Querying Concepts in Product Data by Means of Query Expansion

Silviu Homoceanu, Wolf-Tilo Balke
Institute for Information Systems, Technische Universität Braunschweig, Braunschweig, Germany
{silviu, balke}@ifis.cs.tu-bs.de}

Abstract— The Web has become the primary source of information containing both structured and unstructured information. A good example is e-commerce where products are usually described by technical specifications (structured data) and textual user reviews (unstructured data). Both sources of information complement each other, covering quantifiable as well as perceived aspects of each product. In fact, for most searches users will have more or less abstract concepts in mind, as opposed to clear cut categorical information. In this paper, we develop a novel approach to reveal implicit product features for querying, by combining structured product data with natural-language product reviews. Using a self-supervised learning technique we progressively build a query-aware representation of the product domain under consideration. This representation can then effectively be used for intuitive querying. We performed extensive experiments confirming the effectiveness of our approach over real world product data. In particular, our evaluations show vastly improved precision and recall over the respective IR techniques.

Keywords: product search; e-commerce; query expansion; semantic mismatch.

1. Introduction

During the last decade the Web has evolved into the prime information source for a large variety of topics, particularly fostered by the large amount of user generated content. In the wake of information searches also services like e-commerce have rapidly expanded: Read about something – buy it! Indeed more and more products or services are marketed and sold through online platforms today, and offering a high quality portal is a distinctive advantage in competition. To improve customer experience, popular online shops like Amazon.com or DooYoo, provide users with extensive information regarding products, like technical specifications, expert reviews, and relevant user comments or ratings. The common point of reference is the entity (e.g., product, music or video file, person,…) in question. But due to the mixed nature of the content almost all platforms have to cater for different types of data namely structured data (like product databases) and unstructured data (like product reviews).

However, when it comes to querying such mixed data, today's platforms still face a difficult problem. Most platforms offer navigational interfaces for SQL-style access to structured data (via categories) and then a simple keyword style search for the actual values. Some even offer IR-style keyword search on user comments or product description, but a viable integration of search is still lacking. As an example consider querying Amazon.com for a 'Nokia E72' cell phone. Easily enough, the result is the entity matching the search criteria as in classical databases. But real world user queries tend to be more complex: Often certain aspects of an entity's purpose or application like a 'business cell phone', a 'city car', or 'music for a wedding' are the focus of queries. But trying for example the 'business cell phone' query in Amazon.com, the system only returns 5 devices, entirely missing out on business phone market leaders[1] like the BlackBerry Bold 9700, the Motorola

[1] Cha, B., (Jan. 2010). Best smartphones for business users. CNET, http://reviews.cnet.com/4321-6452_7-6544038.html

Droid, or the HTC Touch Pro2.

How can this happen? The major reason for the catastrophic recall is that whenever a query term is not explicitly mentioned in the stored data, today's systems are not able to interpret the intended information need. As [1] points out, there is a *semantic mismatch* or *gap* between the data presentation and the user perception over entities. In a nutshell, crucial implicit information like 'what explicit features make a cell phone a good business phone?' cannot be derived and this semantic mismatch is unfortunately present in most entity-centric searches.

As a running example for the rest of this paper we will stay with the domain of cell phones. First, this segment of consumer electronics is well-understood. Furthermore, it offers plenty of real-world data since according to Gartner mobile phones sales (especially smart phones) currently is one of the strongly growing markets[2]. In any case, queries on implicit information are not only typical in the cell phone domain, where discussion boards regularly refer to flowery categories like 'ideal for social networkers', 'perfect for fashionistas', 'tough as nails', or 'multimedia marvels'. This basic problem is also consistent with recent results from other domains like the predominant tagging of explicit media features, in contrast to the high number of queries on implicit (usage-based) features in online image repositories or music stores (see [2]). Therefore, being able to transform implicit information needs into explicit terms for querying is generally of vital importance for building successful e-commerce platforms.

The challenge of implicit information needs vs. explicit queries has been discussed before and is directly addressed by some retrieval paradigms. However, experiments on real world data have shown that classical IR techniques like the vector space retrieval model (VSM [3]) and latent semantics (LSI [4]) don't achieve satisfying results [5]. On the other hand, query expansions, i.e. augmenting user queries with relevant semantically related terms, show promising results, if only the expansion terms are chosen in a sophisticated manner. While first approaches only focused on synonymy and term disambiguation, today, domain knowledge is incorporated. Expansion algorithms range from using simple lexical databases [6] like WordNet [21], existing domain ontologies [7] like Medical Subject Headings (a controlled vocabulary thesaurus used for indexing medicine related articles) to extracting language models e.g., probabilistic models based on term co-occurrence [8], directly from text. Following on our running example, a clear semantic connection between the 'business cell phone' concept and technical features like 'email clients', 'organizer', 'calendar', 'notepad' and 'file browser' could be established.

In this paper we present a novel query expansion method, which is able to solve the expansion problem for entity centric search by bridging structured and unstructured data, with the help of a self-supervised learning technique. For evaluation purposes we used cell phones with real world user reviews crawled from the Web, which were subsequently tagged by domain experts with respect to prevalent concepts in the domain. We evaluated our algorithm in terms of precision and recall against three standard IR methods, Latent Semantic Indexing (LSI), Vector Space Model (VSM) and Stephen Robertson's best match (BM25) [9] with Kullback-Leibler Divergence (KLD) [10] as probabilistic term weighting scheme, a widely accepted approach as being the standard method in query expansion. Our experiments show impressive improvements over these baseline methods: On average, our system achieves precision values greater than 0.8 for up to 25% recall and manages to maintain a value of more than 0.5 even for recall values of 90%. In contrast, for such high recall values, VSM and LSI barely reach values of 0.2 in precision. While for some concepts our approach delivers results similar to BM25, improvements are spectacular for concepts with an increased risk of topic drift.

2. Defining the retrieval task

In this section we will lay the foundation of our approach by briefly presenting the application query type and formalizing the actual problem.

2.1. Revisiting entity-centered user queries

As argued before, every query expansion technique reaches its greatest benefit when query terms refer to concepts, which cannot be queried with simple SQL or IR techniques. But is there really a need in today's information portals for specifically answering concept-specific queries? Studying the AOL Web search query logs (comprising logs of all searches done by 650,000 AOL users over the course

[2] http://www.gartner.com/it/page.jsp?id=1372013

of three months in 2006), with regard to our sample domain of cell phones, we observed that the amount of queries on *clear* product features focusing on capabilities e.g., ‘camera’ or ‘voice dial’, is actually smaller than the amount of queries on concepts mainly focusing on usage e.g., ‘cell phone for kids’, ‘cell phone for seniors’, or ‘business cell phone’.

In particular, we extracted 21,650 cell phone relevant entries through the use of regular expressions. After manual inspection we classified all queries into six base categories (see Figure.). The resulting categories deal with:

– **Products:** Represents about 22% of the queries. It contains queries related to brands, product prices, product features, specifications, and types. E.g., ‘Motorola Razr’, ‘cell phone battery’, or ‘cell phone for kids’.
– **Telecom & Pricing Plans:** For example ’Verizon cell phones’ or ‘compare cell phone plans’. This category represents about 30% of the cell phone related queries.
– **Accessories:** Represents 17% of the queries, and refers to products for cell phones e.g., ‘sexy phone wallpaper’ or ‘ringtones’.
– **Phonebook:** 13% in size refers to cell phone numbers, or reverse phonebook lookups, e.g., ‘cell phone number lookup’.
– **Unspecific Queries:** About 15% of all queries representing too general queries, usually simply ‘cell phones’.
– **Other:** About 3% containing more exotic queries like ‘help finding lost cell phone’, or ‘cell phone health risk’.

