Representing Perceptual Product Features in Databases

Joachim Selke
Institut für Informationssysteme
Technische Universität Braunschweig
Braunschweig, Germany

ABSTRACT

Many modern goods have both factual and perceptual features. While factual features such as technical specifications can easily be handled by existing database technology, perceptual features such as design or usage experience are very hard to deal with. However, with the huge success and growing market share of online shopping, retailers face the need to provided detailed and structured information about perceptual product features to their customers. In this paper, we analyze why dealing with perceptual product features in databases is difficult and summarize our current efforts on tackling this problem.

1. INTRODUCTION

Marketing theory distinguishes between two types of product features: Factual and perceptual ones [4, 8]. Factual features are those that can easily be named and specified. Typical factual features are technical specifications (e.g., length, height, and weight) and traditional publication metadata (e.g., authors, number of pages and year of publication). Perceptual features are those that usually are hard to describe and tend to involve an emotional reaction or physical contact to the respective product. Typical perceptual features are artistic or stylistic properties such as the mood of songs, the sophistication of novels, and the character depth in movies.

While factual product features can easily be represented and managed by existing database technology (e.g., by introducing a database attribute per feature), working with perceptual product features is much more complicated. This is mainly because perceptual features tend to be vague and defy precise definitions (e.g. the borders of literary genres). However, paradoxically, there are established ways to express perceptual features using natural language (e.g., *sporty* car or *clunky* cell phone), which surprisingly mostly are not a matter of taste but are based on general agreement. Therefore, we strongly believe that established database technology is indeed able and suited to store, process, analyze, and answer queries based on perceptual product features. We just have to find out how this can be done in practice.

In this paper, we survey existing approaches to handling perceptual product features in databases, point out their limitations, and summarize our own recent work towards solving this problem. In particular, we present a series of use cases illustrating the benefits of our approach.

In this following, we use movies as a running example. Movies are particularly suited for this task as they appeal to a wide range of people and provide a large variety of both factual and perceptual features. In addition, movies perfectly illustrate the problem of lacking support for perceptual features in databases: It has been shown that, when selecting movies, consumers rely far more on perceptual movie features (funny, romantic, scary, ...) than factual ones (actors, directors, release year, ...) [3]. However, the ideas and results presented in this paper can easily be transferred to other types of products.

2. EXISTING APPROACHES

In this section, we take a close look at existing approaches to handling perceptual product features in information systems. We identified three different groups of approaches: those based on explicit data provided by experts, those based on textual data provided by users, and special-purpose approaches that implicitly deal with perceptual product features. In addition, in domains where products can be represented in digital form (e.g., music or movies), (low-level) features can be extracted automatically.

2.1 Explicit Modeling by Experts

Besides the traditional classification of movies into a small number of major genres [2, 9], many movie databases recently adopted more refined classification schemes. While some just introduced a larger number of possible genres (e.g., the rental service Netflix[1] expanded its simple genre list into a taxonomy covering 485 genres), others decided to describe movies using generally applicable description attributes. Popular examples are the metadata provider AllMovie[2], which classifies its 440,000 movies with respect to more than 5,000 different moods, themes, tones, and types (e.g., Ensemble Film, Haunted By the Past, and Intimate), and the recommendation service Clerkdogs[3], which rates each movie with respect to 37 different attributes (e.g., Character Depth, Geek Factor, and Violence) on a 12-point scale. Essentially, all these approaches try to capture a movie's perceptual features by means of a set of predefined databases attributes, which can either contain binary values (as in AllMovie) or numbers (as in Clerkdogs).

Although this approach looks rather straightforward and seems to be easy to implement in practice, it comes with many problems. First of all, clearly identifying and narrowing down the most relevant

[1] `http://www.netflix.com`
[2] `http://www.allmovie.com`
[3] `http://www.clerkdogs.com`

23rd GI Workshop on Foundations of Databases
(Grundlagen von Datenbanken),
31.5.2011–3.6.2011, Obergurgl, Austria.

individual perceptual features tends to be difficult. However, even if a comprehensive and generally understandable classification system has been developed and experts have been trained how to use it correctly and consistently, manually classifying all movies is a huge amount of work.

