Improving Wikipedia with DBpedia

Diego Torres
LIFIA, Fac. Informatica, UNLP
50 y 115, S/N
1900 La Plata, Argentina
diego.torres@lifia.info.unlp.edu.ar

Pascal Molli
LINA, Nantes University
2, rue de la Houssiniere
44322 Nantes, France
Pascal.Molli@univ-nantes.fr

Hala Skaf-Molli
LINA, Nantes University
2, rue de la Houssiniere
44322 Nantes, France
Hala.Skaf@univ-nantes.fr

Alicia Diaz
LIFIA, Fac. Informatica, UNLP
50 y 115, S/N
1900 La Plata, Argentina
alicia.diaz@lifia.info.unlp.edu.ar

ABSTRACT

DBpedia is the semantic mirror of Wikipedia. DBpedia extracts information from Wikipedia and stores it in a semantic knowledge base. This semantic feature allows complex semantic queries, which could infer new relations that are missing in Wikipedia. This is an interesting source of knowledge to increase Wikipedia content. But, what is the best way to add these new relations following the Wikipedia conventions? In this paper, we propose a path indexing algorithm (PIA) which takes the resulting set of a DBPedia query and discovers the best representative path in Wikipedia. We evaluate the algorithm with real data sets from DBpedia.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Selection process; E.2 [**Data Storage Representation**]: Linked Data; E.1 [**Data Structures**]: Graphs and networks—*Bipartite Graph*

General Terms

Algorithms, Experimentation

Keywords

Path query, DBpedia, Wikipedia

1. INTRODUCTION

Semantic web is growing fast and bring better search and navigability on the web. It is mainly built from meta-data extracted from the social web. A good example is DBpedia [2], a knowledge base extracted from Wikipedia[1] infoboxes and wiki markups. DBpedia supports complex semantic queries such as "people born in Berlin before 1900".

The semantic knowledge base of DBpedia is composed of concepts that related by properties. The semantic feature of DBpedia makes possible to deduce new semantic data. For instance, the previous query could produce people that cannot be retrieved by navigating from Berlin article in Wikipedia. There is a gap in information between Wikipedia and DBpedia. This information gap is expressed at level of semantic relations.

In this paper, we introduce an approach that help to inject DBpedia information in Wikipedia. To achieve this, we need first to find out relations in DBpedia that are missing in Wikipedia, and then to find the way to add them following the Wikipedia community usage. The question is how a semantic relation in DBpedia should be expressed in Wikipedia. In order to do this, we use existing relations in Wikipedia as the source to learn how missing relation can be represented following Wikipedia community conventions. Relations in Wikipedia are represented as a navigational path, the main issue of this approach is to discover a navigational path that best represents a missing relation.

In this paper, we define an algorithm called Path Index Algorithm (PIA) that from a semantic relation in DBpedia, finds the best navigational path that represents it in Wikipedia.

Paths are generalized by replacing properties of source and range articles with wildcards forming *path queries*. A path query is a general specification for many paths in Wikipedia. For example, starting from Berlin, the link *people_from_Berlin* is normalized to *people_from_[#from]*. The hypothesis is that the smallest path query in the index that maximally contains the semantic relation in DBpedia is the best expression of this semantic relation in Wikipedia. To validate this hypothesis, we experiments the proposed algorithm on DBpedia queries. The algorithm has demonstrated to be able to discover the navigational path that better fit the conventions of the Wikipedia community.

The paper is organised as follow. Section 2 motivates the work through an example. Section 3 presents the path indexing approach and the PIA algorithm. Section 4 exposes an evaluation of our proposal. Section 5 describes related work. Finally, conclusions and further work are detailed in Section 6.

2. MOTIVATING EXAMPLE

The semantic knowledge base of DBpedia [2] is extracted from Wikipedia infoboxes and wiki markups. For example,

[1] http://www.wikipedia.org

WWW 2012 Companion, April 16–20, 2012, Lyon, France.
ACM 978-1-4503-1230-1/12/04.

DBpedia extracts for each person in Wikipedia the infobox property *birthPlace* and creates an entry in DBpedia knowledge base which relates the person and the city with a semantic property called *birthPlace*. This allows performing semantic queries in DBpedia such as:

```
SELECT ?city , ?person WHERE{
?person a Person.
?city a City.
?person birthplace ?city}
```

Listing 1: DBpedia query (Q1): "is birth place of".

