Voice Search in Mobile Applications and the Use of Linked Open Data

Felix Burkhardt and Hans Ulrich Nägeli

Deutsche Telekom Laboratories, Berlin, Germany

[Felix.Burkhardt@telekom.de, hans.ulrich.naegeli@gmail.com]

Abstract

We describe our approach on voice seach in a mobile context by a TV guide search app that integrates linked open data to identify movies from a search query. A text parser to match keywords against vocabularies and numerical value descriptors is introduced.

Index Terms: voice search, mobile applications, query interpretation

1. Introduction

Two trends make voice search more and more important: on the one hand the widespread use of small mobile devices like smart phones, wearables and soon implants, that make pervasive computing a reality, and on the other hand the development of the TV as a central access point to multi-medial data in the center of a domestic space. Both are used to connect with internet data and do make the use of traditional point, click and type paradigms of user interaction difficult. This paper gives an overview on a mobile app that deals with voice based access to internet content for the TV-program. It's the continuation of our projects regarding voice search, after AskWiki [1] and the AutoScout24 app [2].

Since the acquisition of Siri by Apple in 2010 and successful market introduction, voice enabled search becomes more and more natural to smart phone users. Google voice search provides a similar service. Both offer, in addition to internet search, services like voice dictation, for example to dictate SMS, command and control ("*Call Peter at home, book a table in the Italian restaurant!*"), and question answering ("*What's the capital of Romania?*"), although until now primarily in English.

Under these general services, voice search systems for restricted domains have been developed. Song et al [3] describe a system to voice search for media data. While they incorporate a phonetic similarity model to increase recall on their data, we do this by hand-tuning the vocabularies with synonym entries that were detected when analyzing the user log entries. In [4], Voice search is even used to generate robust SMS text detection by matching the search query against a database of SMS template snippets. Voice search generally might be a technology to overcome problems like illiteracy and information access in the developing world, as described in [5].

For the interpretation of queries in a Question Answering application, in a first step the words must be processed by a natural language interpreting module. Such frameworks require large vocabularies and have a large footprint with respect to hardware resources and computing power. [6] use Google search and WordNet as additional information sources. In [7], categories, typed links and attributes are used to model a semantic structure between the Wikipedia articles.

2. General Voice Search Architecture

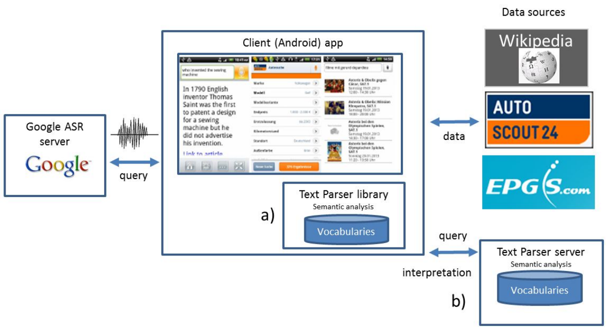

Figure 1: Overview of the general approach.

Figure 1 gives a general overview on our approach to voice search in a mobile application. The process is handled in three steps.

1. We do not use a local speech recognizer but simply interface the Google ASR service that is part of the Android system. The disadvantage of this approach is that the grammar used by the text parser cannot be used by the speech recognizer, which of course would help to get more stable recognition results, but it is more flexible with respect to interfacing different speech recognizers, e.g. when porting to Apple iPhones, and does not require license fees for the ASR component which helps with the business model.

2. The recognized text in form of a N-best list (listing the N most probable recognition results) is handled either by a text parser library, or alternatively sent to a text parser server. The first approach does not require the maintenance of a special server for the client application, which again helps to keep production costs low; the server based approach has the advantage that the vocabularies can be synchronized with the data sources.

3. The query terms can then be searched in the database which might be very different in nature, in our examples it's Wikipedia, the AutoScout24 database or aggregated electronic program guide data from EPGS.com.

Finally, the result is displayed on the app's graphical interface.

