International Workshop on Mining for and from the Semantic Web

Workshop Chairs:

Andreas Hotho
York Sure
Lise Getoor

August 22, 2004
Seattle, USA

Mining for and from the Semantic Web 2004 (SWM 2004)

The intention of this workshop on "Mining for and from the Semantic Web" is to bring together researchers from the two research areas Semantic Web and Knowledge Discovery. According to T. Berners-Lee the Semantic Web is "an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation". Current standardization efforts include e.g. the W3C recommendation for the Web Ontology Language (OWL). Knowledge Discovery is defined by U.M. Fayyad as "the nontrivial process of identifying valid, previously unknown, potentially useful patterns in data".

We foresee two typical ways of combining these areas. On the one hand, mining for the semantic web includes the application of knowledge discovery methods and techniques to support the setting up of the semantic web itself. Prominent examples are here ontology learning and population of ontologies (instance learning). On the other hand, mining from the semantic web emphasizes the usage of semantic web technologies for mining purposes such as e.g. the usage of taxonomies in recommender systems, applying association rules with generalizations or clustering with background knowledge in form of ontologies.

We thank the members of our program committee for their efforts to ensure the quality of accepted papers. We kindly acknowledge the EU integrated project SEKT[1] and EU thematic network Knowledge Web[2] for supporting this workshop.

We are looking forward to having interesting presentations and fruitful discussions.

July 2004

Your MSW team
Andreas Hotho, York Sure, Lise Getoor

[1] http://sekt.semanticweb.org
[2] http://knowledgeweb.semanticweb.org/

Workshop Chairs

Andreas Hotho
KDE Group at University of Kassel
D–34121 Kassel, Germany
http://www.kde.cs.uni-kassel.de/hotho
hotho@cs.uni-kassel.de

York Sure
Institute AIFB at University of Karlsruhe
D–76128 Karlsruhe, Germany
http://www.aifb.uni-karlsruhe.de/WBS/ysu
sure@aifb.uni-karlsruhe.de

Lise Getoor
Computer Science Dept/UMIACS at University of Maryland AV Williams Bldg,
College Park, MD 20742
http://www.cs.umd.edu/~getoor
getoor@cs.umd.edu

Program Committee

AnHai Doan
(University of Illinois)
Bamshad Mobasher
(DePaul University, Chicago)
Bettina Berendt
(Humboldt Universitaet Berlin)
Katia Sycara
(Carnegie Mellon University, Pittsburgh)
Kristina Lerman
(ISI, University of Southern Calofirnia)
Lyle Ungar
(University of Pennsylvania)
Marko Grobelnik
(J. Stefan Institute, Ljubljana)
Natasha Fridman Noy
(Stanford Medical Informatics)
Rudi Studer
(University of Karlsruhe)
Vipul Kashyap
(US National Library of Medicine)

Further Reviewers

Peter Haase
(University of Karlsruhe)
Stephan Bloehdorn
(University of Karlsruhe)

Table of Contents

The Terascale Challenge

Deepak Ravichandran, Patrick Pantel, and Eduard Hovy
Information Sciences Institute
University of Southern California
4676 Admiralty Way
Marina del Rey, CA 90292.
{ravichan,pantel,hovy}@isi.edu

Abstract

Although vast amounts of textual data are freely available, many NLP algorithms exploit only a minute percentage. In this paper, we study the challenges of working at the terascale and survey reasons why researchers are not fully utilizing available resources. As a case study, we present a terascale algorithm for mining *is-a* relations that achieves better performance as compared to a state-of-the-art linguistically-rich method.

1 Introduction

The Natural Language Processing (NLP) community has recently seen a growth in corpus-based methods. Algorithms light in linguistic theories but rich in available training data have been successfully applied to several applications such as machine translation [15], information extraction [9], and question answering [5, 17].

In the last decade, we have seen an explosion in the amount of available digital text resources. It is estimated that the Internet contains hundreds of terabytes of text data, a sizable amount of which is in an unstructured format. State of the art search engines index more than four billion web pages. Yet, many NLP algorithms tap into only megabytes or gigabytes of this information.

In this paper, we study the challenges of working at the terascale and survey reasons why researchers are not fully utilizing available resources. We present an algorithm for extracting *is-a* relations designed for the terascale and compare it, in a preliminary study, to a state of the art method that employs deep analysis of text. We show that by simply utilizing more data on this task, we can achieve similar performance to a linguistically rich approach. *Is-a* relations are roughly characterized by the questions *What/Who is X?*. Examples of *is-a* relation are:

1. *Red* is a *color*.
2. *United States* is a *country*.
3. *Martin Luther King* was a *leader*.

In the above examples, we call *red, United States, and Martin Luther King* instances of the respective concepts *color, country and leader*.

2 Related Work

Banko and Brill [1, 2] investigated the advantages of working with very large corpora. In particular, they worked on the problem of confusion set disambiguation. It is the problem of choosing the correct use of a word from a confusion set such as {*principle, principal*}, {*then, than*}, {*to, two, too*}, and {*weather, whether*}. They empirically proved the following:

1. The learning curve is generally log-linear irrespective of the algorithm.

2. Simple and sophisticated algorithms have comparable performance with very large amounts of data. In particular, the technique of voting by using different classifiers trained on the same corpus seems to be ineffective in improving performance with large amounts of data.

3. One can achieve good performance by using supervised learning techniques and by employing active learning and sample selection.

4. Weakly supervised techniques almost seem to have little effect on performance accuracy.

Curran and Moens [7] experimented with corpus size and complexity of proximity features in building automatic thesauri. The important message to be taken home from these papers is that working with more data definitely helps.

3 Why NLP Researchers Haven't used Terabytes of Data to Date?

Statistical/Empirical techniques employed by NLP researchers to date operate on data on the order of megabytes or gigabytes. We argue that this is because of the following reasons:

1. A lot of NLP researchers have successfully made use of supervised training approaches to build several applications. Examples include POS taggers and syntactic parsers. These algorithms make use of tagged data by humans (e.g. FrameNet, Penn Tree Bank). This is a time consuming and extremely costly process.

2. Many applications such as Question Answering (QA) make use of NLP tools (e.g. syntactic parsers) which require large amounts of processing time and are not easily scalable to terabytes of data.

3. Many unsupervised algorithms (e.g. clustering algorithms) are not linear with respect to the size of the corpus. Hence, they work well only on a small corpus size and cannot be scaled to the terabyte level.

4. Terabytes of text data is not made readily available to NLP researchers by organizations like LDC (Linguistic Data Consortium). Acquiring large collections require downloading data from the Internet which is an extremely time consuming process requiring expertise in networking, distributed computing and fault tolerance.

Table 1: Projected rate of increase for various technologies.

Technology	Rate of increase
Processor Speed	100% every 1.5 years
Hard Disk Capacity	100% every year
Hard Disk Access	10% every year

5. Most of the NLP research groups do not have the necessary infrastructure (e.g. hardware, software, support staff, money) to work with such kinds of data.

4 Challenges of Working with Terabytes of data

Working on terabytes of data poses new challenges which require various engineering and algorithmic changes to the current approaches. Some of the basic challenges are:

1. Algorithms: Algorithms have to be strictly linear with respect to the size of the corpus *O(n)*. It is impossible to work with algorithms which are more than linear with the current computing power. Also, algorithms should involve only unsupervised or semi-supervised machine learning techniques. It is not trivial to hand tag data which is in the order of terabytes.

2. Storage: How would one store terabytes of data? The answer to this question is straightforward – hard disks. It is estimated that data storage capacity doubles every year[1]. A terabyte of data today costs less than $5,000. It is estimated that by the early 2010s we could buy a petabyte of data for the same cost as a terabyte costs today.

3. Data access: What is the rate at which one could access data? The data access rate from hard drives has only been growing a rate 10% a year, thus, growing an order of magnitude slower than the data storage rate. This probably means that we have to rethink the ways in which we access data. As we learnt in basic Computer Science textbooks, accessing a random location on a disk involves an overhead in terms of disk head rotation and seeking. This is a major source of delay. Disks allow roughly 200 accesses per second. So, if one reads only a few kilobytes in every disk access, it will take almost a year to read data from a 20 terabyte disk [10]. To significantly simplify our data access problems we may need to start using our disks as tapes, i.e., start using the inexpensive disks as tape drives by performing sequential access. If one reads and writes large chunks of data, data access speeds can be increased 500 times. Table 1 summarizes the differences in speeds for various technologies.

4. NLP tools: Which NLP tools could one use? One of the biggest achievements in NLP in the 1990s has been the availability of free tools to perform various tasks such as syntactic parsing, dependency parsing, discourse parsing, named-entity identification, part of speech taggers, etc. Almost all of these tools work linearly on an inter-sentence level. This is because they treat each sentence independently from other sentences. (However, in the intra-sentence level these tools may perform non-linearly

[1] This statement holds true only after 1989. Between 1960 and 1989 data storage grew only at the rate of 30%.

Table 2: Approximate processing time on a single Pentium-4 2.5 GHZ machine for a 1 Terabyte text corpus.

Tool	Processing time
POS Tagger	125 days
NP Chunker	216 days
Dependency Parser	10.2 years
Syntactic Parser	388.4 years

Table 3: Examples of *is-a* relation.

Co-occurrence-based system		Pattern-based system	
Instance	Concept	Instance	Concept
azalea	flower	American	airline
bipolar disorder	disease	Bobby Bonds	coach
Bordeaux	wine	radiation therapy	cancer treatment
Flintstones	television show	tiramisu	dessert
salmon	fish	Winona Ryder	actress

as a function of the number of words in the sentence.) We study and apply various off-the-shelf tools to data sets and estimate the amount of time taken to process a terabyte corpus. We take Brill's part of speech tagger [4], noun phrase chunker CASS [3], Lin's dependency parser Minipar [11], and Charniak's syntactic parser [6]. Results are shown in Table 2. It is very clear that terabyte-sized experiments cannot use any NLP tools in the current form.

5. Computing Power: What computer should one use? Computers have been following Moore's law: computer processing speed doubles every 18 months. An exciting development over the past years has been the availability of cluster computers to NLP researchers. Cluster computers are relatively cheaper as compared to Vector computers because they are built from cheap and mass-produced Intel processors with free Linux operating system, installed on them. Cluster computers also have a gigabit switch between them, acting like a cheap context switch. Using a cluster computer with hundreds of nodes, part of speech tagging and noun phrase chunking becomes manageable at the terascale level. However, syntactic parsers and dependency parsers are still too slow.

5 *Is-a* Relation Extraction

As a case study, we now proceed to briefly describe two models to extract of *is-a* relations: **1.** Co-occurrence model which employs linguistically-rich motivated features. **2.** Pattern-based model which employs linguistically-light features such as lexical words and POS tokens. Details of these models appear in [14]. Some examples of extracted *is-a* relations are shown in Table 3.

5.1 Co-occurrence Model

The co-occurrence model as proposed by Pantel and Ravichandran [13] employs clustering technology to extract *is-a* relations. Clustering by Committee (CBC) [12] is used to extract clusters of nouns belonging in the same semantic class. The clustering algorithm employs as features the grammatical contexts of words as output by the dependency parser Minipar [11]. As an example, the output of the clustering algorithm for the fruit sense of orange would contain the following members:

{ *... peach, pear, apricot, strawberry, banana, mango, melon, apple, pineapple, cherry, plum, lemon, grapefruit, orange, berry, raspberry, blueberry, kiwi, ...*}

For each cluster, certain signature features are extracted which are known to signify *is-a* relations. Examples of such features include appositives (e.g. ... *Oracle*, a *company* known for its progressive employment policies, ..) and nominal subjects (e.g. ... *Apple* was a hot young *company*, with Steve Jobs in charge.). These signature features are used to extract the name of each cluster. The highest ranking name of each cluster is used as the concept for each member of the cluster. For example, the top five ranking names for a cluster containing the following elements:

{*...Curtis Joseph, John Vanbiesbrouck, Mike Richter, Tommy Salo..*}

are:

1) goalie
2) goaltender
3) goalkeeper
4) player
5) backup

The syntactical co-occurrence approach has worst case time complexity $O(n^2k)$, where n is the number of words in the corpus and k is the feature-space. Just to parse a 1 TB corpus, this approach requires approximately 10.2 years (see Table 2).

5.2 Pattern-based Model

The pattern based algorithm was specifically designed to be scalable to the terabyte level. It makes use of only POS and surface text patterns. It consists of the following steps:

1. Learn lexico-POS patterns that signify *is-a* relations using a bootstrapping approach. The following patterns are learnt from this procedure along with their underlying part of speech variations:

1. X , or Y
2. X , (a|an) Y
3. X , Y
4. Y , or X
5. X , _DT Y _(WDT|IN)
6. X is a Y
7. X , _RB known as Y
8. X (Y)
9. Y such as X
10. X, _RB called Y

11. Y like X and
12. _NN , X and other Y
13. Y , including X ,
14. Y , such as X
15. Y , especially X

2. Apply the learned patterns to a POS tagged corpus to extract *is-a* relations.

3. Apply a Maximum Entropy based machine learning filter that exploits redundancy, capitalization and other features to weed out bad relations from legitimate ones. Details of the Machine Learning filter are given in the next section.

6 Maximum Entropy Filter

In the next step, each extracted noun phrase is passed through a Machine learning filter which is a model to predict the correctness of the given *is-a* relation. In the following section, we describe the model in detail.

6.1 Model

We model a Maximum entropy model to predict the correctness of a given *is-a* relation using the following equation.

$$p(c|a,b) = \frac{exp(\sum_{m=1}^{M} \lambda_{mc} f_m(a,b))}{\sum_{c'} exp(\sum_{m=1}^{M} \lambda_{mc'} f_m(a,b))} \tag{1}$$

where,
a is the concept part of *is-a* relation.
b is the instance part of *is-a* relation.
$f_m,\ m = \{1, 2..., M\}$ are the M feature functions.
$\lambda_m,\ m = \{1, 2..., M\}$ are the corresponding M model parameters.
$c', c\ \epsilon \{false, true\}$ the decisions to be made for every instance-concept pair.

The features used to model the Eq. 1 can be classified into the following four main categories:

1. **Capitalization features**: These features check to see if certain nouns of the instance-concept begins with a capitalized letter or not. Some features are used to check if the entire instance is capitalized.

2. **Pattern-based features**: These features check to see what kind of pattern triggered this particular instance-concept pair.

3. **Lexicalized features**: These type of features checks to see if the head noun of the concept contains suffixes such as *er, or, ing, ist, man, ary, ant*. Honorific mentions such *Mr., Dr., Ms.* are also checked.

Table 4: Precision and Recall Results on True relations using Machine Learning Filter.

Sample Size	Precision	Recall
500	78%	84%

4. **Co-occurrence based features**: In this category we calculate how many times the instance-concept pair was independently observed in the corpus.

6.2 Training

We randomly sampled 1000 examples from the extracted list of *is-a* relations and asked a human to tag as *correct* or *incorrect*. We used 500 examples from the above set for training and 500 examples for testing and development.

We use Gradient Iterative Scaling algorithm (GIS) [8] to train our Maximum Entropy model implemented by YASMET [2].

6.3 Results

The results of the output of the Machine Learned filter are shown are shown in Table 4. We caution the readers that these are only the precision and recall results for the output of the Machine Learning filter. They do not measure the actual precision wherein we fuse duplicate instance-concept pairs into one output. Similarly they do not measure the actual recall of the system.

The above pattern-based algorithm runs in linear time, *O(n)*, where *n* is the size of the corpus.

7 Experiments with Corpus Size

For a pilot study, we study the task of mining *is-a* relations as a function of corpus size. For this purpose the data set is divided into different sets: 1.5 megabytes, 15 megabytes, 150 megabytes, 1.5 gigabytes, 6 gigabytes and 15 gigabytes. Three systems are evaluated:

1. Co-occurrence based system (as described in subsection 5.1).

2. Pattern-based system **without** the application of the Maximum Entropy based filter (as described in subsection 5.2)

3. Pattern-based system **with** the application of the Maximum Entropy based filter (as described in section 6).

Note that the 15GB corpus was too large to process for the Co-occurrence model. Table 5 tabulates the results. For precision calculations, we extract 50 instances (from the *is-a* list) from each system trained from different corpus size, at random. For each instance we extract the top 3 frequently occurring concepts. These are then judged

[2] YASMET – Yet Another Small Maximum Entropy Toolkit – http://www.isi.edu/~och/YASMET/

Table 5: Approximate corpus size and instance-concept pairs extracted

Corpus	Co-occurrence		Pattern w/o filter		Pattern with filter	
Size	#	Prec.	#	Prec.	#	Prec.
1.5 MB	629	4.3%	494	38.7%	303	63.4%
15 MB	8725	14.6%	4,211	39.1%	2,914	55.6%
150 MB	93725	51.1%	40,967	40.6%	26,467	60.6%
1.5 GB	93,725	56.7%	418,949	40.4%	274,716	65.7%
6 GB	171,066	64.9%	1,398,422	46.3%	981,482	76.9%
15 GB	Too large to process		2,495,598	55.9%	1,809,579	NA
150 GB	??	??	??	??	??	??
1.5 TB	??	??	??	??	??	??

manually by a human as being *correct*, or *incorrect*. The Kappa statistic [16] measures the agreements between a set of judges assessments correcting for chance agreements:

$$K = \frac{P(A) - P(E)}{1 - P(E)} \tag{2}$$

where, *P(A)* is the probability of agreement between the judges and *P(E)* is the probability that the judges agree by chance on an assessment. An experiment with $K = 0.8$ is generally viewed as reliable and $0.67 < K < 0.8$ allows tentative conclusions.

Results for System 1 (Co-occurrence) and System 2 (Pattern based without filter) were evaluated with two human judges. The reported Kappa statistics agreement has a score greater than $k = 0.75$. However, the evaluation of System 3 is preliminary and was performed by only one judge.

The graph in Figure 1 shows that the relation between the number of extracted instance-concept pairs and the corpus size is linear for both pattern-based systems. However, for the co-occurrence based system, the same relation is sub-linear. Note that the x-axis (corpus-size) of the graph is on a log scale while the y-axis (extracted relation-pairs) is on a linear scale.

Figure 2 shows the relationship between the precision of the extracted relations and the corpus size. It is clear that the precision of each system increases with more data. We suspect that the precision curve is log-linear. However, only working on a larger corpus size will prove this point. For small datasets (below 150MB), the pattern-based (without filter) method achieves higher precision compared to the co-occurrence method since the latter requires a certain critical mass of statistics before it can extract useful class signatures. On the other hand, the pattern-based approach has relatively constant precision since most of the *is-a* relations selected by it are fired by a single pattern. Once the co-occurrence system reaches its critical mass (at around 150MB), it generates much more precise *is-a* relations. The pattern-based method with filter shows a lot of promise. However, we again wish to caution the reader that the evaluations for the pattern-based system with filter was performed using only one human judge and hence the results are preliminary.

On the 6 GB corpus, the co-occurrence approach took approximately 47 single Pentium-4 2.5 GHZ processor days to complete, whereas it took the pattern-based ap-

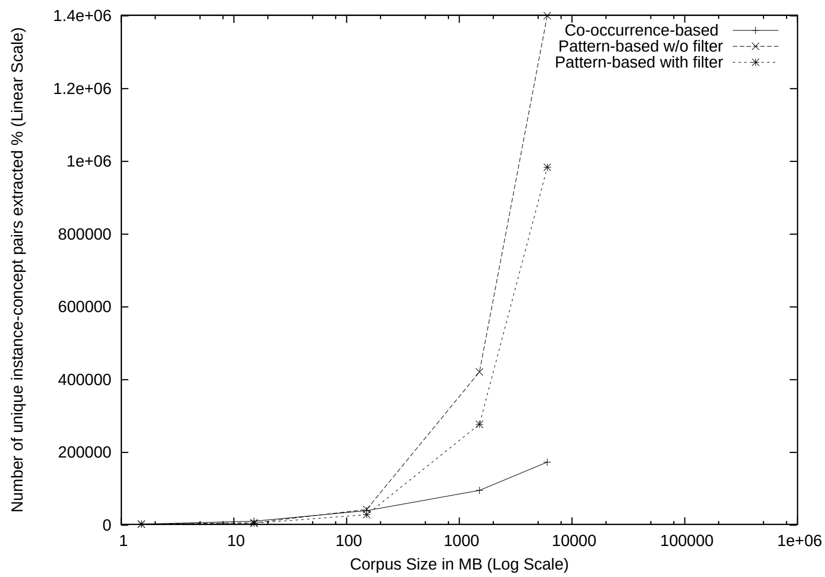


Figure 1: Graph showing the number of unique instance-concept pair extracted as a function of corpus size.

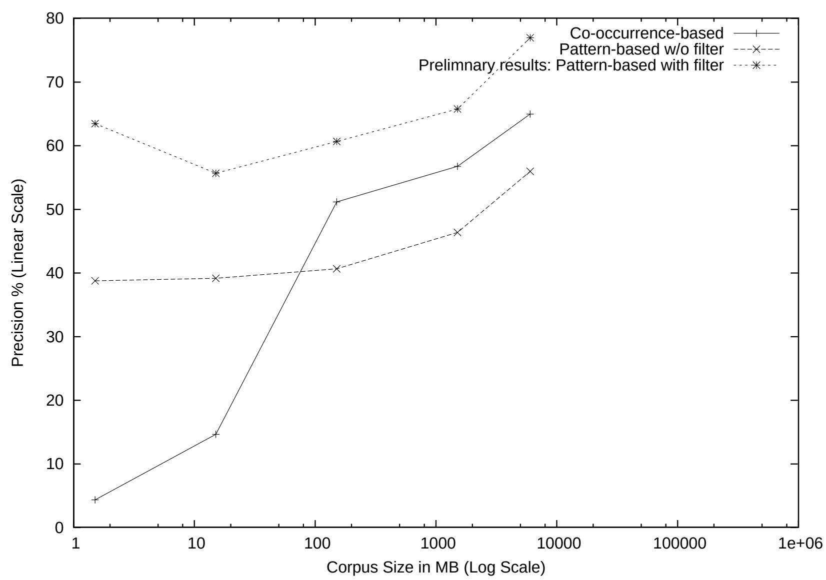


Figure 2: Graph showing the precision of the extracted relations as a function of the corpus size.

proach only four days to complete. It took the pattern-based system 10 days on 15GB corpus.

The results are very encouraging for the linguistically-light pattern based method. The linguistically-rich co-occurrence approach has a problem with respect to scalability. Scaling the entire process to a terabyte holds a lot of promise. We expect to see more relations because proper nouns are potentially an open set and we do learn a lot of proper nouns. The redundancy factor of the knowledge may help to improve precision. We plan to use these extracted relations for knowledge acquisition.

8 Conclusion

In this paper, we explored the various challenges associated with working on terabytes of data. We also made a strong case for working with more data by contrasting two different approaches for extracting *is-a* relations. The shallow pattern based methods with a machine filter has better performance than linguistically rich method. Albeit possible to successfully apply linguistically-light but data-rich approaches to some NLP applications, merely reporting these results often fails to yield insights into the underlying theories of language at play. Our biggest challenge as we venture to the terascale is to use our new found wealth not only to build better systems, but to improve our understanding of language.

References

[1] Banko, Michele and Eric Brill: Mitigating the Paucity of Data Problem. In Proceedings of the *Conference on Human Language Technology*, San Diego, CA. (2001).