An even worse problem is the actual consistency of these movie classifications. We recently compared the genre judgments made by three major movie databases and found that the agreement among them is moderate at best, being just slightly less directed towards to completely random genre assignments than to perfect agreement [11, 12]. As we restricted our analysis only to the most popular movie genres, even worse results can be expected for less established and/or more complex classification schemes.

2.2 Textual Descriptions by Users

An alternative approach to making perceptual movie features available to movie databases has been adopted by movie portals such as the Internet Movie Database[4] (IMDb) or Rotten Tomatoes[5] (RT). Instead of trying to represent movies in a structured fashion by means of explicit database attributes, they focus on textual descriptions, usually in the form of reviews provided by arbitrary users (IMDb) or (semi-)professional critics (RT).

Although textual descriptions give users a comprehensive and helpful characterization of each individual movie, it is difficult to search for movies or provide targeted movie recommendations given only textual data. One the the rare services offering movie search based on movie reviews is Nanocrowd[6], which applies information retrieval methods to extract so-called nanogenres from textual data. Each movie is characterized by a set of nanogenres, where each nanogenre is represented by a three-word group (e.g., sports/ballpark/loves or chemistry/adorable/formulaic). However, these nanogenres tend to be much less informative and understandable than explicit database attributes that have been manually created by experts.

Another drawback of text-based movie descriptions is the lack of data. While blockbusters are commented by a large number people, less popular movies often receive just a very small number of reviews, which tend to provide only a partial movie description and are too short to effectively apply methods of text analysis.

2.3 Implicit Modeling for Special Purposes

The third major approach is collaborative filtering as used in the area of recommender systems [1]. Here, the only data available about movies are numerical ratings provided by users (e.g., on a scale ranging from one to five stars), where each user assigns just a single number to each movie he rated. As rating movies is an almost effortless task, usually there is a large number of ratings from many different users available. For example, in IMDb, there are about a hundred times more ratings than reviews, while even relatively unknown movies still receive a substantial number of ratings.

So far, this kind of data has only been used for special problems such as similarity search (finding those movies that are most similar to a given one) or recommendations (providing a list of movies that are likely to appeal to a given user). Here, the basic idea is to analyze the ratings for systematic patterns indicating similar taste across a group of users or similar properties in a group of movies. For example, to provide recommendations to some user u, one might first look for other users who rated most of the movies rated by u in a similar way, and then recommend those movies to u that have been liked by most of these other users. In a way, movie features and user tastes are modeled implicitly when using collaborative filtering.

Recently, a series of recommendation algorithms has been developed that try to decompose the rating matrix (movies are rows, users are columns, and ratings are entries) into the product of two smaller matrices [6]. These so-called factor models have an important by-product, which usually is neglected by recommendation algorithms: the representation of each movie as points in some abstract coordinate space. Here, movies with similar coordinates tend to be rated similarly by different users, whereas users with very different coordinates tend to be perceived very differently. From this perspective, one can think of these coordinates as a embedding of movies into some abstract *semantic space*.

Our own analysis of the semantic spaces produced by recent recommender algorithms showed that these spaces indeed capture major perceptual features of movies [10, 11, 12]. However, the main problem of semantic spaces hindering their use for general purpose database applications is the total lack of intuitive understandability. To illustrate this problem, Table 1 show the first three dimensions of a 100-dimensional semantic space extracted from the Netflix Prize ratings data set[7] (about 20k movies, 500k users, and 100M ratings). For each dimension, we listed the those popular movies that received the five highest and five lowest scores with respect to this dimension. Clearly, these axes do not offer any intuitive interpretation. However, the relative positions in semantic spaces are indeed meaningful. To give an example, Table 2 shows the five nearest neighbors of three popular movies.

2.4 Content-Based Feature Extraction

In some domains, one can provide a (near-)complete description of each product in digital form. Prime examples are images, music, and movies. In these cases, it is possible to automatically derive so-called low-level features from the products itself, thus avoiding any dependence on external product descriptions. For example, common low-level features of images are color histograms, symmetry properties, and measures for contrast. Low-level features are contrasted by high-level features (concepts), which describe those aspects of content objects a user is interested in.[8] For example, high-level features of images are the types of objects (sun, beach, mother, child, ...), events (playing, talking, ...), or abstract concepts (family, fun, ...) associated with a photo. The multimedia content description standard MPEG-7 defines a large number of low-level features and also provides a language to annotate multimedia content with custom-defined high-level features.