This query retrieves all people and their born city. In the other hand, analyzing in Wikipedia the 119097 pairs $(city, person)$ retrieved from DBpedia by the *is birthPlace of* query, we were surprised that in only 65200 pairs were possible to find in Wikipedia a navigational path from the city to personthe person. The result is even more surprising considering that the information used by DBpedia is the one extracted from Wikipedia.

In order to increase Wikipedia content with knowledge from DBpedia, we have first to learn how wikipedians express the relation *is_birthPlace_of* among cities and people in Wikipedia.

Articles in Wikipedia are connected by wikilinks and are grouped in categories. A wikilink connects an article with another and represents a one-to-one relation. In the other hand, *"The central goal of the category system is to provide links to all Wikipedia articles in a hierarchy of categories which readers can browse, knowing essential, defining characteristics of a topic, and quickly find sets of articles on topics that are defined by those characteristics"*[2]. Therefore, categories are used to define topics (named in singular) or set of elements (named in plural).

Consequently, a one-to-many relation among articles is expressed by grouping the range elements in a category i.e. France page is related to its canals by the category `Cat: Canals in France`[3]. This kind of relations are expressed using the category tree and naming conventions. In the example, the category `Cat:Canals in France` is a subcategory of the category `Cat:France`. The classification is reinforced by the word *in* in the category name to highlight that the articles belonging to the category *Canals* are also part of *France* according to Wikipedia naming conventions.

Although categories in Wikipedia are organized in a hierarchy called the category tree, according to Suchanek et al. [6], "Wikipedia category hierarchy is barely useful for ontological purposes. For example, Zidane is in the category `Football in France`, but Zidane is a football player and not a football“.

Consequently, it is not possible to apply a semantic approach to discover missing relations. We have to map the semantic relations of DBpedia to their syntactical representation in Wikipedia using navigational path. In Wikipedia, we can distinguish between: *One-to-one navigational path* and *one-to-many navigational path*. One-to-many navigational path denotes a navigation only through Wikilink. A one-to-many navigational path denotes a navigation through the category tree. There are established conventions to organize the category tree in Wikipedia. For example most of related pages to a city page in Wikipedia are organized under the category `Cat:CityName`. This can be observed for cities like Boston, New York, London, Paris, etc. People born in a city is denoted by the subcategory called `Cat: People_from_<CityName>`, for Boston page, the `Cat:People_from_Boston` is a subcategory of the category *Cat:Boston*.

This common practice could be generalized by a path like `Cat:<from> / Cat:People_ from_<from>` which represents a path to navigate (written with the symbol `/`) to a category and then continue to its subcategory `Cat: People_from`. This path represents, for example, the case of Boston `Cat: Boston /Cat:People_from_Boston` or the case of Paris `Cat: Paris/Cat:People_from_Paris`. However, there are many paths connecting Cities with People, for example `Cat:Capitals_in_Europe / Cat:Paris/Cat:People_from_Paris`. Having several alternatives, which one is the best representation for the relation *birthPlace*?

The main concern of this paper is to compute automatically the path in the category tree of Wikipedia that best represents a relation of DBpedia. This can be useful for Wikipedia bots to automatically maintain conventions or to assist Wikipedians in the edition of Wikipedia. Imagine a Wikipedian user accesses to *Robin Moore* article[4]. The *birth place* attribute in the infobox of this article is Boston. However, from Boston page, it is impossible to navigate back to Robin Moore article through wikilinks or categories. To increase the information of Boston article, the user could ask the PIA algorithm for retrieving what is the best way to include in the Boston article that Robin Moore was born there. PIA will answer according to Wikipedia conventions, the best way is by defining a path from Boston article to Boston category and then to People from Boston category. In the last category of the path, Robin Moore should belong to.

In next sections we detail the path indexing approach and algorithms to detect the best path representation of a semantic relation from DBpedia.

3. PATH INDEXING APPROACH

Paths were introduced in [4] and can be described, in terms of Wikipedia, as a sequence of pages which connects one article to another. Wikipedia can be considered as a directed graph formed by a set of pages (both articles and categories) which are connected by links. The set of pages that are crossed form a path. Formally, a path between two pages is defined below as:

DEFINITION 3.1 (PATH). *A path in Wikipedia is a sequence of pages* $[v_1, .., v_n]$ *, s.t.* $\forall i, j\ 1 \leqslant i < j \leqslant n$*,* $v_i \neq v_j$ *,* $\forall i, 1 \leqslant i \leqslant n-1$*, where* (v_i, v_{i+1}) *is a link between* v_i *and* v_{i+1}.

