Fuzzy Matching of Web Queries to Structured Data

Tao Cheng *, Hady W. Lauw #, Stelios Paparizos #

*University of Illinois, 201 N Goodwin Ave, Urbana, IL, 61801, USA (work done while at Microsoft)
tcheng3@cs.uiuc.edu

#Microsoft Research, 1065 La Avenida, Mountain View, CA, 94043, USA
{hadylauw, steliosp}@microsoft.com

***Abstract*— Recognizing the alternative ways people use to reference an entity, is important for many Web applications that query structured data. In such applications, there is often a mismatch between how content creators describe entities and how different users try to retrieve them. In this paper, we consider the problem of determining whether a candidate query approximately matches with an entity. We propose an off-line, data-driven, bottom-up approach that mines query logs for instances where Web content creators and Web users apply a variety of strings to refer to the same Web pages. This way, given a set of strings that reference entities, we generate an expanded set of equivalent strings for each entity. The proposed method is verified with experiments on real-life data sets showing that we can dramatically increase the queries that can be matched.**

I. Introduction

Recently, Web search has evolved into an advanced answering mechanism returning relevant facts or content, instead of just links of web pages. For example, a query such as '*Indy 4 near San Fran*', when posed on a major search engine like Bing, produces results for showtimes for the movie '*Indiana Jones and the Kingdom of the Crystal Skull*' near the city of 'San Francisco'. Using only free text to answer such queries can be problematic. On the other hand, structured data sources (*e.g.*, movie databases) often contain appropriate information for this purpose.

Effective usage of such structured data sources requires a fast and accurate match between the various query parts and the underlying structured data. However, there is often a gap between what end users type and how content creators describe the actual data values of the underlying structured entities. Content creators tend to use high-quality and formal descriptions of entities, whereas end users prefer a short, popular, and informal 'synonymous' representation. For example, a movie database lists the full title of '*Indiana Jones and the Kingdom of the Crystal Skull*', whereas Web users may type '*Indy 4*'. This phenomenon exists across virtually all domains. Apple's '*Mac OS X*' is also known as '*Leopard*'. The digital camera '*Canon EOS 350D*' also goes by the name of '*Digital Rebel XT*'.

Existing approaches are not always successful in automatically finding such synonymous strings. Dictionary based approaches, such as Thesaurus or WordNet [1], are insufficient when looking at the semantic alterations necessary for movies, products or company names. Substring matching based approaches work well for some cases ('*Madagascar 2*' from '*Madagascar: Escape 2 Africa*'), fall short in others ('*Escape Africa*' would also be considered incorrectly for '*Madagascar: Escape 2 Africa*') and are hopeless for the rest ('*Canon EOS 350D*' with '*Digital Rebel XT*'). Manual effort based approaches, such as Wikipedia redirect or disambiguation pages, can be of high quality, but are rather limited to only very popular entries, as we will show in Section IV.

The same gap between users and content creators exists in typical Web search. There it is alleviated by the efforts of some content creators, who resort to including known alternative forms within the content of a Web page so as to facilitate a textual match by a search engine. End users resort to trying different queries until they find some Web page that satisfies them. Due to the scale of the Web, there is enough Web page content produced that when considered in unison can handle most of the different query variations.

In this paper, we propose a fully automated solution that can enrich structured data with synonymous or alternative strings. To achieve this goal we leverage the collective wisdom generated by Web page content creators and end users towards closing the above-mentioned gap. At a high level, we use a multi-step data driven approach that relies on query and click logs to capture the Web wisdom. We first retrieve relevant Web page urls that would be good surrogates or representatives of the entity. We then identify the union of all queries that have accessed at least one of these urls by following the edges of a url-query click graph. We qualify which queries are more likely to be true synonyms by inspecting click patterns and click volume on a large subset of such urls.

We formulate the synonym finding problem in Section II, and present our approach in Section III. In Section IV, we perform a comprehensive experimental study to validate the proposed method on large-scale real-life data sets.

II. Problem Definition

In this section, we will first give our formal definitions of synonym, hypernym, and hyponym, before defining the synonym finding problem.