Focusing on the category with references to products, we observed that the majority of queries are either concerned with a specific brand (e.g., ‘Motorola’ or ‘Sony Ericson phone’) or the price (e.g., ‘Nokia 5300 price’ or ‘cheap mobile phone’). Still, about 27% focus on specific product features and the amount of queries for concepts (like ‘multimedia phone’ of ‘cell phone for seniors’) surprisingly, is about 14% of the total feature related queries. Although they represent only a relatively small percentage, answering such queries is vital for the search process. The reason lies mainly in the consumer buying process. According to Engel et. al. [11], [12] users generally first gather information in a task-based manner, i.e. they try to identify the best products for the intended usage. The second phase is purely informational. Here users compare technical specifications or prices of candidate products. Since informational queries usually are posed several times for different products or vendors, it accounts for the significant difference in percentages.

Selecting the appropriate technique for correctly answering a query, needs a more detailed distinction of the query terms. As we have shown in [13] the main differentiating factors are the *clarity* of the query term and the level of *user consensus* over the expected result. These factors span a design space (presented in Figure 2), where typical query terms can be arranged. Considering the techniques for evaluating the respective queries we find that *clear* queries can be answered successfully with simple SQL-based techniques directly relying on product databases. In contrast, for *unclear* query terms, techniques from the field of IR run on product descriptions or reviews would be more appropriate. However, focusing our attention more on the queries in-between, currently there are very few suitable approaches for query evaluation, because a combination of structured/unstructured information has to be exploited for retrieval. Supporting those queries in a satisfying manner is the subject of our research. To be specific, in this paper we focus on answering conceptual queries with large *user consensus* over the expected result. Preference based conceptual queries like ‘best’, ‘beautiful’, etc. are left as a subject for future work.

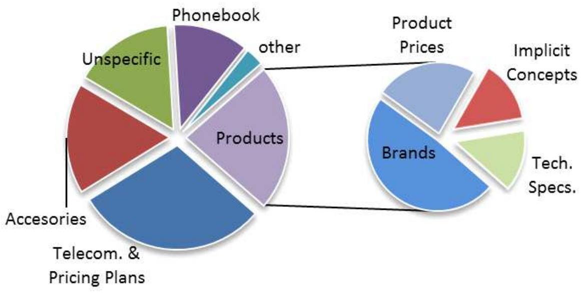


Figure 1. AOL query log.

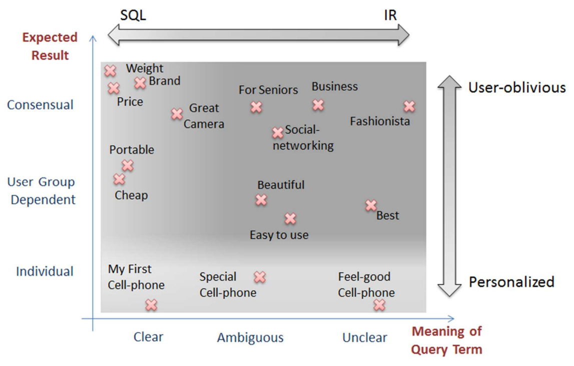


Figure 2. Query design space.

2.2. Bridging the semantic gap

Although many real world queries use conceptual information, it is difficult to define what a concept actually is, and how it can be reliably spotted in queries. Psychology defines concepts as a cognitive unit of meaning, typically associated with a single meaning of a term [14]. Any term can therefore be the representation of a concept. The major importance of specific concepts in practical life comes with the generally consensual notions humans connect with some concepts: Each concept carries connotations that immediately create an intuition about what is meant, and thus enable efficient communication. For example asking about a 'cell phone for kids' will immediately bring up ideas like robustness, ease of use, fun colors, security features, and parental control pricing plans. Explicitly adding exactly these connotations to a query is what makes applying a semantic query expansion technique so promising for good retrieval quality. However, lacking a clear definition, detecting conceptual features in queries is a serious problem. Whereas for *explicit features* like 'weight', 'size', or 'display type', new developments in declarative query languages already allow a mapping of previously unknown attributes to actually existing attributes in the underlying data (e.g., using malleable schemas [16]), the recognition of *implicit conceptual features* like 'portability' or 'design' is much harder. Still, even if implicit conceptual features cannot be clearly defined and the exact disambiguation is beyond the scope of this paper, preparing the underlying data collection to answer at least the most often occurring implicit queries is of strategic advantage.

Therefore, in this paper we will rely on a few simple, yet suitable, heuristics. First, implicit features obviously can never be attribute names of structured data, and also in the respective set of values they should rarely occur. Similarly, in unstructured text documents an implicit concept should occur not too often, either. But since texts are the usual way to communicate connotations and tie concepts to entities, any important implicit concept definitely should occur at least sometimes. In the following we consider that any noun (<*N*>) and nominal phrase (<*NP*>) from the query, is an implicit conceptual feature if it complies with *Observation 1.*

Observation 1: *Implicit Conceptual Feature*

Let x be any query term, S be the collection of structured technical product specifications, $f_S(x)$ be the percentage of entities for which x occurs in values of structured data, D the set of documents presenting products and $f_D(x)$ the percentage of documents grouped by entities explicitly mentioning x.

An *implicit conceptual feature q* is any query term for which $0 \leq f_S(q) \leq r$ and $s \leq f_D(q) \leq t$, where r, s and t are domain specific parameters.

For our cell phone example and the later experiments, we tried different values for r, s, and t. We found that occurrences under 5% in structured data and occurrences in between 2% and 10% of unstructured reviews are sufficient for detecting most implicit conceptual features without generating too many false positives. These parameters are collection-specific, minor adjustments being necessary for other data collections. Such adjustments may be for example performed on the run, by manually inspecting frequently posed queries from the query log.

Now we are ready to address the problem of answering implicit conceptual queries. The entity-related data available online comprises both structured and unstructured data. For our retrieval task this is not a problem, but rather a feature. This is on one hand because most concepts will only be explicitly tied to entities in unstructured texts, thus descriptive vocabularies can be derived by co-occurrence analysis. But also because many concepts are to some degree affected by certain structural characteristics, thus the statistical analysis and exploitation of value distributions can point to similar entities (e.g., 'portable' items will definitely show a bias towards smaller sizes and lighter weight). In fact, starting with a seed vocabulary for some relevant entities, and learning their structured characteristics to find similar entities, which in turn are used to expand the vocabulary and learn even more about the structural bias, will lead to a cyclic improvement of a model that subsequently can be used for effective querying.

In summary, for implementing the query expansion of some initial implicit query term our approach requires the extraction of terms relevant to the intended concept from the underlying data, and thus has to bridge the gap between structured and unstructured information. The retrieval task can be formalized as follows:

Problem Statement: *Query Expansion for Implicit Features*

Given: A relational database S containing data with respect to entities $P_1, \ldots, P_N$. For each entity P_i, there also are text documents $D_{i,1}$, …, $D_{i,n(i)}$ describing P_i.