Example 1. The expression `Boston /Cat:Boston /Cat:People_from_Boston /Robin_Moore` is a path between the articles `Boston` and `Robin_Moore`. And also, `Turin/ Cat:Turin / People_from_Turin /Carla_Bruni` is a path from `Turin` and `Carla_Bruni`.

In this example, the general structure of both paths is coincident: they have a sequence of pages where some of the pages include in its name a reference to the origin page. The same occurs for most cities in Wikipedia. We introduce

[2] http://en.wikipedia.org/wiki/Wikipedia:Categorization

[3] We use the prefix `Cat:` to indicate a category

[4] http://en.wikipedia.org/wiki/Robin_Moore January 2012

Q(d,t)	Path	Path Query
(City1, Bob)	[City1/ Cat:City1/Bob]	[#from/ Cat:#from/#to]
	[City1/ Cat:City1/Cat:Actor_from_City1/ Bob]	[#from/ Cat:#from/Cat:Actor_from_ #from/ #to]
(City2, Alan)	[City2/ Cat:City2/Cat:Actor_from_City2/ Alan]	[#from/ Cat:#from/Cat:Actor_from_#from/ #to]

Table 1: Path Index Algorithm example

the *path query* concept in order to represent a set of similar paths. We define path query in terms of Wikipedia following the path query definition introduced by Abiteboul et al. [1].

DEFINITION 3.2 (PATH QUERY). *A path query is a regular expression to represent paths which uses a finite alphabet (like "*") and is posed relative to a domain page and a range page. The answer of a path query q evaluated on a domain page d is the set of all range pages reachable from d by some path that is compatible with the regular expression in Wikipedia category tree.*

Following the above example, the path query in Wikipedia category tree `PQ1(c,p)= c /Cat:c /Cat: People_from_c / p` is a valid path query for `Boston` to `Robin_Moore`; and `Turin` to `Carla_Bruni`.

The issue now is to discover the path query that best fit to a semantic query of DBpedia. In this work, we focus on queries similar to $Q1$ previously defined.

Let Q a DBpedia query, RS_Q is the result of evaluating Q. RS_Q is defined as a set of pairs *(domain, range)*. For example, for the query $Q1$ in Listing 1, $RS_{Q1} = \{(Paris, Bob), (Boston, Alan), (NewYork\ , John)\}$.

We suppose that the shortest path query that maximally contains the semantic relation expressed by the semantic query in DBpedia is the best expression of this semantic relation in Wikipedia. We formally define the shortest maximally contained path query $SMCPQ$ with length up to a constant $maxL$ for RS_{qw}, where RS_{qw} is the representation of RS_Q in Wikipedia, as:

DEFINITION 3.3. *(SMCPQ) Let a path query $Q' \subseteq RS_{qw}$, where $\forall p \in Q', length(p) \leqslant maxL$. Q' is the SMCPQ if and only if $\nexists$ a path query $Q'' \subseteq RS_{qw}$ such that $Q' \subset Q'' \subseteq RS_{qw}$.*

The PIA algorithm (Algorithm 1) computes SMCPQ for a given semantic query Q. It takes as input the RS_Q and $maxL$ value. The $buildIndex$ function performs a Deep First Search up to $maxL$ starting from a node in the *domain* of RS_Q. For each path reaching the *range*, it normalizes the path into a path query and builds the path index as a bipartite graph linking path queries to elements of RS_Q. The SMCPQ is the first ranked path query in the index.

Algorithm 1 PIA algorithm

```
Input: $RS_Q$: set of (domain,range), k for maxL: int
$index \leftarrow buildIndex(RS_Q, k)$
return $firstRanked(index)$
```

The path index as a bipartite graph is formalized as $index = (PQ, V, E)$ where PQ is a set of path queries, V is a set of pairs $(domain, range)$ and E is a set or pairs (PQ, V). The $pathQueryGen$ function (Algorithm 2) transforms a path into a path query. It replaces occurrences of domain and range by specific wildcards. The $pathQueryGen$ is in charge of the path normalization.

