"I would like to watch something like 'The Terminator'…" Cooperative Query Personalization Based on Perceptual Similarity

Christoph Lofi
Technische Universität Braunschweig
Mühlenpfordtstraße 23
D-38106 Braunschweig
lofi@ifis.cs.tu-bs.de

Christian Nieke
Technische Universität Braunschweig
Mühlenpfordtstraße 23
D-38106 Braunschweig
nieke@ifis.cs.tu-bs.de

ABSTRACT

In this paper, we showcase a privacy-preserving query personalization system for experience items like movies, music, games, or books. Personalizing queries for such items is notoriously difficult as meaningful query attributes are either missing in the database or would require extensive domain knowledge not available to most users. For this reason, state-of-the-art content provision platforms as e.g., Netflix or Amazon usually rely on recommender systems to support their users, and are often working in parallel with traditional SQL-style queries. Unfortunately, recommender systems have several shortcomings as for example high barriers for new users joining the system, which first have to setup a preference profile in a lengthy process, the inability to pose meaningful queries beyond recommendations matching the personal profile, and severe privacy concerns due to storing personal rating data for all users long-term. In order to provide an alternative, we present in this demonstration paper a powerful and intuitive query-by-example (QBE) interaction system. Bayesian Navigation is used to personalize a user's query on the fly. The central challenge when using QBE is the selection of features to represent the items in the database. Here, we rely on a high-dimensional feature space which was mined from rating data of a large number of users, allowing us to measure perceived similarity between items to steer the query process. This also addresses many issues of recommender systems as our query capabilities can be used by any user anonymously in a drive-by fashion. In our proposed demo, users can try our never before presented system hands-on, and can use it to discover interesting movies tailored to their preferences with a pleasantly simple and enjoyable user experience.

Categories and Subject Descriptors

H.2.4 [Data Management]: Systems, Query Processing

Keywords

Personalization, Privacy, Perceptual Similarity, User Modeling, Query-By-Example, Bayesian Navigation.

1. INTRODUCTION

Effective personalization techniques have grown to be an integral and indispensable part of current information systems, and are essential to support users when faced with a flood of different choices. Here, two major approaches are common: a) Using SQL-style personalized queries on meta-data, which however require the users to have extensive domain knowledge in order to formulate precise and efficient queries. Additionally, SQL-style queries are difficult for the large domain of *experience items* like movies, books, music, or games, as the commonly available meta-data is often not describing the items in a suitable fashion (e.g. if they are funny, but with a dry humor, not slapstick). b) Adapting recommender systems which proactively suggest items to users based on their user profile, and which became particularly popular in systems like Amazon or Netflix [1], [2]: While many recommender systems provide recommendations of high quality [3], they have several shortcomings. Especially, for *each user* an elaborate user model needs to be built and stored, requiring up to hundreds of ratings until a user can effectively get meaningful recommendations. This creates a high barrier for new users to join the system. But more severely, this user model contains exhaustive personal information on a user's preferences, her reaction to different items, or her general likes and dislikes. In order to query or use the system, this information must be *clearly associated* with the respective user and needs to be *stored long-term* for the system to work. Such profiles are highly valuable, and can be commercialized, abused, or even stolen. Obviously, this situation raises many privacy concerns and repels privacy-conscious users.

In this proposed demonstration, we therefore present an alternative approach fusing advantages of both recommender systems and SQL-based database personalization techniques, while at the same time avoiding many of the associated privacy risks. We realize this with privacy-preserving *query-by-example personalization*, which allows users to query for items fitting their current preferences easily without providing explicit feedback on attributes or their values. To achieve this, we utilize the *perceived similarity* between given items, which is harvested from user-item ratings, and transformed into a *perceptual space* [4]. However, we avoid the drawbacks of recommender systems: no user profiles are necessary to query the system, allowing situative, personalized, and anonymous ad-hoc queries.

2. SYSTEM DESIGN & FOUNDATIONS

In this section, we briefly outline the general design of our system, and provide some high-level insights into the theoretical foundations. This demonstration proposal is based on the work in [5]

10.5441/002/edbt.2015.51

where all theoretical foundations as well as the resulting performance evaluations are discussed in detail.

Basically, our approach allows users to easily explore a database with experience products by using personalized and privacy preserving query-by-example (QBE) navigation. In this proposed demonstrator, the database will contain around 12,000 movies. Users start the system interaction by providing an example of what they are currently looking for, e.g. "I enjoyed 'The Terminator', and I am now looking for a similar movie." Then, users are presented with a *display* of different items, and can provide a *feedback action* for some or all of them, resulting in a new display. This process continues until the user is satisfied with the results (a screenshot of our demo prototype is shown in Figure 1).