A. Synonym, Hypernym, and Hyponym

Let $\mathcal{E}$ be the set of entities over which the synonyms are to be defined. *An entity is an object with distinct and separate existence from other objects of the same type (those having similar attributes).* For example, "Indiana Jones and the Kingdom of the Crystal Skull" is an entity of type *Movie*.

Let $\mathcal{S}$ be the universal set of strings, where each string is a sequence of one or more words. We assume that there exists

an oracle function $\mathcal{F}(s, \mathcal{E}) \rightarrow E$, which is an ideal mapping from any string $s \in \mathcal{S}$ that users may think of in order to refer to the very subset of entities $E \subseteq \mathcal{E}$.

We now put forward our definitions of synonym, hypernym, and hyponym in the context of entities of a specific domain.

Definition 1 (Synonym): A string $s_1 \in \mathcal{S}$ is a **synonym** of another string $s_2 \in \mathcal{S}$ over the set of entities $\mathcal{E}$ if and only if $\mathcal{F}(s_1, \mathcal{E}) = \mathcal{F}(s_2, \mathcal{E})$. For example, t_1 = "Indiana Jones IV" is a synonym of t_2 = "Indiana Jones 4" since they both cover the same set of entities in the movie domain.

Definition 2 (Hypernym): A string $s_1 \in \mathcal{S}$ is a **hypernym** of another string $s_2 \in \mathcal{S}$ over the set of entities $\mathcal{E}$ if and only if $\mathcal{F}(s_1, \mathcal{E}) \supset \mathcal{F}(s_2, \mathcal{E})$. For example, t_3 = "Indiana Jones series" is a hypernym of t_1, since t_1 only maps to a subset of the entities covered by t_3.

Definition 3 (Hyponym): A string $s_1 \in \mathcal{S}$ is a **hyponym** of another string $s_2 \in \mathcal{S}$ over the set of entities $\mathcal{E}$ if and only if $\mathcal{F}(s_1, \mathcal{E}) \subset \mathcal{F}(s_2, \mathcal{E})$. For example, t_1 is a hyponym of t_3.

B. Synonym Finding Problem

Synonym Finding Problem. Formally, our formulation of the synonym finding problem is as follows. As input, we are given a set of entities $\mathcal{E}$ (*e.g.*, movies) and a set of homogeneous strings (*e.g.*, movie names) $U \subseteq \mathcal{S}$. As output, we would like to produce for each string $u \in U$, its set of synonyms $V_u = \{v \in \mathcal{S} \mid \mathcal{F}(u, \mathcal{E}) = \mathcal{F}(v, \mathcal{E})\}$.

In the problem formulation above, we do not assume that $\mathcal{F}$ is a given. This is because, while we assume that such an oracle function exists (if only abstractly), we do not claim it is obtainable in practice. True $\mathcal{F}$ exists only in the collective minds of all users. Hence, the equality $\mathcal{F}(u, \mathcal{E}) = \mathcal{F}(v, \mathcal{E})$ that underlies Definition 1 cannot be determined exactly.

To resolve this, we propose to relax Definition 1, and instead approximate the equality $\mathcal{F}(u, \mathcal{E}) = \mathcal{F}(v, \mathcal{E})$ using real-life data. We identify the following real-life Web based data sets as especially relevant for this approach:

Search Data A consists of a set of tuples, where each tuple $a = \langle q, p, r \rangle$ denotes the relevance score r of a Web page url p for the search query $q \in \mathcal{S}$. For simplicity, in this paper, we assume r is the relevance rank of p, with rank 1 being the most relevant. A captures the "relevance" relationship between a query string and a Web page as determined by a search engine.

Click Data L is a set of tuples, where each tuple $l = \langle q, p, n \rangle$ denotes the number of times $n \in N^+$ that users click on p after issuing query $q \in \mathcal{S}$ on a search engine. L captures the "relevance" relationship between a query string and a Web page as determined by search engine users.