[2] Banko, Michele and Eric Brill: Scaling to a Very Very Large Corpora for Natural Language Disambiguation. Proceeding of the *Association for Computational Linguistics*, Toulouse, France. (2001).

[3] Berwick, Robert, Steven Abney, and Carol Tenny, editors: Principle-Based Parsing. *Kluwer Academic Publishers.* (1991).

[4] Brill, Eric: Transformation-Based Error-Driven Learning and Natural Language Processing: A Case Study in Part of Speech Tagging. *Computational Linguistics*. (1995).

[5] Brill, E., J. Lin, M. Banko, S. Dumais, and A. Ng: Data-Intensive Question Answering. Proceedings of the *TREC-10 Conference*, pp 183–189. NIST, Gaithersburg, MD. (2001).

[6] Charniak, Eugene: A Maximum-Entropy-Inspired Parser Proceedings of *NAACL*. Seattle, WA. (2000).

[7] Curran, J. and Moens, M.: Scaling context space. In Proceedings of *ACL-02*. pp 231–238, Philadelphia, PA. (2002).

[8] Darroch, J. N., and D. Ratcliff: Generalized iterative scaling for log-linear models. *Annals of Mathematical Statistics*, 43:1470–1480. (1972).

[9] Etzioni, Oren, Michael Cafarella, Doug Downey, Stanley Kok, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S. Weld, and Alexander Yates: Web-scale Information Extraction in KnowItAll (Preliminary Results). To appear in the conference on *WWW*. (2004).

[10] Gray, Jim: A Conversation with Jim Gray *ACM Queue* vol. 1, no. 4 – June 2003.

[11] Lin, Dekang: Principar - an Efficient, Broad-Coverage, Principle-Based Parser. In Proceedings of *COLING-94*. pp. 42-48. Kyoto, Japan. (1994).

[12] Pantel, Patrick and Dekang Lin: Discovering Word Senses from Text. In Proceedings of *SIGKDD-02*. pp. 613–619. Edmonton, Canada. (2002).

[13] Pantel, Patrick and Deepak Ravichandran: Automatically Labeling Semantic Classes. In the Proceedings of *NAACL/HLT*, Boston, MA. (2004).

[14] Pantel, Patrick, Deepak Ravichandran, and Eduard Hovy: Towards Terascale Knowledge Acquisition. To appear in the Proceedings of the *COLING* conference, Geneva, Switzerland. (2004).

[15] Och, Franz Josef and Hermann Ney: Discriminative Training and Maximum Entropy Models for Statistical Machine Translation. In Proceedings of *ACL*, pp. 295–302. Philadelphia, PA. (2002).

[16] Siegel, S. and N. J. Castellan Jr.: Nonparametric Statistics for the Behavioral Sciences. *McGraw-Hill*. (1998).

[17] Wang, B., H. Xu, Z. Yang, Y. Liu, X. Cheng, D. Bu, and S. Bai: TREC-10 Experiments at CAS-ICT: Filtering, Web, and QA. Proceedings of the *TREC-10 Conference*, pp 229–241. NIST, Gaithersburg, MD. (2001).

Large-Scale Extraction of Fine-Grained Semantic Relations between Verbs

Timothy Chklovski and Patrick Pantel

Information Sciences Institute
University of Southern California
4676 Admiralty Way
Marina del Rey, CA 90292
{timc,pantel}@isi.edu

Abstract. Broad-coverage repositories of semantic relations between actions could benefit many NLP tasks, as well as tasks related to reasoning and inference. We present an automatic method for extracting fine-grained semantic relations, addressing relations between verbs. We detect similarity, strength, antonymy, enablement, and temporal relations between pairs of verbs with high mutual information using lexico-syntactic patterns over the Web. On a set of 26,118 strongly associated verb pairs, our extraction algorithm yielded 56.5% accuracy[1]. On the relations *strength* and *similarity*, we achieved 79.6% and 66.7% accuracy respectively.

1 Introduction

Many tasks, such as question answering, summarization, and machine translation could benefit from broad-coverage semantic resources such as WordNet (Miller 1990) and EVCA (English Verb Classes and Alternations) (Levin 1993). These extremely useful resources have very high precision entries but have important limitations when used in real-world tasks due to their limited coverage and prescriptive nature (i.e. they do not include semantic relations that are plausible but not guaranteed). For example, it may be valuable to know that if someone has bought an item, they may sell it at a later time. WordNet does not include the relation "*X buys Y*" *happens-before* "*X sells Y*" since it is possible to sell something without having bought it (e.g. having manufactured or stolen it).

Verbs are the primary vehicle for describing events and expressing relations between entities. Hence, verb semantics could help in many natural language processing (NLP) tasks that deal with events or relations between entities. For NLP as well as reasoning and inference tasks which require canonicalization of natural language statements or derivation of plausible inferences from such statements, a particularly valuable resource is one which (i) relates verbs to one another and (ii) provides broad coverage of the verbs in the target language.

In this paper, we present an algorithm that automatically discovers fine-grained verb semantics by querying the Web using simple lexico-syntactic patterns. The verb

[1] The relations are available for download at *http://semantics.isi.edu/ocean/*.

relations we discover are similarity, strength, antonymy, enablement, and temporal relations. Our approach extends previously formulated ones that use surface patterns as indicators of semantic relations between nouns (Hearst 1992; Etzioni 2003; Ravichandran and Hovy 2002). We extend these approaches in two ways: (i) our patterns indicate verb conjugation to increase their expressiveness and specificity and (ii) we use a measure similar to mutual information to account for both the frequency of the verbs whose semantic relations are being discovered as well as for the frequency of the pattern.

2 Related Work

In this section, we describe application domains that can benefit from a resource of verb semantics. We then introduce some existing resources and describe previous attempts at mining semantics from text.

2.1 Applications

Question answering is often approached by canonicalizing the question text and the answer text into logical forms. This approach is taken, *inter alia*, by a top-performing system (Moldovan et al. 2002). In discussing future work on the system's logical form matching component, Rus (2002 p. 143) points to incorporating entailment and causation verb relations to improve the matcher's performance. In other work, Webber et al. (2002) have argued that successful question answering depends on lexical reasoning, and that lexical reasoning in turn requires fine-grained verb semantics in addition to troponymy (*is-a* relations between verbs) and antonymy.

In multi-document summarization, knowing verb similarities is useful for sentence compression and for determining sentences that have the same meaning (Lin 1997). Knowing that a particular action happens before another or is enabled by another is also useful to determine the order of the events (Barzilay et al. 2002). For example, to order summary sentences properly, it may be useful to know that selling something can be preceded by either buying, manufacturing, or stealing it. Furthermore, knowing that a particular verb has a meaning stronger than another (e.g. *rape* vs. *abuse* and *renovate* vs. *upgrade*) can help a system pick the most general sentence.

In lexical selection of verbs in machine translation and in work on document classification, practitioners have argued for approaches that depend on wide-coverage resources indicating verb similarity and membership of a verb in a certain class. In work on translating verbs with many counterparts in the target language, Palmer and Wu (1995) discuss inherent limitations of approaches which do not examine a verb's class membership, and put forth an approach based on verb similarity. In document classification, Klavans and Kan (1998) demonstrate that document type is correlated with the presence of many verbs of a certain EVCA class (Levin 1993). In discussing future work, Klavans and Kan point to extending coverage of the manually constructed EVCA resource as a way of improving the performance of the system. A wide-coverage repository of verb relations including verbs linked by the similarity relation will provide a way to automatically extend the existing verb classes to cover more of the English lexicon.

2.2 Existing resources

Some existing broad-coverage resources on verbs have focused on organizing verbs into classes or annotating their frames or thematic roles. EVCA (English Verb Classes and Alternations) (Levin 1993) organizes verbs by similarity and participation / non-participation in alternation patterns. It contains 3200 verbs classified into 191 classes. Additional manually constructed resources include PropBank (Kingsbury et al. 2002), FrameNet (Baker et al. 1998), VerbNet (Kipper et al. 2000), and the resource on verb selectional restrictions developed by Gomez (2001).

Our approach differs from the above in its focus. We relate verbs to each other rather than organize them into classes or identify their frames or thematic roles. WordNet does provide relations between verbs, but at a coarser level. We provide finer-grained relations such as strength, enablement and temporal information. Also, in contrast with WordNet, we cover more than the prescriptive cases.

2.3 Mining semantics from text

Previous web mining work has rarely addressed extracting many different semantic relations from Web-sized corpus. Most work on extracting semantic information from large corpora has largely focused on the extraction of *is-a* relations between nouns. Hearst (1992) was the first followed by recent larger-scale and more fully automated efforts (Pantel and Ravichandran 2004; Etzioni et al. 2004; Ravichandran and Hovy 2002).

Turney (2001) studied word relatedness and synonym extraction, while Lin et al. (2003) present an algorithm that queries the Web using lexical patterns for distinguishing noun synonymy and antonymy. Our approach addresses verbs and provides for a richer and finer-grained set of semantics.

Semantic networks have also been extracted from dictionaries and other machine-readable resources. MindNet (Richardson et al. 1998) extracts a collection of triples of the type "*ducks have wings*" and "*duck capable-of flying*". This resource, however, does not relate verbs to each other or provide verb semantics.

3 Semantic relations among verbs

In this section, we introduce and motivate the specific relations that we extract. Whilst the natural language literature is rich in theories of semantics (Barwise and Perry 1985; Schank and Abelson 1977), large-coverage manually created semantic resources typically only organize verbs into a flat or shallow hierarchy of classes (such as those described in Section 2.2). WordNet identifies synonymy, antonymy, troponymy, and cause. As summarized in Figure 1, Fellbaum (1998) discusses a finer-grained analysis of entailment, while the WordNet database does not distinguish be-

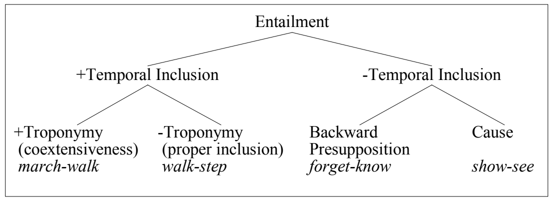


Figure 1. Fellbaum's (1998) entailment hierarchy.

Table 1. Semantic relations we identify. *Siblings* in the WordNet column refers to terms with the same troponymic parent, e.g. *swim* and *fly*.

Semantic Relation	*Example*	*Alignment with WordNet*	*Symmetric*
similarity	transform :: integrate	synonyms or siblings	Y
strength	push :: nudge	synonyms or siblings	N
antonymy	open :: close	antonymy	Y
enablement	wash :: clean	cause	N
happens-before	buy :: have; marry :: divorce	cause; entailment, no temporal inclusion	N
happens-while	chew :: eat snore :: sleep	entailment proper temporal inclusion, no troponymy	N

tween, e.g., proper temporal inclusion (*walk :: step*) from backward presupposition (*forget :: know*). In formulating our set of relations, we have relied on the finer-grained analysis.

In selecting the relations to identify, we aimed at both covering the relations described in WordNet and covering the relations present in our collection of strongly associated verb pairs. We relied on the strongly associated verb pairs, described in Section 4.3, for computational efficiency. The relations we identify were experimentally found to cover 99 out of 100 randomly selected verb pairs.

Our algorithm identifies six semantic relations between verbs. These are summarized in Table 1 along with their closest corresponding WordNet category and the symmetry of the relation (whether V_1 *rel* V_2 is equivalent to V_2 *rel* V_1).

Similarity. As Fellbaum (1998) and the tradition of organizing verbs into similarity classes indicate, verbs do not neatly fit into a unified *is-a* (troponymy) hierarchy. Rather, verbs are often similar or related. Similarity between action verbs, for example, can arise when they differ in connotations about manner or degree of action. Examples extracted by our system include *maximize :: enhance*, *produce :: create*, *reduce :: restrict*.

Strength. When two verbs are similar, one may denote a more intense, thorough, comprehensive or absolute action. In the case of change-of-state verbs, one may denote a more complete change. We identify this as the *strength* relation. Sample verb pairs extracted by our system, in the order *weak :: strong*, are: *taint :: poison, permit :: authorize, surprise :: startle, startle :: shock.*

This subclass of similarity has not been identified in broad-coverage networks of verbs, but may be of particular use in natural language generation and summarization applications.

Antonymy. Also known as semantic opposition, antonymy between verbs has several distinct subtypes. As discussed by Fellbaum (1998), it can arise from switching thematic roles associated with the verb (as in *buy :: sell, lend :: borrow*). There is also antonymy between stative verbs (*live :: die, differ :: equal*) and antonymy between sibling verbs which share a parent (*walk :: run*) or an entailed verb *(fail* :: *succeed* both entail *try*).

Antonymy also systematically interacts with the *happens-before* relation in the case of restitutive opposition (Cruse 1986). This subtype is exemplified by *damage :: repair*, *wrap :: unwrap*. In terms of the relations we recognize, it can be stated that *restitutive-opposition*(V_1, V_2) = *happens-before*(V_1, V_2), and *antonym*(V_1, V_2). Examples of antonymy extracted by our system include: *assemble :: dismantle; ban :: allow; regard :: condemn, roast :: fry*.

Enablement. This relation holds between two verbs V_1 and V_2 when the pair can be glossed as V_1 *is accomplished by* V_2. Enablement is classified as a type of causal relation by Barker and Szpakowicz (1995). Examples of enablement extracted by our system include: *assess :: review* and *accomplish :: complete*.

Happens-before. This relation indicates that the two verbs refer to two temporally disjoint intervals or instances. WordNet's *cause* relation, between a causative and a resultative verb (as in *buy :: own*), would be tagged as instances of *happens-before* by our system. Examples of the *happens-before* relation identified by our system include *marry :: divorce*, *detain :: prosecute, enroll :: graduate*, *schedule :: reschedule*, *tie :: untie*.

Happens-while. This relation indicates proper temporal inclusion, either of a repeating activity (*chew :: eat*) or an event (*find :: study*). In some cases also classified as *happens-while*, it may be difficult to say if the temporal inclusion is necessarily strict, as in *say :: announce*.

4 Approach

We discover the semantic relations described above by querying the Web with Google for lexico-syntactic patterns indicative of each relation. Our approach has two stages. First, we identify pairs of highly associated verbs co-occurring on the Web in sufficient volume. These pairs are extracted using previous work by Lin and Pantel (2001), as described in Section 4.3. Next, for each verb pair, we tested lexico-syntactic patterns, outputting the first detected relation[2].

[2] In effect, we are making the simplifying assumption that at most one relation needs to be detected. This assumption may be relaxed in future work.

Table 2. Semantic relations and samples of the 33 surface patterns used to identify them. In patterns, "*" matches any single word. Punctuation does not count as words by the search engine used (Google). Relations are shown in the order of testing.

SEMANTIC RELATION	*Surface Patterns*	$Hits_{est}$ *for patterns*
happens-while	to X while Ying; Xed while Ying	6,752,541
strength	X and even Y; Yed or at least Xed	2,172,811
happens-before	Xed * and then Yed; to X and eventually Y	4,074,935
enablement	Xed * by Ying the; to X * by Ying or	2,348,392
antonymy	to X * but Y; Xed * * but Yed	18,040,916
nonequivalence-but-similarity*	both Xed and Yed; X rather than Y	1,777,755
broad similarity*	Xed and Yed; Xs and Ys; to X and Y	174,797,897

*nonequivalence-but-similarity and broad-similarity were later combined into a single category, *similarity*, and are treated as a single category in the rest of our discussion.

4.1 Lexico-syntactic patterns

The lexico-syntactic patterns were manually selected by examining pairs of verbs in known semantic relations. They were refined to decrease capturing wrong parts of speech or incorrect semantic relations.

Although many patterns may indicate the relations, we use a total of 33 patterns. Some representatives are shown in Table 2. Note that our patterns specify the tense of the verbs they accept. When instantiating these patterns, we conjugate as needed. For example, "*both Xed and Yed*" instantiates on *sing* and *dance* as "*both sung and danced*".

4.2 Testing for a semantic relation

In this section, we describe how the presence of a semantic relation is detected. We test the relations in the order specified in Table 2. We adopt an approach inspired by mutual information to measure the strength of association, denoted $S_p(V_1, V_2)$, between three entities: a verb pair V_1 and V_2 and a lexico-syntactic pattern p:

$$S_p(V_1, V_2) = \frac{P(V_1, p, V_2)}{P(p) \times P(V_1) \times P(V_2)} \quad \textbf{(1)}$$

The probabilities in the denominator are difficult to calculate directly from search engine results. For a given lexico-syntactic pattern, we need to estimate the frequency of the pattern instantiated with appropriately conjugated verbs. For verbs, we need to estimate the frequency of the verbs, but avoid counting other parts-of-speech (e.g. *chair* as a noun or *painted* as an adjective). Another issue is that some relations are symmetric (we treat *similarity* and *antonymy* as symmetric), while others are not (*strength*, *enablement*, *happens-while*, *happens-before*). For symmetric relations only, the verbs can fill the lexico-syntactic pattern in either order. To address these issues, we estimate $S_p(V_1,V_2)$ using:

$$S_P(V_1,V_2) \approx \frac{\frac{hits(V_1,p,V_2)}{N}}{\frac{hits_{est}(p)}{N} \times \frac{hits(\text{"to } V_1\text{"}) \times C_v}{N} \times \frac{hits(\text{"to } V_2\text{"}) \times C_v}{N}} \tag{2}$$

for asymmetric relations and

$$S_P(V_1,V_2) \approx \frac{\frac{hits(V_1,p,V_2)}{N} + \frac{hits(V_2,p,V_1)}{N}}{\frac{2 * hits_{est}(p)}{N} \times \frac{hits(\text{"to } V_1\text{"}) \times C_v}{N} \times \frac{hits(\text{"to } V_2\text{"}) \times C_v}{N}} \tag{3}$$

for symmetric relations.

Here, *hits(S)* denotes the number of documents containing the string *S*, as returned by Google. *N* is the number of words indexed by the search engine ($N \approx 7.2 \times 10^{11}$), C_v is a correction factor to obtain the frequency of the verb V in all tenses from the frequency of the pattern “to V”. Based on several verbs, we have estimated $C_v = 8.5$. Because pattern counts, when instantiated with verbs, could not be estimated directly, we have computed the frequencies of the patterns in a part-of-speech tagged 500M word corpus and used it to estimate the expected number of hits $hits_{est}(p)$ for each pattern. We estimated the *N* with a similar method.

We say that the semantic relation indicated by lexico-syntactic patterns *p* is present between V_1 and V_2 if

$$\sum_p S_p(V_1,V_2) > C_1 \tag{4}$$

As a result of tuning the system, $C_1 = 8.5$.

Additional test for asymmetric relations. For the asymmetric relations, we require not only that $\sum_p S_P(V_1,V_2)$ exceed a certain threshold, but that there be strong asymmetry of the relation:

$$\frac{\sum_p S_p(V_1,V_2)}{\sum_p S_p(V_1,V_2)} = \frac{\sum_p hits(V_1,p,V_2)}{\sum_p hits(V_2,p,V_1)} > C_2 \tag{5}$$

Tuning on 50 verb pairs has yielded $C_2 = 7$.

4.3 Extracting highly associated verb pairs

To exhaustively test the more than 64 million unordered verb pairs for WordNet’s more than 11,000 verbs would be computationally intractable. Instead, we use a set of highly associated verb pairs output by a paraphrasing algorithm called DIRT. Since we are able to test up to 4000 verb pairs per day on a single machine (we issue at most

40 queries per test and each query takes approximately 0.5 seconds), we are able to test several dozen associated verbs for each verb in WordNet in a matter of weeks.

Lin and Pantel (2001) describe an algorithm called DIRT (Discovery of Inference Rules from Text) that automatically learns paraphrase expressions from text. It is a generalization of previous algorithms that use the distributional hypothesis (Harris 1985) for finding similar words. Instead of applying the hypothesis to words, Lin and Pantel applied it to paths in dependency trees. Essentially, if two paths tend to link the same sets of words, they hypothesized that the meanings of the corresponding paths are similar. It is from paths of the form *subject-verb-object* that we extract our set of associated verb pairs. Hence, this paper is concerned only with relations between transitive verbs.

A path, extracted from a parse tree, is an expression that represents a binary relation between two nouns. A set of paraphrases was generated for each pair of associated paths. For example, using a 1.5GB newspaper corpus, here are the 20 most associated paths to "*X solves Y*" generated by DIRT:

```
Y is solved by X, X resolves Y, X finds a solution to Y, X
tries to solve Y, X deals with Y, Y is resolved by X, X ad-
dresses Y, X seeks a solution to Y, X does something about
Y, X solution to Y, Y is resolved in X, Y is solved through
X, X rectifies Y, X copes with Y, X overcomes Y, X eases Y,
X tackles Y, X alleviates Y, X corrects Y, X is a solution
to Y, X makes Y worse, X irons out Y
```

DIRT only outputs pairs of paths that it has syntactic evidence of being in some semantic relation. We used these as our set to extract finer-grained relations.

5 Experimental results

In this section, we empirically evaluate the accuracy of our system.

5.1 Experimental setup

We studied 26,118 pairs of verbs. Applying DIRT to a 1.5GB newspaper corpus[3], we extracted 4000 paths that consisted of single verbs in the relation *subject-verb-object* (i.e. paths of the form "*X verb Y*") whose verbs occurred in at least 150 documents on the Web. For example, from the 20 most associated paths to "*X solves Y*" shown in Section 4.3, the following verb pairs were extracted:

```
solves :: resolves
solves :: addresses
solves :: rectifies
solves :: overcomes
solves :: eases
solves :: tackles
solves :: corrects
```

[3] The 1.5GB corpus consists of San Jose Mercury, Wall Street Journal and AP Newswire articles from the TREC-9 collection.

Table 3. First five randomly selected pairs along with the system tag (in bold) and the judges' responses.

	CORRECT		PREFERRED SEMANTIC RELATION	
PAIRS WITH SYSTEM TAG (IN BOLD)	JUDGE 1	JUDGE 2	JUDGE 1	JUDGE 2
X rape Y **is stronger than** X abuse Y	Yes	Yes	is stronger than	is stronger than
X accomplish Y **is enabled by** X complete Y	Yes	Yes	is accomplished by	is accomplished by
X achieve Y **is enabled by** X boost Y	Yes	Yes	is accomplished by	is accomplished by
X annotate Y **is similar to** X translate Y	No	Yes	has no relation with	is an alternative to
X further Y **is stronger than** X attain Y	No	No	happens before	happens before

Table 4. Accuracy of system-discovered relations.

	ACCURACY		
	Tags Correct	*Preferred Tags Correct*	*Baseline Correct*
Judge 1	56%	52%	26%
Judge 2	57%	44%	33%
Average	56.5%	48%	29.5%

5.2 Accuracy

To evaluate the accuracy of the system, we ran it on 100 randomly selected pairs and classified each according to the semantic relations described in Section 3. We presented the classifications to two human judges. The adjudicators were asked to judge whether or not the system classification was acceptable. Since the semantic relations are not disjoint (e.g. *mop* is both stronger than and similar to *sweep*), multiple relations may be appropriately acceptable for a given verb pair. The judges were also asked to identify their preferred semantic relations (i.e. that relation which seems most plausible). Table 3 shows the first five randomly selected pairs along with the judges' responses.

