Proceedings of the Twenty-Ninth International
Florida Artificial Intelligence Research Society Conference

# Building User Interest Profiles Using DBpedia in a Question Answering System

**Jonathan Bergeron** and **Aron Schmidt**
Department of Computer Science, Laval University, Quebec, Canada, jonathan.bergeron.6@ulaval.ca,
Department of Software Engineering, Lakehead University, Thunder Bay, Canada, aschmid1@lakeheadu.ca

**Richard Khoury** and **Luc Lamontagne**
Department of Software Engineering, Lakehead University, Thunder Bay, Canada, rkhoury@lakeheadu.ca
Department of Computer Science, Laval University, Quebec, Canada, luc.lamontagne@ift.ulaval.ca

**Abstract**

In this paper, we explore the idea of building an adaptive user interest model. Our proposed system uses implicit data extracted from a user's search queries to select categorical information from DBpedia. By combining the categorical information collected from multiple queries and exploiting the semantic relationships between these categories, it becomes possible for our system to build a model of the user's interests. This model is designed to be responsive to changes in the user's interests over time by including concepts of aging and expiration. Our system also includes mechanisms to pinpoint the correct categories when an ambiguous term is queried. We evaluated our system using a predefined set of test queries and shown to correctly model user short term and long term interests.

## Introduction

As the amount of information on the World Wide Web (WWW) is increasing at an exponential rate, users have to spend much more time searching through it to find the exact information they need. Even though today's search engines work well using keyword-based approaches, they fail to account for the different information needs of different users asking similar queries. For example, a user who has lived in a foreign country for several years will not have the same information needs about that nation as one who has never traveled there, and a user with an interest in national economic policy will expect different answers on questions about that country from one with an interest in grassroots social movements. One promising way of alleviating this problem is to create a model of the user interests, and to use this model to infer the user's intents and provide more appropriate search results.

In this paper, we develop a system to model user interests in the context of an information-retrieval and question-answering system. Since studies have shown that users of these systems are reluctant to provide explicit field of interest (Carroll and Rosson 1987), we designed our system to only use implicit information that can be inferred about the user from the queries asked. To do so, the application extracts important keywords from each query and uses those terms to query DBpedia for category information. DBpedia is an effort aiming to extract structured content from the information created as part of the Wikipedia project (University of Leipzig 2007), and offers a direct mapping from terms to categories. The user interest model is designed so that semantically related categories can reinforce each other. It updates itself by applying an aging function, which enables it both to respond to long-term changes in user interests and to filter out noise from short-term searches. We tested our system with a set of test queries, and evaluated the short term and long term accuracy of the interest model generated qualitatively.



This paper is structured as follows. The paper gives the background information of the project and discuss about related work in the domain of user modeling. Then DBpedia is summarized. After, the paper explains how we use DBpedia to map question's words to categories. We then disclose how we build and update a user interest model from the extracted categories. The next section explains methods used to evaluate the model and expose the results. Concluding thoughts are provided in the last section.

## Background

This project is part of an ongoing research initiative to develop an integrated framework to support the construction of Intelligent Virtual Assistants (IVAs) (Lamontagne et al. 2014). A rich user interest model is an important component for such a system because of its disambiguation and inference potential. However, making a user interest model using only information queries is not an easy task. In recent years, most of the research on user interest modeling had the same goal: to improve the ranking of documents in web searching(Jeh and Widom 2003)(Haveliwala 2002)(Chen, Chen, and Sun 2002). This context offers a lot of ways to gather additional implicit information to build a user interests model(Kelly and Teevan 2003). For example, a model can use the search history of the user(Speretta and Gauch 2005), browsing history (Sugiyama, Hatano, and Yoshikawa 2004), or click-through data(Qiu and Cho 2006). Given this additional information, a bag of words (BOW) model (Sebastiani 2002) can stand in for a user interest model fairly well. However, when this user history is not available, as is our case, this method becomes ineffective.

Our approach instead takes advantage of DBpedia to infer and track the interests of the user by Wikipedia categories. Categories are a much richer representation than words because they represent concept as a whole.