How these data sets may be used to find synonyms can be summarized as follows. Let $\mathcal{P}$ be the union of all Web pages, and $\mathcal{Q}$ be the union of all query strings, in A and L. Since both A and L represent some form of relationship between query strings and Web pages, we can learn from A and L respectively, two functions $\mathcal{G}_A(q, \mathcal{P}) \rightarrow P$ and $\mathcal{G}_L(q, \mathcal{P}) \rightarrow P$, which map a query string $q \in \mathcal{Q}$ to the subset of relevant Web pages $P \subseteq \mathcal{P}$. Assuming that for any entity $e \in \mathcal{E}$, there always exist Web pages that are appropriate and representative surrogates of e (which is a reasonable assumption given the scope of the Web), we consider it probable that two query strings q_1 and q_2 are synonyms if $\mathcal{G}_A(q_1, \mathcal{P}) \approx \mathcal{G}_L(q_2, \mathcal{P})$.

Definition 4 (Web Synonym): A string $s_1 \in \mathcal{S}$ is a **Web synonym** of another string $s_2 \in \mathcal{S}$ over the set of Web pages $\mathcal{P}$ (keeping in mind the actual reference set of entities $\mathcal{E}$) if $\mathcal{G}_A(s_1, \mathcal{P}) \approx \mathcal{G}_L(s_2, \mathcal{P})$.

Web Synonym Finding Problem. In this paper, our specific problem formulation is as follows. As input, we are given a set of homogeneous strings U; the data sets A and L; and the reference set of entities $\mathcal{E}$. As output, we would like to produce for each string $u \in U$, its set of Web synonyms $W_u = \{w \in \mathcal{S} \mid \mathcal{G}_A(u, \mathcal{P}) \approx \mathcal{G}_L(w, \mathcal{P})\}$.

III. A BOTTOM-UP SOLUTION

To solve the Web synonym finding problem, we propose a two-phase solution, consisting of candidate generation and candidate selection.

A. Candidate Generation

To quickly zoom into synonym candidates for a given u, we propose to generate candidate in two steps. First, we seek the Web pages that are good representation for entities referenced by u. We term these Web pages *surrogates* of u. Second, we find out how users refer to these surrogates.

Finding Surrogates. The Web has inarguably become the largest open platform for serving various kinds of data. It is almost certain that entities we have in our entity set $\mathcal{E}$ would have some representation on the Web. This representation (or surrogates) come mostly in the form of Web pages. For a particular digital camera (*e.g.*, Canon EOS 350D), its surrogates may include a page in the manufacturer's site listing its specifications, an eBay page selling it, a Wikipedia page describing it, a page on a review site critiquing it, *etc.*.

Moreover, data appears in various forms on the Web. For instance, a seller on eBay may explicitly list some of the alternative ways to access the data to help increase the chances of her item being retrieved, *e.g.*, "Digital REBEL XT" and "350D". Data, once appearing on the Web, gets enriched in various ways, which enables alternative paths for people to access the same information.

We use the *Search Data* A to find Web page surrogates for a given u. A is derived by issuing each $u \in U$ as a query to the Bing Search API and keeping the top-k results. Based on A, we can define the mapping function $\mathcal{G}_A(u, \mathcal{P})$ between u to the set of top-k pages, as Eq 1 shows.

$$\mathcal{G}_A(u, \mathcal{P}) = \{a.p \mid a \in A,\ a.q = u \wedge a.r \leq k\} \qquad (1)$$

Definition 5 (Surrogate): A Web page $p \in \mathcal{P}$ is a **surrogate** for u if $p \in \mathcal{G}_A(u, \mathcal{P})$.

It may also be possible to use *Click Data* in place of *Search Data*, whereby a Web page is a surrogate if it has attracted many clicks when the entity's data value is used as a query. However, clicks are not always available for this purpose, as the entities' data values usually come in the canonical form (*e.g.*, the full title name of a movie), and therefore may not be used as queries by people.

Referencing Surrogates. Having identified u's surrogates, we next find out how users access those surrogates. Remember that these surrogates are Web pages available for access by the general public. Again, search engine is the primary channel people use for accessing information on the Web. We can therefore regard the queries issued to get to these surrogate pages as the various ways users refer to the entities represented by these pages. Consequently such queries are good synonym candidates for u.