3. "Rootvole": a Text Parser library

For general processing of voice queries we developed a text parsing library named "Rootvole" that can be used to match text with semantic concepts. The algorithm was implemented in Java and can be described as a form of a parsing expression grammar [8], where we generate the expressions to be detected beforehand by regular expressions and store them in a vocabulary.

The central class is the parser class, which is instantiated as a series of vocabularies, simple text lists that describe tokens and synonyms, and value descriptors. Value descriptors describe numerical values and are characterized mainly by a unit string. Furthermore, the information whether the numerical value is postfix or prefix in relation to the unit must be stated explicitly, for example "*500 euro*" vs. "*year 2003*". Via the methods *hasLowerBound* and *getLowerBound*, the parse results for regions can be retrieved, an example would be "*200 to 500 euro*". Via the methods *isMax* and *isMin*, it is possible to determine whether the value is a lower or upper bound, an example would be "*at most 500 euro*". The parsing process itself is programmed in a two step process. Firstly, the values are extracted by detecting the unit strings and extracting nearby numbers as values. Secondly, the remaining bag-of-words set, i.e. all possible string groups given a certain context depth, is used to match against the vocabularies. We plan to release "Rootvole" as an open source project in the near future.

4. TV guide app and Linked Open Data

The TV guide android app lets the user search the TV program for upcoming movies by attributes like scheduled time, or genre, for example *"Action movies on tuesday evening"*. This information is available in the EPG data. For input like *"Movies with Clive Owen"*, the app takes a list of movies starring Clive Owen, looks each of them up in the schedule, and reports the date and time of their broadcasting. Depending on the EPG's provider, this data may be included as well. But as a secondary source, we tapped into some of the available collections of linked open data, primarily DBpedia.

The big advantage of this approach is that it is very extensible. The collections themselves offer ample information about a given movie that could be used to refine the queries (cast, duration, ratings and so on). Additionally, since the knowledge bases are linked, we can always get more data by consulting the corresponding entry in other databases.

DBpedia[1] [9] is a collection of structured information automatically extracted from Wikipedia. Its main sources are Wikipedia infoboxes; besides, it also extracts data like categorization and external links. This knowledge is represented as *Resource Description Framework* (RDF) triples, i.e. as *Subject – Relation – Object* statements. The databases are often called *graphs*, where we think of the entities – represented by an unique resource identifier (URI) – as nodes and of the relations as edges. Databases can be queried using SPARQL, a language derived from SQL.

There are obviously different possible sources for this kind of information. We chose DBpedia as our starting point for several reasons, mainly: 1. DBpedia's international nature makes it particularly suitable for a non-English application. A large German user base ensures that the entries stay up to date and are correctly localized. 2. With *DBpedia Spotlight*[2] [10], there is a powerful tool to do Named Entity recognition in a given natural language text. We thought about incorporating Spotlight as well. While promising, this approach wasn't included in the final version: Queries about the TV schedule tend to be rather explicit and well-structured, so that a more classical parser can handle them. But the use of Spotlight could prove very useful for queries in a less constrained field.

[1] http://dbpedia.org

[2] http://spotlight.dbpedia.org

The German version of DBpedia contains 20.835 objects of type *movie*, and 72.997 distinct entities *starring* in a movie – we will consider them as actors. Of course, we could add that we want only to retain starring entities that are of type *person*; but it turns out we'd lose a lot of persons that don't have this attribute (starring persons: 22.948, starring non-persons: 50.049; a quick search verifies that most of the non-persons are, actually, people). There are 201.501 x *starring* y-relations. One of the huge advantages of Linked Open Data is that different graphs can be interlinked. For a given movie, it is straightforward to find the corresponding entry in some other graph. This gets especially important if we have to deal with ambiguous movie titles (as for remakes); in this case, it's not enough to just start a new query in the next database. As additional source, we used *Freebase*[3], because it's one of the biggest collections, and because its very different data structure presented an interesting challenge. Freebase entities (called *topics*) have a property *key* for different namespaces, among them the German Wikipedia. Starting with a DBpedia key (from which we can directly deduce the Wikipedia key), we can look up the corresponding Freebase topic and get its starring actors. The two sets are then merged. We didn't, however, search for additional movies in Freebase; the assumption was that the German Wikipedia reflects the interests of a German public quite well.