Methods using categories or Wikipedia concept are often aimed at clustering documents or identifying documents topic(Huang et al. 2009)(Schonhofen 2006)(Hu et al. 2009). But some work has been done to use categories to successfully model user interest. For example, in (Min and Jones 2011), the authors use Wikipedia clusters to model the user interests from user's queries and click-through document set. They use the OkapiBM25 weights to rank the importance of the terms and use the top terms in a document as key term to represent user interests. However, in our context, the OkapiBM25 cannot work because of the lack of available data. In (Ramanathan, Giraudi, and Gupta 2008), the authors built an implicit model where interests are represented by Wikipedia article the user has visited. But they do not take into account the existing links between Wikipedia articles, therefore loosing precious information. In our approach we take advantage of the semantic network of DBpedia resources and categories to reinforce the model.

Another major issue that our system will need to deal with is that the context the user works in can change over time. Thus, the need of the model to adapt dynamically to that context is essential. According to (Rich 1983), the user space model can be represented by a three-dimensional space, where one of its dimensions is long term vs. short term characteristics. By contrast, our approach aims to track both long-term and short-term together, as both can help in dealing with the user context, as it has been shown in (Bennett et al. 2012). Moreover, we add an aging function in order to reduce the importance of categories over time.

Although our method is part of an on-going project, it is worth mentioning that this method is completely independent of the project and can be used in any question-answering system, or in any project having only access to a little amount of information about user interests.

## DBpedia

Our application takes advantage of Semantic Web technologies to searching through ontological data in DBpedia. Like all other Semantic Web data, DBpedia is organized into a directed graph of resources linked by predicates. Resources in DBpedia correspond to the actual Wikipedia pages and share the same page names as them. The important DBpedia resources that we use are: Redirect, Disambiguation Page, Article, Page Link and Category. All the listed DBpedia resources, with the exception of Category, have the same URI prefix, making them indistinguishable from each other. One way to determine the resource type is by querying for a property that is unique to one type of resource. For example, if a resource is queried with the predicate `dbpedia-owl`:wikiPageDisambiguates and that query returns a list of resources then we know that the queried resource is a disambiguation page. We can then safely assume that the resources returned by the query are all articles.

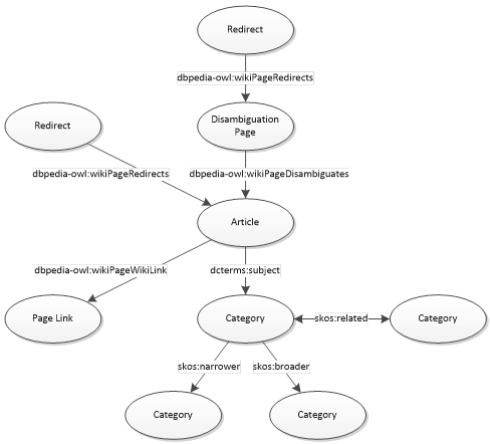


Figure 1: Navigating DBpedia Resources

DBpedia resources have many properties linked by predicates. The properties that are relevant to category extraction are shown in figure 1. Documentation for an RDF predicate can generally be found by visiting its URI in a web browser. Similarly, visiting a resource URI in a web browser will bring up an About Page displaying all of the properties for that resource that are available on the public DBpedia server. Note that page links are not available on the public DBpedia server.

We are using the Simple Protocol and RDF Query Language (SPARQL) for querying Resource Description Framework (RDF) data. SPARQL is SQL-like language designed for pattern matching against RDF triples. SPARQL queries are used in our application for navigating across the predicate links connecting resources, effectively allowing the application to traverse an RDF resource graph.

## Extracting Question Categories

The category extraction process is the first step to build a model. It occurs when the user has made a query to the system. Here's an overview of the process.

1. Extract significant terms from the question.
2. Use the terms to search DBpedia for matching articles.
3. Disambiguate if needed articles and save categories to the model.

### Extracting significant terms

We begin by performing stopword removal and word stemming on the user's question, and extracting tokens from the processed question string which will be used to formulate the DBpedia queries. We create query terms composed of trigram, bigram and unigram of the remaining tokens. We do this because multi-token terms have more meaning than single token term and are less likely to return a disambiguation page when querying DBpedia. If a trigram or bigram

search is successful, we do not further use those token in other tokens, to avoid creating ambiguity in our query results.