Click data L offers us the mapping from candidates to surrogates. Based on L, we can define the mapping function $\mathcal{G}_L(w', \mathcal{P})$ between a potential synonym candidate w' to the set of clicked pages, as shown in Eq 2.

$$\mathcal{G}_L(w', \mathcal{P}) = \{l.p \mid l \in L,\ l.q = w' \ \wedge\ l.n \geq 1\} \quad (2)$$

Definition 6 (Web Synonym Candidate): A string w' is a **synonym candidate** for u if and only if $\mathcal{G}_A(u, \mathcal{P}) \ \cap \ \mathcal{G}_L(w', \mathcal{P}) \neq \emptyset$.

Based on Definition 6, we regard w' as a Web synonym candidate for u if at least one surrogate of u has been clicked when w' is issued as a query. Therefore, the candidate set for u is $W'_u = \{w' \mid \mathcal{G}_A(u, \mathcal{P}) \ \cap \ \mathcal{G}_L(w', \mathcal{P}) \neq \emptyset\}$.

B. Candidate Selection

To estimate the likelihood that a candidate w' is a Web synonym of the input value u, we identify two important measures that can be captured from search data A and click data L. The two measures respectively capture the *strength* and *exclusiveness* of the relationship between a candidate w' and the input value u.

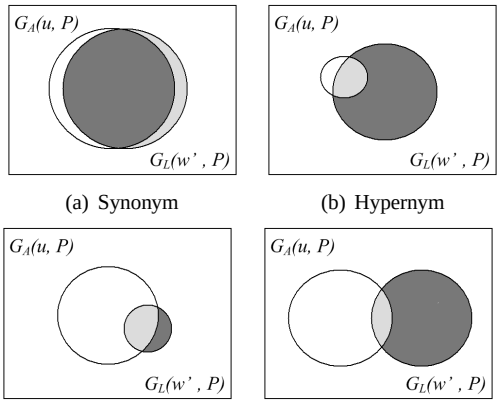
Intersecting Page Count (IPC). Here, we seek to measure the strength of relatedness between an input value u and a candidate w'. In the candidate generation phase, we look for u's candidates by looking at queries (w') for which the following holds: $\mathcal{G}_A(u, \mathcal{P}) \ \cap \ \mathcal{G}_L(w', \mathcal{P}) \ \neq \ \emptyset$. In Eq 3, we derive the $IPC(w', u)$ as the size of this intersection. Intuitively the higher the IPC, the larger the size of the intersection is, the more common pages have been referred to using u and w', and therefore the more likely u and w' would be related to one another.

$$IPC(w', u) = |\mathcal{G}_L(w', \mathcal{P}) \cap \mathcal{G}_A(u, \mathcal{P})| \quad (3)$$

Intersecting Click Ratio (ICR). Another indicator for the strong relationship between w' and u is if a majority of the clicks resulting from w' as a query land on u's surrogate pages more often than on non-surrogate pages. The click ratio measure $ICR(w', u)$ is determined as shown in Eq 4. The higher is $ICR(w', u)$, the more exclusive is the relationship between w' and u, and the more likely w' would be a Web synonym of u.

$$ICR(w', u) = \frac{\sum_{l \in L,\ l.p \in \mathcal{G}_L(w', \mathcal{P}) \cap \mathcal{G}_A(u, \mathcal{P})} l.n}{\sum_{l \in L,\ l.p \in \mathcal{G}_L(w', \mathcal{P})} l.n} \quad (4)$$

We use a Venn diagram illustration in Figure 1 to describe how the above two measures work in selecting the best Web synonyms. Consider the example where the input value u is the movie title "Indiana Jones and Kingdom of the Crystal Skull". Figure 1(a) illustrates the case where a candidate w'



(a) Synonym (b) Hypernym

(c) Hyponym (d) Not Equivalent

Fig. 1. Venn Diagram Illustration

(*e.g.*, "Indiana Jones 4") is a likely Web synonym of u. The sets denote the Web pages that are retrieved by u ($\mathcal{G}_A(u, \mathcal{P})$) and are clicked on for query w' ($\mathcal{G}_L(w', \mathcal{P})$) respectively. In this case, the size of the intersection of the two sets is large, indicating a high IPC value. For the set $\mathcal{G}_L(w', \mathcal{P})$, the darkly shaded (resp. lightly shaded) area indicates the subset of pages getting the most clicks (resp. fewer clicks). In this case, most of the clicks fall within the intersection, as opposed to outside of the intersection, indicating a high ICR value. Thus, w' is likely a Web synonym of u.

