

Efficient Concept Clustering for Ontology Learning using an Event Life Cycle on the Web

Sangsoo Sung
Google Inc.
1600 Amphitheater Parkway
Mt. View, CA 94043, USA
sangsoos@google.com

Seokkyung Chung
Yahoo! Inc.
2821 Mission College Blvd
Santa Clara, CA 95054, USA
schung@yahoo-inc.com

Dennis McLeod
Dept. of Computer Science
Univ. of Southern California
Los Angeles, CA 90089, USA
mcleod@usc.edu

ABSTRACT

Ontology learning integrates many complementary techniques, including machine learning, natural language processing, and data mining. Specifically, clustering techniques facilitate the building of interrelationships between terms by exploiting similarities of concepts. With the rapid growth of the Web, online information has become one of the major information sources. The ontology learning process where traditional clustering algorithms are involved tends to be slow and computationally expensive when the dataset is as large as the Web. To address this problem, we present an efficient concept clustering technique for ontology learning that reduces the number of required pairwise term similarity computations without a loss of quality. Our approach is to identify relevant terms using a computationally inexpensive similarity metric based on an event life cycle in online news articles. Then, we perform more sophisticated similarity computations. Hence, we can build clusters with high precision/recall and high speed. Without a loss of clustering quality, our framework reduces the number of required computations from $O(N^2)$ to $(N + L^2)$ ($L \ll N$) where N is the number of candidate concepts. Our experimental results show that clustering based on our similarity framework can construct concept clusters 1541.07% faster than clustering with all term pair similarity computations.

Categories and Subject Descriptors

H.2.8 [Database Management]: Database Applications—*Data Mining*; I.2.6 [Computing Methodologies]: Artificial Intelligence—*Learning*

Keywords

concept clustering, ontology learning, similarity measures

1. INTRODUCTION

Ontologies, collections of concepts and their interrelationships, play a key role in knowledge management by pro-

viding solutions to multiple problems resulting from computers' inability to understand natural language. Numerous hand-crafted, lexical-semantic databases, such as WordNet [6], have shown that ontologies can facilitate machine understanding of text and the automatic processing of documents. However, this manual approach to extracting semantic meanings cannot scale with the growth of the Web. Therefore, ontology learning, the process of mining taxonomic relations from information sources, has become a significant subfield of ontology engineering.

This research focuses on the efficiency of identifying relevant concept candidates for ontology learning. Typically, identifying semantic structure, where clustering methods have been very involved, is computationally expensive for a large number of data sets since it requires a significant number of computations to measure a similarity among each pair of candidates [4]. To address the computational bottleneck, the reduction of the number of required similarity computations has emerged as a problem awaiting a solution in ontology learning from a large collection of documents.

To address the computational issue, we present a novel similarity computation approach based on the analysis of an event life cycle on the Web. By using an inexpensive similarity metric, the proposed approach identifies rough concept clusters that have high potential to be clustered together, thus avoiding unnecessary similarity computations among concepts that are far from each other. Based on the obtained rough concept clusters, we perform expensive similarity computations to refine the cluster structure. We hypothesize that if an event life cycle based candidate selection in the first phase results in a small number of similarity computations, then it would improve the clustering speed without a loss of quality.

Our approach is based on the analysis of event life cycle on the Web. Certain events generate postings to the Web, such as news articles. The volume of postings typically starts small, grows, and then gradually disappears as time passes [3]. Hence, a new relationship between two concepts can be created when a certain event has happened. Figure 1 depicts an example of term frequency transition over time. The results show that the frequencies of both terms sharply surged up and then dropped around the same time frame (July 16, 2007). Observations of other terms show the same phenomenon. One of the reasons this observation is important is that we can identify the relevant concept candidates if we cluster the terms whose frequencies have a rapid fluctuation at a similar point of time. In other words, it is unnecessary to compute a similarity between the

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

SAC'08 March 16-20, 2008, Fortaleza, Ceará, Brazil

Copyright 2008 ACM 978-1-59593-753-7/08/0003 ...\$5.00.