Table 4 shows the accuracy of the system. The baseline system consists of labeling each pair with the most common semantic relation, *similarity*, which occurs 29 times. The Kappa statistic (Siegel and Castellan 1988) for the task of judging system tags as correct and incorrect is $\kappa = 0.74$ whereas the task of identifying the preferred semantic relation has $\kappa = 0.697$. For the latter task, the two judges agreed on 72 of the 100 semantic relations. 72% gives an idea of an upper bound for humans on this task. Of these 72 relations, the system achieved a higher accuracy of 61.1%.

Table 5 shows the accuracy of the system on each of the relations. The system did particularly well on the *strength* and *similarity* relations. However, the *happens-while* relation was hardly exercised. Only one of the five instances that the system tagged as a *happens-while* relation was judged correct and by only one of the two judges. Also, 35% of the errors the system made on the *no relation* tag were *antonymy* and 23%

Table 5. Accuracy of each semantic relation.

SEMANTIC RELATION	SYSTEM TAGS	*Tags Correct*	*Preferred Tags Correct*
similarity	36	66.7%	58.3%
strength	22	79.6%	65.9
antonymy	4	25.0%	25.0%
enablement	7	57.2%	50.0
happens before	7	28.6%	28.6
happens while	5	10.0%	0%
no relation	19	39.5%	31.6%

were *similarity*. This suggests that other patterns are needed to discover these two relations.

As described in Section 3, WordNet contains verb semantic relations. A significant percentage of our discovered relations are not covered by WordNet's coarser classifications. Of the 50 verb pairs whose system relation was tagged as correct by both judges in our accuracy experiments, only 34% of them existed in a WordNet relation.

5.3 Discussion

The experience of extracting these syntactic relations has clarified certain important challenges.

While relying on a search engine allows us to query a corpus of nearly a trillion words, some issues arise: (i) the number of instances has to be approximated by the number of hits (documents); (ii) the number of hits for the same query may fluctuate over time; and (iii) some needed counts are not directly available. We addressed the latter issue by approximating these counts using a smaller corpus.

We do not detect entailment with lexico-syntactic patterns. In fact, we propose that whether the entailment relation holds between V_1 and V_2 depends on the absence of another verb V_1' in the same relationship with V_2. For example, given the relation *marry happens-before divorce*, we can conclude that *divorce* entails *marry*. But, given the relation *buy happens-before sell*, we cannot conclude entailment since *manufacture* can also happen before *sell*. This also applies to the relations *happens-while* and *enablement*.

Corpus-based methods, including ours, hold the promise of wide coverage but are weak on discriminating senses.

6 Future work

There are several ways to improve the accuracy of the current algorithm and to detect relations between low volume verb pairs. One avenue would be to automatically learn or manually craft more patterns and to extend the pattern vocabulary (when developing the system, we have noticed that different registers and verb types require different patterns). Another possibility would be to use more relaxed patterns when the part of speech confusion is not likely (e.g. "eat" is a common verb which does not have a noun sense, and patterns need not protect against noun senses when testing such verbs).

Our approach can potentially be extended to multiword paths. DIRT actually provides two orders of magnitude more relations than the 26,118 single verb relations (subject-verb-object) we extracted. On the same 1GB corpus described in Section 5.1, DIRT extracted over 200K paths and 6M unique paraphrases. These provide an opportunity to create a much larger corpus of semantic relations, or to construct smaller, in-depth resources for selected subdomains. For example, we could extract that *take a trip to* is similar to *travel to*, and that *board a plane* happens before *deplane*.

Finally, as discussed in Section 5.3, entailment relations can be derived by processing the complete graph of the identified semantic relation.

7 Conclusion

We have demonstrated that certain fine-grained semantic relations between verbs are present on the web, and are extractable with a simple pattern-based approach[4]. In addition to discovering relations identified in WordNet, such as opposition and troponymy, we obtain strong results on enablement and strength relations (for which no wide-coverage resource is available). On a set of 26,118 associated verb pairs, experimental results show an accuracy of 56.5% in assigning similarity, strength, antonymy, enablement, and temporal relations.

Further work may refine extraction methods and further process the mined semantics to derive other relations such as entailment.

We hope to open the way to inferring implied, but not stated assertions and to benefit applications such as question answering, information retrieval, and summarization.

References

1. Baker, C.; Fillmore, C.; and Lowe, J. 1998. The Berkeley FrameNet project. In *Proceedings of COLING-ACL*. Montreal, Canada.
2. Barker, K.; and Szpakowicz, S. 1995. Interactive Semantic Analysis of Clause-Level Relationships. In *Proceedings of PACLING `95*. Brisbane.
3. Barwise, J. and Perry, J. 1985. Semantic innocence and uncompromising situations. In: Martinich, A. P. (ed.) *The Philosophy of Language*. New York: Oxford University Press. pp. 401–413.
4. Barzilay, R.; Elhadad, N.; and McKeown, K. 2002. Inferring strategies for sentence ordering in multidocument summarization. *JAIR*, 17:35–55.

[4] We plan to provide an online resource of verb semantics discovered from this work if accepted.

5. Cruse, D. 1992 Antonymy Revisited: Some Thoughts on the Relationship between Words and Concepts, in A. Lehrer and E.V. Kittay (eds.), Frames, Fields, and Contrasts, Hillsdale, NJ, Lawrence Erlbaum associates, pp. 289-306.
6. Etzioni, O.; Cafarella, M.; Downey, D.; Kok, S.; Popescu, A.; Shaked, T.; Soderland, S.; Weld, D.; and Yates, A. 2004. Web-scale information extraction in KnowItAll. To appear in *WWW-2004*.
7. Fellbaum, C. 1998. Semantic network of English verbs. In Fellbaum, (ed). *WordNet: An Electronic Lexical Database*, MIT Press.
8. Gomez, F. 2001. An Algorithm for Aspects of Semantic Interpretation Using an Enhanced WordNet. In *NAACL-2001*, CMU, Pittsburgh.
9. Harris, Z. 1985. Distributional Structure. In: Katz, J. J. (ed.), *The Philosophy of Linguistics*. New York: Oxford University Press. pp. 26–47.
10. Hearst, M. 1992. Automatic acquisition of hyponyms from large text corpora. In *COLING-92*. pp. 539–545. Nantes, France.
11. Kingsbury, P; Palmer, M.; and Marcus, M. 2002. Adding semantic annotation to the Penn TreeBank. In *Proceedings of HLT-2002*. San Diego, California.
12. Kipper, K.; Dang, H.; and Palmer, M. 2000. Class-based construction of a verb lexicon. In *Proceedings of AAAI-2000*. Austin, TX.
13. Klavans, J. and Kan, M. 1998. Document classification: Role of verb in document analysis. In *Proceedings COLING-ACL '98*. Montreal, Canada.
14. Levin, B. 1993. *English Verb Classes and Alternations: A Preliminary Investigation*. University of Chicago Press, Chicago, IL.
15. Lin, C-Y. 1997. Robust Automated Topic Identification. Ph.D. Thesis. University of Southern California.
16. Lin, D. and Pantel, P. 2001. Discovery of inference rules for question answering. Natural Language Engineering, 7(4):343–360.
17. Lin, D.; Zhao, S.; Qin, L.; and Zhou, M. 2003. Identifying synonyms among distributionally similar words. In *Proceedings of IJCAI-03*. pp.1492–1493. Acapulco, Mexico.
18. Miller, G. 1990. WordNet: An online lexical database. *International Journal of Lexicography*, 3(4).
19. Moldovan, D.; Harabagiu, S.; Girju, R.; Morarescu, P.; Lacatusu, F.; Novischi, A.; Badulescu, A.; and Bolohan, O. 2002. LCC tools for question answering. In *Notebook of the Eleventh Text REtrieval Conference (TREC-2002)*. pp. 144–154.
20. Palmer, M., Wu, Z. 1995. Verb Semantics for English-Chinese Translation Machine Translation, 9(4).
21. Pantel, P. and Ravichandran, D. 2004. Automatically labeling semantic classes. To appear in *Proceedings of HLT/NAACL-2004*. Boston, MA.
22. Ravichandran, D. and Hovy, E., 2002. Learning surface text patterns for a question answering system. In *Proceedings of ACL-02*. Philadelphia, PA.
23. Richardson, S.; Dolan, W.; and Vanderwende, L. 1998. MindNet: acquiring and structuring semantic information from text. In *Proceedings of COLING '98*.
24. Rus, V. 2002. Logic Forms for WordNet Glosses. Ph.D. Thesis. Southern Methodist University.
25. Schank, R. and Abelson, R. 1977. *Scripts, Plans, Goals and Understanding: An Inquiry into Human Knowledge Structures.* Lawrence Erlbaum Associates.
26. Siegel, S. and Castellan Jr., N. 1988. Nonparametric Statistics for the Behavioral Sciences. McGraw-Hill.
27. Turney, P. 2001. Mining the Web for synonyms: PMI-IR versus LSA on TOEFL. In *Proceedings ECML-2001*. Freiburg, Germany.
28. Webber, B.; Gardent, C.; and Bos, J. 2002. Position statement: Inference in question answering. In *Proceedings of LREC-2002*. Las Palmas, Spain.

Sentiment Extraction from Unstructured Text using Tabu Search-Enhanced Markov Blanket

Xue Bai[1 2], Rema Padman[2], and Edoardo Airoldi[3]

[1] Center for Automated Learning and Discovery, School of Computer Science
[2] The John Heinz III School of Public Policy and Management
Carnegie Mellon University, Pittsburgh, PA USA 15213
{xbai,rpadman}@andrew.cmu.edu
[3] Data Privacy Laboratory, School of Computer Science
Carnegie Mellon University, Pittsburgh, PA USA, 15213
{eairoldi}@cs.cmu.edu

Abstract. Extracting sentiments from unstructured text has emerged as an important problem in many disciplines. An accurate method would enable us, for example, to mine on-line opinions from the Internet and learn customers' preferences for economic or marketing research, or for leveraging a strategic advantage. In this paper, we propose a two-stage Bayesian algorithm that is able to capture the dependencies among words, and, at the same time, finds a vocabulary that is efficient for the purpose of extracting sentiments. Experimental results on the Movie Reviews data set show that our algorithm is able to select a parsimonious feature set with substantially fewer predictor variables than in the full data set and leads to better predictions about sentiment orientations than several state-of-the-art machine learning methods. Our findings suggest that sentiments are captured by conditional dependence relations among words, rather than by keywords or high-frequency words.

1 Introduction

Traditionally, researchers have used surveys to collect a limited amount of data in a structured form for their analyses. In recent years, the advent of the Internet, and the widespread use of advanced information technologies in general, have resulted in a surge of information that is freely available on-line in an *unstructured format*. For example, many discussion groups and review sites exist where people post their opinions about a product. The automatic understanding of *sentiments* expressed within the texts of such posts could lead to a number of new applications in the fields of marketing and information retrieval.

Researchers have been investigating the problem of automatic text categorization for the past two decades. Satisfactory solutions have been found for the cases of topic categorization and of authorship attribution; briefly, topics are captured by sets of keywords, whereas authors are identified by their choices about the use of non-contextual, high-frequency words. Pang et al [17] showed that such solutions, or extensions of them, yield cross-validated accuracies and areas under the curve (AUC) in the low 80%s when ported to sentiment extraction. We conjecture that one reason for the failure of such approaches maybe attributed to the fact that the features used in the classification (e.g. the words) are assumed to be pairwise independent. The goal of this paper is to present a machine learning technique for learning predominant sentiments of on-line texts, available in unstructured format, that:

- is able to capture dependencies among words, and
- is able to find a minimal vocabulary, sufficient for categorization purposes.

Our two-stage Markov Blanket Classifier (MBC) learns conditional dependencies among the words and encodes them into a *Markov Blanket Directed Acyclic Graph* (MB DAG) for the sentiment variable (first stage), and then uses a *Tabu Search* (TS) meta-heuristic strategy to fine tune the MB DAG (second stage) in order to yield a higher cross-validated accuracy. Learning dependencies allows us to capture semantic relations and dependent patterns among the words, thus approximating the meaning of sentences, with important applications for many real world applications. Further, performing the classification task using a Markov Blanket (MB) for the sentiment variable (in a Bayesian network) has important properties: (a) it specifies a statistically efficient prediction of the probability distribution of the sentiment variable from the smallest subset of predictors, and (b) it provides accuracy while avoiding over-fitting due to redundant predictors. We test our algorithm on the publicly available Movie Reviews data set and achieve a cross-validated accuracy of 87.5% and a cross-validated AUC of 96.85% respectively, against best performances of competing state-of-the-art classifiers in the low 80%s. This paper is organized as follows: Section 2 surveys related work. Section 3 provides some background about Bayesian networks, Markov Blankets, and Tabu Search. Section 4 contains details about our proposed methodology. Section 5 describes the data and presents the experimental results. Last, Section 6 discusses of our findings and concludes.

2 Related Work on Sentiments

The problem of sentiment extraction is also referred to as opinion extraction or semantic classification in the literature. A related problem is that of studying the semantic orientation, or polarity, of words as defined by Osgood et al. [16]. Hatzivassiloglou and McKeown [10] built a log-linear model to predict the semantic orientation of conjoined adjectives using the conjunctions between them. Huettner and Subasic [11] hand-crafted a cognitive linguistic model for *affection* sentiments based on fuzzy logic. Das and Chen [6] used domain knowledge to manually construct lexicon and grammar rules that aim to capture the "pulse" of financial markets as expressed by on-line news about traded stocks. They categorized news as *buy*, *sell* or *neutral* using five classifiers and various voting schemes to achieve an accuracy of 62% (random guesses would top 33%). Turney and Littman [23] proposed a compelling semi-supervised method to learn the polarity of adjectives starting from a small set of adjectives of known polarity, and Turney [22] used this method to predict the opinions of consumers about various objects (movies, cars, banks) and achieved accuracies between 66% and 84%. Pang et al. [17] used off-the-shelf classification methods on frequent, non-contextual words in combination with various heuristics and annotators, and achieved a maximum cross-validated accuracy of 82.9% on data from IMDB. Dave et al. [7] categorized positive versus negative movie reviews using support vector machines on various types of semantic features based on substitutions and proximity, and achieved an accuracy of at most 88.9% on data from Amazon and Cnn.Net. Last, Liu et al. [14] proposed a framework to categorize emotions based on a large dictionary of common sense knowledge and on linguistic models.

3 Theoretical Background

3.1 Bayesian Networks and Markov Blanket

A Bayesian network is a graphical representation of the joint probability distribution of a set of random variables as nodes in a graph, connected by directed edges. The orientations of the edges encapsulate the notion of parents, ancestors, children, and descendants of any node [18, 20].

More formally, a *Bayesian network* for a set of variables $X = \{X_1, ..., X_n\}$ consists of: (i) a network structure S that encodes a set of conditional independence assertions among variables in X; and (ii) a set $P = \{p_1, ..., p_n\}$ of local conditional probability distributions associated with each node and its parents. Specifically, S is a directed acyclic graph (DAG) which, along with P, entails a joint probability distribution p over the nodes.

We say that P satisfies the *Markov condition* for S if every node X_i in S is independent of its non-descendants, conditional on its parents. The Markov Condition implies that the joint distribution p can be factorized as a product of conditional probabilities, by specifying the distribution of each node conditional on its parents. In particular, for given a structure S, the joint probability distribution for X can be written as

$$p(X) = \prod_{i=1}^{n} p_i(X_i|pa_i) \ , \tag{1}$$

where pa_i denotes the set of parents of X_i.

Given the set of variables X and target variable Y, a *Markov Blanket* (MB) for Y is the smallest subset Q of variables in X such that Y is independent of $X \backslash Q$, conditional on the variables in Q. Intuitively, given a Bayesian network (S, P), the Markov Blanket for Y consists of pa_Y, the set of parents of Y; ch_Y, the set of children of Y; and $pa\,ch_Y$, the set of parents of children of Y.

Example 1. Consider the two DAGs in Figure 1 and Figure 2, below. The factorization of p entailed by the Bayesian network (S, P) is

$$\begin{aligned} p(Y, X_1, ..., X_6) = C \cdot p(Y|X_1) \cdot p(X_4|X_2, Y) \cdot p(X_5|X_3, X_4, Y) \cdot \\ \cdot p(X_2|X_1) \cdot p(X_3|X_1) \cdot p(X_6|X_4) \ , \end{aligned} \tag{2}$$

where C is a normalizing constant.

The factorization of the conditional probability $p(Y|X_1, ..., X_6)$ entailed by the Markov blanket for Y corresponds to the product of those local factors in (2) which contain the term Y, that is

$$p(Y|X_1, ..., X_6) = C' \cdot p(Y|X_1) \cdot p(X_4|X_2, Y) \cdot p(X_5|X_3, X_4, Y) \quad (3)$$

where C' is a different normalizing constant.

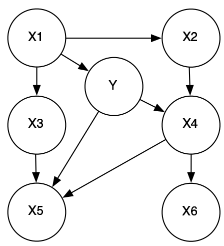


Fig. 1. Bayesian network (S, P).

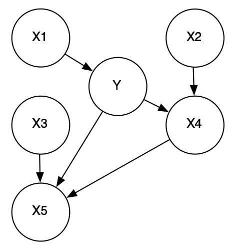


Fig. 2. Markov Blanket for Y in (S, P).

Different MB DAGs that entail the same factorization for $p(Y|X_1, ..., X_6)$ belong to the same *Markov equivalence class*. Our algorithm searches the space of Markov equivalent classes, rather than that of DAGs, thus boosting its efficiency. Markov Blanket classifiers have been recently rediscovered and applied to several domains, but very few studies focus on how to learn the structure of the Markov Blanket from data. Further, the applications in the literature have been limited to data sets with few variables. Theoretically sound algorithms for finding DAGs are known (e.g. see [4]), but none has been tailored to the problem of finding MB DAGs.

3.2 Tabu Search

Tabu Search (TS) is a powerful meta-heuristic strategy that helps local search heuristics explore the space of solutions by guiding them out of local optima [9]. It has been applied successfully to a wide variety of continuous and combinatorial optimization problems, and has been shown to be capable of reducing the complexity of the search process and accelerating the rate of convergence.

The basic Tabu Search starts with a feasible solution and iteratively chooses the *best move*, according to a specified evaluation function, while assuring that solutions previously generated are not revisited in the short-term. This is accomplished by keeping a *tabu list* of restrictions on possible moves, updated at each step, which discourage the repetition of selected moves. Typically tabu restrictions are based on a short-term memory function, called the *tabu tenure*, to prevent loops in the search, but intermediate and long-term memory functions may also be adopted to intensify and diversify the search.

4 Proposed Methodology: Two-Stage MB Classifier

4.1 1^{st} Stage: Learning Dependencies with an Initial MB DAG

The first stage generates an initial MB for Y from the data. This procedure involves the following: It begins by selecting those variables in $\{X_1, ..., X_N\}$ that are associated with Y within two hops in the graphical representation; that is, it finds potential parents and children (L_Y) of Y, and potential parents and children ($\cup_i L_{X_i}$) of nodes $X_i \in L_Y$, using conditional independence tests, representing adjacencies by undirected edges. At this point, the list $Y \cup L_Y \cup_i L_{X_i}$ is a skeleton (an undirected graph) which contains the MB for Y (See above the precise definition of MB(Y) in terms of pa_Y, ch_Y, and $pa\, ch_Y$.) The algorithm then **orients** the edges using six edge orientation rules described in Bai et al. [1]. Finally , it **prunes** the remaining undirected edges and bi-directed edges to avoid cycles, puts them in a list L for Tabu Search, and returns the MB DAG.

The core of the first stage lies in the search for the nodes (L_Y) *associated* with Y, and for those ($\cup_i L_{X_i}$) *associated* with the nodes in L_Y, based on causal discovery theory. [18, 20] This search is non trivial and is performed by two recursive calls to the function **findAdjacencies**(Y), as shown in figure 3: independence tests between Y and each X_i are performed to identify a list (A_Y) of variables associated to Y; then, for $X_i \in A_Y$ and for all distinct subsets $S \subset \{A_Y \backslash X_i\}^d$, where d controls the size of S, conditional independence tests between Y and X_i

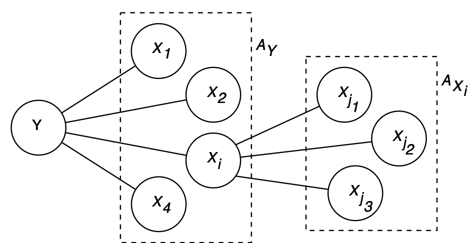


Fig. 3. Illustration of **findAdjacencies** (Y). A_Y and A_{X_i} are shown.

given S are performed to remove unfaithful associations; For more details about *unfaithful* associations and distribution see Spirtes et al. [20]. Then, for all pairs $(X_i, X_j)_{i \neq j}$, independence tests are performed to identify lists of variables (A_{X_i}, i=1,...,N) associated to each X_i; last, for $X_i \in A_Y$ and for all distinct subsets $S \subset \{A_{X_i}\}^d$, conditional independence tests between Y and each X_i given S are again performed to prune unfaithful associations.

4.2 2^{nd} Stage: Tabu Search to Improve the MB Classifier

Tabu Search (TS) is then applied to improve the initial MB DAG. Our algorithm searches for solutions in the space of logical Markov equivalence classes, instead of searching the space of MB DAGs; that is, moves that yield Markov Blankets within the same Markov equivalent class are not considered, and moves that result in cyclic graphs are not valid moves.

Briefly, four kinds of moves are allowed in the TS procedure: edge addition, edge deletion, edge reversal and edge reversal with node pruning. At each stage, and for each allowed move, the corresponding MB DAG is computed, its conditional probability factored, its predictions scored, and the best move is then selected and applied. Best solution and best score at each step are tracked. The tabu list keeps a record of m previous moves, so that moves in the tabu list will not be repeated till their corresponding tabu tenure expires. Details can be found in [2].

4.3 A Sketch of the Algorithm

We present a sketch of the algorithm below. The parameters are: D, a data set with N variables and K examples; Y, the class variable; d, the maximum number of nodes for the conditional independence tests; α, the significance level for the G^2 statistical independence tests (for a definition of G^2 see [20]). The final output is the graphical Markov Blanket structure (MB) for Y.

InitialMBsearch (Data D, Target Y, Depth d, Significance α)
1. L_Y = **findAdjacencies** (Y, $\{X_1, ..., X_N\}$, d, α)
2. **for** $X_i \in L_Y$
   2.1. L_{X_i} = **findAdjacencies** (X_i, $\{X_1, ..., X_N\} \backslash X_i$, d, α)
3. G = **orient** ($Y \cup L_Y \cup_i L_{X_i}$)
4. {MB DAG, L} = **prune** (G)
5. **return** {MB DAG, L}

TabuSearch (Data D, Target Y)
1. **init** ($bestSolution = currentSolution$ = MB DAG, $bestScore = 0$, ...)
2. **repeat until** ($bestScore$ does not improve for k consecutive iterations)
   2.1. form $candidateMoves$ for $currentSolution$
   2.2. **find** $bestMove$ among $candidateMoves$ according to function **score**
   2.3. **if** ($bestScore <$ **score** ($bestMove$))
      2.3.1. **update** $bestSolution$ and $bestScore$ by applying $bestMove$
      2.3.2. **add** $bestMove$ to $tabuList$ // not re-considered in the next t iterations
   2.4. **update** $currentSolution$ by applying $bestMove$
3. **return** $bestSolution$ // an MB DAG

findAdjacencies (Node Y, List of Nodes L, Depth d, Significance α)
1. $A_Y := \{X_i \in L$: X_i is dependent of Y at level $\alpha\}$
2. **for** $X_i \in A_Y$ and **for** all distinct subsets $S \subset \{A_Y \backslash X_i\}^d$
   2.1. **if** X_i is independent of Y given S at level α
   2.2. **then** remove X_i from A_Y
3. **for** $X_i \in A_Y$
   3.1. $A_{X_i} := \{X_j \in L$: X_j is dependent of X_i at level α, $j \neq i\}$
   3.2. **for** all distinct subsets $S \subset \{A_{X_i}\}^d$
      3.2.1. **if** X_i is independent of Y given S at level α
      3.2.2. **then** remove X_i from A_Y
4. **return** A_Y

5 Experiments

5.1 Movie Reviews Data

We tested our method on the data set used in Pang et al [17]. This data set contains approximately 29,000 posts to the rec.arts.movies.reviews newsgroup archived at the Internet Movie Database (IMDb). The original posts are available in the form of HTML pages. Some pre-processing was performed to produce the version of the data we used. Specifically, only reviews where authors' ratings were expressed explicitly (either by stars or by numerical values) were selected. Then explicit ratings were removed and converted into one of three categories: positive, negative, or neutral. Finally, 700 positive reviews and 700 negative reviews, which the authors of the corpus judged to be more extreme, were selected for our study. Various versions of the data are available on-line [24].