Both IPC and ICR also help to weed out candidates that are related, but not synonyms. Figure 1(b) illustrates the case of a hypernym (*e.g.*, "Indiana Jones"). Since a hypernym considers a broader concept, it may be used to refer to many more pages (*e.g.*, concerning other Indiana Jones movies), and consequently most of the clicks fall outside of the intersection (low ICR). A hyponym concerns a narrower concept, where there might be more specific pages about the concept outside of the intersection that receive the most clicks (Figure 1(c)). Finally, a candidate such as "Harrison Ford" is only related, with low IPC and ICR (Figure 1(d)).

We produce the final Web synonym by applying threshold values β and γ on IPC and ICR respectively.

IV. Experiments

Our data sets are: D1) the titles of the top 100 movies of 2008 Box office and D2) a collection of 882 canonical camera names crawled from MSN Shopping [2]. All experiments were done on a single windows 2003 server workstation with 8GB RAM and 2TB disk space. We used query and click logs from Bing Search (July to November 2008).

A. Parameter Sensitivity

In this section we evaluate the effect of **Intersecting Page Count (IPC)** and **Intersecting Click Ratio (ICR)** thresholds on *Precision*, *Weighted Precision* and *Coverage Increase*:

Precision	# of true synonyms over all synonyms generated
Weighted Precision	Weighted by synonym frequency in query log
Coverage Increase	Percentage increase in coverage of queries

We use a precision/recall style figure to show the precision and coverage increase at different thresholds, with x axis for coverage increase, and y axis for precision.

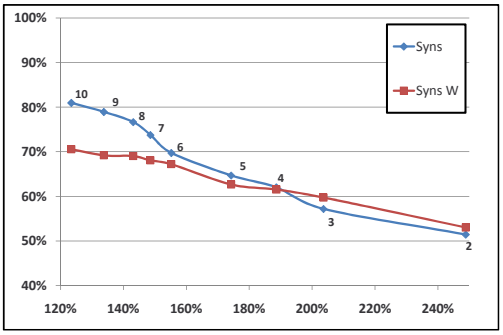


Fig. 2. IPC Precision and Coverage Increase

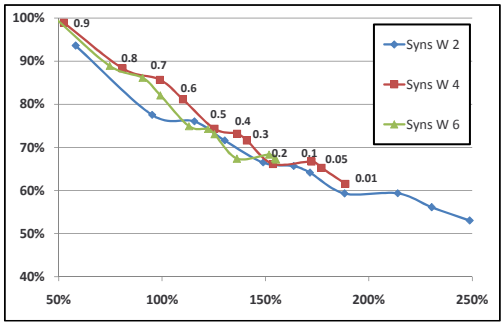


Fig. 3. ICR Precision and Coverage Increase for IPC 2,4,6.

Result for (D1) movies dataset is shown in Figure 2, where IPC threshold β decreases from left to right on the curves from 10 to 2. We see the higher the IPC, the higher the synonym (Syns) precision is. This effect is a bit weaker in weighted precision. Coverage increase reduces as we increase IPC. Yet, even at high IPC value 10, coverage increase is at 120%, more than doubling the original coverage.

To test the combination of IPC and ICR, we used the threshold values of $\beta \in \{2, 4, 6\}$ for IPC, with ICR γ decreasing from left (0.9) to right (0.01) on the curves, as shown in Figure 3 on (D1) dataset. To simplify the figures, we focus on the weighted precision. We see when we increase the ICR parameter, the synonym precision goes up (Syns W 2,4,6). This figure also suggests interesting IPC values around 4, and ICR values (0.1, 0.3, 0.4, 0.7) acting as local maxima with good balance between precision and coverage increase.