In state-of-the-art content-based multimedia retrieval systems, low-level features are usually extracted automatically from the available content, whereas the use of high-level features tends to require a significant amount of human interaction. Although there are initial approaches to automatically derive selected high-level features from low-level features, there is still a large discrepancy between the limited information that one can extract from the available multimedia data and the interpretation that the same data has for users [7]. This problem is usually referred to as *semantic gap*.

When comparing content-based feature extraction to the three approaches discussed previously, we see that low-level features loosely correspond to semantic spaces and high-level features to explicitly modeled attributes. However, there are important differences:

- As low-level features must be extracted by means of spe-

[4] `http://www.imdb.com`
[5] `http://www.rottentomatoes.com`
[6] `http://www.nanocrowd.com`
[7] `http://www.netflixprize.com`
[8] Sometimes, the distinction into low-level and high-level features is refined to a 10-layer pyramid structure for classifying different feature types of multimedia content [5].

Axis	Popular high-scoring movies	Popular low-scoring movies
1	Indiana Jones and the Temple of Doom (1984), The Godfather (1972), American Pie (1999), Top Gun (1986), The Silence of the Lambs (1991)	Eternal Sunshine of the Spotless Mind (2004), Garden State (2004), Two Weeks Notice (2002), Bend It Like Beckham (2002), Miss Congeniality (2000)
2	Twister (1996), Titanic (1997), Lost in Translation (2003), Napoleon Dynamite (2004), Ghost (1990)	Ocean's Twelve (2004), Mission: Impossible (1996), Paycheck (2003), Anger Management (2003), Ocean's Eleven (2001)
3	The League of Extraordinary Gentlemen (2003), Chicago (2002), Van Helsing (2004), Steel Magnolias (1989), Ocean's Twelve (2004)	American Pie (1999), Big Daddy (1999), Mr. Deeds (2002), The General's Daughter (1999), Lethal Weapon 4 (1998)

Table 1: Popular movies receiving high and low scores on the first three coordinate axes.

Rocky (1976)	Dirty Dancing (1987)	The Birds (1963)
Rocky II (1979)	Pretty Woman (1990)	Psycho (1960)
Rocky III (1982)	Footloose (1984)	Vertigo (1958)
Hoosiers (1986)	Grease (1978)	Rear Window (1954)
The Natural (1984)	Ghost (1990)	North By Northwest (1959)
The Karate Kid (1984)	Flashdance (1983)	Dial M for Murder (1954)

Table 2: Three popular movies and their respective five nearest neighbors in semantic space.

cialized extraction algorithms, they are tied to a particular representation of the original content. Consequently, low-level features extracted from images cannot be compared to low-level features extracted from songs. In contrast, semantic spaces are derived from user feedback which can be provided for any product type in the same way, thus enabling the direct comparison of images and music. In addition, the design of effective low-level extraction algorithms is a complex task, which must be hand-crafted for each product domain under consideration.

- Semantic spaces are derived directly from human feedback (e.g., star ratings), which is turn is based on the most relevant perceptual product properties. Low-level features only capture statistical properties of the data representation such as color histograms. Therefore, the semantic gap between semantic spaces and user perception can be expected to be lower than the semantic gap present in current content-based multimedia retrieval systems.

For these reasons, we decided to put aside content-based feature extraction for the moment and focus on the three remaining approaches discussed above. However, in future work we plan to compare the ideas presented in this paper to existing methods from content-based multimedia retrieval where this is possible.

2.5 Conclusion

We can draw the following conclusions from the findings presented in this section:

- Modeling perceptual movie features by explicit attributes requires a huge amount of manual work but still leads to data of questionable quality. However, users can easily understand the meaning of these attributes.
- Capturing perceptual movie features by means of textual descriptions is helpful for users when looking for information about each individual movie. However, this kind of data is difficult to process automatically, cannot be understood as easily as explicit attributes, and the amount of available data is scarce for less popular movies.
- Semantic spaces created from a large number of user-provided ratings capture major perceptual features of movies. However, semantic spaces as such do not offer any intuitive interpretation and thus cannot be used to communicate with users.