5.2 Feature Definition

In our study, we used words as features, where *words* are strings of letters enclosed by non-letters to the left and to the right. Note that our definition excludes punctuation sign even though exclamation signs and question marks may be helpful for our task. Intuitively the task of sentiment extraction is a hybrid task between authorship attribution and topic categorization; we look for frequent words, possibly not related to the context, that help express lexical patterns, as well as low frequency words which may be specific to few review styles, but very indicative of an opinion. We considered all the words that appeared in more than 8 documents as our input features, whereas words with lower counts were discarded since they appear too rarely to be helpful in the classification of many reviews. We were left with a total number of 7,716 words, as input features. In our experiments, we represented each document as a vector, $X := [X_1, ..., X_{7716}]$, of the size of the initial vocabulary, where each X_i is a binary random variable that takes the value of 1 if the i^{th} word in the vocabulary is present in the document and the value of 0 otherwise.

5.3 Experimental Set-Up

In order to compute unbiased estimates for AUC and accuracy we used a nested, stratified, five-fold cross-validation scheme. The parameters in our experiments were the scoring criteria, the maximum size of the condition set to consider for conditional independence tests when learning the MB DAG (i.e. the depth d), and the α level to decide whether to accept or reject each of these tests. We explored 24 configurations of parameter combinations, shown in Table 1. We found the *dominant configuration* of the parameters on the training data and estimated the performance on the testing data, according to the (outer) five-fold cross-validation scheme. In order to find this configuration, within each fold i, we further split the training data in two (TR_{i1} and TR_{i2}), trained the MB classifier on TR_{i1} for each parameter configuration, and tested the performance on TR_{i2}.

Table 1. Experimental Parameter Configurations.

Parameters	Scoring Criteria	Depth of Search	Alpha	C.V. Folds
Configurations	AUC Accuracy	1, 2, 3	0.001, 0.005, 0.01, 0.05	5-fold

The configuration that led to the best MB, in terms of accuracy on TR_{i2} across all five folds $i = 1, ..., 5$, was chosen as the best configuration.

5.4 Results and Analysis

We compared the performances of our two-stage MB classifier with those of four widely used classifiers: a naïve Bayes classifier based on the multivariate Bernoulli distribution with Laplace prior for unseen words, discussed in Nigam et al. [15], a support vector machine (SVM) classifier along with a TF-IDF re-weighting of the vectors of word counts, discussed by Joachims [12], an implementation of the voted Perceptron, discussed in Freund and Schapire [8], and a maximum entropy conditional random field learner, introduced by Lafferty et al. [13].

Table 2 compares the two-stage MBC with the performances of the other classifiers using the *whole feature set* as input. As we expected, more features did not necessarily lead to better results, as the classifiers were not able to distinguish discriminating words from noise. In such a situation we also expected the SVM with TFIDF re-weighting and the voted perceptron to perform better than the other classifiers. As shown in table 2, the two-Stage MB classifier selects 22 relevant words out of 7,716 words in the vocabulary. The feature reduction ratio is 99.71%; the cross-validated AUC based on the 22 words and their dependencies is 96.85%, which is 14.3% higher than the best of the other four methods; the corresponding cross-validated accuracy is 87.5%, which is 3.5% higher than the best of the other four methods. We notice that the two-Stage MB classifier is

Table 2. Average performances on the whole feature set.

Method Method	AUC (%)	Accuracy (%)	# Selected Features	Size Reduction
Two-stage MB	96.85	87.52	22	99.71%
Naïve Bayes	82.61	66.22	7716	0%
SVM + TFIDF	81.32	84.07	7716	0%
Voted perceptron	77.09	70.00	7716	0%
Max. entropy	75.79	79.43	7716	0%

able to automatically identify a very discriminating subset of features (or words) that are relevant to the target variable (Y, the label of the review). Specifically, the selected features are those that form the Markov Blanket for Y. Further, the two-Stage MB classifier yields the best results in terms of both cross-validated AUC and accuracy. Other methods perform worse on the whole feature set and need to be paired with a variable selection strategy.

Table 3 compares the performance of the two-stage MBC with others classifiers using the *same number of features* selected by information gain. We notice that feature selection using information gain criterion does not tell us how many features have to be selected, but rather allows us to rank the features from most to least discriminating instead. Again, the two-Stage MB classifier dominates the other methods both in terms of AUC and accuracy, though it is not clear whether the extra performance comes form the different feature selection strategies, or from the dependencies encoded by the MB.

Table 3. Average performances on the same number of features.

Method Method	AUC (%)	Accuracy (%)	# Selected Features	Size Reduction
Two-stage MB	96.85	87.52	22	99.71%
Naïve Bayes	78.85	72.07	22	99.71%
SVM + TFIDF	67.30	70.43	22	99.71%
Voted perceptron	78.68	71.71	22	99.71%
Max. entropy	68.42	71.93	22	99.71%

To investigate this point, in Table 4 we compare the performance of the two-stage MBC with others classifiers using the *same exact features*. We find that a small part of the difference between the accuracy of the MBC and that of other classifiers in Table 3 arises from the fact that we selected features using information gain; in fact all the four competing classifiers performed better on the set of features in the Markov blanket. We also find that the major portion of such differences is due to the MB classification method itself. We attribute

Table 4. Average performances on the same exact features.

Method Method	AUC (%)	Accuracy (%)	# Selected Features	Size Reduction
Two-stage MB	96.85	87.52	22	99.71%
Naïve Bayes	81.81	73.36	22	99.71%
SVM + TFIDF	69.47	72.00	22	99.71%
Voted perceptron	80.61	73.93	22	99.71%
Max. entropy	69.81	73.44	22	99.71%

the jump in the accuracy and AUC to the fact that the MB classifier encodes and takes advantage of conditional dependencies among words, which all other methods fail to capture.

Finally, in Figure 4 below we show the best MB DAG learned by the two-Stage MB classifier. All the directed edges are robust over at least 4 out of five cross validation runs; the variation is very small. The structure of the final MB DAG does not indicate independence of the words conditional on the sentiment variable, which is the strong assumption underlying all the competing classifiers.

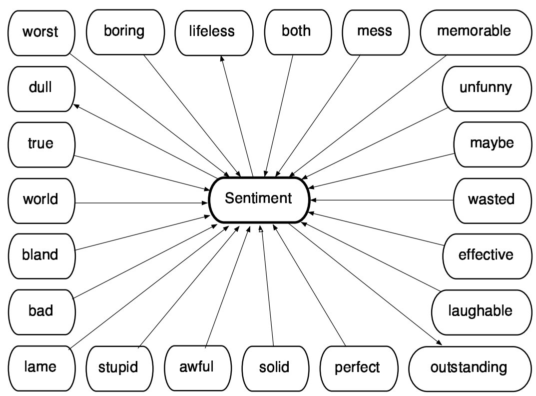


Fig. 4. Best Fitting MB DAG for the Movie Dataset.

These experiments, as well as more results we have obtained on other medical data sets [1], suggest that for problems where the independence assumption is not appropriate, the two-stage MB classifier is a better choice and leads to more robust predictions by: (i) selecting statistically discriminating features for the class variable, and (ii) learning a more realistic model that allows for dependencies among the predictors. Further, according to the empirical findings in Pang et al [17], the baseline accuracy for human-selected vocabularies can be set at about 70%. Comparing the human intuition to our fully automated machine learning technique (two-stage MBC), we observe a non-negligible improvement.

6 Discussion and Conclusions

The two-stage Markov Blanket classifier that we have proposed in this paper

- is able to capture dependencies among words, and
- is a fully automated system able to select a parsimonious vocabulary, customized for the classification task in terms of size and relevant features.

Overall, the two-Stage MB classifier significantly outperforms the four baseline methods and is able to extract the most discriminating features for classification purposes. The main drawbacks of the competing methods are that they cannot automatically select relevant features, and they cannot encode the dependencies among them. While the first issue is easily overcome by combining the classifiers with off-the-shelf feature selection methods, the second issue cannot be addressed. In fact, it is a direct consequence of the assumption of pairwise independence of features underlying all the competing methods. Further, many techniques have been tried in order to automatically capture the way people express their opinions, including models for the contextual effects of negations, the use of feature frequency counts instead of their presence or absence, the use of different probability distributions for different positions of the words in the text, the use of sequences of words or N-grams, the combination of words and part of speech tags, noun-phrase chunks, and so on. However, the empirical results in terms of prediction accuracy and AUC always remain in the same ballpark.

We performed three sets of experiments to compare the methods along various dimensions, in Tables 2, 3, 4. In particular, Table 4 shows that given the *same exact features*, which were identified by the MBC as belonging to the Markov blanket, the MBC leads to significantly higher AUC and accuracy, thus suggesting that taking into account dependencies among words is crucial to perform sentiment extraction. The comparison of results of Table 3 and Table 4 suggests that information gain is not the best criterion to select discriminating variables, but the statistical tests that measure association among features and causal reasoning are better tools to perform the selection. The findings of Bai et al. [1], who obtained similar results on four more data sets form different domains, add strength to our claims. We acknowledge that these are experimental results, and other selection strategies and data sets may tell different stories.

In conclusion, we believe that in order to capture sentiments we have to go beyond the search for richer feature sets and the independence assumption. Rather we need to capture those elements of the text that help identify context and meaning. We believe that a robust model, which would naturally lead to higher performance, is obtained by encoding dependencies among words, and by actively searching for a better dependency structure using heuristic and optimal strategies. Finally, the analysis of the relations among words underlying accurate MBC DAGs may lead to a better understanding of the way contextual meaning arises from the occurrence of words.

References

1. Bai, X., Glymour, C., Padman, R., Spirtis, P., Ramsey, J.: MB Fan Search Classifier for Large Data Sets with Few Cases. Technical Report CMU-CALD-04-102. School of Computer Science, Carnegie Mellon University (2004)
2. Bai, X., Padman, R.: Tabu Search Enhanced Markov Blanket Classifier for High Dimensional Data Sets. Proceedings of INFORMS Computing Society. (to appear) Kluwer Academic Publisher (2005)
3. Chickering, D.M.: Learning Equivalence Classes of Bayesian-Network Structures. Journal of Machine Learning Research **2** MIT Press (2002) 445–498
4. Chickering, D.M.: Optimal Structure Identification with Greedy Search. Journal of Machine Learning Research **3** MIT Press (2002) 507–554

5. Cohen, W.W.: Minor-Third: Methods for Identi-fying Names and Ontological Relations in Text using Heuristics for Inducing Regularities from Data. http://minorthird.sourceforge.net (2004)
6. Das, S., Chen, M.: Sentiment Parsing from Small Talk on the Web. In: Proceedings of the Eighth Asia Pacific Finance Association Annual Conference (2001)
7. Dave, K., Lawrence, S., Pennock, D.M.: Mining the Peanut Gallery: Opinion Extraction and Semantic Classification of Product Reviews. In: Proceedings of the Twelfth International Conference on World Wide Web (2003) 519–528
8. Freund, Y., Schapire, R.E.: Large Margin Classification Using the Perceptron Algorithm. Machine Learning **37** (1999) 277–296
9. Glover, F.: Tabu Search. Kluwer Academic Publishers (1997)
10. Hatzivassiloglou, V., McKeown, K.: Predicting the Semantic Orientation of Adjectives. In: Proceedings of the Eighth Conference on European Chapter of the Association for Computational Linguistics ACL, Madrid, Spain (1997) 174–181.
11. Huettner, A., Subasic, P.: Fuzzy Typing for Document Management. In: Association for Computational Linguistics 2000 Companion Volume: Tutorial Abstracts and Demonstration Notes (2000) 26-27
12. Joachims T.: A Statistical Learning Model of Text Classification with Support Vector Machines. In: Proceedings of the Conference on Research and Development in Information Retrieval ACM (2001) 128–136
13. Lafferty, J., McCallum, A., Pereira, F.: Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In: Proceedings of the Eighteenth International Conference on Machine Learning. Morgan Kaufmann, San Francisco, CA (2001) 282–289
14. Liu, H., Lieberman, H., Selker, T.: A Model of Textual Affect Sensing using Real-World Knowledge. In: Proceedings of the Eighth International Conference on Intelligent User Interfaces (2003) 125–132
15. Nigam, K., McCallum, A., Thrun, S., Mitchell, T.: Text Classification from Labeled and Unlabeled Documents using EM. Machine Learning **39** (2000) 103–134
16. Osgood, C.E., Suci, G.J., Tannenbaum, P.H.: The Measurement of Meaning. University of Illinois Press, Chicago, Illinois (1957)
17. Pang, B., Lee, L., Vaithyanathan, S.: Thumbs up? Sentiment Classification using Machine Learning Techniques. In: Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (2002) 79–86
18. Pearl, J.: Causality: Models, Reasoning, and Inference. Cambridge University Press (2000)
19. Provost, F., Fawcett, T., Kohavi, R.: The Case Against Accuracy Estimation for Comparing Induction Algorithms. In: Proceedings of the Fifteenth International Conference on Machine Learning (1998) 445–453
20. Spirtes, P., Glymour, C., Scheines, R.: Causation, Prediction, and Search. 2nd edn. MIT Press (2000)
21. Spirtes, P., Meek, C.: Learning Bayesian Networks with Discrete Variables from Data. In: Proceedings of the First International Conference on Knowledge Discovery and Data Mining AAAI Press (1995) 294–299
22. Turney, P.: Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews. In: Proceedings Fortieth Annual Meeting of the Association for Computational Linguistics (2002) 417–424
23. Turney, P., Littman, M.: Unsupervised Learning of Semantic Orientation from a Hundred-Billion-Word Corpus. Technical Report EGB-1094. National Research Council, Canada (2002)
24. Movie Review Data: http://www.cs.cornell.edu/people/pabo/movie-review-data/

Mining Structures to Predict Semantics (Invited Talk)

Alon Y. Halevy

Abstract: At a fundamental level, the key challenge in building the Semantic Web is to reconcile the semantics of disparate information sources, each expressed with different structures (e.g., varying ontologies or XML and relational schemas). In this talk, I will argue for a general approach to the reconciliation problem that is based on mining large corpora of schemas and semantic mappings. The intuition behind the approach is that statistics collected over such corpora offer hints about the semantics of the symbols in the structures. Hence, we are able to detect when two symbols, from disparate schemas, should be matched to each other. The same methodology can be applied to several other data management tasks that involve search in a space of complex structures (e.g., searching for web services). I will illustrate several examples where this approach has been successful, and highlight the challenges involved in pursuing it.

Exploiting Recurring Structure in a Semantic Network

Shawn R. Wolfe, Richard M. Keller

Computational Sciences Division, MS 269-2,
NASA Ames Research Center, Moffett Field, CA USA 94035
{Shawn.R.Wolfe, Richard.M.Keller}@nasa.gov

Abstract. With the growing popularity of the Semantic Web, an increasing amount of information is becoming available in machine interpretable, semantically structured networks. Within these semantic networks are recurring structures that could be mined by existing or novel knowledge discovery methods. The mining of these semantic structures represents an interesting area that focuses on mining both *for* and *from* the Semantic Web, with surprising applicability to problems confronting the developers of Semantic Web applications. In this paper, we present representative examples of recurring structures and show how these structures could be used to increase the utility of a semantic repository deployed at NASA.

1 Introduction

The Semantic Web effort, with its emphasis on machine interpretable information, is creating exciting new research possibilities in knowledge discovery. Primarily, this research has focused on adapting known techniques to the Semantic Web, either by mining conventional information sources to augment a semantic network, or by extracting information from a semantic network that is then mined for conventional purposes. The overlap of these areas is an entirely new arena for knowledge discovery: mining the semantic network to enhance the semantic network itself or aid in its use. We are interested in a particular mining problem where both the input and output is a semantic network, namely finding recurring, similar semantic structures in a larger semantic network. A number of Semantic Web applications store information in large networks, notably ODESeW [1], OntoWebber [2], SEAL [3], OntoWeb [4], the KAON suite [5], BrainEKP [6], Semagix Freedom [7] and our own SemanticOrganizer [8]. Our focus in this paper is not an algorithm for discovering recurring semantic structure, but rather how such structures can be used once identified, namely for:

- **Enforcing consistency with rules**. Identifying all the rules needed to enforce logical consistency in a semantic network is a non-trivial task. Patterns of recurring structures can be used to generate candidate rules.
- **Aiding in analysis**. The semantic organization of information makes information easier to find, but the analysis process is still manual. Identifying recurring structures in the semantic network would automate part of the analysis process.

- **Performing ontology maintenance**. Ontology maintenance for persistent and evolving Semantic Web applications is time consuming and difficult, resulting in less than optimal modeling decisions that impact the usability of the system. The identification of recurring structures can indicate patterns that suggest useful changes to the ontology.
- **Reducing network complexity**. As the size of a semantic network grows, it becomes increasingly difficult to navigate or display the information space. Abstracting recurring structures that match the same pattern can reduce the size and complexity of the network representation, making it more manageable with existing navigation and visualization techniques.

2 Exploiting Recurring Semantic Structure

We use semantic templates to define recurring semantic structure. A semantic template consists of a set of abstracted RDF-like triples, and the matches to this template are the recurring semantic structure in the network. Figure 1 gives an example of an abstract graph pattern in an RDQL format and a matching set of statements. In the following subsections, we describe various ways in which recurring semantic structure can be exploited to improve the utility of systems that use semantic networks.

```
(?x researcher-in ?y)
(?x authored ?z)
(?z submitted-to ?w)
(?w has-topic-area ?y)
```

matches

```
("Shawn Wolfe" researcher-in "Semantic Web")
("Shawn Wolfe" authored "Exploiting Recurring Structure…")
("Exploiting Recurring Structure…" submitted-to "SW Mining Workshop")
("SW mining Workshop" has-topic-area "Semantic Web")
```

Figure 1. Example of a semantic template and a corresponding match in the semantic network

2.1 Enforcing consistency with rules

Users of a semantically structured repository cannot be expected to create every relevant link between nodes. However, failure to create all such links leads to a less complete, less accurate and subsequently less useful network. We feel that it is necessary to augment the semantic network by providing additional links through inference. Some of the supporting inference rules can be derived from the structure of the ontology (e.g., deriving a property from a sub-property), whereas other rules are based on domain knowledge. Figure 2 gives an example of a rule based on domain knowledge, stating that samples gathered during an experiment must be collected at the site of that experiment. It is these domain-specific rules that we seek to discover through the identification of recurring semantic structure.

We regard inference rules as composed of semantic templates, with the antecedent and consequent sections each consisting of a semantic template. Assuming a relatively complete and representative semantic network, it should be possible identify possible domain-specific inference rules by finding a significant number of matches to candidate antecedents and consequents. Even a fairly unsophisticated technique that generates a large number of undesirable candidate rules would be helpful, since identifying correct rules from a large set of candidates is easier than deriving them through manual domain analysis.

```
(?sample gathered-during ?experiment)
(?experiment conducted-at ?site)
->
(?sample collected-from ?site)
```

Figure 2. Example of an inference rule from a biology domain

2.2 Aiding in analysis

Recurring structures can also reveal interesting features in the semantic network. For example, consider a semantic network modeling a biological experiment measuring the effects of salinity and pH level on stored cultures. An algorithm that generates candidate inference rules by identifying recurring structure could generate the rules in Figures 2-3. However, the rule in Figure 3 would reveal a result of the experiment, thus aiding the biologist in analyzing the results. The difference between these two candidate rules is that the rule in Figure 2 would be used to enforce semantic consistency, whereas the rule in Figure 3 reveals something interesting about the domain.

```
(?culture salinity "high")
(?culture pH-level "9.0")
->
(?culture exhibits "speckling")
```

Figure 3. Example of an unexpected rule that reveals a previously unknown correlation

Statistical analysis on the recurring semantic structure can also reveal interesting features in a semantic network. Consider a semantic network for an investigation domain that has information on 1000 total mishaps. Figures 4-7 show three semantic templates for this domain and the number of matches for each. Since one out of ten mishaps involves a jackscrew in this example, we would have expected only four or so MD-80 mishaps to involve jackscrews. Since this number is significantly higher, an investigator may deduce that there is an issue with reliability of jackscrews in MD-80 airplanes.

```
(?mishap involves ?plane)
(?plane model "MD-80")
```

Figure 4. A semantic template that has 40 matches.

```
(?mishap involves ?plane)
(?mishap concerns "jackscrew-failure")
```

Figure 5. A semantic template that has 100 matches.

```
(?mishap involves ?plane)
(?plane model "MD-80")
(?mishap concerns "jackscrew-failure")
```

Figure 6. A semantic template that has 16 matches, indicating a correlation between jackscrew failures and MD-80 mishaps

2.3 Performing ontology maintenance

The identification of recurring structure can also be benefit ontology development. In our experience, ontologies require significant maintenance as application requirements change over time. The identification of recurring semantic structure can suggest approaches to revising an existing ontology based on this evolving pattern of usage. One form of ontology change supported by semantic template identification is specialization, where a single concept in an ontology is elaborated by adding several more specific subconcepts beneath the original, thus providing for more accurate and therefore more meaningful modeling.

Consider the patterns described in Figures 7-9 from a project management ontology. Three different subconcepts of document are suggested by the documents that would match these templates: a submitted publication concept, an experimental procedure concept, and software documentation concept. Additional analysis of the recurring structure could reveal that no document matches more than one of these patterns: after all, software documentation is not submitted to conferences, experiment procedures do not describe software, and so on. Such realizations may suggest to the ontology maintainer that the document concept should be split into several subconcepts: publications, experimental procedures and software documentation. This specialization would lead to a more constrained domain model that prevents some illogical pairings (such as a given document describing software and following an experimental protocol), and indeed manual analysis lead us to a similar specialization in our ontology.

```
(?document submitted-to ?conference)
(?document acceptance-status ?status)
```

Figure 7. A publication document template

```
(?document tests ?hypothesis)
(?document follows ?experimental-protocol)
```

Figure 8. An experiment procedure template

```
(?document describes ?software-module)
(?document has-version ?software-version)
```

Figure 9. A software documentation template

2.4 Reducing network complexity

Finally, repeating patterns can serve as an aid to visualization and navigation. We have found that our semantic networks have quickly grown to the point where people have trouble navigating them [9]. A display of the immediate neighborhood of a semantic node is often insufficient context for users, but displaying the entire network is infeasible due to the large number of nodes and edges. One approach to solve this problem is to combine similar nodes into a composite node, thus reducing the complexity of the space and making it possible to visualize with conventional techniques.

