Linked Movie Data Base

Oktie Hassanzadeh
University of Toronto
10 King's College Rd., Toronto
Ontario, M5S-3G4, Canada
oktie@cs.toronto.edu

Mariano Consens
University of Toronto
10 King's College Rd., Toronto
Ontario, M5S-3G4, Canada
consens@cs.toronto.edu

ABSTRACT

The Linked Movie Database (LinkedMDB) project provides a demonstration of the first open linked dataset connecting several major existing (and highly popular) movie web resources. The database exposed by LinkedMDB contains millions of RDF triples with hundreds of thousands of RDF links to existing web data sources that are part of the growing Linking Open Data cloud, as well as to popular movie-related web pages such as IMDb. LinkedMDB uses a novel way of creating and maintaining large quantities of high quality links by employing state-of-the-art approximate join techniques for finding links, and providing additional RDF metadata about the quality of the links and the techniques used for deriving them.

Keywords

Linked Data, Semantic Web, Semantic Link Discovery, Record Linkage, Movie Database

1. INTRODUCTION

Movies are highly popular on the Web. There are several web resources dedicated to movies and many others containing movie-related information. Creating a single source of information about movies that contains information from existing open web data sources and links to other related data sources is a challenging task and the goal of the Linked Movie Data Base (LinkedMDB) project. LinkedMDB provides a high quality source of RDF data about movies (`http://linkedmdb.org`) that appeals to a wide audience, enabling further demonstrations of the linked data capabilities. Furthermore, LinkedMDB demonstrates the value of a novel class of tool to facilitate high volume and dense interlinking of RDF datasets.

Figure 1 shows an example of the entities and the interlinking in LinkedMDB. There are several challenges involved in identification of the entities in different data sources that should be interlinked. In some cases, the access to the data in target data source is limited. For example, only the title of the movies with their associated URLs can be obtained from the data source. In such cases, matching only the titles may not be sufficient due to different representations of the same title. Matching the movie title "The Shining" in LinkedMDB would miss the `owl:sameAs` link to the movie title "The_Shining_(film)" in DBpedia. Similarly, movie titles "A Thousand and One Nights" and "1001 Nights" would not match. Many non-English movie titles have different spellings in English, e.g., the titles "Adu Puli Attam" and "Sacco and Vanzetti" in LinkedMDB are written as "Aadu_Puli_Aattam" and "Sacco_e_Vanzetti" in DBpedia. This calls for *approximate (or fuzzy) string matching* for finding `owl:sameAs` links between the two sources.

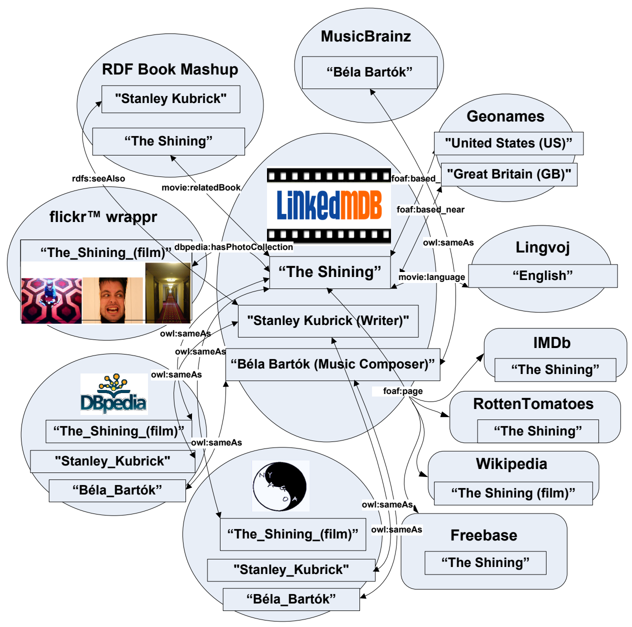


Figure 1: Sample LinkedMDB Entities

However, exact or approximate matching of movie titles could results in false matches. The movie "Chicago" (1927 movie) would link to the movie "Chicago" (2002 movie) using exact matching. By approximate matching, movie titles "Spiderman 1" and "Spiderman 2" have similar titles but are not the same. There is a similar case for the movies "Face to Face" and "Face to Fate", and some adult movies that have names very similar to popular Hollywood movies. Although using proper string similarity function and specific record matching techniques (e.g., using additional structural and co-occurrence information as in [3, 7]) could significantly reduce the amount of false matches, achieving 100% accuracy is not always possible. Also, higher accuracy may result in fewer correct links, as shown in the accuracy evaluation of Section 3 in this paper. Therefore, it is plausible for the publisher to include *metadata* about the links and how are they