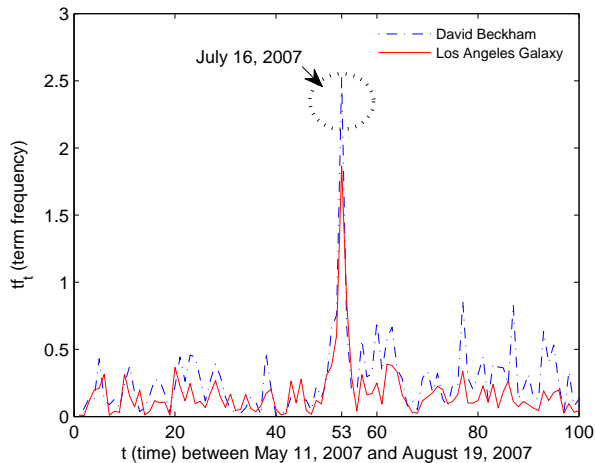


Figure 1: Term frequencies have been collected from online news articles for 100 days between May 11, 2007 and August 19, 2007 for both “David Beckham”, who is a famous football (soccer) player, and the “Los Angeles Galaxy”, which he joined in July 2007 and had his first official soccer practice session on July 16, 2007

terms before the event happens because the similarity between them probably would be low prior to the event. Our approach can be coupled with any type of clustering algorithms and can be utilized for making algorithms scalable with respect to the millions of documents.

Once concept clusters are identified, we utilize the obtained clusters to enrich existing ontologies. As many thousands of articles are published daily on the Web, neologisms or concepts appear as time passes. Thus, it is essential to maintain ontology to reflect up-to-date knowledge. To address this problem, we present how to utilize refined clusters to enrich existing ontologies.

The remainder of this paper is organized as follows. Section 2 presents the ontology learning problems that we consider in this paper. Section 3 discusses our similarity framework based on event life cycle, and Section 4 describes how to enrich ontologies using term similarity. Section 5 presents our experimental results, and Section 6 reviews existing solutions. Finally, Section 7 summarizes this paper, and discusses future work.

2. ONTOLOGY LEARNING PROBLEM

Ontologies provide an explicit model for structuring concepts, together with their interrelationships. Most existing ontologies, such as WordNet [6], were manually created and are being maintained by ontology engineers. Instead of employing manual construction, many recent studies have investigated ontology learning that automatically or semi-automatically extracts information from texts and builds a structured organization using extracted information. Inherently, it integrates multiple complementary techniques, such as machine learning, natural language processing, and data mining. Particularly, clustering techniques facilitate building interrelationships between terms by exploiting similarities of concepts to propose a hierarchy of concept categories.

With the rapid growth of the Web in the past decade, on-line information has become one of the major information sources. Traditional clustering algorithms have a scalability limitation in that similarity computation needs to be performed on all term pairs. Thereby, it would be inefficient when we have numerous terms which are extracted from the overwhelming number of documents on the Web. To address this problem, the following two sections present a method for ontology learning that reduces the number of required computations without sacrificing the precision/recall.

3. ROUGH CLUSTER IDENTIFICATION

We now illustrate the key ideas of our approach that one can greatly reduce the number of similarity computations required for ontology learning. We first identify rough cluster structure from documents using inexpensive similarity metric, and then only measure the similarity among pairs of concepts in rough clusters to identify their internal semantic structure. Hence, our first step in ontology learning from Web documents is to develop a solid solution for rough cluster identification.

Most research in concept extraction has incorporated both natural language processing and information retrieval techniques for term indexing. These methods include parsing HTML, tokenizing, stemming words¹, eliminating stopwords² and detecting phrases. A set of terms (phrases) is defined as $X = \{x_i : 1 \leq i \leq N\}$ where x denotes a concept candidate. A term is a concept candidate and can consist of multiple words.