Algorithm 2 pathQueryGen

```
Input: path: [v_1,...,v_n], $domain, range$: WP article
Output: A path query
for all $v$ in path do
    StringReplace($v$, domain, range)
end for
return path
```

Applying $pathQueryGen$ on `path = [Paris /Cat:Paris, Cat:People_from_Paris/ Pierre Curie]` produces `[#from /Cat:#from/ Cat:People_from_#from/ #to]`.

To illustrate the algorithm, we calculate the SMCPQ for a hypothetical query Q in DBpedia using the Wikipedia category tree. Suppose that the RS_Q contains the two pairs enumerated in the first column of the table 1 ((City1, Bob), (City2, Alan)). We execute PIA with this RS_Q as input and calculate for each pair the path and its corresponding path query which are listed in the second and third column respectively.

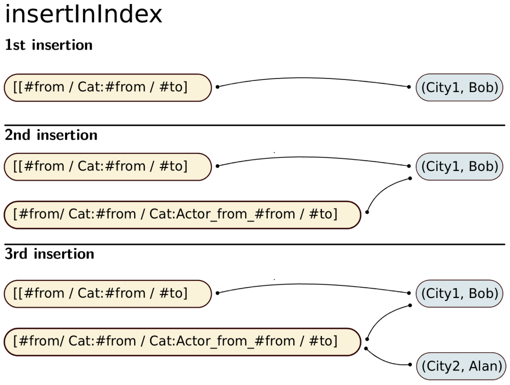
insertInIndex
1st insertion
[[#from / Cat:#from / #to]
(City1, Bob)
2nd insertion
[[#from / Cat:#from / #to]
(City1, Bob)
[#from/ Cat:#from / Cat:Actor_from_#from / #to]
3rd insertion
[[#from / Cat:#from / #to]
(City1, Bob)
[#from/ Cat:#from / Cat:Actor_from_#from / #to]
(City2, Alan)

Figure 1: Insertion example

For each path query, PIA updates the bipartite graph by adding the new path queries and the correct edge. In our example, we have to insert three path query, two for (City1, Bob) and one for (City2, Alan). Figure 1 describes the state of the index after each insertion. The first one inserts a node for the path query `[#from/ Cat:#from/ #to]` , a node for pair (City1,Bob)) and an edge between them. In the second insertion, the algorithm inserts only the path query `[#from/ Cat:#from/ Cat:Actor_from_#from -/#to]` and an edge between the path and the pair (City1,Bob). Finally, PIA inserts the path query corresponding to the pair (City2,Alan). As the corresponding path query already exists in the index, the algorithm inserts only the pair (City2,Alan) and the edge between the path query `[#from/ Cat:#from/Cat:Actor_from_#from/ #to]` and the new inserted pair.

In terms of the bipartite graph, the most representative path query is the one with the highest degree. In this example, the degree of `[#from/ Cat: #from/Cat:Actor_from_`

Domain	Range	Cases	Found	Not Found	Errors
City	Person	119097	65200	49899	3998
City	Philosopher	171	103	61	7
France	Philosopher	21	21	0	0

Table 2: Evaluation datasets for "is birthPlace of" using different Domain and Range

Pos	Path Query	#
1	[#from]/ [Cat:from]/ [Cat:People_from_#from/ [#to]	34008
2	[#from]/[#to]	16312
3	[#from]/[Cat:Wikipedia_semi-protected_pages]/ [Cat:Wikipedia semi-protected categories]/ [Cat:Living_people]/ [#to]	3188
4	[#from]/ [Cat:Articles_containing-_Japanese_language_text]/ [#to]	2422
5	[#from]/ [Cat:Cities_in_New_York/ [Cat:#from]/ [Cat:People_from_#from]/ [#to]	2104
6	[#from]/ [Cat:Capitals_in_Europe]/ [Cat:#from]/ [People_from_#from]/ [#to]	2028

Table 3: Index results for the Query1: is birthPlace of (City, People)

#from/ #to] is 2, because it has an edge to (City1, Bob) and other to (City2, Alan). The degree of the path query [#from/ Cat:#from/#to] is 1, because it has only one edge.