Total number of triples	3,579,616
Number of interlinks to LOD cloud	162,199
Number of links to movie websites	271,671
Number of entities in LinkedMDB[1]	233,103

Table 1: Overall Statistics

are obtained. This approach has several advantages. The users will be able to determine the type of the links and level of accuracy depending on the application. Furthermore, this will facilitate the process of judging the quality of the links by the users and therefore allowing the users to only judge the quality of the links as opposed to *User Contributed Interlinking* [6].

In this paper, we present an overview of the movie data triplification effort showcased in LinkedMDB (Section 2). We then overview the interlinking of the data sources (Section 3), and provide a brief overview the approximate string matching techniques used for link discovery in relational data and present an evaluation of the performance of some of the techniques in LinkedMDB (Section 4). The need for linkage metadata and our approach for providing such data in LinkedMDB is discussed (Section 5). We conclude the paper by a brief discussion of a few future directions (Section 6).

2. TRIPLIFICATION OF MOVIE DATA

2.1 Data sources

Currently there are several sources of information on the web of documents:

- IMDb is the biggest database of movies on the Web that provides a huge variety of up-to-date information about movies. Although IMDb data is available for download and personal use, it is strongly protected by copyright laws. Although we did transform the IMDb data to RDF, we could not get permission for publishing it and therefore our implementation does not include any information from IMDb although we include external links to IMDb pages whenever possible..

- FreeBase is an open, shared database of the world's knowledge. The "film" category of freebase is one of the biggest and most complete domains in this database with more than 38,000 movies and thousands of other data items related to movies. Freebase has open data and has recently made its data available for download. Therefore, we use freebase as the nucleus of our database, although we do not limit our data source to the information available on freebase.

- OMDB is another open data source of movies. The dataset currently contains information about more than 9,000 movies, and its data is available for public use.

- DBpedia (Wikipedia) Movies: DBpedia contains a huge amount of information about more than 36,000 movies and thousands of related data items. We provide owl:sameAs links to DBpedia. Apart from extra information available in freebase such as movie characters and many other user-contributed data, we hope to serve additional information about movies and links to other data sources. This can be achieved due to the fact that unlike DBpedia, our database is specifically designed for movie data and also our information is not limited to that available on Wikipedia articles.

- RottenTomatoes.com is another movie website with information about movies. RottenTomatoes data is not available for download and public use, however, we include `foaf:page` links to RottenTomatoes website as well.

- Stanford Movie Database is a free database of movie information initially provided as a real test data for students. This database is relatively old, last updated in November 1999. Therefore it includes only a few data items that are not present in FreeBase. We however plan to extend our database with the additional information that can be obtained from this source.

Entity	Count
Film	38,064
Actor	29,361
Director	8,367
Writer	12,990
Producer	9,637
Music Contributor	3,995
Cinematographer	2,169
Interlink	162,199

Table 2: Sample Entities

2.2 Entities and Facts

Our database currently contains information about several entities including but not limited to movies, actors, movie characters, directors, producers, editors, writers, music composers and soundtracks, movie ratings and festivals. Table 2 shows the statistics for major entities in LinkedMDB. A list of all entities and facts in LinkedMDB will be made available in the extended version of this paper.

3. INTERLINKING DATA SOURCES

LinkedMDB provides links to several Linking Open Data (LOD) cloud datasets. Among these links are links to DBpedia, YAGO, flickr-wrapper, Geonames and lingvoj. Moreover, several data items are linked to external web pages such as pages on freebase, IMDb, OMDB, RottenTomotoes and Wikipedia.

LinkedMDB is connected to the following LOD data sources:

- DBpedia/YAGO: Apart from the movie titles, person names (such as actors, writers and composers) are linked the related resources in DBpedia and YAGO data sources with `owl:sameAs` links.

- Geonames: We interlink the countries of the movies to Geonames dataset by `foaf:based_near` type of links. This is done by matching name of the countries in the two datasets. These links could be extended by matching featured locations of the movies to Geonames items.

- FlickrWrapper: The moviess are linked to their photo collections using FlickrWrapper web service. These links are derived from the corresponding DBpedia URIs of the movies.

- RDF book mashup: Movies are linked to their related stories.