### Searching DBpedia

After extracting the set of terms from the sentence, the search handler queries the DBpedia graph with the trigram, bigram and unigram term, in that order, to retrieve relevant articles. We save the categories of the articles along with any other useful information, such as article page links (which will be used later on for category disambiguation filtering). We then filter out from this set any category that has less than two or more than 10,000 articles pointing it. These thresholds were set to account for the varying granularity of category coverage in DBpedia. Too broad categories have no discriminating potential to determine user interest. On the other hand, too narrow categories tend to be isolated in the category graph and not useful for interest inference; they would simply be eliminated by our aging process later on.

### Article Disambiguation

It is sometimes the case that ambiguous terms are extracted from the user's question. When using an ambiguous term to query DBpedia, a disambiguation page will be returned. Selecting the appropriate disambiguation link(s) to select from that page requires understanding the context of how the term was used in the original question. We designed a disambiguation filter for that purpose that scores the target of each link on the disambiguation page, so that the top-scored link will be the appropriate one to use given the user's original question. The formula for that filter is:

$$(\alpha \times pCount) + (\beta \times lCount) + (\gamma \times cCount) + (\mu \times sim(q,p))$$

where

- $pCount$ is the number of DBpedia pages belonging to another term in the same question that link to the target page.
- $lCount$ is the number of links belonging to another term in the question that link to the target page.
- $cCount$ is the number of categories belonging to another term in the question that match a category belonging to the target page.
- $sim(q,p)$ is the cosine similarity between the question words and the short abstract of the target page (after stopword removal and word stemming).
- $\alpha, \beta, \gamma$ and $\mu$ are weights.

Our initial experimenting found that $pCount$ is the best predictor of the correct disambiguation page for the question context followed by $sim(q,p)$ and $lCount$ while $cCount$ is the weakest predictor. Based on these findings, the default values of the coefficients are: $\alpha = 100$, $\beta = 10$, $\gamma = 1$ and $\mu = 10$. It is also worth noting that the filter will only select the top-scoring page only if its score is greater than a confidence threshold. If it is below that threshold, the filter selects all disambiguation pages.

## User Interest Model

The primary goal of computing the set of question categories is to use them as building blocks for the user interest model. However, the user interest model is designed to be more than just a collection of categories. Categories in the model are linked together to form semantically-related groups whose members are boosted whenever one member of the group is updated. The model also responds to long term changes in user behaviour by aging stale categories. To give an overview of the model, here is the pseudo code of what happens when a question is asked.

**Algorithm 1** Update User Interest Model

```
Q ← User question
L ← Long term user model categories
S ← Short term user model categories
K = extractSignificantTerms(Q)
P = searchDBpedia(K)
C = ∅
for DBpediaPage page in P do
    if page is ambiguous then C = C ∪ disambiguationFilter(page)
    else
        C = C ∪ page categories
    end if
end for
Set all categories score in C to initial score.
BoostRelatedCategories(U,C)
S = S ∪ C
PromoteDemoteExpireCategories(S,L)
for Category category in S do
    category score -= shortTermAgingFunction
end for
for Category category in L do
    category score -= longTermAgingFunction
end for
```

### Expanding Categories

One of our first findings during the development of our system was that the granularity of categories can vary greatly between similar articles. It is often the case that one Wikipedia article has been assigned to a generic category while another article about a related topic has been assigned to a more specific category. We find it desirable for general categories to have their scores influenced by the specific categories they contain. Furthermore, DBpedia includes the relationships between categories as SKOS links, such as the `skos:broader`, `skos:narrower`, and `skos:related` predicates as previously shown in figure 1.

This can be demonstrated with an example. Suppose that the user is interested in the country of Syria. The user first asks a general question about Syria that contains the term "Syria" which leads to the article dbpedia:Syria, and its categories are added to the user interest model. The second question more specifically asks about the "Demographics of Syria" and that page's categories are also added to the model. The third question is even more specific and asks about the "Refugees of the Syrian Civil War". Figure

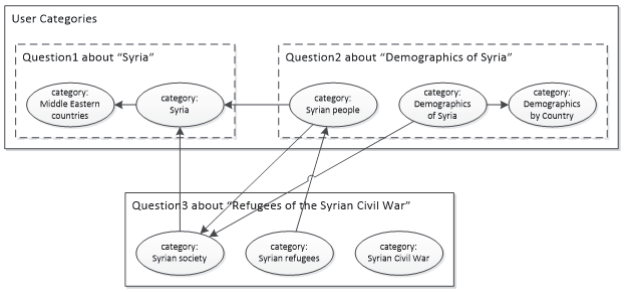


Figure 2: Returned Categories(circles) per question(dotted squares) with `skos:broader` links (arrows)

2 shows a sample of the actual categories found for each question. The categories in the model have been grouped by their question of origin for clarity. Notice that even though all three questions are about the topic of Syria, there are no categories in common between them. However, direct links can be found using the unidirectional `skos:broader` relation, marked as arrows in the figure. It thus becomes immediately clear not only that the questions are related, but what the hierarchical relationship between them is. Note that including the other SKOS relations would make the links bidirectional.