Task: Given a user query Q containing an implicit conceptual feature q, derive an expanded query $Q' := q \cup \{q_1, q_2, \ldots, q_k\}$, where $q_1, q_2, \ldots, q_k$ are terms from S and D which explicitly describe q (with corresponding weights $w_1, \ldots, w_k$).

3. The query expansion process

The problem of querying for implicit conceptual features is typically solved by using a query expansion technique. The key task however, is the selection of the right terms for expanding the query. An intuitive approach would be to consider for the expansion all the terms occurring together with the queried concept in the *product data*. (Note that by product data we understand structured and unstructured data i.e. technical specifications and product reviews respectively). But the number of such terms is quite high, and although the query expanded in this manner leads to high recall, the precision is catastrophic with almost any product qualifying as a result. In consequence, we first choose a set of candidates that appear together with the query in product data, then we calculate the weight of each candidate term based on a function similar to the term co-occurrence and finally we select only those terms with the highest weights.

3.1. Choosing the candidate terms for query expansion

The query expansion is performed based on a corpus of products. Each product in the corpus is described through one or more text documents and one tuple in the technical specifications table. Of course any term appearing in documents or the tuples could be of interest for the expansion. But particularly in the case of the documents, it's obvious that many of the terms have no relation with the queried concept. Actually when referring products, concepts mostly relate to some product features. Thus any term expressing a product feature and co-occurring with the queried concept in the product data is considered a candidate for the query expansion. Two steps have to be performed for choosing the candidate terms: First, select the query relevant product data (data in which the queried concept appears) and second extract the product features from the selected data.

3.1.1. Query relevant product data

A document is likely to be relevant if the query is mentioned in it. But not the same can be said about the structured data. We found numerous cases where a product manufacturer would include some task based concept in the product name, model or series, although the product is not a good candidate for the concept. However, if the query is explicitly mentioned in a document, then the technical specification of the corresponding product is also relevant with respect to the query.

The process of selecting query relevant product data works as follows: First, all product descriptions containing the query are considered relevant. The technical specifications of the products corresponding to the descriptions found as relevant are also considered relevant to the query. In a second pass, in a boosting fashion, descriptions being similar to the relevant descriptions are also added to group in transitive fashion along with the corresponding technical specifications. The product data is separated this way in two classes: Class c representing data - documents (D_c) and tuples from the structured data (S_c) - being highly relevant with respect to the query and $\bar{c}$ representing the remainder of the data. In technical terms, in order to distinguish between relevant and irrelevant documents we represent each document as a vector according to the vector space model: Terms of all documents represent axes of the space and projections of documents on each axis are computed with the help of the Term Frequency-Inverse Document Frequency (TF-IDF) measure [26]. The similarity between two documents is computed according to the well-known cosine metric (further denoted as *cos*) [3]. A more elaborate, technical presentation of how these standard techniques are applied is presented in [29].

This being said, we can proceed to elaborating on D_c and S_c:

$D_c = \{d_i | d_i \in D \wedge \exists d_j \in D'_c \text{ s.t.} \, cos(d_i, d_j) \geq \theta\}$,

where θ is a collection specific parameter regulating the precision of c, and *cos* represents the cosine similarity measure;

$D'_c = \{d_j | d_j \in D \wedge d_j \text{ contains } q\}$;

P_c is the set of products whose textual descriptions have been found as relevant to the query and S_c are the corresponding technical specifications:

$P_c = \{p_i | p_i \in P \wedge \exists d \in D_c \, with \, d \, describing \, p_i\}$;

$S_c = \{s_i | s_i \in S \wedge \exists p_i \in P_c \text{ s.t.} \, s_i \, tech.specs.of \, p_i\}$.

Accordingly $\bar{c}$ comprises $D_{\bar{c}} = D - D_c$ and $S_{\bar{c}} = S - S_c$.

3.1.2. Product features

In the case of unstructured data product features are usually represented through nouns and nominal phrases [22]. Some adjectives can also imply product features e.g., 'heavy' may imply the 'weight', but these are rather infrequent cases. Consequently, in order to extract the candidate terms, we applied standard natural language processing (NLP) techniques like part-of-speech tagging (POS) and chunking. Word inflections have been eliminated by means of stemming.

In structured data, products are described through table attributes and the corresponding values. While all attributes are product features, from the values we only considered the ones corresponding to categorical attributes. Obviously all values in the product table define a certain aspect of the product but the categorical attributes bear most of the differentiating force. Typical examples of such values are 'nokia', 'apple', etc., for the 'brand' attribute, or 'candy bar', 'clam shell' for the 'form factor' attribute. Numerical values like in the case of the 'price' or 'weight', have dynamically been reduced to the ordinal values 'low', 'average' and 'high'. We established the 'average' interval of the values for an attribute as being between [average of the values – one standard deviation, and average of the values + one standard deviation]. We then set the 'low' and 'high' intervals accordingly. Although they are not candidate terms, and will not be included amongst the query expansion terms, these ordinal values allow us to establish the weight of their corresponding attribute. In this manner we can for example find out that the 'weight' is an important factor since most of the devices which are explicitly relevant toward the conceptual query, fall into just one of these intervals (say 'low weight'), while the remaining products are spread amongst, or fall into the other two intervals.

Finally, after establishing what *product features* and *query relevant product data* stand for, we can formally define the set of candidate terms:

Definition 1: *Candidate Terms (CT)*
Let CT_D and CT_S be the set of query expansion candidate terms from documents and structured data respectively, with:

$$CT_D = \{t_i | (POS(t_i) =< N > \vee POS(t_i) =< NP >) \wedge (t_i \subseteq d, \text{with } d \in D_c)\} \text{ and}$$

$$CT_S = \{t_i | t_i \; table\; attr.from\; S\} \cup \cup \{v_{t_i} | (\exists v_{t_i}\; value\; of\; attr.t_i\; in\; S_c) \wedge \wedge (t_i\; categorical\; attr.)\}$$

where, *POS* (*t*$_i$) represents the part of speech of term *t*$_i$, and <*N*> and <*NP*> tags represent the noun and respectively nominal phrase parts of speech.

We define the set of *candidate terms* as:
$CT = CT_D \cup CT_S$.

3.2. Calculating the weight of candidate terms

Associating the candidate terms with the *right* weights is crucial for the entire process. The weight of a term must reflect the term's contribution to describing the queried concept.

In this paper we estimate the weight of a candidate term by relying on a document classification approach introduced in [25]. The basic idea is to give higher weight to candidate terms appearing quite often in data from *c* and not that often in data from $\bar{c}$.

Definition 2: *Weighting Function (W)*
Let ct_i be any candidate term from the candidate list *CT*. $n_c(ct_i)$ and $n_{\bar{c}}(ct_i)$ represent the number of documents (if ct_i was extracted from unstructured data) or tuples (if ct_i was extracted from structured data) that contain ct_i from *c* and respectively $\bar{c}$.

The weight of ct_i, denoted $W(ct_i)$ is estimated by calculating the difference between the normalized frequencies of ct_i in *c* and $\bar{c}$:

$$W(ct_i) = \frac{n_c(ct_i) - min_c}{max_c - min_c} - \frac{n_{\bar{c}}(ct_i) - min_{\bar{c}}}{max_{\bar{c}} - min_{\bar{c}}},$$

where the components of the normalizing factors max_c and min_c are the number of documents, or by case tuples, containing the most frequent and respectively least frequent product feature from *c*. $max_{\bar{c}}$ and $min_{\bar{c}}$ are analogously defined, with the most frequent and respectively least frequent product feature from $\bar{c}$.