Target	Type	Count
DBpedia	`owl:sameAs`	30,354
YAGO	`owl:sameAs`	30,354
flickr wrappr	`DBpedia:hasPhotoCollection`	30,354
RDF Book Mashup (Books)	`movie:relatedBook`	700
RDF Book Mashup (Authors)	`rdfs:seeAlso`	12,990
MusicBrainz	`owl:sameAs`	2,207
GeoNames	foaf:based_near	27,272
GeoNames	`owl:sameAs`	272
lingvoj	`movie:language`	28,253
IMDb, RottenTomatoes Freebase.com	`foaf:page`	271,671

Table 3: Interlinking Statistics

- Musicbrainz: Music composers and soundtracks are linked to muzicbrainz.
- Revyu.com: Movie reviews on Revyu can be linked to movies (and vice versa).

Apart from links to LOD datasets, we also have setup `foaf:page` links to external webpages:

- Freebase.com pages.
- IMDb.com movies and actor profiles.
- RottenTomatoes.com movie information and reviews.

Other potential links include links to external webpages from OMDB, boxoffice and movie show-times website and also homepages of the movies.

4. APPROXIMATE STRING MATCHING FOR LINK DISCOVERY

As mentioned earlier, link discovery often requires approximate matching of strings. In LinkedMDB, several links to other data sources are found using string matching. In this Section, we first briefly overview a set of string similarity functions and state-of-the art approximate string join techniques that are used (or can be used) in link discovery in LinkedMDB and similar link discovery settings. We then present the results of the evaluation of the quality of the links found using different similarity functions.

4.1 String Similarity Measures

There exists a wide variety of similarity functions for comparing similarity of the strings. The similarity measures we discuss here share one or both of the following properties:

- High scalability: There are various techniques proposed in the literature as described in Section 4.2 for enhancing the performance of the similarity join operation using q-grams along with these measures.
- High accuracy: Previous work has proved that in most scenarios these measures perform better or equally well in terms of accuracy comparing with other string similarity measures. Specifically, these measures have shown good accuracy in name-matching tasks [4] or in approximate selection [5].

Let r be the set of q-grams (i.e., sequences of length q of consecutive characters of a string) in string record r. For example, for $r = \text{`}dblab\text{'}$, $r = \{\text{`}d\text{'}, \text{`}db\text{'}, \text{`}b\text{'}, \text{`}l\text{'}, \text{`}la\text{'}, \text{`}ab\text{'}, \text{`}b\text{'}\}$ for tokenization using 2-grams . In certain cases, a weight may be associated with each token.

4.1.1 Edit Similarity

Edit distance between two string records r_1 and r_2 is defined as the transformation cost of r_1 to r_2, $tc(r_1, r_2)$, which is equal to the minimum cost of edit operations applied to r_1 to transform it to r_2. Edit operations include character *copy*, *insert*, *delete* and *substitute*. The edit similarity is defined as:

$$sim_{edit}(r_1, r_2) = 1 - \frac{tc(r_1, r_2)}{\max\{|r_1|, |r_2|\}} \qquad (1)$$

There is a cost associated with each edit operation. There are several cost models proposed for edit operations for this measure. In the most commonly used measure called Levenshtein edit distance, which we will refer to as edit distance in this paper, uses unit cost for all operations except copy which has cost zero.

4.1.2 Jaccard and WeightedJaccard

Jaccard similarity is the fraction of tokens in r_1 and r_2 that are present in both. Weighted Jaccard similarity is the weighted version of Jaccard similarity, i.e.,

$$sim_{WJaccard}(r_1, r_2) = \frac{\sum_{t \in \mathbf{r_1} \cap \mathbf{r_2}} w_R(t)}{\sum_{t \in \mathbf{r_1} \cup \mathbf{r_2}} w_R(t)} \qquad (2)$$

where $w(t, R)$ is a weight function that reflects the commonality of the token t in the relation R. We choose RSJ (Robertson-Sparck Jones) weight for the tokens which was shown to be more effective than the commonly-used Inverse Document Frequency (IDF) weights [5]:

$$w_R(t) \quad = \quad log\left(\frac{N - n_t + 0.5}{n_t + 0.5}\right) \qquad (3)$$

where N is the number of tuples in the base relation R and n_t is the number of tuples in R containing the token t.