While this workflow could be achieved by simple similarity search, we strive to personalize this query process. Therefore, we build a disposable user profile which is only valid during the query's runtime, and the next display depends on the history of all feedback actions in the current query session instead of only the last feedback. Therefore, for two users, the same feedback action on one screen can result in different displays depending on their previous feedback within that query.

Accordingly, three major challenges are discussed in this section:

a) How can experience items as for example movies, books, music, games, software, or even hotels or restaurants be represented in a high-dimensional feature space such that meaningful similarity measurements and QBE navigation are possible?
b) How can we personalize an example-based query in such a way that it respects the user's feedback actions?
c) Which implications does such a system have on the user experience, and how does our approach compare to SQL-style systems and recommender systems?

Most popular QBE approaches in multi-media databases tried to operate on features generated from the actual multimedia file itself, which could either be low-features (e.g. color histograms or pattern-based features), or so-called high-level features as for example in scene composition [6] or content-based semantic features [7] (e.g., presence of explosions, or a mountain, or a flag, etc.) Here, our approach takes a completely different route, as our feature space results from external user ratings instead of being extracted from the media.

Information mined from user ratings has been shown to be very informative, and semantically more meaningful to users than traditional meta data as, e.g. information about the director or actors (as shown in e.g. [8] for movies). In our prototype, we demonstrate how such semantically rich rating data can represent each item of an experience product database within a high-dimensional feature space. The idea is that the resulting space implicitly encodes how users perceived a movie, e.g., if it was funny, or if certain plot elements or tropes were present. For this task, we adapt *perceptual spaces*. Perceptual spaces have been introduced in [4], and are built on the basic assumption that each user who provides ratings on items has certain personal interests, likes, and dislikes, which steer and influence her rating behavior [9]. For example, with respect to movies, a given user might have a bias towards furious action; therefore, she will see movies featuring good action in a slightly more positive light than the average user who doesn't care for action. The sum of all these likes and dislikes will lead to the user's overall perception of that movie, and will ultimately determine how much she enjoyed the movie, and how she rates it on a social movie site. Moreover, the rating will share this bias with other action movies in a systematic way. Therefore, one can claim that a perceptual space captures the "essence" of all user feedback, and represents

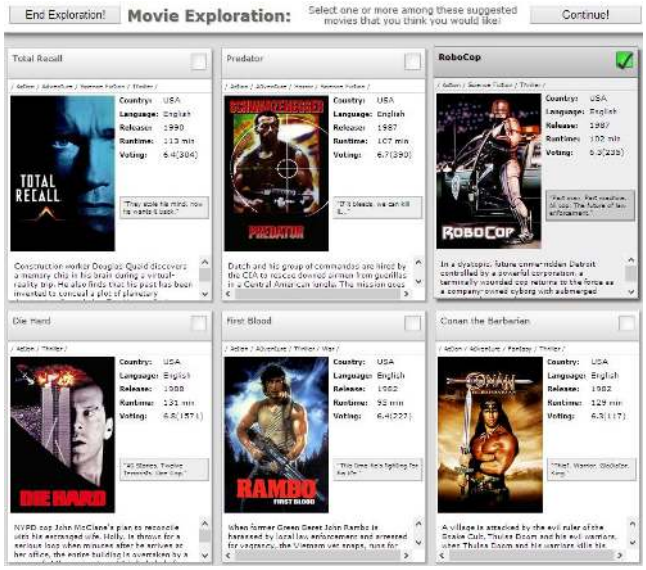

Figure 1. Screenshot of Prototype Implementation
First display, using "The Terminator (1984)" as start example

the shared as well as individual views of all users. A similar reasoning is also successfully used by recommender systems, e.g. [3], [10]. Now, the challenge of perceptual spaces is to reverse a user's rating process: For each item which was rated, commented, or discussed by a large number of users, we approximate the actual characteristics (i.e., the systematic bias) which led to each user's opinion as numeric features.

The general system design of our approach is shown in Figure 2: in an offline system initialization phase, a large number of user ratings (as e.g. obtained from a co-located recommender system, or from sites like e.g. IMDb or Netflix) is processed into a perceptual space. This process is described in detail in [4] and [5], but for clearer illustration of our demonstrator, we will briefly summarize it in the following. Then, personalized QBE queries are used to query that space.

