautern, 1998.
2. R. L. de Mántaras, "Case-Based Reasoning," *Machine Learning and Its Application*, vol. 2049, 2001, pp. 127–145.
3. F. Lorenzi and F. Ricci, "Case-Based Recommender Systems: A Unifying View," *Intelligent Techniques for Web Personalization*, Springer, 2005, pp. 89–113.
4. G. Finnie and Z. Sun, "Similarity and Metrics in Case-Based Reasoning," *Int'l J. Intelligent Systems*, vol. 17, no. 3, 2002, pp. 273–287.
5. F. Ricci et al., "ITR: A Case-Based Travel Advisory System," *Case-Based Reasoning*, Springer, vol. 2416, 2002, pp. 613–627.
6. C. Baccigalupo and E. Plaza, *Case-Based Sequential Ordering of Songs for Playlist Recommendation*, LNCS 4106, T. Roth-Berghofer, M.H. Göker, and H.A. Güvenir, eds., Springer, 2006, pp. 286–300.

accuracy than rule-based options and are able to better handle sparsity problems.[8]

Due to the evolution of mobile devices and the use of recommender systems in applications highly depended on context (location, time, weather, movement state, emotional state, and so on), context-aware recommenders are receiving more attention. They involve much more than grouping users or items based on their ratings or characteristics—instead, they group users or items associated with similar context information.[9]

Approach Description

The majority of recommendation techniques ignore the fact that, in many situations, the utility a user obtains depends on the set of items selected together more than on the selected item's isolated attributes. Sometimes these "relations" among items seem obvious, while in other situations, they must be inferred from the underlying patterns.

CF techniques evaluate only the ratings assigned by users to items, whereas CB techniques and the majority of case-based recommenders focus on the characteristics of items that a user has liked or requested. The commonly used approach in MBA—the ARs mining methodology—evaluates the presence or absence of items within a transaction to extract patterns and generate recommendations based on them. Recently, latent semantic analysis and probabilistic topic models have been applied to RSs for recommendations on sets of items, evaluating the similarities of latent topics to recommend those items.

Our proposed recommender uses case-based reasoning (CBR) to identify and recommend the items that

seem more suitable for completing a user's buying experience provided that he or she has already selected some items. The system models complete transactions as cases and recommended items come from the evaluation of those transactions. Because the cases aren't restricted to the user who purchased them, the developed system can generate accurate item recommendations for joint item selections, both for new and existing users. Having analyzed the previous transactions and identified the concepts within which concrete items appear, the given part of a new transaction is matched over the existing ones to find the more adequate solution—that is, the best way to fill this basket.

Following the formalization used in MBA, let $I = \{i_1, i_2, \ldots, i_l\}$ be the set of l distinct elements called items that can be found in a database. Let D be a transactional database that contains a set of transactions $T = \{t_1, t_2, \ldots, t_m\}$, where each transaction $t_j = \{i_a, i_b, \ldots\}$ contains a subset of items from I that were purchased together at a specific period of time by a user from the set of users $U = \{u_1, u_2, \ldots, u_k\}$. In contrast to most CBR item recommenders[7,10] that trade items as cases and their attributes form the case description $i[a_1, \ldots, a_t]$, we model every transaction as a case and its items as attributes of the case $c[i_1, \ldots, i_j]$, to capture the sets of items selected together. A case can be denoted as an ordered pair $c = (p, q)$, where $p = \{i_{i1}, \ldots, i_{ij}\} \in P$ is the problem description (the set of already selected items) and $q = \{i_{il}, \ldots, i_{i(l+n-1)}\} \in Q$ is the problem solution of size n (the set of items that will complete this transaction), both subsets of I with $p \cap q = \varnothing$. The case base $C = (P, Q)$ is the set of transactions that have been performed, where all the included items are known.

The treated items aren't (directly) associated with quality attributes that would either enable a content-based comparison or affect their acceptance by users. Therefore, we view all the items as attributes of the same importance for the user's buying experience. The global similarity of two cases is the aggregated similarity of the local similarities of the items included in them with equal weighting factors, therefore

$$Sim\left(c^I, c^R\right) = \sum_{i=1}^{t} sim\left(i_i^I, i_i^R\right).$$

Currently, a lot of alternative items/services able to cover the same, or very similar, needs can be found, with their differences mainly encountered in marketing characteristics rather than in core specifications. In addition, although the number of registered users is usually high, each one experiences only a small percentage of the available items. Consequently, the resulting buying patterns are sparse and may represent situations that occur by chance. To identify user habits, it's important to identify the specification of an item that appears within a concrete concept. For instance, the set of items bought together before a football game, such as soda, beer, chips, and so on, can be seen as the "snacks for watching TV" concept.