Given concept candidates extracted by these methods, we consider an event life cycle on the Web to cluster relevant concept candidates. Key terms of certain events on the Web, like online news articles become popular and then diminish as time passes. In other words, the term frequency of those key terms greatly increases and decreases as an event appears, grows, and disappears. A term which is highly relevant to the key term of the events also appears more frequently in the documents and peaks during the same time frame.

We are interested in the term frequency valued time series denoted by $tf_t(x), t = 0, 1, 2, \dots, T$, where x is a term and t is a time variable. A function $count(x)$ computes the number of x 's occurrences in a document d_i . To account for different length of documents, we normalize the term frequency ($freq$) of term x in a document d_i as follows:

$$freq_{d_i}(x) = \frac{count(x)}{l_i} \quad (1)$$

where l_i is the length of d_i .

Let D_t be a set of documents that are published in time slot t . The time slot t that we use is a day. We formulate $tf_t(x)$ as follows:

$$tf_t(x) = \sum_{d_i \in D_t} freq_{d_i}(x) \quad (2)$$

We now present a solid algorithm that captures the important fluctuations in the tf_t over time. The algorithm is referred to as the Gallistel change point finding algorithm [2].

¹We combine the Porter stemmer with the lexical database [5] since this combination deals with irregular plural/tense [1].

²We employ the stopword list used in the Smart project [8].

We would like to find a change point where a certain attribute of the distribution from which tf_t are drawn has the important fluctuations. We divide this algorithm into the following steps:

We first define a function that associates with each point in tf_t as follows:

$$f(t_i) = \frac{tf_{t_i}(x) - tf_{t_0}(x)}{t_i - t_0} \text{ where } i > 1 \quad (3)$$

We then identify a putative change point (pcp) by finding the earliest point t that deviates maximally from $f(t)$.

$$pcp_t = \max_{j=0}^i (|f(t_j) - tf_{t_j}(x)|) \text{ where } 0 \leq i \leq T \quad (4)$$

To verify that pcp_t satisfies a decision criterion, we divide the set of tf_t up to that point into two subsets, the subset of S_1 for the tf_t values before pcp_t , and the subset of S_2 for the tf_t values after it. We then perform an *unequal variance t-test* [7]. The t statistic (z) to test whether the means are different can be calculated as follows:

$$z = \frac{\bar{S}_1 - \bar{S}_2}{\sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}} \quad (5)$$

where \bar{S}_1 and \bar{S}_2 are the sample means of the tf_t values (g) in S_1 and S_2 , and s_1^2 and s_2^2 are the standard errors as follows:

$$s_1^2 = \frac{\sum_i^{n_1} (g_i - \bar{g}_1)^2}{n_1 - 1} \text{ and } s_2^2 = \frac{\sum_i^{n_2} (g_i - \bar{g}_2)^2}{n_2 - 1}$$

We can identify a change point (cp) when t is significantly far from zero (null), rejecting the *null hypothesis* [7] that the tf_t values in S_1 and S_2 have been generated based on the same mean. The user can specify the threshold for rejecting the null hypothesis. Thereby, a change point can be expressed as:

$$cp_k = t \text{ of } pcp_t \quad (6)$$

We recursively execute Eq. (3) through Eq. (6) by starting immediately after the most recently identified change cp_k . Given the above change point finding algorithm, we define a set of terms (ω_t) whose elements have the same cp_k as follows:

$$\omega_t = \{x_i \in X : cp(x_i) = t, 0 \leq i \leq N\} \quad (7)$$

where $t = cp$ and $1 \leq i \leq N$. We also define a set of ω_t as follows:

$$\Omega = \{\omega_t : 0 \leq t \leq T\} \quad (8)$$

To reduce the number of expensive similarity computations in the next phrase, we cluster Ω into overlapping subsets like *canopies* [4] using a computationally inexpensive distance metric. Let α and β ($\alpha > \beta$) be distance thresholds (difference of days), which are specified by the user. We start with time t of ω_t with two distance thresholds. We then assign all the terms ω_t at t that are within the distance threshold $t - 0 \leq \alpha$ to a subset Ω_p . Next, we remove from the list all points that are within the distance threshold β in Ω . A collection of Ω_p can be obtain by repeating these clustering steps until Ω becomes empty. Consequently, each term would be assigned to smaller groups in which similarity measures would be performed in the next phase.

4. CLUSTER REFINEMENT USING AN EXPENSIVE SIMILARITY METRIC

This section presents how we refine rough cluster structure using an expensive similarity metric. We present a robust similarity model to identify the closeness of the candidate terms in Ω_p . Since concepts in ontologies commonly are expressed in few words, it is difficult to quantify the similarity of candidate terms. Our approach to this problem is to use documents in order to capture a richer semantic context of candidate terms instead of simply quantifying their string-wise similarity. Specifically, we are only interested in the documents which are posted on the Web during the time frame of Ω_p . Let cp_i and cp_j be the time range of Ω_p . Λ_p denotes the set of documents which are published between cp_i and cp_j .

We now provide precise definitions of the similarity measures that we use. The algorithm is based on *tf-idf* vector weighting scheme [8]. We incorporate Λ_p to generate a *tf-idf* vector for each candidate term $x \in \Omega_p$. The vector includes terms which co-occurred in the documents with x . Let v_i be a term vector for each document $d_i \in \Lambda_p$ where the weight w_{x,d_i} associated with term x in document d is defined to be:

$$w_{x,d_i} = freq_{d_i}(x) \times \log\left(\frac{|\Lambda_p|}{|\{d_i : x \in \Lambda_p\}|}\right) \quad (9)$$

where $freq_{d_i}(x)$ is Eq. (1), $|\Lambda_p|$ is the total number of documents in Λ_p , and $|\{d_i : x \in \Lambda_p\}|$ is the number of documents where the term x appears.

We eliminate all the elements in v_i except m highest w_{x,d_i} terms using the optimal trade-off curve between sophistication and efficiency. We then employ a cosine metric [8], which has been widely used in a vector space model. It measures the similarity of two vectors according to the angle between them. Let $\gamma(x)$ be the centroid vector of all v_i 's. The next step is to measure the relatedness between them. Given two terms x_i and x_j , the similarity between x_i and x_j is defined as the inner product as follows:

$$Sim(x_i, x_j) = \text{sigmoid}\left(\frac{\gamma(x_i)}{\|\gamma(x_i)\|_2} \cdot \frac{\gamma(x_j)}{\|\gamma(x_j)\|_2}\right) \quad (10)$$

where $\text{sigmoid}(x)$ is defined as follows:

$$\text{sigmoid}(x) = \frac{1}{1 + e^{-x}}$$

To obtain a desirable similarity prediction, we use the sigmoid transfer function because it can divide the whole input space smoothly into a few regions. Using Eq. (10), we generate a vector R_i for each term $x_i \in \Omega_p$ by quantifying the closeness of candidate terms. R_i contains $x_j \in \Omega_p$ ($i \neq j$). We then truncate each vector R_i to include its δ highest related terms based on $Sim(x_i, x_j)$.