Figure 10 presents a semantic template from a biological domain. In this domain, scientists perform experiments collecting measurements on samples. Any set of measurements that match the template with the same values for `?experiment`, `?date`, and `?sample` would be indistinguishable with respect to this template, thereby forming an equivalence class. We envision developing a technique, either by explicitly choosing important and unimportant differences or through some implicit analysis, which would allow us to collapse such similar nodes in appropriate situations, as illustrated in Figure 11.

```
(?experiment produces ?measurement)
(?measurement collected-on ?date)
(?measurement measures ?sample)
```

Figure 10. A template defining an equivalence class

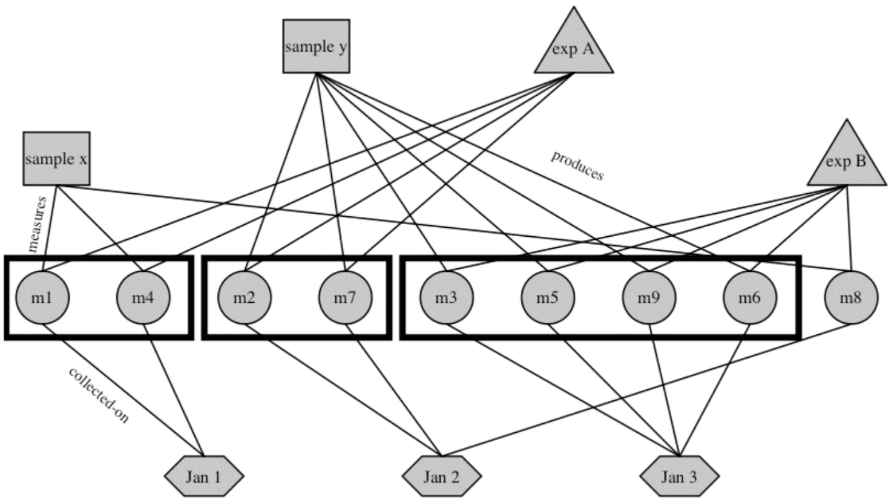


Figure 11. A semantic network with boxes around nodes that could be combined into composite nodes according to the semantic template given in Figure 10

3 Conclusion

We have presented a simple definition of recurring semantic structure and discussed several ways in which it could used to improve a repository that stores information in a semantic network. Our analysis has led us to advocate mining for recurring semantic structure as a fruitful area of research: the problem lies in an area relatively unexplored and the simple definition of semantic structure should be amenable to straightforward knowledge discovery methods. Furthermore, even unsophisticated techniques could be beneficial, as relatively inaccurate and imprecise results still offer some automated assistance where there currently is none.

4 Acknowledgements

We would like to thank the ScienceDesk team and Deepak Kulkarni for their contributions to this paper. Our work on SemanticOrganizer is funded by the NASA Intelligent Systems Project of the Computing, Information, and Communications Technology Program.

5 References

1. O. Corcho, A. Gomez-Perez, A. Lopez-Cima, V. Lopez-Garcia, and M. Suarez-Figueroa, "ODESeW. Automatic generation of knowledge portals for Intranets and Extranets," The Semantic Web - ISWC 2003, vol. LNCS 2870, pp. 802-817, 2003.
2. Y. Jin, S. Xu, S. Decker, and G. Wiederhold, "OntoWebber: a novel approach for managing data on the Web," International Conference on Data Engineering, 2002.
3. N. Stojanovic, A. Maedche, S. Staab, R. Studer, and Y. Sure, "SEAL - a framework for developing semantic portals," Proceedings of the International Conference on Knowledge capture, pp. 155-162, 2001.
4. P. Spyns, D. Oberle, R. Volz, J. Zheng, M. Jarrar, Y. Sure, R. Studer, and R. Meersman, "OntoWeb - a semantic Web community portal," Fourth International Conference on Practical Aspects of Knowledge Management, 2002.
5. E. Bozsak, M. Ehrig, S. Handschuh, A. Hotho, A. Maedche, B. Motik, D. Oberle, C. Schmitz, S. Staab, L. Stojanovic, N. Stojanovic, R. Studer, G. Stumme, Y. Sure, J. Tane, R. Volz, and V. Zacharias, "KAON-towards a large scale Semantic Web," Proceedings of EC-Web, 2002.
6. "BrainEKP." Santa Monica, CA: TheBrain Technologies Corporation, 2004, http://www.thebrain.com.
7. A. P. Sheth and C. Ramakrishnan, "Semantic (Web) Technology In Action: Ontology Driven Information Systems for Search, Integration and Analysis," IEEE Data Engineering Bulletin, vol. 26, pp. 40-48, 2003.
8. R. M. Keller, D. C. Berrios, R. E. Carvalho, D. R. Hall, S. J. Rich, I. B. Sturken, K. J. Swanson, and S. R. Wolfe, "SemanticOrganizer: A Customizable Semantic Repository for Distributed NASA Project Teams," ISWC-2004, 2004.
9. R. M. Keller, and D. R. Hall, "Developing Visualization Techniques for Semantics-based Information Networks," Workshop on Visualization in Knowledge Engineering, 2nd International Conference on Knowledge Capture October.

SEMEX: Mining for Personal Information Integration

Xin Dong, Alon Halevy, Ema Nemes, Stephan B. Sigurdsson, and Pedro Domingos

University of Washington, Seattle
{lunadong, alon, enemes, stebbi, pedrod}@cs.washington.edu

Abstract. Personal information management is one of the key applications of the semantic web. Whereas today's devices store data according to applications, ideal personal information management system should treat all data as a set of meaningful objects and associations between the objects. To ensure extensibility, a personal information management system should automatically incorporate associations generated in multiple ways: mining specific personal data sources, or integrating with external data. As a first step in this direction, we describe the SEMEX system that provides a logical and integrated view of one's personal information.

1 Introduction

The advent of modern networking technology has enabled numerous opportunities for sharing data among multiple parties. Today, data sharing and integration is crucial in large enterprises, government agencies, collaborative scientific projects, and in our personal information management where individuals need to share data from various sources. The pervasive applications of data sharing and integration have led to a very fruitful line of research and recently to a significant industry as well. The vision of the Semantic Web is even more ambitious: web-scale data and knowledge integration.

Despite the immense progress, building an information integration application is still a major undertaking that requires significant resources, upfront effort, and technical expertise. Today, information integration projects proceed by identifying needs in an organization and the appropriate set of data sources that support these needs, typically focusing on frequently recurring queries throughout the organization. As a result, current information integration systems have two major drawbacks. First, evolving the system is hard as the requirements in the organization change. Second, many smaller-scale and more transient information integration tasks that we face on a daily basis are not supported. In particular, integration that involves personal data sources on one's desktop or in one's laboratory is not supported. On the Semantic Web front, it has been observed on several occasions that the growth of the Semantic Web is rather slow, and that personal information management has the potential of fueling faster growth [**?**].

The vision of *on-the-fly information integration* is to fundamentally change the cost-benefit equation associated with integrating information sources. The goal is to aid non-technical users to easily integrate diverse information sources. To achieve this goal, we posit that information integration systems should incorporate two principles:

– The information integration environment should be closely aligned with and be an extension of users' *personal* information space, *i.e.*, the information they store on the desktop (*e.g.*, files, emails, contact lists, spreadsheets, personal databases). In that way, users can extend their personal information views with public data resources.
– Information integration should happen as a *side effect* of people doing their daily jobs, by continuous accumulation of the solutions they produce for their needs of the moment, and by leveraging experiences from previous integration tasks. In short, information integration should be *woven into the fabric* of the organization.

We are building the SEMEX System (short for SEMantic EXplorer), that embodies the vision of on-the-fly integration. With SEMEX, users can access a set of information sources, spanning from personal to public, and from unstructured to structured. Users interact with SEMEX through a domain ontology that offers a set of meaningful domain objects and relationships between these objects. Information sources are related to the ontology through a set of mappings, thereby enabling queries that span multiple sources. Users can personalize their domain models, share domain models with other users, and import fragments of public domain models in order to increase the coverage of their information space. When users are faced with an information integration task, SEMEX aids them by trying to leverage from previous tasks performed by the user or by others with similar goals. Hence, the effort expended by one user later benefits others.

There are three main thrusts to the SEMEX System. This paper focuses on the first of these.

Personal information management (PIM) and integration: Today, the personal information on our desktop is organized by applications (*e.g.*, email, calendar, files, spreadsheets). Finding a specific piece of information involves either searching a file directory or employing a particular application. Integration of multiple pieces of information can only be done manually. Nevertheless, even as early as 1945, Vannevar Bush pointed out in his vision of the *Personal Memex* [Bus45] that our mind works by connecting disparate data items with *associations*, which are not naturally supported by directory and application structures. Hence, an ideal personal information management system should provide a *logical view* of our data so that it can support search through associations between multiple items. A key for its success is that personal information should be populated automatically. This requirement poses an important challenge to the data mining and information extraction communities. The bulk of this paper describes a system that automatically creates such a view, and describes the main technical challenges in doing so.

Personal information as a platform for information integration: Once we have a logical view of our personal information, we can relate external sources to it, thereby facilitate personal tasks that require integration of multiple external sources. Using an architecture such as peer-data management [TIM+03,TH04], we can share data among multiple users. The challenges involved in building this component of SEMEX are to develop tools that make it easy to incorporate external sources (by non-technical users), to personalize the domain model of one's data, and to share these personalized views of data.

Leveraging previous integration tasks: Information integration tasks are often repetitive or closely related to each other. Hence, the final component of SEMEX is to leverage previous integration tasks to facilitate future ones. In this way, users can benefit from integrations performed by colleagues interacting with the same data sources. Our past work on schema matching using machine learning [DDH01] has shown that previous experience can be used to boost the performance of semi-automatic schema matching. Following the same line, mining previous information integration tasks poses several exciting challenges to the data mining community.

In the remainder of the paper we discuss how SEMEX creates a database of instances and associations from one's personal information, thereby offering a logical view of this data. This database complements current storage of personal information, and will form the basis for a variety of services relating to personal information and to information integration. The main technical challenge we address in this component of SEMEX is to reconcile multiple references to the same real-world data item. In contrast to previous work on object-matching (a.k.a. *record linkage, reference reconciliation*), here the references we need to consider (1) do not conform to a single schema, (2) may have multiple values for a single attribute, and (3) typically have very few attributes, thereby exacerbating the challenges involved.

The paper is organized as follows. Section 2 describes the architecture of SEMEX. Section 3 describes the reference reconciliation algorithm and discusses the experimental results on a significant personal data set. Section 4 discusses related work and concludes.

2 Personal Information Management

The first goal of SEMEX is to create a database that consists of objects and relationships between objects obtained from one's personal information (see Figure 1). Objects come from a variety of sources, such as email, contacts, calendar, Latex and Bibtex, Word documents, Powerpoint presentations, pages in the user's web cache, other files in a person's personal or shared file directory, and data in more structured sources, such as spreadsheets and databases. Associations are binary relationships between objects, such as AuthorOf, Sender, Cites, *etc.* Given this logical model of one's personal information, users can seamlessly browse or query their data.

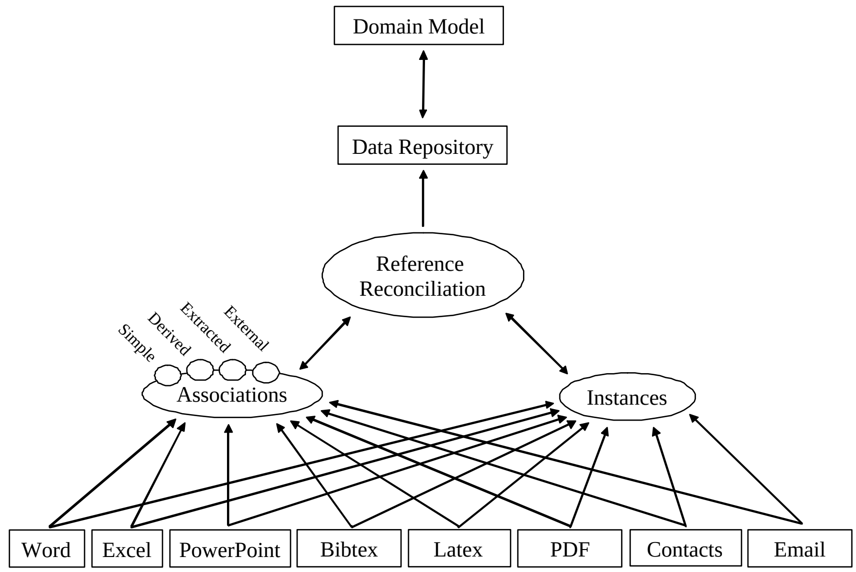


Fig. 1. The architecture of Semex. Semex begins by extracting data from multiple sources. Such extractions create instances of classes in the domain model. Semex employs multiple modules for extracting associations, as well as allowing associations to be given by external sources or to be defined as views over other sets of associations. To combine all these associations seamlessly, Semex automatically reconciles multiple references to the same real-world object. The user browses and queries all this information through the domain model.

Semex stores the objects in a domain ontology, which includes a set of *classes* such as Person, Publication and Event, and *relationships* (which we refer to as *associations*). At the moment the Semex uses a simple data model of classes and associations, but there is a clear need for supporting subclasses and subproperties (*e.g.*, AuthorOf is a subclass of MentionedIn). We also note that our domain model is not a proposal for a standard schema for personal information; it will evolve from several base models by modification and personalization, and we will have to support mappings between the various schemas. The instances and associations that Semex extracts are stored in a separate database. While we have not implemented any sophisticated update mechanisms yet, we envision a module that periodically updates the database and makes the process transparent to the user.

Associations and instances: The key architectural premise in Semex is that it should support a variety of mechanisms for obtaining class and association instances. Semex currently supports the following:

1. *Simple:* In many cases, objects and associations are already stored conveniently in the data sources and they only need to be extracted into the

domain model. For example, a contact list already contains several important attributes of persons, and email messages contain several key fields indicating their senders and receivers.

2. *Extracted:* A rich set of objects and associations can be extracted by analyzing specific file formats. For example, authors can be extracted from Latex files and Powerpoint presentations, and citations can be computed from the combination of Latex and Bibtex files.
3. *External:* External sources can explicitly define many associations. For example, if CiteSeer were to publish a web interface, one could extract citation associations directly from there. Alternatively, a professor may wish to create a class MyGradStudents and populate the class with data in a department database.
4. *Defined:* In the same way as views define interesting relations in a database, we can define objects and associations from simpler ones. As simple examples, we can define the association coAuthor, or the concept emailFromFamily.

In a sense, the domain ontology of SEMEX can be viewed as a *mediated schema* over the set of personal information sources. Instances of the classes and the associations in the domain ontology are obtained from multiple sources. The distinguishing aspect of our context from other information integration settings is that we expect the ontology to be significantly evolved by the user through adding new classes and arbitrary associations.

To make such a system useful, we must ensure that all the data mesh together seamlessly. Specifically, if the same object in the real world (*e.g.*, a person) is referred to in two ways, the system must be able to determine that the two references are to the same object. Otherwise, we will not be able to query effectively on associations, let alone follow chains of associations. In personal data, reference reconciliation is extremely challenging. For example, in the personal data of one author of this paper, there were over 100 distinct ways in which the author was referred. The next section describes the reference reconciliation algorithm of SEMEX.

Browsing and querying interface: SEMEX offers an interface that combines intuitive browsing and a range of querying options. Figure 2 shows a sample screenshot from browsing SEMEX database. Initially, a user can simply type keywords into a search box and SEMEX will return all the objects that are somehow associated with the keyword. For example, typing `Bernstein` in the search box will produce a set of objects that mention Bernstein. Note that the answers to such a query can be a heterogeneous set of objects; SEMEX already classifies these objects into their classes (Person, Publication, *etc.*). When the `Bernstein` person object is selected, the user can see *all* the information related to the person, and the relationship is explicitly specified. (*e.g.*, AuthorOf, CitedIn). The user can then browse any of Bernstein's emails, papers (and then to the objects corresponding to other authors), *etc.* An alternative way to begin browsing is to choose a particular property in the domain model (e.g., AuthorOf) and enter a specific value, thereby specifying an association query.

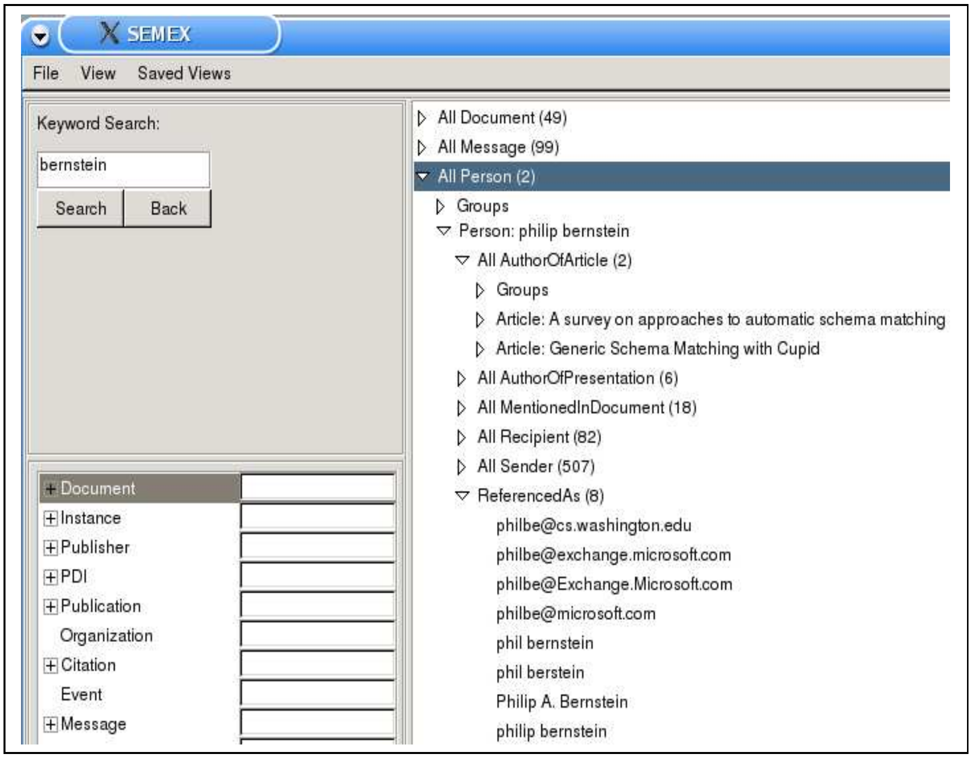


Fig. 2. A sample screenshot from browsing the SEMEX database. Note that the ReferencedAs attribute lists the different ways in which Phil Bernstein is referenced in this personal data set.

3 Reference Reconciliation in Semex

In this section we describe how SEMEX reconciles multiple references to the same real-world object. Our discussion focuses on the hardest reconciliation problem, namely references to persons. We leave the generalization of our algorithm to other objects and domains for further study.

The following example shows three references of persons derived from contact, email and Bibtex data.

$$\text{name, phone} : \texttt{Mike Carey}, (\texttt{123})\texttt{456} - \texttt{7890}$$
$$\text{email} : \texttt{carey@almaden.ibm.edu}$$
$$\text{name} : \texttt{M. Carey}$$

Earlier approaches (see [BMC$^+$03] for a recent survey) to reference reconciliation focus on reconciling tuple references from a single database table; these tuples share attributes and each attribute allows a single value. These approaches do not directly apply to Semex for four reasons. First, the data sources in Semex are heterogeneous, containing different sets of attributes; as the above example shows, the attributes of the first and the second references even do not overlap. Second, each attribute of a person object may contain multiple values: it is common for a person to have multiple email accounts and phone numbers. Furthermore, some of the statistical techniques that have been considered are difficult to apply because of the relatively small size of the personal data sets. Finally, training data is also not readily available, which limits the application of supervised learning. On the other hand, the size of the data sets allows for more computationally intensive matching algorithms.

3.1 Reference reconciliation algorithm

Traditionally, the reference reconciliation problem was solved by independently matching each pair of references, and taking a transitive closure over matching pairs. In the case of people, each single reference is rather weak (*i.e.*, contains relatively little information). To tackle this problem, our algorithm repeats the comparing-and-clustering process several times, each time considering a result cluster obtained from the preceding pass as a single reference, and recomputing the distances between new references based on a different distance measure. The stronger reference may potentially be matched with other instances with which its constitutes could not be matched before.

Specifically, the algorithm begins by assigning each reference to a class of cardinality one and then successively refines the relation in four passes.

Step 1: Reconciling based on shared keys. The first step merges references that share exact values on keys. For person instances, `name` and `email` can each serve as a key.

Step 2: Reconciling based on string similarity. The second iteration combines string matching features with domain-specific heuristics. We employ edit distance [BMC$^+$03] to measure string similarity. In some cases we exploit the specific data types and apply domain heuristics. For example, we compare email addresses by exploiting knowledge of the different components of the address and recognizing certain mail software idiosyncrasies. In the case of phone numbers, we allow for missing area codes or additional extension numbers.

Step 3: Applying global knowledge. Now that we have grouped multiple references into clusters, we can extract global information to perform additional merging. We give two important examples of such global knowledge. In the first case, the knowledge is extracted *within* the cluster, and in the second case we use *external* information. We note that the algorithm is conservative when applying global knowledge, as we consider avoiding false positives more important to guarantee quality browsing of personal information.

– *Time-series comparison:* The time-series analyzer selects pairs that were judged similar in the previous passes, but not combined. It then collects for each reference a set of time stamps associated with its email messages. If the time series have little or no overlap, the references are merged. This heuristic works well for detecting people who move from one institution to another. In our experiments, this method was very effective.

– *Search-engine analysis:* Our search-engine analyzer feeds the texts of two references into the Google search engine (via their web-service interface) and compares the top hits. Two references to the same person object tend to obtain similar top hits in Google search. In our experiments, this technique also helped resolve a significant number of references.

The result of the reconciliation algorithm is a high-quality reference list of people mentioned in one's personal data. We then leverage this list to obtain additional associations within the data set. For example, we search for occurrences of the names in the reference list in spreadsheets and the top portions of Word and PDF files to create associations to these types of files. We do not discuss the details of this step due to space limitations.

	Count	%	Size [kb]	%
Messages	18037	—	—	—
Contacts	240	—	—	—
Files	7085	100%	886836	100%
Latex	582	8%	7332	1%
Bibtex	25	0.9%	2236	0.3%
PDF	97	1.3%	24768	2.8%
PostScript	668	9.4%	215584	24%
Plain text	51	0.7%	940	0.1%
Rich text	31	0.4%	104	0.0%
HTML/XML	666	9.4%	7060	0.8%
Word	400	5.6%	12092	1.3%
PowerPoint	777	11%	151045	17%
Excel	55	0.7%	1396	0.2%
Multimedia	539	7.6%	123521	14%
Archives	475	6.7%	15754	1.8%
Other	1809	32%	194112	22%

Table 1. The characteristics of our experimental data set.

3.2 Experiments

We describe the results of experiments applied to a personal data set of one author of this paper[1]. The data set spans six years of activities and consists of

[1] To further complicate matters, this author changed his name from Levy to Halevy a few years ago.