A preprocessing phase that analyzes items and transforms them to a convenient representation for further comparison is important. More than classifying items based on the exact values of their characteristics or the ratings assigned to them, our approach categorizes items based on the types of their hierarchical attributes (the categories that they belong to). As the tree structure in Figure 2 shows, starting from the generic item concept, we specify at each level one more category that an item belongs to, dividing the existing sets of items into smaller, more coherent subsets. As we go lower in the tree, these sets get smaller until we reach the leaves with the unique items. At each level, the items that have common ancestors and aren't differentiated at the current level are grouped together and treated as the "same." For instance, the items regarded as being the "same" item at the third level are the items belonging to the same categories at the first two levels with the same characteristics at the third level. So, in a physical store at the second level, all types of milk would be grouped together, while at the third level, based on the type of milk, full and semi-skimmed milk will be distinguished; at the forth level, we'll have distinctions based on the distribution package, and the brand name will specify the unique item.

Because meaningful information about the items can't be extracted from their initial unique IDs, the new groups of items are labeled with appropriate "IDs/tags" to enable their comparison. These IDs are generated based on the categories that the items belong to, following the path from the tree root to its position. Having specified the level of detail needed, the required IDs are generated as in Figure 2; the similarity of items in the new and existing cases can be calculated based on those IDs. Two items i_i^I, i_i^R from the input and a retrieved case are thought to be similar to the extent to which they share the same path from the tree root to their position. Thus, their level of similarity can be calculated from the following equation:

$$sim\left(i_i^I, i_i^R\right) = \frac{LengthOfCommonPath\left(i_i^I, i_i^R\right)}{level\left(i_i^I\right)}.$$

Depending on the application domain, the hierarchical attributes and their classification may be extracted