4.1.3 Measures from IR

A well-studied problem in information retrieval is that given a query and a collection of documents, return the most *relevant* documents to the query. In the measures in this part, records are treated as documents and q-grams are seen as words (tokens) of the documents. Therefore same techniques for finding relevant documents to a query can be used to return *similar* records to a query string. In the rest of this section, we present three measures that previous work has shown their higher performance for approximate selection problem [5].

Cosine w/tf-idf The *tf-idf cosine* similarity is a well established measure in the IR community which leverages the vector space model. This measure determines the closeness of the input strings r_1 and r_2 by first transforming the strings into unit vectors and then measuring the angle between their corresponding vectors. The *cosine* similarity with tf-idf weights is given by:

$$sim_{Cosine}(r_1, r_2) = \sum_{t \in \mathbf{r_1} \cap \mathbf{r_2}} w_{r_1}(t) \cdot w_{r_2}(t) \qquad (4)$$

where $w_{r_1}(t)$ and $w_{r_2}(t)$ are the normalized *tf-idf* weights for each common token in r_1 and r_2 respectively. The normalized *tf-idf* weight of token t in a given string record r is defined as follows:

$$w_r(t) = \frac{w'_r(t)}{\sqrt{\sum_{t' \in \mathbf{r}} w'_r(t')^2}} \quad , \quad w'_r(t) = tf_r(t) \cdot idf(t)$$

where $tf_r(t)$ is the term frequency of token t within string r and $idf(t)$ is the inverse document frequency with respect to the entire relation R.

4.1.4 BM25

The BM25 similarity score for a query r_1 and a string record r_2 is defined as follows:

$$sim_{BM25}(r_1, r_2) = \sum_{t \in \mathbf{r_1} \cap \mathbf{r_2}} \hat{w}_{r_1}(t) \cdot w_{r_2}(t) \tag{5}$$

where

$$\hat{w}_{r_1}(t) = \frac{(k_3+1) \cdot tf_{r_1}(t)}{k_3 + tf_{r_1}(t)}$$
$$w_{r_2}(t) = w_R^{(1)}(t) \frac{(k_1+1) \cdot tf_{r_2}(t)}{K(r_2) + tf_{r_2}(t)}$$

and $w_R^{(1)}$ is the RSJ weight:

$$w_R^{(1)}(t) = \log\left(\frac{N - n_t + 0.5}{n_t + 0.5}\right)$$
$$K(r) = k_1\left((1-b) + b\frac{|r|}{avg_{rl}}\right)$$

where $tf_r(t)$ is the frequency of the token t in string record r, $|r|$ is the number of tokens in r, avg_{rl} is the average number of tokens per record, N is the number of records in the relation R, n_t is the number of record containing the token t and k_1, k_3 and b are set of independent parameters. We set these parameters as described in [5] where $k \in [1, 2]$, $k_3 = 8$ and $b \in [0.6, 0.75]$.

4.1.5 Hidden Markov Model

The approximate string matching could be modeled by a discrete Hidden Markov process which has shown better performance than Cosine w/tf-idf in the IR literature, and high accuracy and running time for approximate selection [5]. The HMM similarity function accepts two string records r_1 and r_2 and returns the probability of generating r_1 given r_2 is a similar record:

$$sim_{HMM}(r_1, r_2) = \prod_{t \in \mathbf{r_1}} (a_0 P(t|GE) + a_1 P(t|r_2)) \tag{6}$$

where a_0 and $a_1 = 1 - a_0$ are the transition states probabilities of the Markov model and $P(t|GE)$ and $P(t|r_2)$ is given by:

$$P(t|r_2) = \frac{\text{number of times } t \text{ appears in } r_2}{|\mathbf{r}_2|}$$
$$P(t|GE) = \frac{\sum_{r \in R} \text{number of times } t \text{ appears in } r}{\sum_{r \in R} |\mathbf{r}|}$$

4.2 Approximate String Join Techniques

An advantage of the similarity predicates described above is that they can be implemented declaratively using standard SQL queries over any relational DBMS. This is in particular useful considering the fact that many existing linked data sources are published using linked data publication tools that operate over relational data sources, such as D2R server, Triplify or OpenLink Vituoso. Some of the similarity predicates can be made scalable to huge web data sources using some of the specialized, high performance, approximate join algorithms. Specifically, Enumeration (Enum) and Weighted Enumeration (WtEnum) signature generation algorithm can be used to significantly improve the running time of the join with Jaccard and weighted Jaccard predicates [1]. In addition, novel indexing and optimization techniques can be utilized to make the join even faster [2].