	Before Reconciliation	%
Instances	23318	100%
Person	5014	22%
Message	17322	74%
Document	805	3%
Publication	177	1%
Associations	38318	100%
senderOf	17316	45%
recipientOf	20530	54%
authorOf	472	1%

Table 2. The number of instances extracted from the raw data for classes in the domain model. For example, after scanning all the sources, we have 5014 person references, and these need to be reconciled.

the usual variety of personal data (though probably more Latex files than typical computer users). Table 1 details the characteristics of the raw data, and Table 2 shows the number of instances extracted from the raw data for several of the classes in the domain model.

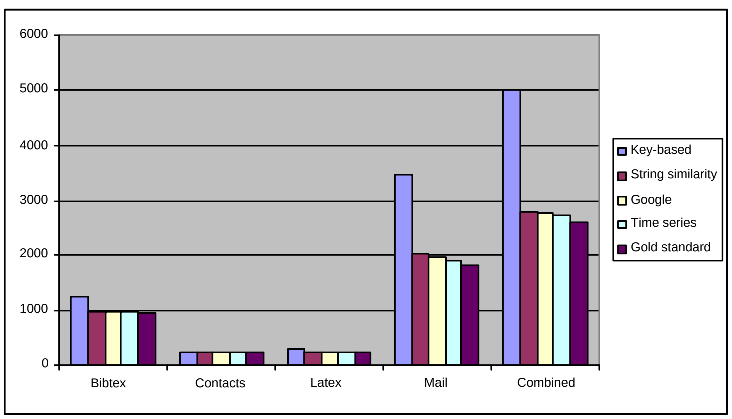


Fig. 3. This figure shows the progress of the reference reconciliation algorithm w.r.t. its different steps. The right-most set of bars concerns the entire data set, while the other sets consider individual components of the data set.

We limit the following discussion to person instances. Figure 3 shows the progress of the matching algorithm for each component of the data set in isolation (*i.e.*, for Bibtex, contacts, email, latex), and then the results for all these components combined. The rightmost column (labeled *gold standard*) in each group indicates the *actual* number of distinct objects in the domain. The other columns report the numbers of clusters after each reconciliation step.

We observe from the experiment that the first two steps of the algorithm remove 91% of the extra references (*i.e.*, differences between the references extracted directly from the raw data set and the distinct ones in the gold standard). The time-series and Google analyzers successively remove an additional 1.7% of the beginning total of extra references each, but more importantly, these correspond to 18% and 29% of the references that still need to be reconciled. We also observed that changing the order of the time-series and Google analyzers does not change the results substantially.

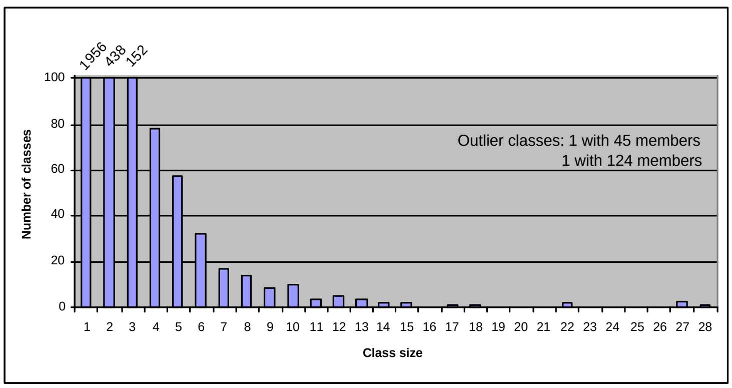


Fig. 4. The number of different references per person after the reconciliation algorithm is applied.

Another perspective on the quality of the reference reconciliation is shown in Figure 4. Each bar shows the number of persons for whom there are n references, where n labels the bar (therefore, when users browse the data they could expand the single collapsed reference to see the n original references to that person).

In conclusion, while the current reconciliation algorithm already provides a reasonable start, we believe that techniques for reference reconciliation by growing clusters of references merits additional study.

4 Related Work and Conclusions

A number of PIM projects studied the method to organize and search information effectively. They all discard the traditional hierarchical directory model. Haystack [QHK03] and MyLifeBits [GBL$^+$02] resort to annotations in building a graph model of information; Haystack puts more emphasis on personalization. Placeless Documents [DEL$^+$00] annotates documents with property/value pair, and group documents into overlapping collections according to the property value. Stuff I've Seen (SIS) [DCC+03] indexes all types of information and provides a unique full-text search interface. Finally, LifeStreams [FG96] organizes documents based on a chronological order. All of the above projects manage information at the document level. Our approach distinguishes from them by taking objects as the search and organization unit and facilitating the search with associations between objects. The system uses an ontology to guide information management, allowing manipulation and personalization of the ontology.

This paper serves to bring personal information management closer to the mainstream of data management research, and as a platform for the next generation of information integration systems. Specifically, we have argued that the keys to research on personal information management are to seamlessly integrate users' personal information views with organizational data sources and to integrate information on-the-fly. We described the current implementation of SEMEX that performs personal information management and integration. We described a novel reference reconciliation algorithm for personal information, and showed that it performs well on a sizable data set.

Personal information management is a rich area for further research. In the immediate future, our goal is to improve the reference reconciliation algorithm. We believe that rich probabilistic models hold great promise in this context because there is a clear need to combine evidences from multiple sources during the reconciliation. Further down the road, we plan to use the SEMEX database to discover useful patterns in one's data set, such as clusters of people who are related in ways that are not explicit in one's data. Finally, we will use SEMEX to coordinate multiple PIM devices and provide a flexible tool for merging multiple data sets of a user.

References

[BMC+03] Mikhail Bilenko, Raymond Mooney, William Cohen, Pradeep Ravikumar, and Stephen Fienberg. Adaptive name matching in information integration. *IEEE Intelligent Systems Special Issue on Information Integration on the Web*, September 2003.

[Bus45] Vannevar Bush. As we may think. *The Atlantic Monthly*, July 1945.

[DCC+03] Susan Dumais, Edward Cutrell, JJ Cadiz, Gavin Jancke, Raman Sarin, and Daniel C. Robbins. Stuff i've seen: A system for personal information retrieval and re-use. In *SIGIR*, 2003.

[DDH01] Anhai Doan, Pedro Domingos, and Alon Halevy. Reconciling schemas of disparate data sources: a machine learning approach. In *Proc. of SIGMOD*, 2001.

[DEL+00] Paul Dourish, W. Keith Edwards, Anthony LaMarca, John Lamping, Karin Petersen, Michael Salisbury, Douglas B. Terry, and James Thornton. Extending document management systems with user-specific active properties. *ACM TOIS*, 18(2), 2000.

[FG96] Eric Freeman and David Gelernter. Lifestreams: a storage model for personal data. *SIGMOD Bulletin*, 1996.

[GBL+02] Jim Gemmell, Gordon Bell, Roger Lueder, Steven Drucker, and Curtis Wong. Mylifebits: Fulfilling the memex vision. In *ACM Multimedia*, 2002.

[QHK03] Dennis Quan, David Huynh, and David R. Karger. Haystack: A platform for authoring end user semantic web applications. In *ISWC*, 2003.

[TH04] Igor Tatarinov and Alon Halevy. Efficient query reformulation in peer data management systems. In *Proc. of SIGMOD*, 2004.

[TIM+03] Igor Tatarinov, Zachary G. Ives, Jayant Madhavan, Alon Y. Halevy, Dan Suciu, Nilesh N. Dalvi, Xin Dong, Yana Kadiyska, Gerome Miklau, and Peter Mork. The piazza peer data management project. *SIGMOD Record*, 32(3):47–52, 2003.

A Knowledge Discovery Workbench for the Semantic Web

Jens Hartmann and York Sure
Institute AIFB, University of Karlsruhe, 76128 Karlsruhe, Germany
http://www.aifb.uni-karlsruhe.de/WBS/
email:{hartmann,sure}@aifb.uni-karlsruhe.de

Abstract

We present a workbench for integrating Web documents into semantically enriched representations suitable on the Semantic Web. The approach benefits on the one hand from the facilities provided by Semantic Web technologies and on the other hand from the applicability of well-known knowledge discovery techniques. The main achievement of our contribution is an up-and-running, open and component based prototype which can be easily extended by 3rd parties.

1 Introduction

The World Wide Web consists of information concerning nearly every imaginable topic represented by weakly structured Web documents. The process of searching and accessing relevant information on the Web leads often to a practical problem [1] hampered by the lack of semantic markup and missing inference capabilities [2, 3].

As an evolutionary step the Semantic Web [4] tends to overcome these problems by applying formal knowledge representation languages such as OWL [5] and enabling inferencing capabilities. Consequently, existing Web documents have to be translated into knowledge representations suitable for the Semantic Web, e.g. RDF(S) [6] or OWL [5]. Hence, we argue that the task of integrating Web documents for the Semantic Web acts a key challenge for the Semantic Web.

Our approach relies on a combination of knowledge discovery and semantic web technologies. It is built on top of the knowledge discovery process by [7]. Each step of the process is implemented by a component of our system. The developed system ARTEMIS is freely available[1]. We argue that extensibility of knowledge discovery systems and data mining algorithms is essential for successful real-world applications, as discussed in [8]. Hence, ARTEMIS is open and can be easily extended by 3rd parties. Further, we extend existing data mining methods with ontologies as background knowledge to improve (i) the mining task and (ii) the quality of created data models. This philosophy is also reflected by the software architecture itself: ARTEMIS uses semantic technologies in a component oriented software architecture.

[1] see http://artemis.ontoware.org

2 An Example of learned Document Models

We illustrate the impact of the ARTEMIS approach using results we achieved on classifying the Web site of the University of Bremen[2]. The goal was to learn classification rules that uniquely identify pages of the research group on theoretical computer science. For this purpose we used about 150 pages of that group as positive and about 300 other pages from the university Web site as negative examples. Table 1 shows generated rules for the different mode declarations and the accuracy of the rules.

Experiment A1-0		TrainingSet0
	TZI - Theory	
Mode Dec.	*Hypotheses*	*Acc.*
H 1	document(A) :- doctitle(A,research).	100
H 2	document(A) :- metatag(A,keywords, theoretical).	100
H 3	document(A) :- relation(A,B), relation(B,C), mail(C,helga,'informatik.uni-bremen.de').	86,82
H 4	document(A) :- relation(A,B), url(B,'[URL]/cs/ref.num.html'). document(A) :- relation(A,B), url(B,'[URL]/projects.html').	86,82
URL: `http://www.tzi.de/theorie`		

Table 1: An Example of Document Models

The results show the different kinds of classification rules (models) we get when using different elements of Web documents. Using the page title as a criterion, we find out that the pages of the theoretical computer science group are exactly those that contain the word 'research' in their title (H1). An analysis of metadata (H2) shows that the keyword 'theoretical' uniquely identifies the pages we are interested in. We get even more interesting results that still have an accuracy of more than 85% when analyzing e-mail addresses and links to other pages (H3). For the case of e-mail addresses we find out that most pages are linked over steps with a page that contains the mail address of the secretary of the group. If we only consider links (H4), we see that most pages are linked to pages containing references and to a page listing projects of the group.

3 ARTEMIS Workbench

The ARTEMIS Workbench represents a tool for knowledge engineers and industrial practitioners required to integrate large and heterogenous sets of documents whereby it provides functionalities of well-known knowledge discovery tools to generate semantic enabled document models to apply them on the Semantic Web.

To avoid such intricateness, ARTEMIS combines well-known knowledge discovery methods on the one hand and semantic technologies such as ontology-based knowledge engineering and reasoning techniques on the other hand. This combination is realized by an expressive and easily extendable component architecture with semantic enriched interfaces.

[2]http://www.uni-bremen.de

3.1 General Overview

The workbench consists of three main blocks: (i) The **ARTEMIS Core System (ACS)** as surrounding technology, (ii) the **Workflow Model (WM)** providing a knowledge discovery workflow and (iii) the **Component Model (CM)** instantiates the workflow by extensible components as presented in figure 1.

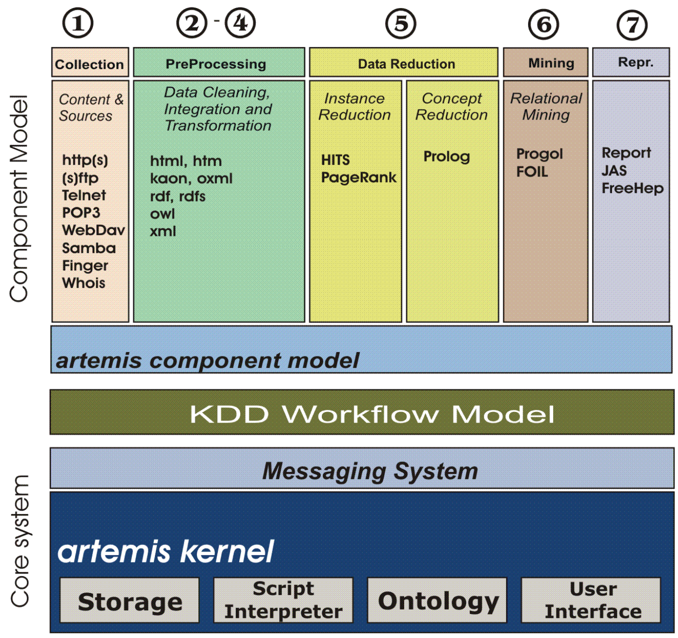


Figure 1: ARTEMIS Architecture

The ARTEMIS *Core System* contains the main system functionalities which are subdivided into the *kernel* and the *messaging system*. The *kernel* provides core functionalities for the workbench like realising **storage mechanisms**, running a **script interpreter** and providing the **ARTEMIS ontology** for the components.

The *Component Model* provided by ARTEMIS instantiates the knowledge discovery process of the *Workflow Model* and provides components for each process step. A component used within ARTEMIS provides a semantic description in form of an ontology which (i) allows to classify the type of component according the workflow model and (ii) provides a set of services to the ARTEMIS workbench, e.g. a text classification algorithm.

3.2 Workflow Model

The accomplishment of a knowledge discovery process is handled by the **Workflow Model** which provides a workflow manager to monitor the flow of data and extracted information. Further, it assures the application of components depending on the current process step. Our approach instantiates the knowledge discovery process presented in [7, 9]. As indicated in Figure 2 ARTEMIS provides for each step of the process specialised components.

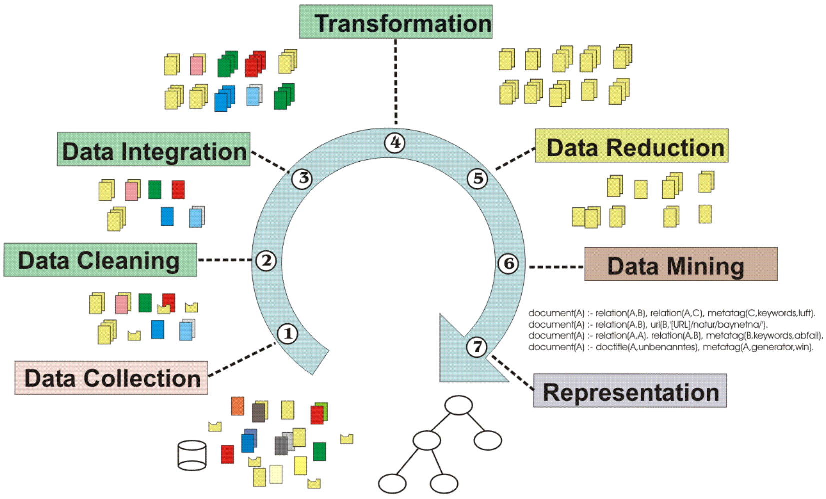


Figure 2: ARTEMIS Workflow

4 Knowledge Representation

In order to use the PROGOL system for generating document models , we have to encode knowledge about Web documents and their internal structure in PROLOG. For this purpose, we developed a representation scheme consisting of a set of pre-defined predicates.

- `document(object)`: the constant 'object' represents a document
- `url(object, ADRESS)`: the document represented by 'object' has the URL 'ADDRESS'
- `relation(doc1, doc2)`: there is a directed link between the document 'doc1' and 'doc2'
- `structure(object, CLASS)`: the constant 'object' represents an element tag of type 'CLASS'
- `contains(doc, object)`: the document contains the tag 'object' as a top level element.
- `attribute(parent, object)` the element tag 'parent' contains the attribute 'object'
- `contains(parent, object)` the element 'parent' contains the element 'object' as a child element
- `value(object, 'VALUE')`: 'object' is an element or attribute and it has the value 'VALUE'
- `text_value(object, 'TEXT')`: 'object' is an element or attribute and it has the text 'TEXT'

In order to be able to use an ILP learner for the acquisition of document models, the structure of the documents serving as positive and negative examples have to be translated into the representation described above. Unfortunately, most of the documents came in less standardized form, partly containing syntactic errors. Therefore all training examples were semi-automatically cleaned and tidied up. We use HTML Tidy[3] and its Java pendant JTidy[4] for this task.

The next step to obtain a usable training set is the *syntactical translation* of the training examples. A Web document like a HTML or an XML Document contains predefined tags which describes structure (in particular relations inside a document or between other documents) and layout of documents. The complete translation process is described here in a very abstract way: (i) Every document is parsed into a DOM tree. We use Apache JXERCES 2.0 for this task. (ii) ARTEMIS then walks through the DOM tree. Depending on a predefined translation scheme all desired tags are translated into PROLOG clauses. (iii) The positive and negative examples are stored into a database which represents the training set. (iv) In order to enable the system to perform a restricted kind of learning on the text of a page, simple normalization techniques are applied that convert the words of a text into lower case letters, removes special symbols as well as words from a stop list and inserts a list of the remaining words in the PROLOG notation. More details can be found in [10].

5 Conclusion

We presented an approach for automatically acquiring models from Web documents applicable on the Semantic Web. The approach can be used to integrate Web documents with semantic markup in terms of an assignment to certain ontologies for building repositories or data warehouses. We discussed the architecture and its provided component model extensible by 3rd parties.

Acknowledgements

Research reported in this paper has been partially funded by EU in the IST project SEKT (IST-2003-506826), the network of excellence Knowledge Web (IST-2004-507482) and has been funded by the german BMBF (federal ministry of education and research) project SemIPort. We thank all our colleagues at AIFB for fruitful discussions.

References

[1] Chakrabarti, S.: Mining the Web: Discovering knowledge from hypertext data. Morgan Kaufmann, San Francisco (2003)

[2] Rindflesch, T., Aronson, A.: Semantic processing in information retrieval. In Safran, C., ed.: Seventeenth Annual Symposium on Computer Applications in Medical Care (SCAMC 93), McGraw-Hill Inc., New York (1993) 611–615

[3] http://www.w3.org/People/Raggett/tidy/
[4] http://lempinen.net/sami/jtidy/

[3] Zweigenbaum, P., Bouaud, J., Bachimont, B., Charlet, J., Séroussi, B., Boisvieux, J.: From text to knowledge: a unifying document-oriented view of analyzed medical language. Methods of Information in Medicine **37(4-5)** (1998) 384–393

[4] Berners-Lee, T.: Weaving the Web. Harper (1999)

[5] Smith, M.K., Welty, C., McGuinness, D.: OWL Web Ontology Language Guide (2004) W3C Recommendation 10 February 2004, available at http://www.w3.org/TR/owl-guide/.

[6] Brickley, D., Guha, R.V.: RDF Vocabulary Description Language 1.0: RDF Schema. W3C Recommendation 10 February 2004 (2004) available at http://www.w3.org/TR/rdf-schema/.

[7] Fayyad, U.M., Piatetsky-Shapiro, G., Smyth, P.: Knowledge discovery and data mining: Towards a unifying framework. In: Knowledge Discovery and Data Mining. (1996) 82–88

[8] Wrobel, S., Wettschereck, D., Sommer, E., Emde, W.: Extensibility in data mining systems. In Simoudis, E., Han, J.W., Fayyad, U., eds.: Proc. 2nd International Conference On Knowledge Discovery and Data Mining, Menlo Park, CA, USA, AAAI Press (1996) 214–219

[9] Chang, G., Healey, M.J., McHugh, J.A.M., Wang, J.T.L.: Mining the world wide web (2001)

[10] Stuckenschmidt, H., Hartmann, J., van Harmelen, F.: Learning structural classification rules for web-page categorization. In: Proceedings of FLAIRS 2002, special track on Semantic Web. (2002)

A Framework for Image Annotation Using Semantic Web

Ahmed Bashir and Latifur Khan

University of Texas at Dallas
{ahmedb, lkhan}@utdallas.edu

Abstract. The impetus behind Semantic Web research remains the vision of supplementing availability with utility; that is, the World Wide Web provides availability of digital media, but the Semantic Web will allow presently available digital media to be used in unseen ways. An example of such an application is multimedia retrieval. At present, there are vast amounts of digital media available on the web. Once this media gets associated with machine-understandable metadata, the web can serve as a potentially unlimited supplier for multimedia web services, which could populate themselves by searching for keywords and subsequently retrieving images or articles, which is precisely the type of system that is proposed in this paper. Such a system requires solid interoperability, a central ontology, semantic agent search capabilities, and standards. Specifically, this paper explores this cross-section of image annotation and Semantic Web services, models the web service components that constitute such a system, discusses the sequential, cooperative execution of these semantic web services, and introduces intelligent storage of image semantics as part of a semantic link space.

1 Introduction

The impetus behind Semantic Web research remains the vision of supplementing availability with utility; that is, the World Wide Web provides availability of digital media, but the Semantic Web will allow presently available digital media to be used to serve new purposes, an example of which is image retrieval.

The Semantic Web is an extension of today's Web technology; it boasts the ability to make Web resources accessible by their semantic contents rather than merely by keywords and their syntactic forms. Due to its well-established mechanisms for expressing machine-interpretable information, information and Web services previously available for human consumption can be created in a well-defined, structured format from which machines can comprehend, process and interoperate in an open, distributed computing environment.

This proves to be quite advantageous with respect to data collection; intelligent software entities, or agents, can effectively search the web for items of interest, which they can determine with new semantic knowledge. For instance, sports images or articles can be retrieved from around the web and processed by the respective web services to enhance a website in terms of the sheer multimedia content available. In such a system, the semantic web serves as a large, automated image collection that may be used to populate an annotated image “gallery”. This image “gallery” would be represented as a semantic link space that organizes like images together based on known image semantics; for example, all basketball images would be grouped as an image network, and so on.

Combining image retrieval with the Semantic Web, however, is not merely beneficial due to the availability of raw data or the potential for automated image annotation, but there is also the added benefit of using a web ontology, or a set of concepts and their interrelations. By using such ontologies not only to search for multimedia but also to classify it, the system ensures consistency in terminology, leading to more accurate and precise query results.

1.1 The Approach

At present, there are vast amounts of digital media available on the web. Once this media gets associated with machine-understandable metadata, the web can serve as a potentially unlimited supplier for multimedia web services, which could populate themselves by searching via keywords and subsequently retrieving images or articles. This presents a novel approach to semi-automatic image annotation or classification. In this case, not only is the annotation done automatically once both the support vector machine and the Bayesian network are trained, but the source is replenished automatically, as well.

The image annotation task has been decomposed into classification of low-level, or atomic, concepts and classification of high-level concepts in a domain-specific ontology. In the general sense, concepts are atomic if they are terms that can describe specific objects or image segments. Examples would be ball, stick, net, and other well-defined objects. High-level semantic concepts, on the other hand, are used to describe an environment with a set of existing atomic concepts associated to it. For example, an image that contains a ball, a net, shoes, and humans can be described as a basketball game. The framework takes advantage of this natural gap in semantics, classifying atomic concepts using support vector machines and high-level concepts using Bayesian belief networks.