NB: For the candidate terms which were extracted from structured data, the weight of an attribute is calculated by considering also the corresponding attribute values: *ct*$_i$ is being extended in this case to the attribute-value pair. To clarify, the weight of the 'price' attribute, which may be selected as a candidate term, will be calculated as the maximum out of three weights, one for 'low price' one for 'average price' and one for 'high price'. Of course in the case of numerical attributes, this is only possible if the values have previously been transformed to ordinals based on their average values and standard deviation (as presented in Subsection 3.1.2).

The key factor in the weighting function is that the weight of each term is normalized with respect to typical terms (the most frequent product features) from both c and $\bar{c}$. This is critical because $|c| \ll |\bar{c}|$. In this way important candidate terms with implicit connection to the queried concept aren't severely penalized despite appearing also in $\bar{c}$.

But since we have split the product data into two classes why not apply classical supervised machine learning techniques on this automatically generated training set and train a classifier? As argued in [19] and as shown in the evaluation section, classical IR techniques like VSM are not able to retrieve many of the eligible products. Therefore both $D_{\bar{c}}$ and $S_{\bar{c}}$ contain data which is implicitly relevant regarding the query. For this reason, classifiers like SVM or decision trees are not an option (see [25] for further details). Furthermore, typical weight measures associated with *discriminative feature weighting* like term co-occurrence, mutual information or information gain tend to excessively penalize important terms due to the noisy classification.

3.3. Selecting the expansion terms

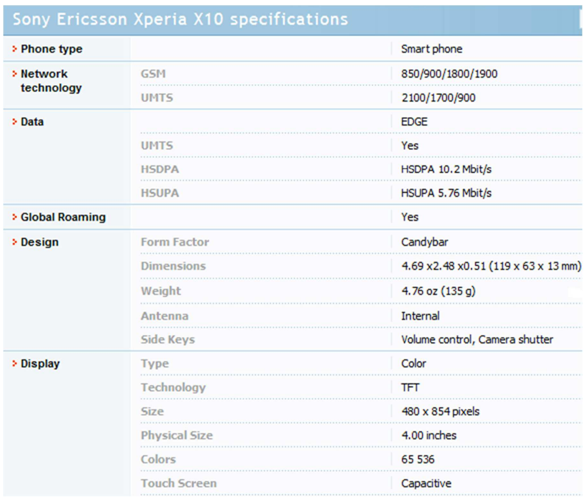
Having calculated the weight of all candidate terms we are now ready to choose the most appropriate terms for query expansion. Taking a closer look at the weighting function, the candidate terms are associated values between [-1; 1]. As intuitively expected, there are few very week candidate terms, with weights close to -1, many general terms, with similar normalized appearances in c and $\bar{c}$ and weights close to 0, and some strong candidate terms with values closer to 1. For the query expansion, we chose the candidates with the highest weights according to the 'three-sigma rule' [15] (average plus three standard deviations).

4. Evaluation

In this section we present the methodology for evaluating our approach. We first introduce the metrics and the baseline method, in Section 4.1. In Section 4.2 we describe the data used for the tests, while the discussion of the results is presented in Section 4.3. Finally in Section 4.4 we discuss efficiency related aspects.

4.1. Evaluation methodology

Query expansion is a classical method for



Sony Ericsson Xperia X10 specifications

Phone type		Smart phone
Network technology	GSM	850/900/1800/1900
	UMTS	2100/1700/900
Data		EDGE
	UMTS	Yes
	HSDPA	HSDPA 10.2 Mbit/s
	HSUPA	HSUPA 5.76 Mbit/s
Global Roaming		Yes
Design	Form Factor	Candybar
	Dimensions	4.69 x2.48 x0.51 (119 x 63 x 13 mm)
	Weight	4.76 oz (135 g)
	Antenna	Internal
	Side Keys	Volume control, Camera shutter
Display	Type	Color
	Technology	TFT
	Size	480 x 854 pixels
	Physical Size	4.00 inches
	Colors	65 536
	Touch Screen	Capacitive

Figure 3. Technical specifications snippet.

improving the retrieval performance of IR techniques. For evaluating purposes, we compared results with the well-known VSM featuring TF-IDF with cosine similarity. LSI is a promising technique for indexing and retrieving documents in a low-dimensional concept space by making use of semantic connections between terms. We address queries containing implicit concepts and as such we considered LSI is an important reference for our tests. Since the proposed approach is a query expansion technique, we also compared our method against the well-known BM25 with KLD as weighting scheme for query expansion. As metric we relied on the well-known Precision/Recall curves [27].

The evaluation process was the following: For each conceptual query, candidate terms for expansion were extracted according to Definition 1. All candidates were weighted with the function presented in Definition 2 and only those terms having weights greater than average plus three standard deviations were considered for the query expansion. With the query in expanded form, all products were ranked based on their relevance to the expanded query. The relevance of a product was computed as the sum of weights of the query expansion terms appearing in the unstructured data of the product. Products for which product reviews have been previously tagged by domain experts with respect to prevalent concepts in the domain were used as a gold standard.

4.2. Data

For our tests we analyzed product data from the field of cell phones. If when it comes to structured data, the only possibility is to use technical

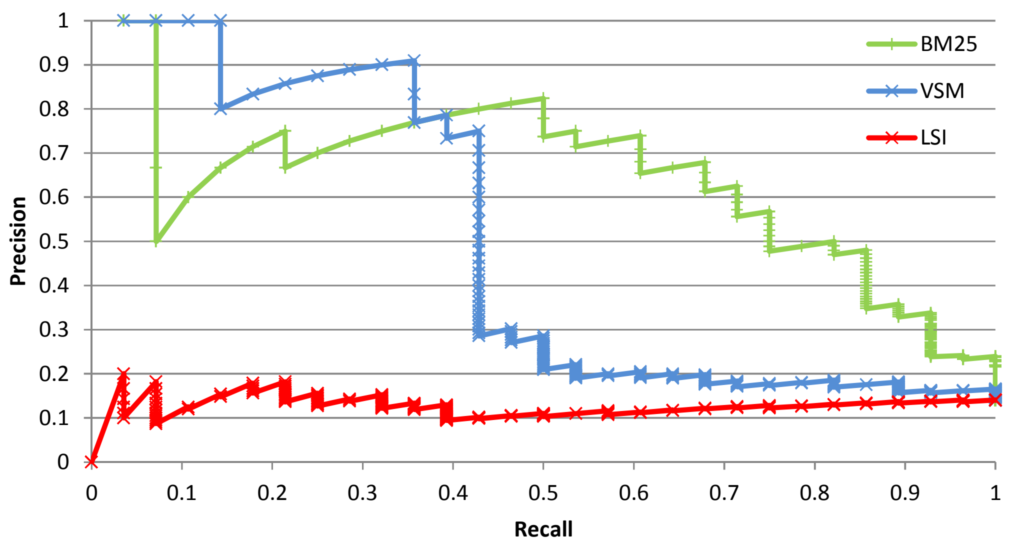


Figure 4. 'Business' - LSI vs. VSM vs. BM25.

specifications of the products (like the ones presented in Figure 3), text documents come in more flavors like for example editor's reviews, user reviews or blogs. Analyzing these information sources we observed that they offer different perspectives of the products. If editor's reviews presented the features and facts in a more objective manner, with extensive but field-relevant vocabulary, the user comments were smaller in size, concentrated on a reduced number of features, and were strongly influenced by the user's interests and point of view towards the entity. Blogs were even more emotional than user reviews making sentiment analysis an absolute requirement. Sentiment analysis however remains very unreliable when the text uses slang, sarcasm, emoticons, prolonged letter usage, capitalization, punctuation, etc. For this reason, we performed the query expansion process on a collection of 350 products with the corresponding technical specifications and 500 editor's reviews. The data has been crawled from phonearena.com, a top Web publication in the field.