5. EVALUATION

The main purpose of this paper is to reduce the number of necessary similarity computations for concept clustering. That is, we want to produce the same quality of clusters by using a significantly small amount of information, instead of using all pairwise term similarity. Thus, assuming that clusters resulting from using all possible pairwise similarity computation is ground truth data (O_b), we want to measure how many clusters produced by our approach (O_a) are close

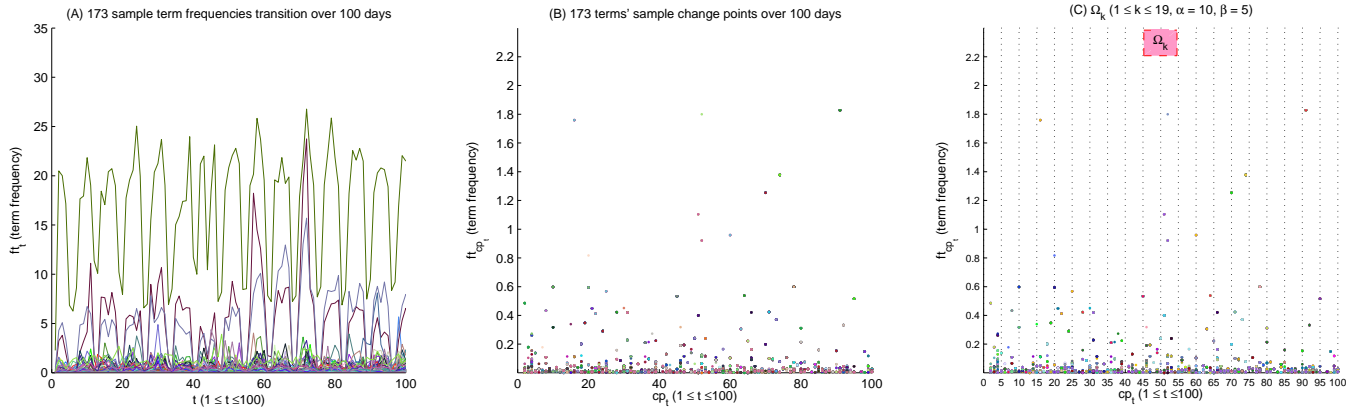


Figure 2: We quantified ft_t for each term by executing Eq. (1) and (2) as shown in (A). (B) illustrates the change points of ft_t for each term that we found by running Eq. (3) through Eq. (6). As depicted in (C), the terms were assigned to overlapping smaller groups by creating Ω_k with two tunable tight (α) and loose (β) thresholds for an inexpensive distance measure.

to O_b . Towards this end, we use standard precision/recall metrics as follows:

$$\text{precision} = \frac{1}{K} \sum_{i=1}^K \frac{|O_{a_i} \cap O_{b_j}|}{|O_{a_i}|} \quad (11)$$

$$\text{recall} = \frac{1}{K} \sum_{i=1}^K \frac{|O_{a_i} \cap O_{b_j}|}{|O_{b_j}|} \quad (12)$$

$$F = \frac{2 \cdot (\text{precision} \cdot \text{recall})}{(\text{precision} + \text{recall})} \quad (13)$$

where K is the number of clusters in O_a , and a cluster O_{a_i} is referred to as a concept O_{b_j} cluster if and only if the majority of concepts for O_{a_i} belong to O_{b_j} .

Experiment: We conducted our experiment using a web search engine³ to crawl online US news articles. We extracted 173 sample terms from 1,269,940 online news articles for 100 days between May 3, 2007 and August 11, 2007 by the methods that we described in Section 3. We produced two types of output. On the one hand, we constructed O_a by computing similarities that are described in Eq. (9) through Eq. (10) after identifying all terms in X using inexpensive similarity metric. As shown in Figure 2, average 34 terms were evenly assigned to each Ω_k ($1 \leq k \leq 19$) when we set α to 10 days and β to 5 days. On the other hand, we performed the complete similarity computations of 173 terms for structuring O_b by skipping the creation of Ω_k . All methods were implemented in C++ and Matlab, and experiments were performed on a Pentium4 CoreDuo 3.40GHz with 3GB memory, enough such that there was no paging activity.