Linking the categories found for the questions produces a sub graph of the DBpedia resource graph. An important feature of this graph is that it is possible to form a path from the more specific categories about Syria to the general category of `category:Syria`. The exception to this, `category:Syrian_Civil_War`, has no direct links to the other categories belonging to these questions. However, If we allow for 2 degrees of separation in our search through the DBpedia resource graph, we find that `category:Syrian_Civil_War` links to a category not found in the questions, `category:Politics_of_Syria`, which then links to `category:Syria`. The links between categories are found by recursively querying DBpedia with the desired SKOS predicates to retrieve a list of expansions for each category. Presently, we expand a category by taking the union of all three SKOS relations. Assuming that the DBpedia resource graph is represented by an adjacency list with a branching factor b, and we assign a depth limit d, to represent the maximum degrees of separation allowed between category links, then both the space and time complexity of traversing the graph to find all the links for a single category is $O(b^d)$.

## Boosting Related Categories

When a new question is asked, the expansions list for each relevant category is retrieved. Then the sub graph is created by adding edges between categories and their expansions. Next, the sub-graph is traversed starting from each question category. When a link to a category already in the user profile is found, the score of that user category is boosted by the score of the initial question category. This boosting

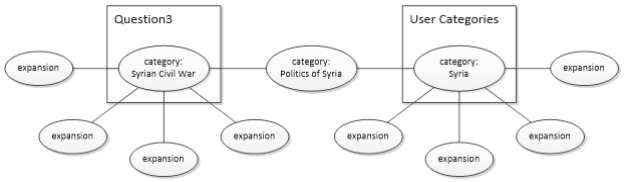


Figure 3: Indirect Path between Syrian Civil War and Syria after Expansions

only occurs once per question, so if more than one question category links to the same user category, the user category score is only boosted once by the maximum of the scores.

Note that the category expansions are all included in the graph. It allows for more traversable paths to be created with fewer DBpedia queries. To use the previous Syria example, if we set the expansion depth to 1 and expand the question category `category:Syrian_Civil_War` then several categories not found in either the question set or user set will be added to the graph, including for example `category:Politics_of_Syria`. Hence, when the user category `category:Syria` is expanded, an edge will be added from `category:Syria` to `category:Politics_of_Syria`. Assuming that all the edges are bidirectional, it is now possible to traverse from `category:Syrian_Civil_War` to `category:Politics_of_Syria` to `category:Syria` as long as the $traverse\_depth$ parameter is greater than the $expand\_depth$ parameter. Figure 3 shows a diagram of this path including placeholders for the irrelevant expansions.

## Category Aging

It is often the case that a single question produces a diverse set of categories, many of which are not actually relevant to the main topic of interest. Over time, irrelevant categories will accumulate and clutter up the user interest model. Another case to consider is that it is possible for the interests of the user to change over time. In this case, the categories associated with the older interests should be given lower priority.

A part of the solution is to create two types of categories: short term and long term categories. Typically, short term categories represent ephemeral interest of the user. Conversely, long term categories hold the constant interest of the user. Our method includes a mechanism that promotes, demotes, or expires categories depending on their score and certain thresholds. Those thresholds have been set empirically. A category that expires is a short term categoy removed from the model. Promoting and demoting simply refers to short term category becoming long term category and vice-versa.

Additionnaly, our method decreases the score of categories not related to a user's question. We refer to this as aging. Our category score includes an age variable

defined by an aging function. This function keeps track of the number of questions asked by the user since the category's score was last increased, $l_n$. We have defined two aging functions, one for long term categories and one for short term categories. For long term categories we use a simple constant function, $A(n) = -1$, in order to have a slower drop; a user's long-term interests should be stable, even after a period working on something unrelated. Short term categories are more likely to be noise and should be eliminated as fast as possible, so a squared function is used that will increase the penalty in relation to the time since the category was last increased. We define the aging function as: $A(n) = -\sqrt{l_n}$.