The quality of the expansion terms was tested on the more challenging user reviews, to question suitability of the expansion terms for non-expert typical user language. The test set collection, comprised 200 user reviews regarding the latest cell phones, we crawled from CNET (http://www.cnet.com - a leading technology oriented Web site offering large amounts of both editor and user reviews for different products). These reviews were manually labeled by experts in the field, as either being relevant or not with respect to three most important[3] features: 'business', 'social networking' and 'camera'. We chose these features to cover different levels of clarity regarding the meaning of the query terms: 'business' represents ambiguous, classical concepts; 'social networking' stands for emerging concepts with well-defined use and finally 'camera' represents clear cut technical characteristics.

4.3. Discussion of the results

4.3.1. The baseline methods

First we tested the base line methods i.e. VSM with TF-IDF, LSI and BM25 with the available data. To our surprise, LSI always obtained poor results even compared to VSM (see Figure 4). Varying the number of dimensions for LSI (we evaluated with 10, 20 and 100 dimensions which according to [4] typically provide for good results) for all our test scenarios, didn't bring any improvements. The reason for this behavior is the small amount of data available for training the LSI. The collection of 500 documents seems rather limited for the latent semantics needs. Editor's reviews are rather scarce resource, so we then increased the document base for LSI to 6000 documents, supplementing with user reviews. However, user generated documents do not offer similar advantages as editor's reviews do. Even with this large collection, LSI is still unable to achieve notable results. Collecting editor's reviews

[3] http://tech.uk.msn.com/features/photos.aspx?cp-documentid=149711759

over long periods of time is also not a solution. The cell phone domain is a great example in showing how fast concepts evolve with time.

The TF-IDF based VSM retrieved all the products for which the conceptual query is explicitly mentioned in the description of products. This provided for quite good precision for low recall rates. But the precision deteriorated heavily in the case of products for which the query concept is only implied in the description. In the case of conceptual query ‘business’ presented in Figure 4, VSM achieved good precision up to a recall of about 40%. The behavior of VSM becomes clear after taking a closer look at the data: 43% of the reviews the experts labeled as relevant towards the ‘business’ concept, explicitly mentioned the conceptual feature. VSM identified with a high precision exactly these documents. It is interesting to notice that there was a drop in precision at a recall of about 15%. The reason for this drop is that the word ‘business’ appeared in the description of some non-business products, e.g., “allows you to locate businesses nearby” tricking VSM into retrieving the product as relevant.

The BM25 ranking model assigns weights to all terms according to the KLD probabilistic weighting scheme. Only terms having the KLD weight above a certain threshold are used for expansion. In order to establish this threshold, we conducted a series of tests. Expanding the query with terms weighting more than the average KLD weight of all terms, provided the best results in terms of precision and recall for BM25. For the case presented in Figure 4, BM25 with KLD identified 192 expansion terms out of which the top 10 terms were: ‘business’, ‘bold’, ‘nexus’, ‘pure’, ‘webo’, ‘she’, ‘exchange’, ‘control’, ‘offer’ and ‘storm’. In terms of precision and recall, BM25 achieved less precision than VSM in the low recall area (up to 40% recall), but compensated more than enough by obtaining pretty high precision (about 50%) for recall rates as high as 80%. Since LSI doesn’t even come close to the results of the other two techniques, in any experiment we performed, in the following graphs we will display only the more successful VSM and BM25.

Also worth mentioning is the ‘saw-tooth shape’ effect [28], common in precision/recall curves.

4.3.2. The query expansion technique

The query expansion comprises terms which have orthogonally been extracted from structured and unstructured data. But is there a real need to use both of the underlying sources? To answer this question, we evaluated the results obtained by expanding the query with terms originating from structured data only, then from unstructured data only, and then from both data sources. In Table 1 we present the query expansion terms extracted from the technical specifications, along with the top 10 out of a total of 153 terms extracted from the unstructured data.

Table 1. Query expansion terms.

Structured data	Unstructured data
phone Type	windows mobile
smart phone	business
phonebook features	work
picture id, multiple numbers	letters
Phonebook	notes
phonebook capacity	fields
	qwerty keyboard
	navigation
	outlook
	task

As shown in Figure 5, expanding the query only on the technical specifications (Structured data Query Expansion, further denoted as SQE), leads to poor results in terms of precision and recall. The same test performed with the expansion terms from the unstructured data (Unstructured data Query Expansion, further denoted as UQE) already delivers much better results. Finally, since the structured and unstructured data cover different aspects of products, by considering both data sources, Conceptual Query Expansion (further denoted as CQE) achieved even better results. Not only did the precision for low recall values drastically improve, but it was also maintained above 50% up to a recall above 90%. Also worth mentioning is the fact that at 100% recall, precision was of approximately 40%. In fact, CQE has consistently achieved better results than considering only structured or unstructured data alone for all experiments. Taking this into consideration, for the subsequent experiments we present the results for CQE only.

Comparing the results to the baseline methods (Figure 6), besides some marginal cases in low recall conditions, VSM was always dominated by CQE. On the other hand, BM25 achieved results that were quite comparable with our approach. Between the recall rates of 30% to 60% (middle area of the recall range) it even managed to obtain higher precision. However, for the low (up to 30%) and high (above

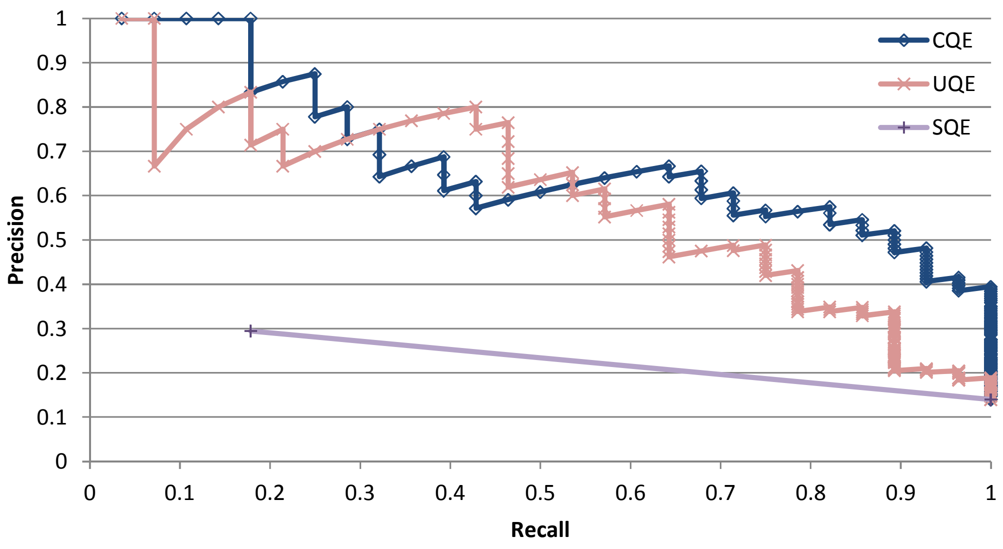


Figure 5. 'Business' – CQE vs. UQE vs. SQE.