4.3 Evaluation

In this Section, we provide a summary of the evaluation of the accuracy of the linkage in only one of the linkage scenarios. More detailed comparison of the techniques in other scenarios (including other type of links such as `rdfs:seeAlso` links) will be made available in the extended version of this paper.

4.3.1 Matching Algorithm for `owl:sameAs` Links

In order to match movie titles between two sources to discover `owl:sameAs` links, we perform the following steps:

- We choose one source as the base source and the other as query source. We pre-process the base source to calculate weights.
- For each string in the query source, we find all those strings in the base source which have similarity score above a threshold θ by performing an approximate selection.
- If there is only one string matched, then we output the query and base strings as certain matches. If there is more than one string or no string with similarity score above θ with the query string, then we do not output a match for the query string.

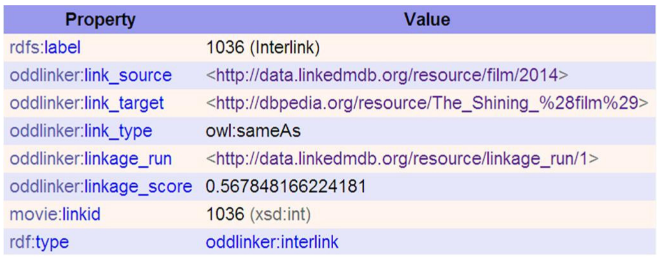
4.3.2 Accuracy Results

In our experiments we used $q = 2$ for generating q-grams as it showed better performance comparing with other values of q. Here, we present brief accuracy results for matching movie titles from DBpedia to movie titles in our database. We matched 38,064 movie titles in our database with 25,424 movie titles from DBpedia using the similarity predicates described above. We need to inspect different thresholds to see find the optimal threshold. Table 4 shows the number of matches obtained with different values of threshold as well as the accuracy obtained. Note that accuracy reported is the precision of the links, i.e., percentage of the output links that are correct. The recall is hard to find since the correct number of matches is not known. However, the number of links returned reflects the value of recall. The ground truth is obtained by manually finding all the *rules* for matching in this scenario. For example, all underscores are replaced with whitespaces, and the substring "(film)" is removed from the DBpedia movie titles. These rules themselves are discovered by running the similarity join and manually inspecting thousands of the links returned.

The results in Table 4 show that the weighted Jaccard similarity outperforms other predicates in this scenario in terms of the number of correct links found. Based on these results we chose threshold $\theta = 0.7$ with weighted Jaccard similarity for the existing links in our database.

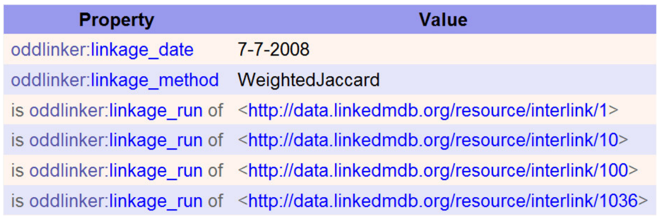
Measure	Threshold	#Total	#Wrong	Accuracy	Measure	Threshold	#Total	#Wrong	Accuracy
Weighted Jaccard	0.6	12,756	693	94.57%	Cosine w/tf-idf	0.6	24,549	7,189	70.72%
	0.65	9,823	350	96.44%		0.65	21,068	4,546	78.42%
	0.7	7,130	169	97.63%		0.7	17,623	2,541	85.58%
	0.75	4,874	86	98.24%		0.75	14,082	1,243	91.17%
	0.8	3,018	36	98.81%		0.8	10,671	571	94.65%
	0.85	1,913	15	99.22%		0.85	7,169	197	97.25%
	0.9	1,505	6	99.60%		0.9	3,886	61	98.43%
Jaccard	0.6	10,476	785	92.51%	BM25	0.6	7,889	1,258	84.05%
	0.65	7,798	353	95.47%		0.65	5,565	695	87.51%
	0.7	5,545	189	96.59%		0.7	3,760	407	89.18%
	0.75	3,909	104	97.34%		0.75	2,485	254	89.78%
	0.8	2,117	43	97.97%		0.8	1,658	160	90.35%
	0.85	1,531	25	98.37%		0.85	1,092	98	91.03%
	0.9	1,432	7	99.51%		0.9	715	67	90.63%
Edit Similarity	0.6	16,137	3,260	79.80%	HMM	0.6	3,737	374	89.99%
	0.65	12,550	1,219	90.29%		0.65	2,396	226	90.57%
	0.7	9,423	519	94.49%		0.7	1,534	154	89.96%
	0.75	5,848	227	96.12%		0.75	992	92	90.73%
	0.8	2,719	93	96.58%		0.8	646	61	90.56%
	0.85	1,043	7	99.33%		0.85	447	45	89.93%
	0.9	334	4	98.80%		0.9	306	35	88.56%