4. EXPERIMENTATION

We implemented and executed the PIA algorithm against the following three queries [5]:

```
PREFIX db:<http://dbpedia.org/resource/>
PREFIX db-owl:<http://dbpedia.org/ontology/>
PREFIX db-p:<http://dbpedia.org/property/>

#Query 1
SELECT ?city, ?person WHERE{
?person a db-owl:Person.
?city a db-owl:City.
?person db-p:birthplace ?city}

#Query 2
SELECT ?city, ?philosopher WHERE{
?philosopher a db-owl:Philosopher.
?city a db-owl:City.
?philosopher db-p:birthplace ?city}

#Query 3
SELECT db:France, ?philosopher WHERE{
?philosopher a db-owl:Philosopher.
?philosopher db-p:birthplace db:France}
```

The queries have been executed on a DBpedia snapshot produced in January 2012. The PIA algorithm has been executed on a copy of the English version of Wikipedia produced in October 2011[6].

Pos	Path Query	#
1	[#from] /[Cat:from]/[Cat:People_from_#from/ [#to]	60
2	[#from] /[Cat:Capitals_in_Europe]/ [Cat:#from]/[People_from_#from]/[#to]	15
3	[#from] /[#to]	14

Table 4: Index results for the Query 2: is birthPlace of (City, Philosopher)

Pos	Path Query	#
1	[#from]/ [Cat:#from]/[Cat:French people]/ [Cat:#French people by occupation]/ [French philosophers]/ [#to]	21
2	[#from]/ [Cat:#from]/ [Cat:French people]/ [Cat:#French people by occupation]/ [French sociologists]/ [#to]	3
3	[#from]/ [Cat:#from]/ [Cat:French people]/ [Cat:#French people by religion]/ [French atheists]/ [#to]	2

Table 5: Index results Query 3: is birthPlace of (France, Philosopher)

Table 2 shows the number of pairs produced by each query on DBpedia. For each pair *(domain, range)*, it evaluates the existence of paths between the domain and the range in Wikipedia: If it exists, then the pair is computed as a found path (Found column), if not it is computed as a non found path (Not Found column). The *Errors* column shows the number of pairs where elements of DBpedia are not present in Wikipedia dump. This comes mainly from desynchronization between Wikipedia and DBpedia.

Tables 3, 4 and 5 show the path index. The first column of each table points out the rank position, in the second column appears the path query and the last column indicates the number of pairs (domain, range) of the path query result set. The index is sorted and the first path query is the SMCPQ returned by PIA .

Table 3 shows the path index for the Query 1. The SMCPQ is the first path query in the table 3. It describes a path built by a category with the same name of the city and a subcategory called Cat: People_from_#from where #from is a wildcard which replace city name. Additionally, table 3 includes the administrative category *Wikipedia semi-protected categories* in the third path query. Administrative categories are not reachable by regular Wikipedia user and these cases must be pruned from the index.

Table 4 presents the path index for Query 2. The SMCPQ is the same as in the previous case. The main difference in this index is that we have pruned administrative categories from the index ranking.

Finally, table 5 presents the index path obtained for Query 3. We can notice that the SMCPQ covers all pairs in the dataset.

The SMCPQ were the same for the first and second queries, even having into account that the second case was more specific than the first one. This path queries show us that according to Wikipedia conventions, the most used path to express the relation birth place among cities and people is by

[5] All queries share the same set of prefix enumerated at the beginning of the listing.

[6] http://dumps.wikimedia.org/enwiki/20111007/

the query path `[#from]/ [Cat:from]/ [Cat:People_from_#from/ [#to]`.

In summary, now we are able to say that the best path to represent the relation between Boston and Robin_Moore is `Boston/ Cat:Boston/Cat:People_from_Boston / Robin_Moore` based on 34008 paths which represents the same relation in Wikipedia.

5. RELATED WORK

There are some existing approaches to increase the content of Wikipedia by using external resources. Wigipedia [3] allows regular Wikipedia's users to introduce semantic relations from DBpedia into Wikipedia. This is done by means of an interface which can be embedded in a Wikipedia page. Wigipedia generates from the current article a graph of relations with no more than 2 hops. By inspecting this graph, the user would be able to add eventual missing relations. Our approach is more general, it is not related to one article but it uses the result of a semantic query. In addition, Wigipedia does not take in consideration the Wikipedia community convention in a whole as our approach.