3. PROPOSED SOLUTION

At first view, the result of our above analysis is rather disillusioning. Intuitively understandable models of perceptual movie properties are expensive to create and lack data quality, semantically meaningful models cannot be understood, and the third option seems to combine both disadvantages.

However, there is still hope. In [11] we introduced a data model that tries to combine the strengths of the approaches mentioned above. To be more precise, we propose to represent each movie by three different types of database attributes:

A. attributes describing factual movie properties,

B. attributes making a selected number of perceptual properties explicit (manual classification), and

C. attributes containing the movie's coordinates in some semantic space.

This approach brings several advantages. Probably most important is that the three different types of attributes can work together to reduce the weaknesses of each of them. In the following, we show a series of examples illustrating this idea (for technical details, please see [11]).

Enhancing the data quality in type-B attributes.

By aligning the manual classification of movies as expressed in type-B attributes to the semantic space, we are able to detect are large number of possibly misclassified movies. The basic idea is that movies that are classified into the same category should also be located close together in the semantic space. If we find a movie *m* that has the same value with respect to some type-B attribute but is very different from other movies having this value with respect to the semantic space, then *m* is likely to be misclassified by the experts. By identifying such movies and giving human experts

Figure 1: Genre clouds for Rocky (1976) and Star Trek (1979).

a chance carefully re-check problematic movies, the data quality can be increased. In our experiments on genre classifications [11], we have been able to detect possibly misclassified movies with a mean precision of about 55% and a mean recall of about 25%, which is significantly better than drawing random samples (the only alternative approach available). In summary, with the help of type-C attributes we are able to reduce a significant weakness of type-B attributes (data quality).

Saving manual work in creating type-B attributes.

To significantly reduce the amount of work required to manually classify all movies with respect to the type-B attributes, automatic classification can be applied. Here, given a binary type-B attribute (e.g., the genre *Action*), a human experts provides a small number (e.g., 10) of clearly positive examples (i.e., typical Action movies) and the same number of clearly negative examples (i.e., obvious non-Action movies). Using a support vector machine classifier that categorizes all remaining movies based on the training data and the type-C semantic space representation of movies, we have been able to produce results being only of slightly lower quality than those created by human experts [11]. By means of the method described previously, the data quality can easily be increased incrementally. In summary, with the help of type-C attributes we are able to reduce another significant weakness of type-B attributes (amount of work).

Enriching type-B attributes.

Again, by comparing type-B attributes to the semantic space represented by type-C attributes we are able to determine to what degree a type-B attribute value applies to each movie. For example, IMDb only assigns binary genre judgments to its movies, which leads to the classification Drama/Romance/Sport for the movie Rocky (1976) and Action/Adventure/Mystery/Sci-Fi for the movie Star Trek (1976). Although this classification is justified, there are several problems: Rocky contains romantic elements but it is a highly untypical Romance movie. It is most well-known for being a typical sports movie with dramatic activities. Similarly, Sci-Fi is widely recognized as Star Trek's most prominent genre, while it is a rather untypical Mystery movie. By analyzing the semantic space for where typical movies of genre *X* are located, we are able to judge how typical an assigned genre for each movie really is. To illustrate this, Figure 1 depicts a "genre clouds" for the above two movies. We automatically generated it from IMDb's binary genre assignments (type B) in combination with a semantic space extracted from ratings (type C) [11].

Enabling conceptual queries.

When describing their movie preferences, users often refer to factual movie properties as means attributes that approximately characterize an intuitive concepts that they are unable to express otherwise. For example, movies in the style typically associated with the director Quentin Tarantino could be called *Tarantino-ish* movies. In fact, Google counts 4530 Web pages mentioning this term. We refer to database queries in this style as conceptual queries. We are able to answer such queries by first finding out where movies directed by Quentin Tarantino are typically located in the semantic space (by identifying a small continuous region in space), and then looking for other movies that are located close to the center to this region. By applying a simple weighting scheme, we are able to produce a, say, top-10 list of the most Tarantino-ish movies. To give an example, Table 3 shows our results for Tarantino and two popular actors. Here, we used a support vector machine to learn where movies directed by Tarantino tend to be located in semantic space and used this information to find very similar movies that have not been directed by Tarantino [11]. Apart from minor exceptions (in particular, The Professional and Dragon: The Bruce Lee Story), these results look very promising. In summary, we have been able to understand users' implicit concepts by of mapping type-A attributes to the semantic space.