Appearances	Mean	Total
Known by both	3.25	5,373
Only known by DBpedia	6.28	10,377
Only known by Freebase	8.37	13,842
Found totally	17.90	29,592

Table 1: Comparison of movie-actor information in DBpedia and Freebase. For 1,653 of 2,000 movies, both databases know at least one starring actor.

Table 1 compares the data we pulled from the two sources. These results are only to be taken as approximation. We didn't check the identities of starring actors with the same mechanism as we did for the movies. Therefore, an actor may be listed twice for the same movie, having a different name in the two knowledge bases. Some observed differences include omission of the middle name, different spelling and accentuation, different apostrophe forms.

For better performance and reliability, the TV guide app doesn't query DBpedia or Freebase directly. Instead, the query goes to a server where the extracted data is stored. This also allows for some adequate normalization of the movie titles and actor names. Of the 1781 programs appearing for one week and 28 stations, 87 movies were matched against Wikipedia in a trial run of the app.

5. Conclusions and Outlook

The next steps will be to further inquiry the use of external data sources like Wikipedia or WordNet for voice query expansion as well as experiment with the advantages and drawbacks of distributed versus local (client based) architectures. Another exciting topic in this field is the combination of explicit semantic modeling by expert ontologies with statistical approaches based on large databases like DBpedia or Freebase.

[3] http://www.freebase.com

6. References

[1] F. Burkhardt and J. Zhou, "Askwiki: Shallow semantic processing to query wikipedia," *Proc. EUSIPCO*, 2012.

[2] F. Burkhardt, J. Zhou, S. Seide, T. Scheerbarth, B. Jäkel, and T. Buchner, "Voice enabling the autoscout24 car search app," *Proc. of the ESSV, Elektronische Sprachsignalverarbeitung, Bielefeld*, 2013.

[3] Y.-I. Song, Y.-Y. Wang, Y.-C. Ju, M. Seltzer, I. Tashev, and A. Acero, "Voice search of structured media data," *International Conference on Acoustics, Speech and Signal Processing*, 2009.

[4] Y.-C. Ju and T. Paek, "A voice search approach to replying to sms messages in automobiles," *Proc. Interspeech*, 2009.

[5] E. Barnard, J. Schalkwyk, C. Van Heerden, and P. Moreno, "Voice search for development," *Proc. Interspeech*, 2010.

[6] D. Buscaldi and P. Rosso, "Mining knowledge from wikipedia from the question answering task," *Proceedings of the 5th International Conference on Language Resources and Evaluation*, 2006.

[7] M. Völkel, M. Krötzsch, D. Vrandečić, H. Haller, and R. Studer, "Semantic wikipedia," in *Proceedings of the 15th international conference on World Wide Web, WWW 2006, Edinburgh, Scotland, May 23-26, 2006*, MAY 2006. [Online]. Available: http://www.aifb.uni-karlsruhe.de/WBS/hha/papers/SemanticWikipedia.pdf

[8] B. Ford, "Parsing expression grammars: A recognition based syntactic foundation," *Proceedings of the 31st ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages. ACM*, 2004.

[9] C. Bizer, J. Lehmann, G. Kobilarov, S. Auer, C. Becker, R. Cyganiak, and S. Hellmann, "Dbpedia - a crystallization point for the web of data," *Web Semant.*, vol. 7, pp. 154–165, September 2009. [Online]. Available: http://portal.acm.org/citation.cfm?id=1640541.1640848

[10] P. N. Mendes, M. Jakob, A. Garcia-Silva, and C. Bizer, "Dbpedia spotlight: Shedding light on the web of documents," in *Proceedings of the 7th International Conference on Semantic Systems (I-Semantics)*, 2011.