B. Other approaches to synonyms

To measure our solution *Us* (thresholds IPC 4, ICR 0.1) against other approaches, we looked at Wikipedia and random walk on the click graph to produce synonyms. We focus on *Hit Ratio* and *Expansion Ratio*:

Hit Ratio	Percentage of entries producing at least 1 synonym
Expansion Ratio	Sum of synonyms and orig entries over orig entries

Wikipedia. We use redirection and disambiguation pages in Wikipedia for producing synonyms (*e.g.*, the entry for 'LOTR' redirects to 'Lord of The Rings'). As shown in Table I, Wikipedia performs poorly for less popular entries (*e.g.*, cameras). Our approach consistently creates more synonyms (expansion) and for more entries (hit) for both datasets.

	Orig	Hits	Ratio	Synonyms	Expansion
Movies Us	100	99	99%	437	537%
Movies Wiki	100	96	96%	270	370%
Movies Walk(0.8)	100	100	100%	229	329%
Cameras Us	882	767	87%	4286	586%
Cameras Wiki	882	101	11.5%	576	165%
Cameras Walk(0.8)	882	479	54%	697	179%

TABLE I
HITS AND EXPANSION

Random Walk on a Click Graph. We used the random walk solution in [3] to evaluate the potential of generating synonyms with default parameters. We see in Table I that the random walk has low hit ratio on cameras, since the random walk operates completely on the click graph. So if a query has not been asked then no synonym will be produced.

V. RELATED WORK

Our work is related to reference reconciliation or record matching techniques ([4], [5]), which try to resolve different references or records in a dataset to the same real world entity. There are a few differences from our work. First, they normally assume the "references" are given, while we have to generate the candidate "references" ourselves. Second, such approaches usually rely on multiple attributes to be present to produce high quality results(*e.g.*, name, age, gender for person record). Yet, Web queries normally lack multi-attribute semantic context.

There are previous works to measure similarity between queries [6] by using Web data, for various purposes such as document ranking [7], semantic relation discovery [8], keyword generation for advertisement [3] and query suggestion ([9], [10]). These similarity based approaches do not work well for our problem for several reasons. First, they may discover many pairs of related queries that are not synonyms (*e.g.*, "Windows Vista" and "PC"). Second, the input for which we seek to derive synonyms are generally well-formed strings as full movie titles or digital camera names, which real users seldom use and may not appear frequently as queries.

VI. CONCLUSION

In this paper, we address the problem of discovering entity synonyms for fuzzy matching of Web queries to structural data, by mining query and click logs to generate synonym candidates and select true synonyms from candidates. Experiments validate the effectiveness of our proposed approach.

REFERENCES

[1] G. A. Miller, "Wordnet: a lexical database for english," *Commun. ACM*, vol. 38, no. 11, pp. 39–41, 1995.
[2] Microsoft, "MSN Shopping XML data access API." http://shopping.msn.com/xml/ v1/getresults.aspx?text=digital+camera.
[3] A. Fuxman, P. Tsaparas, K. Achan, and R. Agrawal, "Using the wisdom of the crowds for keyword generation," in *WWW*, 2008, pp. 61–70.
[4] X. Dong, A. Halevy, and J. Madhavan, "Reference reconciliation in complex information spaces," in *SIGMOD*, 2005.
[5] O. Benjelloun, H. Garcia-Molina, D. Menestrina, Q. Su, S. E. Whang, and J. Widom, "Swoosh: a generic approach to entity resolution," *The VLDB Journal*, vol. 18, 2009.
[6] J.-R. Wen, J.-Y. Nie, and H.-J. Zhang, "Clustering user queries of a search engine," in *WWW*, 2001, pp. 162–168.
[7] N. Craswell and M. Szummer, "Random walks on the click graph," in *SIGIR*, 2007, pp. 239–246.
[8] R. Baeza-Yates and A. Tiberi, "Extracting semantic relations from query logs," in *SIGKDD*, 2007.
[9] R. Jones, B. Rey, O. Madani, and W. Greiner, "Generating query substitutions," in *WWW*, 2006, pp. 387–396.
[10] I. Antonellis, H. Garcia-Molina, and C. Chang, "Simrank++: Query rewriting through link analysis of the click graph," in *VLDB*, 2008.