Complexity Analysis: We determined the complexity of the methods to build both O_a and O_b as a function of the number of extracted terms (N). Since the different computational phases contributes to complexity, time complexity (C) of O_a and O_b can be expressed as:

$$C(O_a) = O(\text{Eq.1}) + O(\text{Eq.}\{2 : 6\}) + O(\text{Eq.10}) \quad (14)$$

$$C(O_b) = O(\text{Eq.1}) + O(\text{Eq.10}) \quad (15)$$

The complexity $C(O_a)$ is different from $C(O_b)$ in that $C(O_a)$ has rough cluster identification step (Eq. (2) through Eq. (6)),

³<http://code.google.com/apis/>

which runs in $O(N)$. Thereby, $C(O_a)$ involves Eq. (10) computation time, which is $O(L^2)$ ($L \ll N$) since T is assigned to Ω_k , while $C(O_b)$ corresponds to $O(N^2)$ where L is the number of candidate terms after rough cluster identification. Therefore, structuring O_a is more efficient than constructing O_b because $O(N + L^2) \ll O(N^2)$.

Experimental Results: Figure 3 describes the distributions of the similarity for identifying related terms in different term-sets: Ω_k for structuring O_a and all terms in X for building O_b . The results clearly show that the former Figure 3(A) has the log-normal distribution while the latter Figure 3(B) has a distribution with long-tailed behavior. Hence, almost 99% of similarity computation in Figure 3(B) was wasteful. Table 1 presents a summary of the experimental results that computational time increases since more pairwise similarity computations are needed, as we increase α and decrease β . Construction of O_a was 1541.07% faster than O_b where the distance thresholds α and β were respectively 10 and 5. Therefore, it demonstrates that our framework remarkably reduces the number of required computations for clustering without loss of the precision/recall.

6. RELATED WORK

An extensive amount of literature has addressed ontology learning in the context of natural language processing and data clustering [1]. Most research efforts have profited from clustering methods where terms are associated by specified similarity measures. However, most methods use a single similarity metric which are usually slow/accurate or fast/less accurate.

In recent data mining literature, McCallum *et al.* [4] used two different similarity measures: using quick, cruder metrics, they generated smaller clusters, and then used expensive metrics to refine the smaller clusters. These inspiring works provided us with a concrete foundation to further enhance the contemporary ontology learning framework. We develop their foundation in the following ways.

Our research presented in this paper focused on improving the efficiency of concept clustering process, which is essential in ontology learning. Our approach is to build a rough cluster structure by using a computationally inexpensive similarity metric [4] based on an event life cycle [3] in online

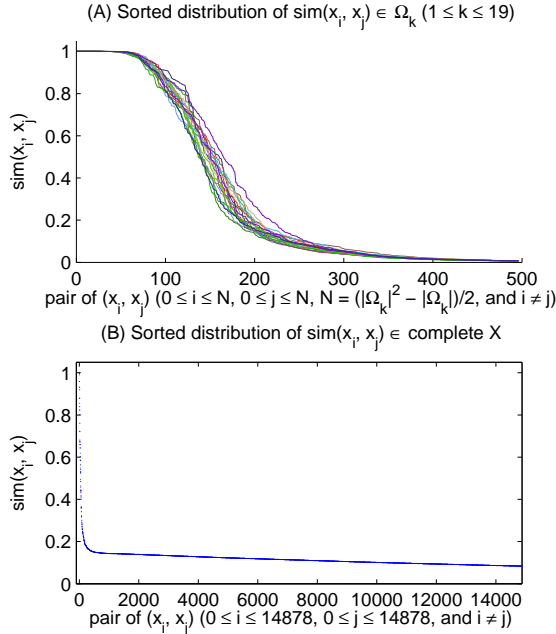


Figure 3: We compared distributions of similarities for structuring both O_a and O_b using Eq. (10). (A) shows the sorted distribution of pairwise similarities of terms in overlapping smaller groups. (B) presents the sorted distribution of complete pairwise similarities of 173 terms.

news articles before performing more sophisticated similarity computations. Hence, our approach is distinguished from the above studies in that the number of pairwise similarity computations is drastically reduced based on formal and practical analysis of event life cycle of news articles. Therefore, we can obtain the concept clusters with high precision/recall and high speed.