Given is a large user-item-rating matrix, which usually is very sparse, containing only rating for around 1-2% of all user-item pairs. The goal is to find a matrix $A = (a_{m,k}) \in \mathbb{R}^{n_M \times d}$ representing movies as d-dimensional coordinates. To achieve this, we also need a helper matrix $B = (b_{u,k}) \in \mathbb{R}^{n_U \times d}$, representing user in the same space. Then, we use a factor model representing a rating function $f : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}$. Basically, this function can predict missing ratings given user and item vectors. We approximate this function and the involved vectors/matrices, we use Euclidian Embedding (as in [11]), and we want the distance between a movie vector a_m and user vector b_u to be small if user u likes movie m; otherwise, it should be large. To account for general effects independent of personal preferences, for each movie m and user u, we introduce the model parameters δ_m and δ_u, which represent a generic movie rating bias relative to the average rating μ. Then, a rating of a movie m by a user u can be predicted by $\hat{r}_{m,u} = \mu + \delta_m + \delta_u - dis_E^2(a_m, b_u)$, i.e. it is average rating of all movies (e.g., μ=6.2 out of 1..10) plus the user bias (e.g., δ_u=-1.6 representing a critical user always rating worse than others) and the movie bias (e.g., it's a overall good movie with an average rating of 8.4, so δ_m=2.2). The last term, $dis_E(\cdot,\cdot)$, represents the distance of the movie vector and the user vector in a d-dimensional space. Finally, all movie vectors (and therefore the matrix A) can be approximated by solving a large least squares optimization problem with all instances of the above

equation for which a rating is known (with some correcting terms accounting for noise added). Now, this matrix A represents our perceptual space.

Unfortunately, the resulting features in this space are implicit and have no direct real-world interpretation, therefore rendering SQL-style queries useless (i.e. you could ask for a value >0.8 on feature 5, but we don't know what feature 5 is). However, they allow for measuring perceived similarity effectively (i.e. the distance between the feature vectors). This now allows using the query-by-example paradigm, which provides simple query formulization without the need to explicitly refer to any features.

We adapt Bayesian Navigation [12] for this purpose. Here, for each query, a disposable user model is created during the interaction and discarded afterwards. Simplified, this model describes for each database item the probability that this particular item is the target item the user is looking for. Please note that usually the user is not explicitly aware of his target; if a user knew exactly which movie he was searching, an SQL-based query would be more efficient. However, we assume that there is at least one implicit target movie which the user simply does not know yet, nor can she describe exactly what her target looks like until she finds it. Our system is designed to guide a user to this implicit target, which is represented by adjusting the respective probabilities in each step. More formally, this is given by $P(M = M_i | H_t)$, i.e. the probability of a specific movie M_i being the intended implicit target M given the users current query history H_t. This history contains all displays (i.e. selection of movies a user has seen) and user actions on these displays (i.e. a binary selection of one or more of these movies). Together with a user prediction model, which predicts which movie a user will likely select out of a given display, (soft-min binary feedback in our case [12]), a selection strategy, which determines how the next display of the current interaction is selected, (most-probable strategy in our demo [5]), and a suitable start-up probability distribution, all probabilities for each movie can be recursively recomputed after each user interaction. This results in a new display and a new user interaction until the user is satisfied. Finally, a personalized list of top-k database items ranked by their probability is returned, and the temporary user model is disposed. The central challenge in computation is that Bayesian navigation requires an update of the modeled probabilities for each user interaction and each movie in the database, and each update of a single probability requires considering all other probabilities. This, of course, poses severe threats to the scalability of our system. Therefore, we presented an heuristic optimization technique in [5] which restricts these updates to the locality of the query. As a result, memory consumption and speed could be improved significantly.

The semantics of our approach are complemental to both SQL-style personalization as well as to recommender systems: recommender systems have precise knowledge of a user's likes and dislikes (basically: they "know" each user), and they use this knowledge to proactively provide personalized recommendations. However, they do not support dynamic queries, and cannot easily cater to changing moods and requirements. SQL-style personalization in contrast offers powerful queries on the "normal" available meta-data for users who know exactly what they are looking for. Our approach is in the middle-ground between both: for querying, a user just needs a vague idea / example of what she is looking for, and can navigate through items by simply pointing out good suggestions in the displays created using her feedback (which can be seen as situative recommendations). No direct interaction with attribute values is necessary. All three approaches serve their own purpose, and are useful in different situations.