Upon classifying the image, the system would reflect the image semantics, its features, content, and semantic category, as part of a semantic link space. Figure 1 illustrates the layered architecture. The bottom-most layer represents the original image, and the layer directly above it will represent the image semantics using an ontology. The semantic space can then, as mentioned, prepare image networks based on the available image semantics, and the features that correspond to the respective images will constitute the feature space. As part of the operation interface, a user or a web agent can query the system, which would search and retrieve image information from the underlying layer [11].

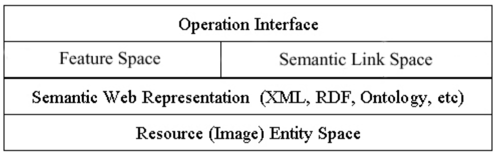


Fig. 1. Semantic space architecture

1.2 Experimental Context

Results have shown that separating atomic classification from high-level classification improves Bayesian classification by reducing the complexity of the directed acyclic graph associated with the Bayesian network. This way, image features are abstracted away and the Bayesian structure includes only semantic concepts. With a reasonably strong segmentation algorithm, results are promising. To test the strengths and weaknesses of this system, a classic segmentation algorithm has been combined with a support vector machine and a Bayesian network. The training set consists of 3,000 feature sets, and over 300 images have been classified.

1.3 Contributions

This paper explores this cross-section of image annotation and Semantic Web services, models the web service components that constitute such a system, discusses the sequential, cooperative execution of these semantic web services, and presents the technical challenges. The main contributions deal with the integration of these new service-building technologies, the use of two classification methods to separate high-level and low-level semantic concepts, the hyperlink-based search and collection of fresh raw images using intelligent web agents, and of course the representation of key image information in terms of a semantic link space rather than a local image repository. Another key contribution is the use of a web ontology as a multipurpose tool that seamlessly integrates different service components; the ontology can serve as the Bayesian structure to classify images or text, a translator to understand user queries, and an instructor for agents that gather multimedia.

2 Proposed Architecture for Prototype System

2.1 Framework Description

The prototype system will consist of an interface through which users can query the system regarding specific sports. This request would be processed by a service that would retrieve the relevant images and articles from their respective repositories and present them to the user.

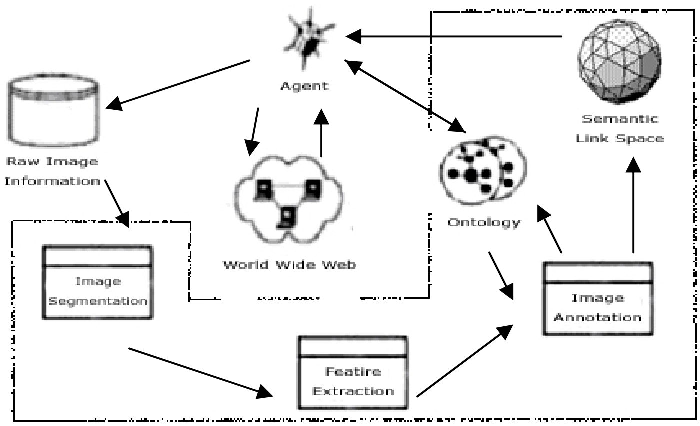


Fig. 2. Proposed architecture

As depicted in Figure 2, web agents collect unclassified images using a hyperlink-based approach in order to build the semantic link space containing image semantics and classifications; agents will discard images that cannot be supported by the system. Supported images are segmented into various objects, and the objects' features are subsequently extracted; features include hue, intensity, saturation, and shape. Feature vectors are sent into the support vector machine, which will identify low-level concepts that exist in each image. The set of low-level concepts are sent as known evidence into a trained Bayesian network, which will classify the images according to high-level semantic concepts. The image semantics can then be stores as part of the semantic link space.

Lastly, the system contains the central ontology. This ontology will contain detailed information regarding the sports domain, specifically the different types of sports, the equipment involved, and so on. Moreover, the service ontology will be continually changing as agents are able to discover more sports and so on. For example, agents may discover a new sport, tennis for example, and add it into the ontology. Alternatively, agents may find additional gear that is associated to an existing sport, such as a baseball helmet. However, the system as discussed in this paper assumes a single ontology without any ontology merging, which is beyond the scope of this paper.

2.2 Technical Challenges

To implement such a system involves an integration of several up and coming technologies. This section highlights some of the technologies involved and the challenges presented by each of them.

2.2.1 Image Annotation

Annotation has been decomposed into identifying low-level and high-level semantic concepts, respectively. The former will be determined using support vector machines, and Bayesians networks will determine the latter.

The support vector method, or SVM, is a technique which is designed for efficient multidimensional function approximation; the aim is to determine a classifier or regression machine which minimizes the training set error. The basic procedure is to fix the empirical risk associated with an architecture and then to use a method to minimbize the generalization error. The primary advantage of support vector machines as adaptive models for binary classification and regression is that they provide a classifier with a low expected probability of generalization errors. This approach can be trivially extended to multi-class classification by getting a binary response with respect to each atomic classification.

Bayesian networks are used to model the causal relationships that exist in the context of image semantics. This will allow a system to associate query keywords with other semantic concepts to some degree of belief. Hence, image queries will be more intelligently handled and will yield better results, a direct result of understanding the dependencies and relationships between different semantic concepts. The causal relationships can be patterned using the provided web ontology, which already represents a set of concepts and their interrelations. The variables that will make up the network will be a combination of high-level semantic classifiers as well as atomic level classifiers sent from the Support Vector Machine. For the purpose of this project, the Bayesian structure is precisely the web ontology.

2.2.2 OWL-S

To make use of a Web service, a software agent needs a computer-interpretable description of the service, and the means by which it is accessed. Semantic Web markup languages must not only establish a framework within which these descriptions are made and shared but also enable one web service to automatically locate and utilize services offered by other web services. OWL-S provides the solution, providing facilities for describing service capabilities, properties, pre-/post-conditions, and input/output specifications.

2.2.3 WSDL

Web Services Description Language (WSDL) is a new specification to describe networked XML-based services. It provides a simple way for service providers to describe the basic format of requests to their systems regardless of the underlying protocol, in our case SOAP. Under the WSDL standard, network services are viewed as a set of endpoints operating on messages containing either document-oriented or procedure-oriented information. The operations and messages are described abstractly, and then bound to a concrete network protocol and message format to define an endpoint. Related concrete endpoints are combined into abstract endpoints, or services. WSDL is extensible to allow description of endpoints and their messages regardless of what message formats or network protocols are used to communicate [4].

WSDL documents describe operations, messages, datatypes, and communication protocols specific to a web service. To carry out the communication between web services, SOAP will be used. SOAP provides the framework by which application-specific information may be conveyed in an extensible manner. Also, SOAP provides a full description of the required actions taken by a SOAP node on receiving a SOAP message. The SOAP stack will convert SOAP requests into native requests that the web service can make use of. Similarly, the web services' responses must be designed as SOAP responses.

2.2.4 Semantic Link Space

Information regarding classified images will be organized using a semantic link space [12]. By associating like images together, image networks are created; similarity will be judged based on image semantics, including hue, saturation, intensity, and shape. By using this idea in conjunction with the hyperlink-based approach, user queries would be satisfied.

3 Results and Conclusions

Results from image annotation have proved that a properly trained support vector machine and Bayesian network can work alongside one another to produce satisfactory results. The SVM was trained with a mix of basketball, baseball, bat, soccer, hoop, and grass images that total 3000 training objects. The training set captured key characteristics of each image segment: hue, saturation, intensity, and shape. Once the support vector machine was trained, the training set was also used as test data in order to judge the training accuracy, which averaged at 98.5%.

The recall values indicate how well the system fared in recognizing all the segments that depict the same object. For instance, in the case of grass, the system is able to recognize 74 of the 80 grass objects, resulting in a 93% recall. However, there are 88 grass objects recognized, so the precision value is used to indicate how many of the retrieved positively classified images truly depict the object in question. In this case, 74 of the 88 positively classified grass objects are actually grass objects, leading to an 84% precision. Some of the problems with atomic classification are intuitive. In the case of a basketball hoop, the support vector machine is attempting to recognize an object that lacks a definite shape or color. Soccer balls are also tough to recognize because they are composed of two distinct colors: black and white. Complete results are presented in Tables 1 and 2.

Table 1. SVM classification results

Object	**# Expected**	**# Retrieved**	**Recall**	**Precision**
Soccer	80	61	39%	51%
Grass	80	88	93%	84%
Basket-	150	134	89%	100%
Baseball	100	123	83%	67%
Bat	80	81	100%	99%
Hoop	30	81	100%	37%

Table 2. Bayesian classification results

Classification	**# Expected**	**# Retrieved**	**Recall**	**Precision**
Basketball	150	181	92.7%	76.8%
Soccer game	80	111	100%	72.1%
Baseball game	100	135	96%	71.1%

Higher-level classification suffers in all instances where atomic classification falls short; however, one idea that deserves mention is the difference between misclassification and unclassification. For example, even if a basketball is not recognized as a basketball, there is still an inherent benefit of not recognizing the basketball as another type of object, a soccer ball for instance. In the case of the 150 basketball pictures, 136 were retrieved due to limited misclassification. On the other hand, of the 80 soccer images, 35 were misclassified at the atomic level as containing baseballs. In this case, the Bayesian network will be unable to accurately classify the image, which will subsequently be discarded.

The results, particularly the precision values, show that there are too many multiple classifications. For example, an image that contains a soccer ball, a bat, and a baseball will be retrieved both as a baseball image as well as a soccer image. Another important note is that, due to a simple Bayesian structure, images were often classified correctly if one of two objects were recognized.

The system will be extended to recognize details pertaining to the environment so images can be classified on the basis of indoors or outdoors and team or individual. Extracting such detailed information from an image will again require a strong segmentation algorithm coupled with some preprocessing that will give way to more intelligent segmentation so that regions can be identified more accurately.

References

1. O. Marques and N. Barman. Semi-automatic Semantic Annotation of Images Using Machine Learning Techniques. In *The Semantic Web - ISWC 2003 Proceedings*, pages 550-565, 2003.
2. Protégé-2000. http://protege.stanford.edu/
3. R. E. Neapolitan. *Learning Bayesian Networks*. Prentice-Hall, Upper Saddle River, NJ, 2004.
4. E. Christensen, F. Curbera, G. Meredith, and S. Weerawarana. Web Services Description Language (WSDL) 1.1. http://www.w3.org/TR/wsdl
5. The OWL Services Coalition. OWL-S: Semantic Markup for Web Services. http:// www.daml.org/ services/owl-s/1.0/owl-s.html
6. C. Cortes and V. Vapnik. Support vector networks. *Machine Learning*, 20:1-25, 1995.
7. L. Khan, D. McLeod, and E. Hovy. Retrieval Effectiveness of Ontology-based Model for Information Selection. The VLDB Journal: The International Journal on Very Large Databases, ACM/Springer-Verlag Publishing, Vol. 13(1): 71-85 (2004).
8. A. Benitez and Shih-Fu Chang. Multimedia Knowledge Integration, Summarization and Evaluation. http://citeseer.ist.psu.edu/benitez02multimedia.html"
9. Oge Marques, Nitish Barman. Semi-automatic Semantic Annotation of Images Using Machine Learning Techniques. International Semantic Web Conference 2003: 550-565.
11. Hai Zhuge. Semantic-Based Web Image Retrieval. http://www2003.org/cdrom/papers/poster/p172/p172-zhuge/p172-zhuge.htm
12. Eero Hyvönen, Samppa Saarela, Avril Styrman, and Kim Viljanen. Ontology-Baed Image Retrieval. http://www.cs.helsinki.fi/group/seco/presentations/www2003/p199-hyvonen.html

Boosting for Text Classification with Semantic Features

Stephan Bloehdorn, Andreas Hotho

University of Karlsruhe, Institute AIFB, Knowledge Management Group, Germany
sbl@aifb.uni-karlsruhe.de
University of Kassel, Knowledge and Data Engineering Group, Germany
hotho@cs.uni-kassel.de

Abstract. Current text classification systems typically use term stems for representing document content. Semantic Web technologies allow the usage of features on a higher semantic level than single words for text classification purposes. In this paper we propose such an enhancement of the classical document representation through concepts extracted from background knowledge. Boosting, a successful machine learning technique is used for classification. Comparative experimental evaluations in three different settings support our approach through consistent improvement of the results. An analysis of the results shows that this improvement is due to two separate effects.

1 Introduction

Most of the explicit knowledge assets of today's organizations consist of unstructured textual information in electronic form. Users are facing the challenge of organizing, analyzing and searching the ever growing amounts of documents. Systems that automatically classify text documents into predefined thematic classes and thereby contextualize information offer a promising approach to tackle this complexity [17]. During the last decades, a large number of machine learning methods have been proposed for text classification tasks. Recently, especially Support Vector Machines [9] and Boosting Algorithms [16] have produced promising results.

So far, however, existing text classification systems have typically used the *Bag-of-Words model* known from information retrieval, where single words or word stems are used as features for representing document content. By doing so, the chosen learning algorithms are restricted to detecting patterns in the used *terminology* only, while *conceptual* patterns remain ignored. Specifically, systems using only words as features exhibit a number of inherent deficiencies:

1. *Multi-Word Expressions* with an own meaning like *"European Union"* are chunked into pieces with possibly very different meanings like *"union"*.
2. *Synonymous Words* like *"tungsten"* and *"wolfram"* are mapped into different features.
3. *Polysemous Words* are treated as one single feature while they may actually have multiple distinct meanings.
4. *Lack of Generalization*: there is no way to generalize similar terms like "beef" and "pork" to their common hypernym "meat".

While items 1 – 3 directly address issues that arise on the lexical level, items 4 rather addresses an issue that is situated on a conceptual level.

In this paper, we show how background knowledge in form of simple ontologies can improve text classification results by directly addressing these problems. We propose a hybrid approach for document representation based on the common term stem representation which is enhanced with concepts extracted from the used ontologies. For actual classification we suggest to use the AdaBoost algorithm which has proven to produce accurate classification results in many experimental evaluations and seems to be well suited to integrate different types of features. Evaluation experiments on three text corpora, namely the Reuters-21578, OHSUMED and FAODOC collections show that our approach leads to consistent improvements. We also show that in most cases the improvement can be traced to two distinct effects, one being situated mainly on the lexical level and the generalization on the conceptual level.

This paper is organized as follows. We introduce some preliminaries, namely the classical bag-of-words document representation and ontologies in section 2. A detailed process for compiling conceptual features into an enhanced document representation is presented in section 3. In section 4 we review the AdaBoost algorithm and its inner workings. Evaluation Measures for text classification are reviewed in section 5. In the following, experimental evaluation results of our approach are presented for the Reuters-21578, OHSUMED, and FAODOC corpora under varying parameter combinations. It turns out that combined feature representations perform consistently better than the pure term-based approach. We review related work in section 7 and conclude with a summary and outlook in section 8.

2 Preliminaries

The Bag-Of-Words Paradigm In the common term-based representation, documents are considered to be bags of terms, each term being an independent feature of its own. Let D be the set of documents and $T = \{t_1, \ldots, t_m\}$ the set of all different terms occurring in D. For each term $t \in T$ in document $d \in D$ one can define feature values functions like binary indicator variables, absolute frequencies or more elaborated measures like TFIDF [15].

Typically, whole words are not used as features. Instead, documents are first processed with stemming algorithms, e.g. the Porter stemmer for English [14]. In addition, *Stopwords*, i.e. words which are considered as non–descriptive within a bag–of–words approach, are typically removed. In our experiments later on, we removed stopwords from T, using a standard list with 571 stopwords.

Ontologies The background knowledge we have exploited is given through simple ontologies. We first describe the structure of these ontologies and then discuss their usage for the extraction of conceptual feature representations for text documents. The background knowledge we will exploit further on is encoded in a *core ontology*. For the purpose of this paper, we present only those parts of our more extensive ontology definition [2] that we need within this paper.

Definition 1 (Core Ontology). *A core ontology is a structure* $\mathcal{O} := (C, <_C)$ *consisting of a set C, whose elements are called concept identifiers, and a partial order* $<_C$ *on C, called concept hierarchy or taxonomy.*

Definition 2 (Subconcepts and Superconcepts). *If* $c_1 <_C c_2$ *for any* $c_1, c_2 \in C$*, then* c_1 *is a subconcept (specialization) of* c_2 *and* c_2 *is a superconcept (generalization) of* c_1*. If* $c_1 <_C c_2$ *and there exists no* $c_3 \in C$ *with* $c_1 <_C c_3 <_C c_2$*, then* c_1 *is a direct subconcept of* c_2*, and* c_2 *is a direct superconcept of* c_1*, denoted by* $c_1 \prec c_2$*.*

These specialization/generalization relationships correspond to what we know as is-a vs. is-a-special-kind-of, resulting in a hierarchical arrangement of concepts[1]. In ontologies that are more loosely defined, the hierarchy may, however, not be as explicit as is-a relationships but rather correspond to the notion of narrower-than vs. broader-than[2]

According to the international standard ISO 704, we provide names for the concepts (and relations). Instead of 'name', we here call them 'sign' or 'lexical entries' to better describe the functions for which they are used.

Definition 3 (Lexicon for an Ontology). *A lexicon for an ontology* $\mathcal{O}$ *is a tuple* $Lex := (S_C, Ref_C)$ *consisting of a set* S_C*, whose elements are called signs for concepts (symbols), and a relation* $Ref_C \subseteq S_C \times C$ *called lexical reference for concepts, where* $(c, c) \in Ref_C$ *holds for all* $c \in C \cap S_C$*. Based on* Ref_C*, for* $s \in S_C$ *we define* $Ref_C(s) := \{c \in C | (s, c) \in Ref_C\}$*. Analogously, for* $c \in C$ *it is* $Ref_C^{-1}(c) := \{s \in S_C | (s, c) \in Ref_C\}$*. An ontology with lexicon is a pair* $(\mathcal{O}, Lex)$ *where* $\mathcal{O}$ *is an ontology and* Lex *is a lexicon for* $\mathcal{O}$*.*

Ontologies for the experimental evaluation For the purpose of actual evaluation in the experiments, we have used three different resources, namely WordNet and the MeSH Tree Structures Ontology and the AGROVOC ontology.

Although not explicitly designed as an ontology, *WordNet* [13] largely fits into the ontology definitions given above. The WordNet database organizes simple words and multi-word expressions of different syntactic categories into so called *synonym sets (synsets)*, each of which represents an underlying concept and links these through semantic relations. The current version 2.0 of WordNet comprises a total of 115,424 synsets and 144,309 lexical index terms. The noun category, which was the main focus of our attention[3], contains nearly 70 % of the total synsets, links from 114,648 index terms to 79,689 synsets in a total of 141,690 mappings. The collection of index terms in WordNet comprises base forms of terms and their exceptional derivations. The retrieval of base forms for inflected forms is guided by a set of category-specific morphological

[1] Note that this hierarchical structure is not necessarily a tree structure. It may also be a *directed acyclic graph* possibly linking concepts to multiple superconcepts at the same time.

[2] In many settings this view is considered as a very bad practice as it may lead to inconsistencies when reasoning with ontologies. However, this problem does not arise in the context of this work.

[3] Beside the noun category, we have also exploited verb synsets, however, without making use of any semantic links,

transformations, which ensure a high precision in the mapping of word forms to index words.

The MeSH Tree Structures Ontology is an ontology that has been compiled out of the Medical Subject Headings (MeSH) controlled vocabulary thesaurus of the United States National Library of Medicine (NLM). The ontology contains more than 22,000 concepts, each enriched with synonymous and quasi-synonymous language expressions. The underlying hierarchical structure is in large parts consistent with real hypernym relations but also comprises other forms of hierarchical arrangements. The ontology itself was ported into and accessed through the Karlsruhe Ontology and Semantic Web Infrastructure (KAON) infrastructure[4].

The third ontology that has been used is the AGROVOC ontology [11], based on AGROVOC, a multilingual agricultural thesaurus[5] developed by the United Nations Food and Agricultural Organization (FAO). In total, the ontology comprises 17,506 concepts from the agricultural domain. The lexicon contains label and synonym entries for each concept in English and six additional languages. The concept hierarchy in the AGROVOC ontology is based on broader-term relationships thus not necessarily on strict superconcept relations in some cases.

3 Conceptual Document Representation

To extract concepts from texts, we have developed a detailed process, that can be used with any ontology with lexicon. The overall process comprises five processing steps that are described in this section.

Candidate Term Detection Due to the existence of multi-word expressions, the mapping of terms to concepts cannot be accomplished by querying the lexicon directly for the single words in the document.

We have addressed this issue by defining a candidate term detection strategy that builds on the basic assumption that finding the longest multi-word expressions that appear in the text and the lexicon will lead to a mapping to the most specific concepts. The candidate expression detection algorithm we have applied for this lookup procedure is given in algorithm 1[6].

The algorithm works by moving a window over the input text, analyze the window content and either decrease the window size if unsuccessful or move the window further. For English, a window size of 4 is sufficient to detect virtually all multi-word expressions.

Syntactical Patterns Querying the lexicon directly for any expression in the window will result in many unnecessary searches and thereby in high computational requirements. Luckily, unnecessary search queries can be identified and avoided through an analysis of the part-of-speech (POS) tags of the words contained in the current window. Concepts are typically symbolized in texts within *noun phrases*. By defining appropriate

[4] see `http://kaon.semanticweb.org/`
[5] see `http://www.fao.org/agrovoc/`
[6] The algorithm here is an improved version of one proposed in [18].

Algorithm 1 The candidate expression detection algorithm

Input: document $d = \{w_1, w_2, \dots, w_n\}$,
  $Lex = (S_C, Ref_C)$ and window size $k \geq 1$.
  $i \leftarrow 1$
  list L_s
  index-term s
  while $i \leq n$ **do**
    for $j = min(k, n - i + 1)$ to 1 **do**
      $s \leftarrow \{w_i \dots w_{i+j-1}\})$
      if $s \in S_C$ **then**
        save s in L_s
        $i \leftarrow i + j$
        break
      else if $j = 1$ **then**
        $i \leftarrow i + j$
      end if
    end for
  end while
  return L_s

POS patterns and matching the window content against these, multi-word combinations that will surely not symbolize concepts can be excluded in the first hand and different syntactic categories can be disambiguated.

Morphological Transformations Typically the lexicon will not contain all inflected forms of its entries. If the lexicon interface or separate software modules are capable of performing base form reduction on the submitted query string, queries can be processed directly. For example, this is the case with WordNet. If the lexicon, as in most cases, does not contain such functionalities, a simple fallback strategy can be applied. Here, a separate index of stemmed forms is maintained. If a first query for the inflected forms on the original lexicon turned out unsuccessful, a second query for the stemmed expression is performed.

Word Sense Disambiguation Having detected a lexical entry for an expression, this does not necessarily imply a one-to-one mapping to a concept in the ontology. Although multi-word-expression support and pos pattern matching reduce ambiguity, there may arise the need to disambiguate an expression versus multiple possible concepts. The *word sense disambiguation (WSD)* task is a problem in its own right [8] and was not the focus of our work.

In our experiments, we have used three simple strategies proposed in [7] to process polysemous terms:

– The ‘‘all” strategy leaves actual disambiguation aside and uses all possible concepts.
– The ‘‘first” strategy exploits WordNet’s capability to return synsets ordered with respect to usage frequency. This strategy chooses the most frequent concept in case of ambiguities.