80%) recall areas, CQE was superior. Taking a closer look at the results we observed that the behavior of BM25 was much more similar to the results we obtained by expanding the query based only on the unstructured data (UQE in Figure 5). By considering also the structured data, the precision is then improved in low and high recall areas, at the cost of precision in the middle recall area.

The positive behavior of BM25 confirms that query expansion is indeed a suitable and most powerful technique for dealing with more sophisticated queries as is the case for concepts. However, as we will present in the following section, BM25 doesn't always achieve such good results.

4.3.3. *The 'Social Networking' concept*

The 'social networking' concept is an exceptional example of how the syntactical representation of concepts can be misleading. From a linguistic perspective this concept is represented by a nominal phrase with strong syntactic relation to the 'networking/network' technical feature. This relation however doesn't reflect human perception. For instance, the concept of 'social networking' and the

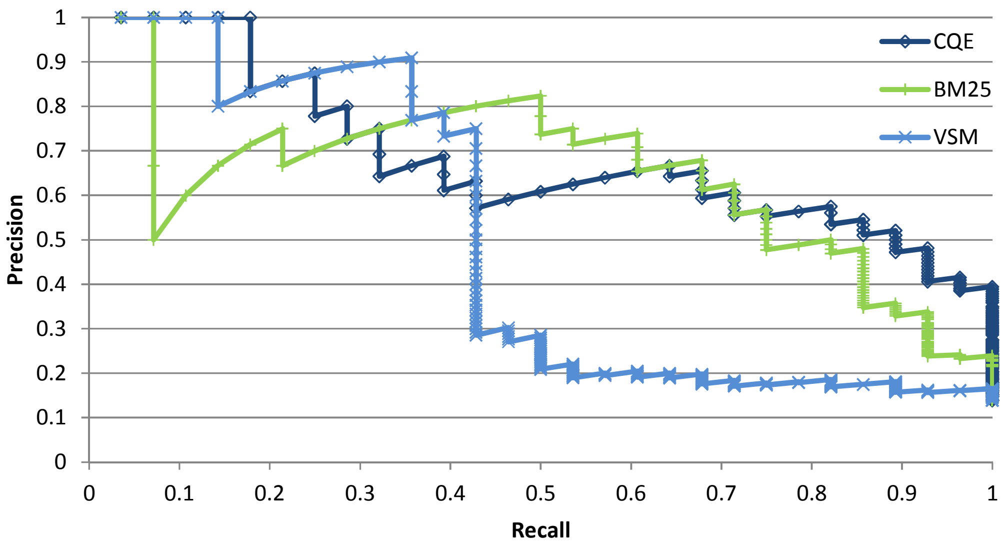


Figure 6. 'Business' – CQE vs. VSM vs. BM25.

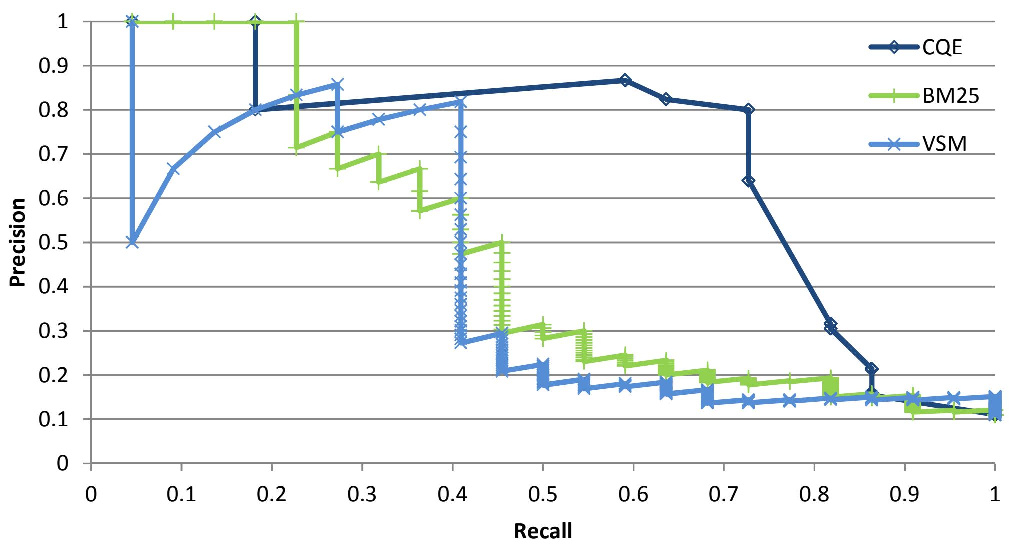


Figure 7. ‘Social Networking’ concept

‘UMTS network’ technical specification show no semantic connection whatsoever. In such cases, both VSM and BM25 have a very difficult time in providing for correct results (Figure 7). In fact looking into the behavior of both methods, we observed a very powerful topic drift towards the ‘networking’ features of products.

Since the number of such conceptual queries relying on nominal phrase constructs is not neglectable (e.g., ‘tough as nails’, ‘packed with value’, ‘multimedia marvel’ to mention just a few) we decided to take a closer look at the ‘social networking’ case. In the case of VSM, every product containing the terms ‘social’ or ‘networking’ in its textual description was considered relevant. Of course, products for which both terms co-occur in the textual description were ranked higher. This way VSM was able to identify the explicit cases, achieving some precision for the top 20% of the products. Unfortunately, the remainder of the products was ranked based on their mentioning of the term ‘networking’. This led to catastrophic precision.

BM25 was also not able to provide notable results. Similar to the case of VSM, most of the products were considered relevant, due to their description containing the term ‘networking’. As a consequence, the query expansion terms seemed to have been selected randomly. Besides ‘social’ and ‘networking’, other top expansion terms were ‘nexus’, ‘hero’, ‘release’, ‘widget’ and ‘bluetooth’.

In the case of CQE first, the set of documents containing the complete concept were selected. In a boosting fashion, this set of documents was expanded to include all other documents being highly similar to them. At this stage however the topic drift doesn’t take place for two reasons: On the one side documents selected in the first stage are relevant since they include the complete concept as a noun phrase and not a part of it; on the other side the highly selective threshold θ for the similarity between selected documents and the rest, prohibits from expanding the relevant document base with product descriptions which are only vaguely similar to relevant documents.

As shown in Figure 7, our results were, in this tricky case indeed much better than the ones achieved by VSM and BM25. The curve is also different from the ‘business’ concept. This is due to the fact that more user reviews share the same strength, i.e. the recall was improved without significantly lowering precision. Actually, it is a consequence of the reduced number of terms selected for query expansion, which characterizes this concept.