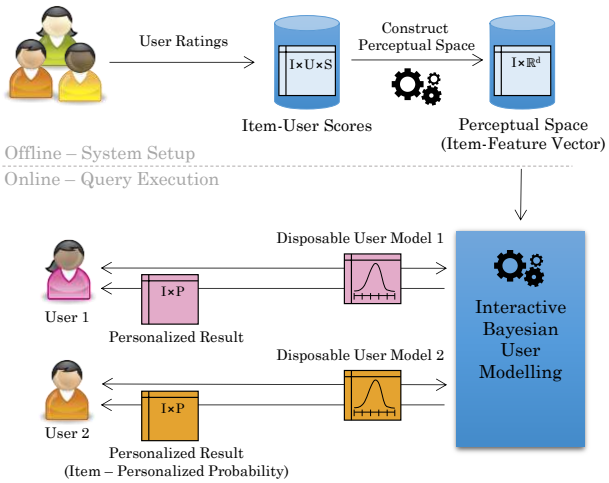

Figure 2. Basic System Design

3. PERSONALIZATION AND PRIVACY

Privacy concerns severely impact a user's overall satisfaction with a Web-based system (as argued using the example of recommender system in [13]), and might even prevent them from using it altogether, if the balance between privacy concerns and perceived system utility becomes unfavorable.

The central focus of our system in terms of privacy is to allow all users to use the personalized query capabilities as anonymously as possible, without requiring a user profile or pre-query preference elicitation. Especially in contrast to recommender systems, this means that for browsing or querying, no long term user profiles are required – only feedback (selection history) with regard to the current query needs to be temporarily retained and does not have to be connected to a user id. This history of a query session is deleted after the query is completed, thus removing the need to store and protect this sensitive information. Even if a single query history was analyzed, you would need an extensive model of a user's preferences to convincingly match it to her, in which case you would not gain any new information from this query, and her remaining queries are not connected to this one.

But still, our system will require a small group of enthusiast users to provide identifiable rating data in order to construct the perceptual space, not unlike a recommender system. However, this construction process is completely decoupled from executing queries, and even the users which contributed ratings can later use the system anonymously for querying. The perceptual space itself does not contain any user related information, not even in an anonymized or masked form or even just the number of users that participated in its creation. It is basically just a matrix of movie ids and their major perceptual dimensions (n=100 in our case). Therefore, approaches de-anonymizing ratings similar to the ones detailed in [14] cannot be applied. This could allow a "trusted platform" like IMDb or MovieLens to use its users' ratings to construct a perceptual space, which then could be used by a 3rd party system like ours. In contrast to publishing anonymized rating data, publishing a perceptual space carries only minimal risks to the user's privacy. But in any case, even if users did decide to contribute ratings to build the space, all users can use the query capabilities of our system without leaving trails of personal information in an ad-hoc fashion.

4. EXPERIENCING THE DEMO SYSTEM

Our proposed demonstrator allows users to directly interact with the prototype implementation of our system. Users may freely issue

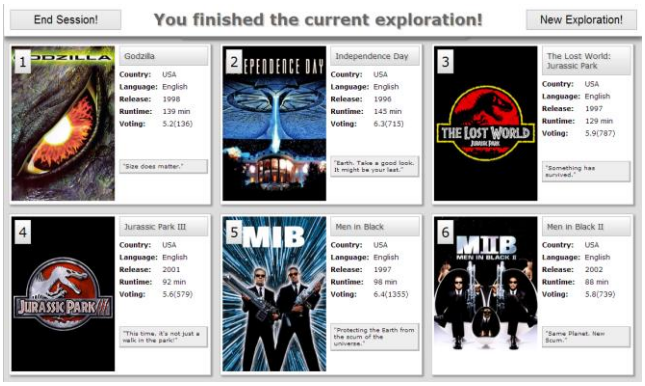

Figure 3. Screenshot of Prototype Implementation
Best matches movies after query is completed

their own queries, and may explore the system as they wish, using a pleasantly simple query interface. The prototype will include a wide selection of 11.976 movies from early 1900s up to 2005, and the underlying perceptual space was created by analyzing more than 103M user-movie ratings from over 480k users.

The users can interact with the system using a web-based user interface, which will resemble the screenshot in Figure 1. Additionally, we will prepare some query scenarios which highlight some interesting semantic aspects of our system. We will provide the required hardware and software to allow users to test the system, including at least two client devices. We also invite users to use their own mobile devices to access our web service. According to the user study we performed in [5], using our system was considered being an enjoyable and fun experience by most of the ~180 participating users.