7. CONCLUSION

Given large quantities of online information with many billions of terms, quantifying all pairwise similarities of terms is very computationally expensive. Hence, the development of efficient techniques for clustering is crucial. In this paper, we have focused on efficiency in concept clustering. Our approach is to produce a rough cluster structure by reducing the number of required computations based on an event life cycle on the Web and then to refine the cluster structure. We also presented how refined clusters can be used to enrich existing ontology.

The proposed method can be utilized for a broad range of ontology-driven applications that require up-to-date ontologies. The generated ontologies continuously maintain up-to-date interrelationships among concepts by detecting an event life cycle on the Web. Exploiting a manually developed ontology with a controlled vocabulary is helpful in diverse applications such as query expansion. However, although ontology-authoring tools have been developed in the past decades, manually constructing ontologies whenever new domains are encountered is a time-consuming process. Moreover, the constructed ontology should evolve over time

Table 1: The precision, recall, F-measure, and time costs of different distance thresholds α and β for structuring O_a . Note that the time cost of structuring O_b was 394.67 mins. Based on our observations, news articles have a near-weekly cycle so that the best performing α and β were 10 and 5 respectively, which are indicated in bold.

α	β	Precision	Recall	F-Measure	Minutes
3	1	0.761	0.387	0.513	15.59
5	1	0.709	0.497	0.584	17.47
5	3	0.899	0.572	0.699	23.18
7	1	0.880	0.775	0.824	19.42
7	3	0.864	0.809	0.836	21.18
7	5	0.869	0.838	0.853	23.15
10	1	0.841	0.838	0.840	22.01
10	3	0.956	0.919	0.937	24.52
10	5	1.00	0.965	0.982	25.61
15	1	0.920	0.867	0.892	37.26
15	3	0.962	0.908	0.934	27.05
15	5	0.962	0.908	0.934	30.17
15	10	1.00	0.965	0.982	32.05

given that neologisms or concepts appear as time passes. Thus, we envision these ontologies will be an important resource for query refinement in search engines and ontology-driven matching solutions [9].

8. ACKNOWLEDGMENTS

This research was conducted when the first and second authors were at the University of Southern California. This work was funded in part by a grant from the Department of Homeland Security, ONR Grant number N00014-07-1-0149.

9. REFERENCES

- [1] S. Chung, J. Jun, and D. McLeod. A web-based novel term similarity framework for ontology learning. In *OTM Conferences (1)*, pages 1092–1109, 2006.
- [2] C. R. Gallistel, T. A. Mark, A. P. King, and P. E. Latham. The rat approximates an ideal detector of changes in rates of reward: Implications for the law of effect. *Journal of Experimental Psychology: Animal Behavior Processes*, (27):354–372, 2001.
- [3] J. M. Kleinberg. Bursty and hierarchical structure in streams. *Data Min. Knowl. Discov.*, 7(4), 2003.
- [4] A. McCallum, K. Nigam, and L. H. Ungar. Efficient clustering of high-dimensional data sets with application to reference matching. In *KDD*, pages 169–178, 2000.
- [5] I. D. Melamed. Automatic evaluation and uniform filter cascades for inducing n-best translation lexicons. *CoRR*, cmp-lg/9505044, 1995.
- [6] G. A. Miller. Wordnet: A lexical database for english. In *HLT*, 1994.
- [7] M. O’Mahony. *Sensory Evaluation of Food: Statistical Methods and Procedures*. 1986.
- [8] G. Salton and M. McGill. *Introduction to modern information retrieval*. 1983.
- [9] S. Sung and D. McLeod. Ontology-driven semantic matches between database schemas. In *ICDE Workshops*, page 6, 2006.