4.3.4. The ‘Camera’ technical feature

Finally, inspired by the contradicting terms obtained when considering also the ‘network’ feature as seed for expansion, the last of our tests, investigated a query purely based on a technical feature. The results show that our approach is at the present time indeed limited to expanding implicit

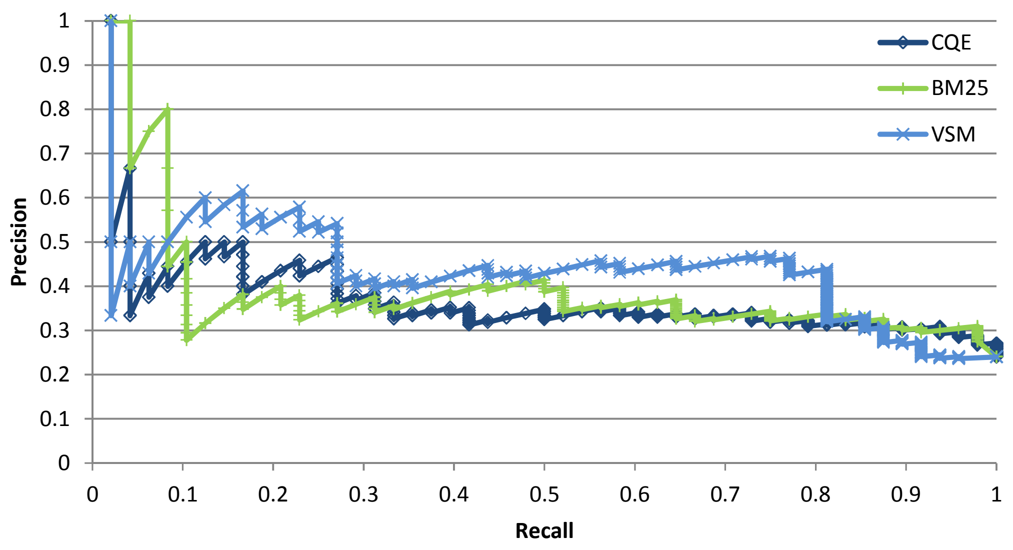


Figure 8. ‘Camera’ feature.

conceptual features (see Figure 8). The retrieval performance for technical features was merely comparable to VSM and BM25. The reason is that technical features are always explicitly mentioned in most of the editor reviews, as well as the technical specifications, regardless of the product. For example, the ‘camera’ technical feature was present in 80% of the documents from the collection used for query expansion. This clearly calls for standard techniques and our approach cannot offer any additional benefits here.

4.4. Performance results

Since query expansion should be conducted in real time, we inspected the feasibility of the proposed method also in terms of performance. As expected, the NLP techniques, respectively the chunking process and the POS detector represent important performance killers. Considering a collection of 500 documents with an average of 1500 words each, the parsing process took about 100 seconds, which is not acceptable for real-time constraints. By comparison BM25 needed about 7 seconds to prepare the documents (The preparation includes text tokenization, word stemming and building inverted indices). Even if by comparison the time of 7 seconds seems quite good, it still doesn’t fulfill real-time expectations.

Surely by optimizing the implementation of the NLP components or by means of parallelization, one could achieve better performance. The solution we propose is a system which makes use of the caching principle. Two major components are necessary: An on-line retrieval component which establishes the workflow and performs the actual Web search, and one off-line component, which maintains a database of products together with the technical specifications and corresponding editor’s reviews. The off-line component also performs the NLP tasks on the documents, storing the resulting noun phrases into the database. This reduces the computation time of a query expansion model to less than 2 seconds on regular hardware (for our tests we used a Core I7 QM with 2.4 Ghz and 16 GB RAM). Run on the same collection of preprocessed documents, also BM25, needed between 1 and 2 seconds for query expansion.

Building further on the caching solution, one could even store the expansion models (expansion terms and corresponding weights) of the most queried concepts, just by periodically inspecting query logs for the most frequent terms complying with *Observation 1*. This reduces the on-line process to ranking new products based on pre-cached models, operation which can be easily executed in real-time. A visual representation of the proposed system is presented in Figure 9.

Deciding upon the course of action is left to the query processor. Actually the most important decision it has to make is whether to expand the input query or not. This decision is entirely based on *Observation 1*: If the query term is an attribute, or appears often in structured data, then performing SQL on the database is enough. If the term occurs

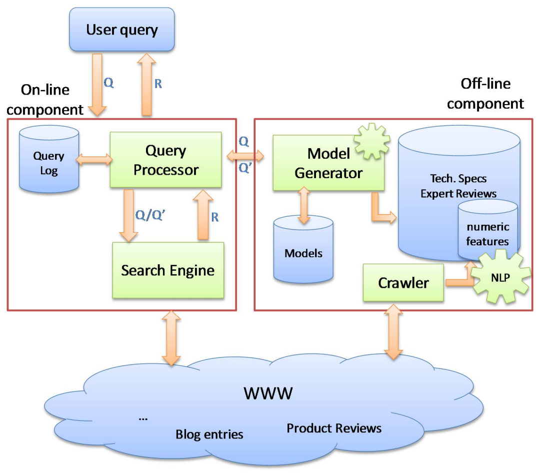


Figure 9. Proposed system architecture.

rarely in the structured data, and not too often in the unstructured data, then query expansion is performed. Otherwise, classical web search is performed with the original query.

5. Related work

Recently, several search engines have been proposed, which can retrieve products even if the query keywords don't match the product tuples in the database [17][18]. Such engines extract the entities which co-occur with keywords from the query, in documents on the web. But for concept driven querying this approach is likely to suffer from *incompleteness* since most of the concepts are mentioned only in a few documents. The reason is that concepts are rather implied by means of related terms. We tackle this problem by further expanding the query with terms related to the concept. Such search engines may also suffer from *impreciseness* of the results. In some of the documents the concept may be present but with a different meaning than the one intended by the user. Searching for a 'business' cell phone, one would also encounter cell phones with a description similar to '…it has GPS, so you can locate businesses nearby!'. By adding weights to the query expansion terms we are able to maintain a higher precision even for high recall.

On the other hand, approaches like [19][20] follow a query transformation technique. They translate the user query to a SQL statement to be executed on the product database. The query terms are mapped to predicates on the table attributes. This approach is able to tackle queries like 'small IBM laptop' with clear meaning (map on the size and brand attributes). However complex concepts i.e. 'business' for which the meaning is rather ambiguous, are associated with a textual predicate ('contains') over attributes like the product name or description. Again this approach suffers from *incompleteness* and *impreciseness*.

Our work is also related to the field of product feature extraction. In this context, Hu and Liu [23], introduce a method for considering product features implied through adjectives like 'heavy', or 'big'. For this purpose, they use a human labeled training set, and generate rules with association rule mining for the features and adjective mappings. As in the case of approaches translating the user query to SQL, this method is only feasible for queries where a clear-cut mapping between the query and table attributes can be performed. This is not the case for conceptual queries.

Turning to the field of concept extraction, in [24], Weld Hoffman and Wu propose Kylin, a self-supervised open information extraction technique. Kylin relies on information from Wikipedia to learn extractors for concepts. Wikipedia is only used as a seed, with the extractors being learned by means of bootstrapping on the Web and with the support of WordNet providing for the semantic term relations. But the extracted concepts are rather general and cannot cope with the closed vocabulary of product descriptions.

An interesting approach is presented in [29]. The authors build on the theory of Formal Concept Analysis and caching mechanisms to improve precision and recall for conceptual queries. A similar approach but in the context of Linked Open Data is presented in [30]. Both approaches assume that a shared, domain-specific vocabulary is available. However, in the context of web search, and especially in the case of users who can't express their needs in clear cut technical specifications, such vocabularies have to be extracted first. The method we presented in this paper is not affected by such problems as it dynamically extracts the needed vocabulary if enough structured and unstructured data is available.