Users can start a query by either providing a free example themselves, or by using one of twelve hand-picked examples provided by the system. In each display, 9 movies are displayed together with their respective poster, short synopsis, and general Meta data. From this display, users can select any number of movies as a positive example for the general "direction" in which they want to continue the query process. Then, after users are satisfied with the movies they encountered during the interaction, the query can be closed, and the final query result is presented in form of a list of movies ordered by their final Bayesian probability (see Figure 3).

Of course, we will also provide posters which explain the system design in detail, and discuss our design decision with interested visitors. This covers the general architecture, and also the theory behind building perceptual spaces, the Bayesian navigation technique, and the applied optimization techniques.

5. SUMMARY AND OUTLOOK

In this demonstration proposal, we described a personalized and privacy preserving query-by-example system for experience products as for example movies, books, or music. Our innovative system uses perceptual spaces built from a large number of user-item ratings to represent all database objects in a high dimensional feature space. We used Bayesian Navigation to allow users to issue queries in this space. The resulting system is thus very well suited to support users who only have a vague idea about what they are looking for, and helps them to explore the space in a personalized and guided fashion. Therefore, our system perfectly complements the features of SQL-based systems as well as those provided by recommender systems (which either support the case that the user knows exactly what she is looking for, or the case she does not know at all and therefore relies on a proactive recommendation).

In our prototype implementation, users can freely issue queries to a database containing around 12,000 movies in order to discover new and interesting titles, while at the same time learning about the inner working of our system.

6. REFERENCES

[1] G. Linden, B. Smith, and J. York, "Amazon.com recommendations: item-to-item collaborative filtering," *IEEE Internet Comput.*, vol. 7, no. 1, pp. 76–80, Jan. 2003.

[2] R. M. Bell, Y. Koren, and C. Volinsky, "All together now: A perspective on the Netflix Price," *CHANCE*, vol. 23, no. 1, pp. 24–24, Apr. 2010.

[3] Y. Koren and R. Bell, "Advances in Collaborative Filtering," in *Recommender Systems Handbook*, 2011, pp. 145–186.

[4] J. Selke, C. Lofi, and W.-T. Balke, "Pushing the Boundaries of Crowd-Enabled Databases with Query-Driven Schema Expansion," *Proc. VLDB*, vol. 5, no. 2, pp. 538–549, 2012.

[5] C. Lofi and C. Nieke, "Exploiting Perceptual Similarity: Privacy-Preserving Cooperative Query Personalization," in *Int. Conf. on Web Information System Engineering (WISE)*, 2014.

[6] H. Sundaram and S.-F. Chang, "Computable scenes and structures in films," *IEEE Trans. Multimed.*, vol. 4, no. 4, pp. 482–491, 2002.

[7] S.-Y. Neo, J. Zhao, M.-Y. Kan, and T.-S. Chua, "Video Retrieval Using High Level Features: Exploiting Query Matching and Confidence-Based Weighting," *Lect. Notes Comput. Sci.*, vol. 4071, pp. 143–152, 2006.

[8] I. Pilászy and D. Tikk, "Recommending new movies: even a few ratings are more valuable than metadata," in *ACM Conf. on Recom. Systems (RecSys)*, 2009.

[9] D. Kahneman and A. Tversky, "Psychology of Preferences," *Sci. Am.*, vol. 246, no. 1, pp. 160–173, 1982.

[10] T. Hofmann, "Latent semantic models for collaborative filtering," *ACM Trans. Inf. Syst.*, vol. 22, no. 1, pp. 89–115, Jan. 2004.

[11] M. Khoshneshin and W. Street, "Collaborative filtering via euclidean embedding," in *4th ACM Conf. on Recommender Systems (RecSys)*, 2010.

[12] I.J. Cox, M. L. Miller, T. P. Minka, T. V. Papathomas, and P. N. Yianilos, "The Bayesian Image Retrieval System, PicHunter: Theory, Implementation, and Psychophysical Experiments," *IEEE Trans. Image Process.*, 2000.

[13] B. P. Knijnenburg, M. C. Willemsen, Z. Gantner, H. Soncu, and C. Newell, "Explaining the user experience of recommender systems," *UMUAI*, vol. 22, no. 4–5, pp. 441–504, 2012.

[14] A. Narayanan and V. Shmatikov, "Robust De-anonymization of Large Sparse Dataset," in *IEEE Symposium on Security and Privacy*, 2008.