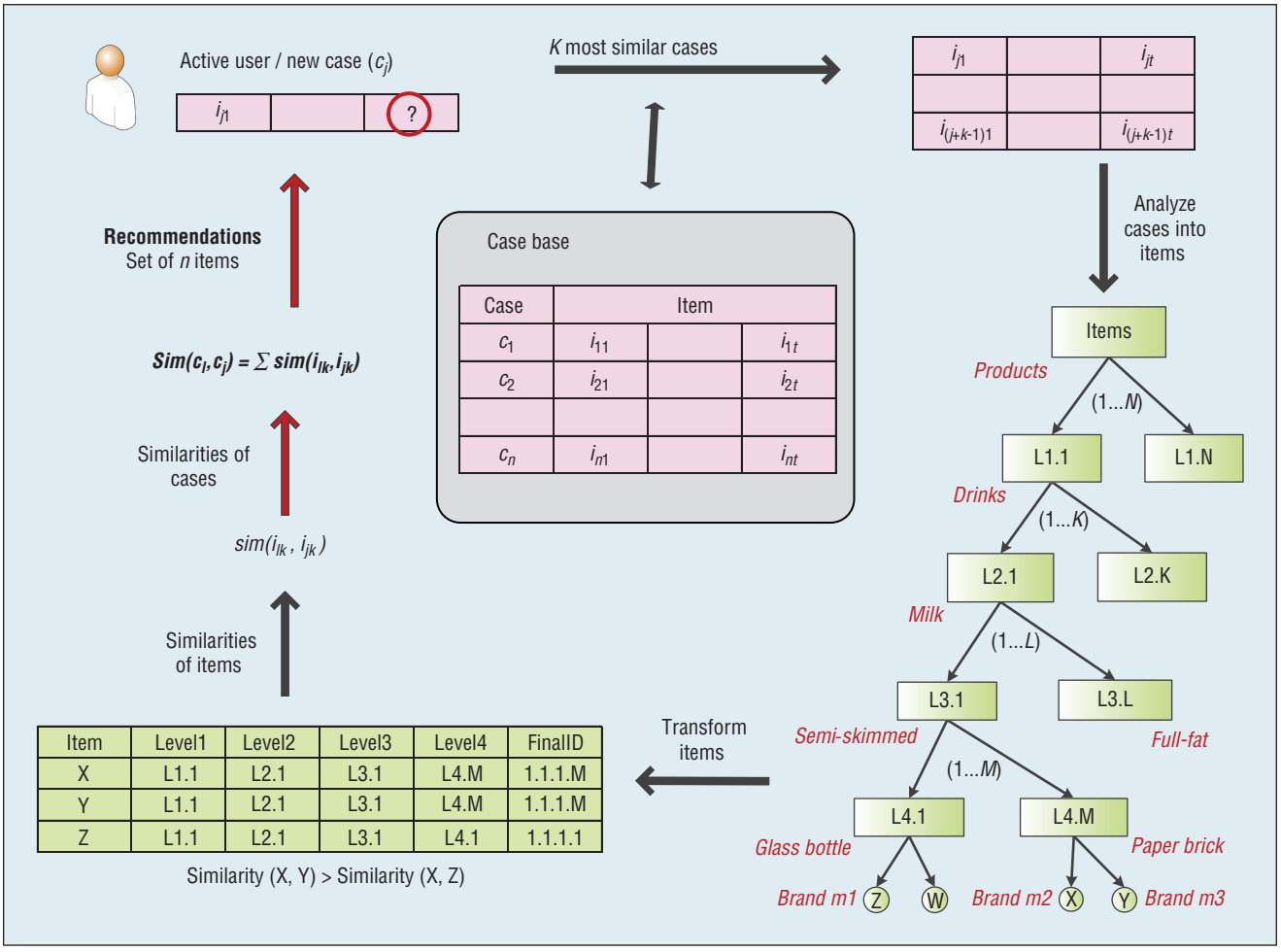

Figure 2. Recommendation process. The similarity of cases is calculated as the aggregated similarity of the items these cases are composed of, while the items are compared based on their hierarchical characteristics. Following the general CBR cycle, the most similar cases are retrieved and the sets of items completing these cases form the set of recommended items.

from a proper ontology or a simpler categorization. The described classification of items forms part of the recommendation preprocessing phase and is used to calculate similarities between cases and to highlight the type of items appearing in different cases. Nevertheless, the implemented recommendation methodology doesn't generate recommendations based on a clustering of items.

When a new user comes, the set of already selected items is compared to those in the existing cases, to identify past cases with similar descriptions and the way they were structured. The k most similar cases are retrieved, and the items that most frequently appear in their solution parts are recommended to the user. Therefore, even if the new case belongs to an unknown user, or if it contains items that appear only under certain circumstances, the recommender is able to generate accurate recommendations.

Experimental Results and Evaluation

To evaluate the implemented recommender, we used a transactional dataset with real market data from a European supermarket. The number of offered items was 102,142, with 1,057,076 transactions performed by 17,672 customers. Each transaction is associated with the user who purchased it and the included items. Each item, apart from its name and unique ID, is associated with the various categories that it belongs to (general category, item group, and two item subgroups that, for example, would be, drink, milk, semi-skimmed, glass bottle of 1 liter, brand name). This information was used to transform the item representation into a proper depiction, as described above, before generating recommendations. Demographic and subscription information about users can also be found in this dataset.

The recommender was tested for different values of parameters that affect its performance, such as the

Table 1. Recommendation results in terms of precision (Pr), recall (R), and f-measure (F1) for the use of level 2 and level 3 item representations.

No. items	Association Rules			Latent Dirichlet Allocation			Collaborative Filtering			Case-Based Reasoning		
	Pr	R	F1	Pr	R	F1	Pr	R	F1	Pr	R	F1
Level 2												
5	0.226	0.089	0.128	0.418	0.084	0.139	0.242	0.205	0.222	0.446	0.177	0.253
7	0.282	0.080	0.125	0.493	0.099	0.164	0.308	0.252	0.277	0.554	0.206	0.300
9	0.337	0.075	0.123	0.531	0.106	0.177	0.366	0.296	0.327	0.629	0.226	0.333
11	0.390	0.071	0.120	0.592	0.118	0.197	0.417	0.321	0.363	0.678	0.239	0.353
Level 3												
5	0.137	0.040	0.062	0.086	0.017	0.029	0.072	0.064	0.068	0.318	0.110	0.163
7	0.168	0.038	0.062	0.135	0.019	0.034	0.093	0.082	0.087	0.445	0.154	0.229
9	0.198	0.037	0.062	0.190	0.021	0.038	0.114	0.099	0.106	0.519	0.176	0.263
11	0.225	0.036	0.062	0.197	0.018	0.033	0.140	0.119	0.129	0.570	0.185	0.279

number of similar cases retrieved (k), the number of recommended items, and the level of detail at which the items are represented. To specify the number of similar cases that would be used, recommendations were generated using 1, 5, 10, 20, and 30 similar cases. The experimental results show that the best option is to use only the most similar case, thus k was set equal to 1.

We compared the performance of the developed recommender system to three of the widely used techniques, namely, AR, probabilistic topic models, and CF. However, only the first two techniques address exactly the same problem: recommendations of sets of items. CF recommendations focus on the ratings users assign to items and don't take into account joint item selections. But because the CF technique is widely used in commercial applications, its results are also presented. Item descriptions in the transactional database don't contain quality attributes, so we didn't use a CB or a CBR item-based approach. The Apriori algorithm was used to extract ARs from the given transactional database, based on which the recommendations were generated. In addition, we used a topic model recommender (Latent Dirichlet Allocation, or LDA). In this approach, the offered items are seen as words of a vocabulary, and transactions are treated as documents formed from a combination of topics (item concepts) with some probability distribution.[11] In the CF approach, item selection means the item has a high user rating.