6. Conclusions & future work

In this paper we presented a novel approach for supporting product search on conceptual features, combining structured product data with natural-language product reviews. Starting from the AOL query log we identified concepts as an essential building block for feature based product evaluation. The major problem with classical retrieval approaches for this task is that user reviews mostly do not mention these features explicitly, but only hint at them in more or less semantically related terms.

Starting from a small set of comprehensive editorial reviews our novel self-learning-based approach allows identifying implicit conceptual features even in short pieces of unstructured data like e.g. user reviews. In our evaluation against classical IR baselines (VSM with TF-IDF, LSI and BM25 with KLD weighting scheme) we have shown that our system definitely achieves superior results and can even deal with overlapping concepts. Indeed our approach outperforms classical IR methods especially for high recall values up to 90% where on average a precision of still over 50% has been proven.

Although for the discussion in this paper we only applied our approach to the restricted domain of cell phones, the results should be applicable in other product domains, too. We are currently in the process of experimenting with other product fields like laptops and cars to get a better intuition about the specific needs and the existence of conceptual features in different domains. Moreover, we also want to address conceptual queries with lower *consensus* over the result e.g. 'special', 'beautiful', 'best'. Such concepts have to be answered in a more personalized (or user profile-based) fashion.

References

[1] D. Felix, C. Niederberger, P. Steiger, and M. Stolze, *Feature-oriented vs. needs-oriented product access for non-expert-online shoppers*, in Proceedings of the 1st International Federation for Information Processing Conference on E-commerce, E-business, E-government, Vol. 74, pp. 399–406, 2001.

[2] K. Bischoff, C. Firan, W. Nejdl, and R. Paiu, *Can all tags be used for search?*, in Proceedings of the 17th ACM Conference on Information and Knowledge Management, 2008.

[3] G. Salton, A. Wong, and C. Yang, *A vector space model for automatic indexing*, in Communications of the ACM, Vol. 18(11), pp. 613–620. 1975.

[4] S. Deerwester, S. Dumais, G.W. Furnas, T.K. Landauer, and R. Harshman, *Indexing by latent semantic analysis*, in Journal of the American Society for Information Science, Vol. 41(6), pp. 391–407, 1990.

[5] P. Cimiano, A. Schultz, S. Sizov, P. Sorg, and S. Staab, *Explicit versus latent concept models for cross-language information retrieval*, in Proceedings of the 21st International Joint Conference on Artificial Intelligence, pp. 1513–1518, 2009.

[6] E. M. Voorhees, *Query expansion using lexical-semantic relations*, in Proceedings of the 17th International ACM Special Interest Group on Information Retrieval, pp. 61–69, 1994.

[7] P. Srinivasan, *Query expansion and MEDLINE*, in Journal of Information Processing and Management, Vol. 32(4), pp. 431–443, 1996.

[8] Y. Qiu, and H. Frei, *Concept based query expansion*, in Proceedings of the 16th International ACM Special Interest Group on Information Retrieval, pp. 160–169, 1993.

[9] S. E. Robertson and K. S. Jones, *Relevance Weighting of Search Terms*, in Journal of the American Society for Information Science, Vol. 27(3), pp. 129-146. 1976.

[10] S. Kullback and R. A. Leibler, *On Information and Sufficiency*, The Annals of Mathematical Statistics, Vol. 22(1), pp. 79-86, 1951.

[11] J. F. Engel, R. D. Blackwell, and P. W. Miniard, *Consumer Behaviour*, 6th Edition, Dryden Press, Chicago, Illinois, USA. 1990.

[12] J. Rowley, *Product search in e-shopping: a review and research propositions*, in Journal of Consumer Marketing, Vol. 17(1), pp. 20–35, 2000.

[13] J. Selke, S. Homoceanu, W. T. Balke, *Conceptual Views for Entity-Centric Search: Turning Data into Meaningful Concepts*, in Journal of Computer Science: Research and Development, vol. 27(1), pp. 65-79, 2012.

[14] G. L. Murphy, *The Big Book of Concepts*, MIT Press. 2002.

[15] F. Pukelsheim, *The Three Sigma Rule*, The American Statistician, Vol. 48, pp. 88–91, 1994.

[16] X. Zhou, J. Gaugaz, W. T. Balke, and W. Nejdl, *Query relaxation using malleable schemas*, in Proceedings of the ACM Special Interest Group on Management of Data, pp. 545–556, 2007.

[17] S. Agrawal, K. Chakrabarti, S. Chaudhuri, V. Ganti, C. König, and D. Xin, *Exploiting web search engines to search structured databases*, in Proceedings of the 18th International World Wide Web Conference, pp. 501–510, 2009.

[18] N. Zaiqing, W. Ji-Rong, and M. Wei-Ying, *Object-level vertical search*, in Proceedings of the 3rd Conference on Innovative Data Systems Research, pp. 235–246, 2007.

[19] V. Ganti, Y. He, and D. Xin, *Keyword++: A framework to improve keyword search over entity databases*, in Proceedings of the Very Large Database Endowment Vol. 3, pp. 711–722, 2010.

[20] N. Sarkas, S. Paparizos, and P. Tsaparas, *Structured annotations of web queries*, in Proceedings of the ACM Special Interest Group on Management of Data, pp. 771–782, 2010.

[21] C. Fellbaum, *WordNet: An Electronic Lexical Database*, The MIT Press, 1998.

[22] M. Hu, and B. Liu, *Mining Opinion Features in Customer Reviews*, in Proceedings of the 19th International Conference on Artificial Intelligence, pp. 755–760, 2004.

[23] B. Liu, M. Hu, and J. Cheng, *Opinion Observer: Analyzing and Comparing Opinions on the Web*, in Proceedings of the 14th International World Wide Web Conference, pp. 342–351, 2005.

[24] D. Weld, R. Hoffmann, and F. Wu, *Using Wikipedia to Bootstrap Open Information Extraction*, in Proceedings of the ACM Special Interest Group on Management of Data Vol. 37(4), pp. 62–68, 2008.

[25] C. P. Fung, J. Yu, H. Lu, and P. Yu, *Text Classification without Negative Examples Revisit*, in IEEE Transactions on Knowledge and Data Engineering, Vol. 18(1), pp. 6–20, 2006.

[26] G. Salton, and C. Buckley, *Term-weighting approaches in automatic text retrieval*, in Journal of Information Processing and Management, pp. 513–523, 1988.

[27] V. Raghavan, P. Bollmann, and G. S. Jung, *A critical investigation of recall and precision as measures of retrieval system performance*, in ACM Transactions on Information Systems, Vol. 7, pp. 205–229, 1989.

[28] C. Peters, M. Braschler, J. Gonzalo, and M. Kluck, *Comparative Evaluation of Multilingual Information Access Systems*, in Proceedings of the 4th Workshop of the Cross-Language Evaluation Forum, 2003.

[29] W. C. Cho, and D. Richards, *Ontology Construction and Concept Reuse with Formal Concept Analysis for Improved Web Document Retrieval*, in Web Intelligence and Agent Systems, Vol. 5(1), 109-126, 2007.

[30] S. Homoceanu, P. Wille, and W.-T. Balke, *ProSWIP: Property-based Data Access for Semantic Web Interactive Programming*, in Proceedings of the 12th International Semantic Web Conference, 2013.