## Experimentation

We tested our system using a set of 170 questions selected from the TREC 2007 QA track. These questions cover a large variety of topics, but we selected them to keep mainly questions about companies, political figures, and major events, with a few unrelated "noise" questions. The questions are grouped by topic so that an interest can be established in the model before the topic is changed.

For every question asked, the application generates a report to keep track of the new categories found for the question and the state of the user interest model after the new categories have been added. The generated report is evaluated qualitatively based on the following criteria.

- Are the highest ranking categories found for a question similar to the topic of the question?
- Do the unrelated categories from individual questions get filtered out over time?
- After a group of questions about the same topic are asked, are the highest ranked categories in the user interest model similar to the common topic?

Samples of the top 10 categories at different points of the experiment are given in Table 1, Table 2, and Table 3. The short term and long term categories are separated by a line in the results.

In the light of our results, the model is perfectly adapting itself to each of the topics. Indeed, we can observe that the top category of each of the topic is `Management`, this is not surprising because all three topics have in common the larger concept of management. Also, even though the questions have been asked consecutively, the model quickly adapt itself to new subject. We can observe that in each sample the short term categories are specific to the topic of interest. Also, the short term categories of the last topic does not appear in the top ten of the model, showing the adaptive capability of the user interest model. After 170 questions, the long term categories of the model represent perfectly the general topic of the TREC 2007 questions. Also, it is important to note that noise question categories have been eliminated quickly.

Table 1: Categories after 170 Questions

| Rank | Category | Score |
|---|---|---|
| 1 | Management | 198.5 |
| 2 | Management_occupations | 153.5 |
| 3 | Companies_listed_on_the_New_York_Stock_Exchange | 115.6 |
| 4 | Presidents | 96.0 |
| 5 | Republics | 95.5 |
| 6 | Positions_of_authority | 86.0 |
| 7 | Heads_of_state | 81.0 |
| 8 | Titles | 76.0 |
| 9 | Middle_Eastern_countries | 70.5 |
| 10 | Member_states_of_the_United_Nations | 63.5 |
| 11 | Human_geography | 61.8 |
| 12 | Liberal_democracies | 54.6 |
| 13 | Gas_stations_of_the_United_States | 54.1 |
| 14 | International_volunteer_organizations | 54.0 |
| 15 | Human_habitats | 54.0 |
| 16 | Landscape_ecology | 54.0 |
| 17 | Environment | 54.0 |
| 18 | Systems_ecology | 54.0 |
| 19 | Development_charities | 54.0 |
| 20 | Comtism | 54.0 |

Table 2: Categories after 7 questions about IMG

| Rank | Category | Score |
|---|---|---|
| 1 | Management | 65.0 |
| 2 | Management_occupations | 65.0 |
| 3 | Sports_event_promotion_companies | 54.0 |
| 4 | Sports_agents | 54.0 |
| 5 | Sports_management_companies | 54.0 |
| 6 | Public_relations | 50.0 |
| 7 | Political_geography | 50.0 |
| 8 | Group_theory | 44.0 |
| 9 | Symmetry | 41.5 |
| 10 | Companies_based_in_Cleveland,_Ohio | 41.5 |

Table 3: Categories after 7 questions about US Mint

| Rank | Category | Score |
|---|---|---|
| 1 | Management | 78.0 |
| 2 | Management_occupations | 68.0 |
| 3 | Numismatics | 59.0 |
| 4 | Political_geography | 54.0 |
| 5 | Mints_(currency) | 54.0 |
| 6 | Collecting | 50.1 |
| 7 | Science_education | 47.6 |
| 8 | Mints_of_the_United_Kingdom | 47.6 |
| 9 | Former_British_colonies | 46.5 |
| 10 | Banks_of_the_United_Kingdom | 45.1 |

Table 4: Categories after 7 questions about 3M

| Rank | Category | Score |
|---|---|---|
| 1 | Management | 88.5 |
| 2 | Companies_listed_on_the_New_York_Stock_Exchange | 83.1 |
| 3 | Management_occupations | 76.0 |
| 4 | Nanotechnology_companies | 50.0 |
| 5 | Telecommunications_equipment_vendors | 50.0 |
| 6 | Multinational_companies_headquartered_in_the_United_States | 50.0 |
| 7 | Renewable_energy_technology | 50.0 |
| 8 | National_Medal_of_Technology_recipients | 50.0 |
| 9 | Companies_in_the_Dow_Jones_Industrial_Average | 50.0 |
| 10 | 3M | 50.0 |