We ran various experiments, randomly selecting each time 20 percent of the transactional database for new cases (test set) and the rest as the case base (training set). Using only the most similar case ($k = 1$), Table 1 shows the average results for the recommendation of 5, 7, 9, and 11 items for different representation levels. Because our intention was to evaluate the recommender's ability to identify and recommend the missing items in the transactions, information retrieval metrics (precision, recall, and f-measure) were used for the evaluation, with our focus being on the precision value.

As you can see, the accuracy of all the methodologies highly improves when using more abstract descriptions. However, as the CBR recommender evaluates the degree of an item's similarity with the items in the target case and not just an item's presence or absence, it outperforms the other recommenders at both representation levels. In contrast, the LDA recommender evaluates similarities among item concepts (topics), while the ARs recommender evaluates only the presence or absence of items within transactions to extract the buying patterns and generate recommendations. Finally, CF recommenders take into account only the presence of items in the user profiles without evaluating the items' co-occurrences within transactions.

Recommender systems have become an important part of numerous commercial applications, enabling customers and providers in their decision-making processes while pursuing their buying and selling strategies. The identification of item selection patterns is thus of high importance. One of our approach's main advantages is its ability to recommend complementary items to the already selected ones. Additionally, it can recommend less popular items and generate recommendations to new users, reducing the cold start and the overspecialization problems of CF and CB techniques.

This work could be further extended by incorporating a second processing level into the recommendation methodology. At this level, additional quality characteristics of the items and constraints related to them could be incorporated (for example, to recommend only new items). In addition, it could be

applied to other domains where the outcome of a user's experience depends on the total set of items used together. Finally we plan to examine the temporal characteristics of the selected items. ■

References

1. F. Ricci, L. Rokach, and B. Shapira, "Introduction to Recommender Systems Handbook," *Recommender Systems Handbook*, F. Ricci et al., eds., Springer, 2011, pp. 1–35.
2. L. Cavique, "A Scalable Algorithm for the Market Basket Analysis," *J. Retailing and Consumer Services*, vol. 14, no. 6, 2007, pp. 400–407.
3. R. Agrawal, T. Imielin´ski, and A. Swami, "Mining Association Rules between Sets of Items in Large Databases," *ACM SIGMOD Record*, vol. 22, 1993, pp. 207–216.
4. P. Melville and V. Sindhwani, "Recommender Systems," *Encyclopedia of Machine Learning*, Springer, 2010, pp. 829-838.
5. Z. Huang, D. Zeng, and H. Chen, "A Comparison of Collaborative-Filtering Recommendation Algorithms for E-commerce," *IEEE Intelligent Systems*, vol. 22, no. 5, 2007, pp. 68–78.
6. G. Adomavicius and A. Tuzhilin, "Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions," *IEEE Trans. Knowledge and Data Eng.*, vol. 17, no. 6, 2005, pp. 734–749.
7. F. Lorenzi and F. Ricci, "Case-Based Recommender Systems: A Unifying View," *Intelligent Techniques for Web Personalization*, Springer, 2005, pp. 89–113.
8. M. Steyvers and T. Griffiths, "Probabilistic Topic Models," *Handbook of Latent Semantic Analysis*, vol. 427, 2007, pp. 424–440.
9. J.H. Su et al., "Music Recommendation Using Content and Context Information Mining," *IEEE Intelligent Systems*, vol. 25, no.1, 2010, pp. 16–26.
10. D. Bridge et al., "Case-Based Recommender Systems," *Knowledge Eng. Rev.*, vol. 20, no. 3, 2006, pp. 315–320.
11. K. Christidis, D. Apostolou, and G. Mentzas, "Exploring Customer Preferences with Probabilistic Topic Models," *Proc. European Conf. Machine Learning*, 2010; www.ke.tu-darmstadt.de/events/PL-10/papers/3-Christidis.pdf.

THE AUTHORS

Anna Gatzioura is a PhD student at the Universitat Politècnica de Catalunya · BarcelonaTech. Her research interests include intelligent decision support systems, recommender systems, machine learning, and artificial intelligence applications. Contact her at gatzioura@cs.upc.edu.

Miquel Sànchez-Marrè is an associate professor in the computer science department at Universitat Politècnica de Catalunya · BarcelonaTech. His research interests include case-based reasoning, machine learning, data mining, intelligent decision support systems, integrated architectures for AI, and applications of IDSS. Sànchez-Marrè has a PhD in computer science from the Universitat Politècnica de Catalunya · BarcelonaTech. Contact him at miquel@cs.upc.edu.