4. CONCLUSION AND OUTLOOK

In this paper, we have discussed the problem of representing perceptual product features in databases. We concluded that each existing approach alone does not provide an acceptable solution to this problem as it comes with severe disadvantages. However, by combining several methods into a joint data model, we have been able to reduce the weaknesses of each individual approach and boost its strengths. Our examples show promising results, which we are going to analyze in detail in future work. In addition, as already indicated in Section 2.4, we plan to compare our work to approaches from content-based multimedia retrieval. For example, for genre classification tasks, it would be interesting to compare semantic spaces derived from ratings to low-level features extracted from the actual movies.

5. REFERENCES

[1] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, 17(6):734–749, 2005.

[2] D. Chandler. An introduction to genre theory, 1997. Available from `http://www.aber.ac.uk/media/Documents/intgenre`.

[3] E. Cooper-Martin. Consumers and movies: Some findings on experiential products. In *Advances in Consumer Research*, volume 18, pages 372–378. 1991.

[4] E. C. Hirschman and M. B. Holbrook. Hedonic consumption: Emerging concepts, methods and propositions. *Journal of Marketing*, 46(3):92–101, 1982.

[5] C. Jörgensen, A. Jaimes, A. B. Benitez, and S.-F. Chang. A conceptual framework and empirical research for classifying visual descriptors. *Journal of the American Society for Information Science and Technology*, 52(11):938–947, 2001.

[6] Y. Koren and R. Bell. Advances in collaborative filtering. In F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, editors, *Recommender Systems Handbook*, pages 145–186. Springer, 2011.

[7] Y. Liu, D. Zhang, G. Lu, and W.-Y. Ma. A survey of content-based image retrieval with high-level semantics. *Pattern Recognition*, 40(1):262–282, 2007.

[8] P. Nelson. Information and consumer behavior. *Journal of Political Economy*, 78(2):311–329, 1970.

[9] C. Preston. Film genres. In W. Donsbach, editor, *The International Encyclopedia of Communication*. Blackwell, 2008.

[10] J. Selke and W.-T. Balke. Extracting features from ratings: The role of factor models. In *Proceedings of M-PREF 2010*, pages 61–66, 2010.

Stallone-ish	Tarantino-ish	Jim Carrey-ish
Universal Soldier (1992)	True Romance (1993)	EDtv (1999)
Commando (1985)	GoodFellas (1990)	Innerspace (1987)
Missing in Action (1984)	The Usual Suspects (1995)	Bedazzled (2000)
Red Heat (1988)	Casino (1995)	Cadillac Man (1989))
Raw Deal (1986)	Desperado (1995)	Pleasantville (1998)
Bloodsport (1988)	The Professional (1994)	Dragon: The Bruce Lee Story (1993)
The Last Boy Scout (1991)	Killing Zoe (1994)	Honey, I Shrunk the Kids (1989)
The Running Man (1987)	Full Metal Jacket (1987)	Alive (1993)
Kickboxer (1989)	2 Days in the Valley (1996)	Shallow Hal (2001)
The Delta Force (1986)	Go (1999)	Punchline (1988)

Table 3: Top 10 results for three different conceptual queries.

[11] J. Selke and W.-T. Balke. TEAMWORK: A data model for experience products. ifis technical report, Institut für Informationssysteme at Technische Universität Braunschweig, 2011.

[12] J. Selke, S. Homoceanu, and W.-T. Balke. Conceptual views for entity-centric search: Turning data into meaningful concepts. In *Proceedings of BTW 2011*, pages 327–346, 2011.

Acknowledgments

I am very grateful to Prof. Dr. Wolf-Tilo Balke for providing valuable guidance and supervising my doctoral thesis, which will be based partly on the work presented in this paper.