Hoffman et al. [5] developed human-machine collaborative system to complete infobox information by taking sentences from the Wikipage using Kylin [8]. Kylin is an information extraction system that extracts training data by analysing Wikipedia infoboxes and collects the best sentences in the article body that represents a field in the infobox. Finally, the system makes suggestions to the user for adding missing fields in the infobox. As in our approach, the machine effort is used to detect missing relations in Wikipedia. By contrast, we are able to fix relations among articles which does not have necessary sentences in their text body.

Madwiki [4] stores information in structured databases using slots pairing attribute/values. Views of these databases are generated as wikipages where users can add and manage information. When changes are generated, Madwiki takes the control and synchronize users information with the structured database fixing inconsistencies. Finally, Madwiki uses the idea of path view language to navigate from a node to another in the database representation. We were inspired from Madwiki navigational path to define the navigational path in Wikipedia. The main difference with our approach is that we use DBpedia.

Related with the algorithmic field there are several works in link and patterns predictions. For example, Thor et al. in [7] developed a link prediction framework for a controlled vocabulary ontology using tripartite and bipartite graph approach for biological datasets. In their work, they developed a prediction link framework for an annotated graph using topological shortest path. They used graph summarization to transform dense graph into a simpler one to analyse graphs. The idea of generates shortest path in a topological order in a graph is similar, but we focus on the Wikipedia graph.

6. CONCLUSIONS AND FURTHER WORK

In this article, we proposed a novel approach to increase Wikipedia content by using semantic data of DBpedia. We developed an algorithm, called PIA, that indexes path queries of Wikipedia based on results sets generated by queries on DBpedia. We also made a preliminary evaluation of our approach. During the evaluation, we have noticed that the path queries are not always coincident with Wikipedia categorization usage. In some cases the analysed DBpedia query does not have a direct categorization rule which involves it, for example *birthplace* between Cities and People instead of Countries and People with Occupation. A combination with the obtained path query and the confidence level bring us important information to determine the evolution in Wikipedia Community of that path query. Now, we are working on more experimentation.

7. ACKNOWLEDGEMENTS

This work is supported by the French National Research agency (ANR) through the KolFlow project (code: ANR-10-CONTINT-025), part of the CONTINT research program.

This work was also funded by: the PAE 37279-PICT 02203 which is sponsored by the ANPCyT, Argentina.

8. REFERENCES

[1] S. Abiteboul and V. Vianu. Regular path queries with constraints. In *Proceedings of the sixteenth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems*, PODS '97, pages 122–133, New York, NY, USA, 1997. ACM.

[2] 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 Semantics: Science, Services and Agents on the World Wide Web*, 7(3):154 – 165, 2009.

[3] S. Bostandjiev, J. O'Donovan, C. Hall, B. Gretarsson, and T. Höllerer. Wigipedia: A tool for improving structured data in wikipedia. In *ICSC*, pages 328–335. IEEE, 2011.

[4] P. DeRose, X. Chai, B. J. Gao, W. Shen, A. Doan, P. Bohannon, and X. Zhu. Building community wikipedias: A machine-human partnership approach. In G. Alonso, J. A. Blakeley, and A. L. P. Chen, editors, *ICDE*, pages 646–655. IEEE, 2008.

[5] R. Hoffmann, S. Amershi, K. Patel, F. Wu, J. Fogarty, and D. S. Weld. Amplifying community content creation with mixed initiative information extraction. In *Proceedings of the 27th international conference on Human factors in computing systems*, CHI '09, pages 1849–1858, New York, NY, USA, 2009. ACM.

[6] F. M. Suchanek, G. Kasneci, and G. Weikum. Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*, WWW '07, pages 697–706, New York, NY, USA, 2007. ACM.

[7] A. Thor, P. Anderson, L. Raschid, S. Navlakha, B. Saha, S. Khuller, and X.-N. Zhang. Link prediction for annotation graphs using graph summarization. In L. Aroyo, C. Welty, H. Alani, J. Taylor, A. Bernstein, L. Kagal, N. F. Noy, and E. Blomqvist, editors, *International Semantic Web Conference (1)*, volume 7031 of *Lecture Notes in Computer Science*, pages 714–729. Springer, 2011.

[8] F. Wu and D. S. Weld. Autonomously semantifying wikipedia. In M. J. Silva, A. H. F. Laender, R. A. Baeza-Yates, D. L. McGuinness, B. Olstad, Ø. H. Olsen, and A. O. Falcão, editors, *CIKM*, pages 41–50. ACM, 2007.