Table 4: The number and the accuracy of the owl:sameAs links between LinkedMDB and DBpedia/YAGO



Property	Value
rdfs:label	1036 (Interlink)
oddlinker:link_source	<http://data.linkedmdb.org/resource/film/2014>
oddlinker:link_target	<http://dbpedia.org/resource/The_Shining_%28film%29>
oddlinker:link_type	owl:sameAs
oddlinker:linkage_run	<http://data.linkedmdb.org/resource/linkage_run/1>
oddlinker:linkage_score	0.567848166224181
movie:linkid	1036 (xsd:int)
rdf:type	oddlinker:interlink

Figure 2: Sample `interlink` Entity



Property	Value
oddlinker:linkage_date	7-7-2008
oddlinker:linkage_method	WeightedJaccard
is oddlinker:linkage_run of	<http://data.linkedmdb.org/resource/interlink/1>
is oddlinker:linkage_run of	<http://data.linkedmdb.org/resource/interlink/10>
is oddlinker:linkage_run of	<http://data.linkedmdb.org/resource/interlink/100>
is oddlinker:linkage_run of	<http://data.linkedmdb.org/resource/interlink/1036>

Figure 3: Sample `linakge_run` Entity

5. LINKAGE METADATA

As shown in the accuracy evaluation in previous Section, although using proper string matching techniques and string similarity function could significantly reduce the amount of false matches, achieving 100% accuracy is not always possible or may result in fewer correct links. Therefore, it is plausible for the publisher to include *metadata* about the links and how are they are obtained. In LinkedMDB, we provide to entities, namely `interlink` and `linkage_run` for this purpose. Figures 2 and 3 show examples of these entities. This approach has several exciting advantages. The users will be able to determine the type of the links and level of accuracy depending on the application. Furthermore, this will facilitate the process of judging the quality of the links by the users and therefore allowing the users to provide feedback on the quality of the links.

6. CONCLUSION

LinkedMDB provides a high quality source of RDF data about movies that appeals to a wide audience, enabling further demonstrations of the linked data capabilities. Furthermore, LinkedMDB demonstrates the value of a novel way of link discovery and publishing linkage metadata to facilitate high volume and dense interlinking of RDF datasets. We plan to extend LinkedMDB in several aspects. Our plan is to provide an easy-to-use interface to allow the users to provide feedback on the quality of the links. In this way, users will only need to report the quality of the links as opposed to manually providing the links, as proposed in *User Contributed Interlinking* framework of [6]. Apart from extending the number of external links, we plan to provide internal links (of type `rdfs:seeAlso` or a similar type) between related entities, such as movies with similar titles. Such links can be found using similar approximate matching techniques, and will further facilitate automatic mining of the data sources.

7. REFERENCES

[1] A. Arasu, V. Ganti, and R. Kaushik. Efficient exact set-similarity joins. In *VLDB '06 - Proceedings of the 32nd international conference on Very large data bases*, pages 918–929, 2006.

[2] R. J. Bayardo, Y. Ma, and R. Srikant. Scaling up all pairs similarity search. In *WWW'07 - Proceedings of the 16th International World Wide Web Conference*, pages 131–140, 2007.

[3] I. Bhattacharya and L. Getoor. Collective entity resolution in relational data. *IEEE Data Eng. Bull*, 29(2):4–12, 2006.

[4] W. W. Cohen, P. Ravikumar, and S. E. Fienberg. A comparison of string distance metrics for name-matching tasks. In *IIWeb'03*, pages 73–78, 2003.

[5] O. Hassanzadeh. Benchmarking declarative approximate selection predicates. Master's thesis, University of Toronto, Feb 2007.

[6] M. Hausenblas and W. Halb. Interlinking of resources with semantics. In *ESWC'08(Posters)*.

[7] Y. Raimond, C. Sutton, and M. Sandler. Automatic interlinking of music datasets on the semantic web. In *LDOW'08*.