## Conclusion

We developed a system to dynamically model user interests based only on user search queries. The terms of the user's question are used to query DBpedia and retrieve a single page, or a disambiguation page from which we extract the correct link using our disambiguation filter. Once the page is found, its categories are added to the user interest model, and existing semantically-related categories in the model have their scores boosted. The model then applies an aging function to its categories for the dual purpose of filtering out those the user has lost interest in and for noise reduction. The accuracy of the model was evaluated against a set of test questions and is shown to both properly reinforce categories related to topics in common with a group of questions and respond to long term changes in the common topics of the questions asked. Thus, we conclude that our system for modeling user interests shows promise and, after more quantitative testing and further improvements; machine learning algorithm could be applied to determine better weights of the disambiguation filter formula and query expansion process could be implemented to retrieve more categorie, may become part of any system that needs to track and adapt to changing user interest.

## References


Bennett, P. N.; White, R. W.; Chu, W.; Dumais, S. T.; Bailey, P.; Borisyuk, F.; and Cui, X. 2012. Modeling the impact of short-and long-term behavior on search personalization. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, 185–194. ACM.

Carroll, J. M., and Rosson, M. B. 1987. *Paradox of the active user.* The MIT Press.

Chen, C. C.; Chen, M. C.; and Sun, Y. 2002. Pva: A self-adaptive personal view agent. *Journal of Intelligent Information Systems* 18(2-3):173–194.

Haveliwala, T. H. 2002. Topic-sensitive pagerank. In *Proceedings of the 11th international conference on World Wide Web*, 517–526. ACM.

Hu, X.; Zhang, X.; Lu, C.; Park, E. K.; and Zhou, X. 2009. Exploiting wikipedia as external knowledge for document clustering. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, 389–396. ACM.

Huang, A.; Milne, D.; Frank, E.; and Witten, I. H. 2009. Clustering documents using a wikipedia-based concept representation. In *Advances in Knowledge Discovery and Data Mining*. Springer. 628–636.

Jeh, G., and Widom, J. 2003. Scaling personalized web search. In *Proceedings of the 12th international conference on World Wide Web*, 271–279. ACM.

Kelly, D., and Teevan, J. 2003. Implicit feedback for inferring user preference: a bibliography. In *ACM SIGIR Forum*, volume 37, 18–28. ACM.

Lamontagne, L.; Laviolette, F.; Khoury, R.; and Bergeron-Guyard, A. 2014. A framework for building adaptive intelligent virtual assistants. In *Proc. of the 13th IASTED Intl Conf. on Artificial Intelligence and Applications*, 17–19.

Min, J., and Jones, G. J. 2011. Building user interest profiles from wikipedia clusters.

Qiu, F., and Cho, J. 2006. Automatic identification of user interest for personalized search. In *Proceedings of the 15th international conference on World Wide Web*, 727–736. ACM.

Ramanathan, K.; Giraudi, J.; and Gupta, A. 2008. Creating hierarchical user profiles using wikipedia. *HP Labs*.

Rich, E. 1983. Users are individuals: individualizing user models. *International journal of man-machine studies* 18(3):199–214.

Schonhofen, P. 2006. Identifying document topics using the wikipedia category network. In *Web Intelligence, 2006. WI 2006. IEEE/WIC/ACM International Conference on*, 456–462. IEEE.

Sebastiani, F. 2002. Machine learning in automated text categorization. *ACM computing surveys (CSUR)* 34(1):1–47.

Speretta, M., and Gauch, S. 2005. Personalized search based on user search histories. In *Web Intelligence, 2005. Proceedings. The 2005 IEEE/WIC/ACM International Conference on*, 622–628. IEEE.

Sugiyama, K.; Hatano, K.; and Yoshikawa, M. 2004. Adaptive web search based on user profile constructed without any effort from users. In *Proceedings of the 13th international conference on World Wide Web*, 675–684. ACM.

University of Leipzig, University of Mannheim, O. S. 2007. Dbpedia.