– The ‘‘context” strategy performs disambiguation based on the degree of overlap of lexical entries for the semantic vicinity of candidate concepts and the document content as proposed in [7].

Generalization The last step in the process is about going from the specific concepts found in the text to more general concept representations. Its principal idea is that if a term like ‘beef’ appears, one does not only represent the document by the concept corresponding to ‘arrythmia’, but also by the concepts corresponding to ‘heart disease’ and ‘cardiovascular Diseases’ etc. up to a certain level of generality. This is realized by compiling, for every concept, all superconcept up to a maximal distance h into the concept representation. Note that the parameter h needs to be chosen carefully as climbing up the taxonomy too far is likely to obfuscating the concept representation.

4 Boosting

Boosting is a relatively young, yet extremely powerful machine learning technique. The main idea behind boosting algorithms is to combine multiple *weak learners* – classification algorithms that perform only slightly better than random guessing – into a powerful composite classifier.

Although being refined subsequently, the main idea of all boosting algorithms can be traced to the first practical boosting algorithm, AdaBoost [4], which we will concentrate on in this paper. AdaBoost and related algorithms have proved to produce extremely competitive results in many settings, most notably for text classification [16]. At the beginning, the inner workings of boosting algorithms were not well understood. Subsequent research in boosting algorithms made them rest on a well developed theoretical framework and has recently provided interesting links to other successful learning algorithms, most notably to Support Vector Machines, and to linear optimization techniques [12].

AdaBoost The idea behind “boosting” weak learners stems from the observation that it is usually much easier to build many simple “rules of thumb” than a single highly complex decision rule. Very precise overall decisions can be achieved if these weak learners are appropriately combined.

This idea is reflected in the output of the boosting procedure: for AdaBoost the aggregate decisions are formed in an *additive model* of the form: $\hat{f}(x) = \mathrm{sign}(\sum_{t=1}^{T} \alpha_t \, h_t(x))$ with $h_t : \mathbb{X} \to \{-1, 1\},\ \alpha_t \in \mathbb{R}$, where α_t denotes the weight of the ensemble member h_t in the aggregate decision and where the output values $\hat{f}(x) \in \{1, -1\}$ denote positive and negative predictions respectively. In such a model, AdaBoost has to solve two questions: How should the set of base hypotheses h_t be determined ? How should the weights α_t determined, i.e. which base hypotheses should contribute more than others and how much ? The AdaBoost algorithm, described in algorithm 2 aims at coming up with an optimal parameter assignment for h_t and α_t.

AdaBoost maintains a set of weights D_t over the training instances $x_1 \ldots x_i \ldots x_n$. At each iteration step t, a base classifier is chosen that performs best on the *weighted* training instances. Based on the performance of this base classifier, the final weight

Algorithm 2 The AdaBoost algorithm.

Input: training sample $\mathcal{S}_{train} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$
with $(x_i, y_i) \in \mathbb{X} \times \{-1, 1\}$ and $y_i = f(x_i)$,
number of iterations T.

Initialize: $D_1(i) = \frac{1}{n}$ for all $i = 1, \ldots, n$.

for $t = 1$ to T **do**

train base classifier h_t on weighted training set

calculate the weighted training error:

$$\epsilon_t \leftarrow \sum_{i=1}^{n} D_t(i)\, I_{y_i \neq h_t(x_i)} \tag{1}$$

compute the optimal update step as:

$$\alpha_t \leftarrow \frac{1}{2} \ln \frac{1 - \epsilon_t}{\epsilon_t} \tag{2}$$

update the distribution as:

$$D_{t+1}(i) \leftarrow \frac{D_t(i)\, e^{-\alpha_t\, y_i\, h_t(x_i)}}{Z_t} \tag{3}$$

where Z_t is a normalization factor to ensure that $\sum_{i=1}^{n} D_{t+1}(i) = 1$

if $\epsilon_t = 0$ or $\epsilon_t = \frac{1}{2}$ **then**

break

end if

end for

Result: composite classifier given by:

$$\hat{f}(x) = \operatorname{sign}\left(\hat{f}_{soft}(x)\right) = \operatorname{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right) \tag{4}$$

parameter α_t is calculated in equation (2) and the distribution weights D_{t+1} for the next iteration are updated. The weight update in equation (3) assigns higher weights to training instances that have been misclassified, while correctly classified instances will receive smaller weights in the next iteration. Thereby, AdaBoost kind of “focusing in” on those examples that are more difficult while the weight each base classifier receives in the final additive model depends on its performance on the weighted training set at the respective iteration step.

Weak Lerners for AdaBoost In theory, AdaBoost can be used with *any* base learner capable of handling weighted training instances. Although the base classifiers are not restricted to belong to a certain classifier family, virtually all work with boosting algorithms has used the very simple class of *decision stumps* as base learners. In this presentation, we focus on simple indicator function decision stumps of the form

$$h(x) = \begin{cases} c \text{ if } x^j = 1 \\ -c \text{ else.} \end{cases} \tag{5}$$

with $c \in \{-1, 1\}$. A decision stump of this form takes binary features (e.g. word or concept occurrences) as inputs. The index j identifies a specific binary feature whose presence either supports a positive classification decision, i.e. $c = 1$ or a negative decision, i.e. $c = -1$.

5 Evaluation Metrics

A standard set of performance metrics is commonly used to assess classifier performance which we will review shortly in this section.

Classification Metrics Given a set of test documents $\mathcal{S} = \{x_1, \ldots, x_n\}$ with binary labels $\{y_1, \ldots, y_n\}$ where $y_i \in \{-1, 1\}$ codes the membership in a class in question. Given further a classifier $\hat{f}$ trained on an independent training set with $\hat{f}(x) \in \{-1, 1\}$ indicating the binary decisions of the classifier. Then the test sample can be partitioned into sets $\mathcal{S} = \mathcal{S}^+ \cup \mathcal{S}^-$, i.e. the set of positive and negative test documents. These partitions can be decomposed further into $\mathcal{S}^+ = TP \cup FN$ and $\mathcal{S}^+ = FP \cup TN$ with: $TP = \{x_i \in \mathcal{S} | \hat{f}(x_i) = 1 \wedge y_i = 1\}$, $FP := \{x_i \in \mathcal{S} | \hat{f}(x_i) = 1 \wedge y_i = -1\}$, $TN := \{x_i \in \mathcal{S} | \hat{f}(x_i) = -1 \wedge y_i = -1\}$ and $FN := \{x_i \in \mathcal{S} | \hat{f}(x_i) = -1 \wedge y_i = 1\}$ called the sets of documents classified *true positive*, *false positive*, *true negative* and *false negative*, often referred to as the classification contingency table.

Based on these definitions, different evaluation measures have been defined [19]. Commonly used classification measures in text classification and information retrieval are the *classification error*, *precision*, *recall* and the F_β *measure*:

1. Classification Error

$$err(\hat{f}, \mathcal{S}) := \frac{|FP| + |FN|}{|TP| + |FP| + |TN| + |FN|} \; . \tag{6}$$

2. Precision

$$prec(\hat{f}, \mathcal{S}) := \frac{|TP|}{|TP| + |FP|} \; . \tag{7}$$

3. Recall

$$rec(\hat{f}, \mathcal{S}) := \frac{|TP|}{|TP| + |FN|} \; . \tag{8}$$

4. F_1 measure

$$F_1(\hat{f}, \mathcal{S}) := \frac{2 \, prec(\hat{f}, \mathcal{S}) \; rec(\hat{f}, \mathcal{S})}{prec(\hat{f}, \mathcal{S}) + rec(\hat{f}, \mathcal{S})} \; . \tag{9}$$

Ranking Metrics The ensemble classifiers produced by AdaBoost are capable of returning a real-valued output $\hat{f}_{soft}(x) \in [-1, 1]$. The magnitude $|\hat{f}_{soft}(x)|$ reflects the "confidence" of the classifier in a decision and allows to rank documents. Consequently, a parameterized classifier $\hat{f}_k$ can be defined that returns $\hat{f}_k(x) = 1$ if $\hat{f}_{soft}(x)$ ranks among the first k documents and $\hat{f}_k(x) = -1$ otherwise. On this basis, values for precision and recall can be calculated and tuned with respect to different values of k. When precision and recall coincide at some k, this value is called the break-even point (BEP). It can be shown that this is necessarily the case at $k = |\mathcal{S}^+|$[7].

Micro- and Macro Averaging To average evaluation results over binary classifications on the per-class level, two conventional methods exist. The *macro-averaged* figures are meant to be averages on the class level and are calculated as simple averages of the scores achieved for the different classes. In contrast, *micro-averaged* figures are computed by summing the cells of per-class contingency tables together and then computing performance scores based on these global figures. These can consequently be seen as averages on the document level.

Statistical Significance Tests Statistical significance tests are useful in order to verify to which extent the claim of an improvement can be backed by the observations on the test set. For the experiments we report in this paper, we focused on two statistical significance tests, a sign test ("S-test") and a paired t-test ("T-test") on an improvement of individual F_1 scores for the different classes that have been evaluated in each experiment described in detail in [20]. Following common statistical practice, we have required a significance level $\alpha = 0.05$ is required for claiming an improvement to be *significant*. The significance level of $\alpha = 0.01$ was used for the claim that an improvement was *very significant*.

6 Experiments

The focus of our evaluation experiments was directed towards comparing whether AdaBoost using the enhanced document representation would outperform the classical term representation.

6.1 Evaluation on the Reuters-21578 Corpus

A first set of evaluation experiments was conducted on the well-known Reuters-21578 collection. We used the "ModApte" split which divides the collection into 9,603 training documents, 3,299 test documents and 8,676 unused documents.

Experimental Setup In the first stage of the experiment, terms and concepts were extracted as features from the documents in the training and test corpus. For terms, the feature extraction stage consisted of the stages described in section 2, namely chunking, removal of the standard stopwords for English defined in the SMART stopword list and stemming using the porter stemming algorithm, resulting in a total number of 17,525 distinct term features. Conceptual features were then extracted for noun and verb phrases using WordNet as background ontology. Different sets of concept features were

[7] This follows from the fact that if there are m negative documents among the first $|\mathcal{S}^+|$ documents in the ranked list, there must also be exactly m positive examples in the remainder of the list, thus: $FP_k = FN_k = m$, which guarantees precision and recall to be equal according to the formulas given above.

extracted based on varying parameters for disambiguation strategy and maximal hypernym distance ranging from 10,259 to 27,236 distinct concept features.

In the next stage of the experiment, classification was performed using AdaBoost. We performed binary classification on the top 50 categories containing the highest number of positive training documents. The number of boosting iterations for training was fixed at 200 rounds for all feature combinations.

Results As a general finding, the results obtained in the experiments suggest that AdaBoost typically achieves better classification for both macro- and micro-averaged results when used with a combination of term-based and concept-based features. Table 1 summarizes the results of the experiments for different feature types with the best values being highlighted. The relative gains on the F_1 value, which is influenced both by precision and recall, compared to the baseline show that in all but one cases the performance can be improved by including conceptual features, peaking at an relative improvement of 3.29 % for macro-averaged values and 2.00 % for micro-averaged values. Moderate improvements are achieved through simple concept integration, while larger improvements are achieved in most cases through additional integration of more general concepts.

The results of the significance tests allow us to conclude that these improvements are significant in at least half of the cases. In general, the improvements of macro-averaged F_1 are higher than with micro-averaging which seems to suggest that the additional concepts are particularly helpful for smaller classes.

		macro-averaged (in percentages)			
Feature Type	**Error**	**Prec**	**Rec**	**F$_1$**	**BEP**
term	00.65	80.59	66.30	72.75	74.29
term & synset.first	00.64	80.66	67.39	73.43	75.08
term & synset.first.hyp5	00.60	80.67	**69.57**	74.71	74.84
term & synset.first.hyp10	00.62	80.43	68.40	73.93	**75.58**
term & synset.context	00.63	79.96	68.51	73.79	74.46
term & synset.context.hyp5	00.62	79.48	68.34	73.49	74.71
term & synset.all	00.64	80.02	66.44	72.60	73.62
term & synset.all.hyp5	**00.59**	**83.76**	68.12	**75.14**	75.55

		micro-averaged (in percentages)			
Feature Type	**Error**	**Prec**	**Rec**	**F$_1$**	**BEP**
term	00.65	89.12	79.82	84.21	85.77
term & synset.first	00.64	88.75	80.79	84.58	85.97
term & synset.first.hyp5	00.60	89.16	**82.46**	85.68	85.91
term & synset.first.hyp10	00.62	88.78	81.74	85.11	86.14
term & synset.context	00.63	88.86	81.46	85.00	85.91
term & synset.context.hyp5	00.62	89.09	81.40	85.07	85.97
term & synset.all	00.64	88.82	80.99	84.72	85.69
term & synset.all.hyp5	**00.59**	**89.92**	82.21	**85.89**	**86.44**

Table 1. Evaluation Results for Reuters-21578.

6.2 Evaluation on the OHSUMED Corpus

A second series of experiments was conducted using the OHSUMED collection, initially compiled by Hersh et al. [6]. It consists of titles and abstracts from medical journals, each being indexed with multiple MeSH descriptors. We have used the 1987 portion of the collection containing a total of 54,708 entries. Two thirds of the entries were randomly selected as training documents while the remainder was used as test set, resulting in a training corpus containing 36,369 documents and a test corpus containing 18,341 documents.

Experimental Setup Term stems were extracted as with Reuters-21578 resulting in a total number of 38,047 distinct features. WordNet and the MeSH Tree Structures Ontology were used to extract conceptual features. For WordNet, noun and verb phrases were considered while for the MeSH Tree Structures Ontology, only noun phrases were considered. For WordNet, the same disambiguation strategies were used as in the Reuters-21578 experiments. For the MeSH Tree Structures Ontology, only the "all" strategy was used due to the observation that polysemy problems occur extremely rarely with this ontology as descriptor terms are most naturally unique. For both ontologies, different degrees of depth were used for hypernym or superconcept integration, resulting in a total of 16,442 to 34,529 synset features and 11,572 to 13,663 MeSH concept features.

On the documents of the OHSUMED dataset — as on Reuters-21578 — binary classification with AdaBoost was performed on the top 50 categories that contained the highest number of positive training documents. To cope with the on average larger number of features and the much higher number of documents compared to the Reuters-21578 corpus, the number of boosting iterations for all experiments with the OHSUMED collection was set to 1000 rounds.

Results Different runs of the classification stage were performed based on the different features, leading to often substantially different results. Again, the general finding is that complementing the term stem representation with conceptual features significantly improves classification performance.

Table 2 summarizes the macro- and micro-averaged results. The relative improvements for the F_1 scores compared to the term stem baseline are depicted in figure 6.2 for WordNet as background knowledge resource. These range from about 2% to a maximum of about 7 %. The relative F_1 improvements when using the MeSH Tree Structure Ontology, were on the 3% to 5% level in all cases.

The statistical significance tests revealed that in virtually all cases, these improvements can be claimed to be significant and actually even very significant in most cases.

Again, the integration of conceptual features improved text classification results. The relative improvements achieved on OHSUMED are generally higher than those achieved on the Reuters-21578 corpus. This makes intuitively sense as the documents in the OHSUMED corpus are taken from the medical domain. Documents from this domain typically suffer heavily from the problems described in section 2, especially synonymous terms and multi-word expressions. But this is only a first effect. The even better results achieved through hypernym integration with WordNet indicate that also the highly specialized language is a problem that can be remedied through integration of more general concepts.

Feature Type	Error	macro-averaged (in percentages) Prec	Rec	F_1	BEP
term	00.53	52.60	35.74	42.56	45.68
term & synset.first	00.52	53.08	36.98	43.59	46.46
term & synset.first.hyp5	00.52	53.82	38.66	45.00	48.01
term & synset.context	00.52	52.83	37.09	43.58	46.88
term & synset.context.hyp5	**00.51**	**54.55**	**39.06**	**45.53**	**48.10**
term & synset.all	00.52	52.89	37.09	43.60	46.82
term & synset.all.hyp5	00.52	53.33	38.24	44.42	46.73
term & mesh	00.52	53.65	37.56	44.19	47.31
term & mesh.sc1	00.52	52.91	37.59	43.95	46.93
term & mesh.sc3	00.52	52.77	38.06	44.22	46.90
term & mesh.sc5	00.52	52.72	37.57	43.87	47.16

Feature Type	Error	micro-averaged (in percentages) Prec	Rec	F_1	BEP
term	00.53	55.77	36.25	43.94	46.17
term & synset.first	00.52	56.07	37.30	44.80	47.01
term & synset.first.hyp5	00.52	56.84	38.76	46.09	48.31
term & synset.context	00.52	56.30	37.46	44.99	47.34
term & synset.context.hyp5	**00.51**	**58.10**	**39.18**	**46.81**	**48.45**
term & synset.all	00.52	56.19	37.44	44.94	47.32
term & synset.all.hyp5	00.52	56.29	38.24	45.54	46.73
term & mesh	00.52	56.81	37.84	45.43	47.78
term & mesh.sc1	00.52	56.00	37.90	45.20	47.49
term & mesh.sc3	00.52	55.87	38.26	45.42	47.45
term & mesh.sc5	00.52	55.94	37.94	45.21	47.63

Table 2. Evaluation Results for OHSUMED.

A comparison between WordNet and the MeSH Descriptor Ontology is hard. On the one hand, without generalization, the domain specific MeSH Tree Structures Ontology is able to achieve slightly better results. Taking into account that the extraction was here bases solely on noun phrases and that WordNet's coverage is much broader, this is a positive surprise. On the other hand, WordNet achieves much better results when generalization comes into play. In contrast to WordNet, superconcept integration for MeSH does not really improve the results and varying levels of superconcept integration lead to similar or even worse results. Apparently, the broader-term relation of the MeSH thesaurus is indeed not well suited to improve the results. Also note that in contrast to the Reuters-21578 experiments, "context" word sense disambiguation strategy performs best in combination with hypernym integration. Apparently, it is easier to disambiguate polysemous words in the medical context.

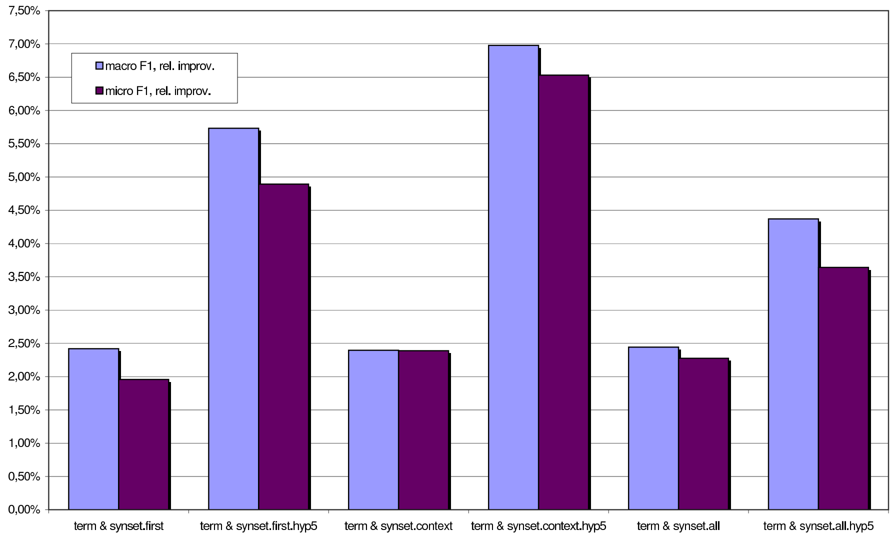


Fig. 1. Relative Improvements of F_1 Scores on OHSUMED for combined Term-Synset Features vs. Term Stems.

6.3 Evaluation on the FAODOC Corpus

The third and last series of experiments uses a collection of documents from the FAO Document Online Catalogue (FAODOC)[8], managed by the United Nations Food and Agricultural Organization. The FAODOC database houses articles and other publications from the agricultural domain together with metadata information, including subject and category elements.

Experimental Setup The FAODOC collection contains English, French and Spanish HTML documents. All documents are indexed with one or multiple category codes, each of which refers to one of 115 FAODOC subject categories (details in [11]). In the experiments, only the subset of English documents has been used where each of the categories has at least 50 positive documents. In total, this document set contains 1 501 HTML documents each indexed with one to three labels from 21 distinct subject categories. From the total number of 1 501 documents, the first 1 000 documents were used for training while the remainder of 501 documents were held out as test set.

The FAODOC dataset is very different from the other datasets encountered so far. Besides being taken from a different domain, the total number of documents is much smaller. The documents in the FAODOC dataset are typically much larger in size, ranging from 1.5 kilobytes to over 600 kilobytes, which is also reflected in the resulting feature representations with 68 608 word stems. Besides the extraction of term stems as usual, conceptual features were extracted again, this time using the AGROVOC ontology as background knowledge resource. For both types of features, the documents were

[8] see `http://www4.fao.org/faobib/index.html`

first converted from HTML to plain text, then proceeding in the same way as with the documents in the other corpora. Again as by the OHSUMED corpus only the all strategy was apply to disambiguate word stems if necessary.

As in the other experiments, each of the 21 different labels resulted in a binary classification run of its own, each time using DiscreteAdaBoost.MH was as learning algorithm with decision stump classifier based on the binary feature weights as base learners. The chosen number of 500 boosting iterations is based on a trade-off between the smaller number of training documents on the one hand and a typically larger size per document on the other. In all experiments, the results on the 21 individual labels were eventually macro- and micro-averaged.

Results Different runs of the classification stage were performed based on different features: term stems and again combinations of both types of features.

Table 3 summarizes the results of the experiments with the FAODOC for the different feature representations, evaluation metrics and averaging variants. For each performance metric, the best result is highlighted.

		macro-averaged				
Feature Type	**Error**	**Prec**	**Rec**	$\mathbf{F}_1$	**BEP**
term	06.87	45.47	27.11	33.97	36.93
term & agrovoc	**06.66**	**50.96**	28.63	36.66	39.84
term & agrovoc.sc1	06.76	49.26	27.48	35.28	39.40
term & agrovoc.sc3	06.79	49.08	**30.41**	**37.55**	**41.69**
		micro-averaged				
Feature Type	**Error**	**Prec**	**Rec**	$\mathbf{F}_1$	**BEP**
term	06.87	50.44	31.22	38.57	44.29
term & agrovoc	**06.66**	**52.91**	32.46	**40.24**	**48.01**
term & agrovoc.sc1	06.76	51.75	**32.60**	40.00	46.77
term & agrovoc.sc3	06.79	51.47	31.36	38.97	47.73

Table 3. Results on FAODOC

Again, combinations of terms and concepts as features also achieve considerable improvements over the classic term stem representation in all scores, most notably in respect to precision. Figure 2 undermines the good performance of the term and ‘agrovoc’ concept representation achieving an impressive relative improvement of 10.54 % on the macro-averaged F_1 value compared to the ‘term’ representation. The relative improvement on the micro-averaged F_1 lies at 4.33 %. Again, one observes a heavy discrepancy between the macro- and micro-averaged scores. Again, macro-averaged performance gains are higher than those for micro-averaging, which makes sense taking into account the fairly unequal category sizes. In contrast to the other experiments, the amount of deviation however varies considerably among the different feature representations. Furthermore, the question which superconcept integration depth leads to the best improvement cannot be answered easily because the effects vary between micro- and macro-averaging.

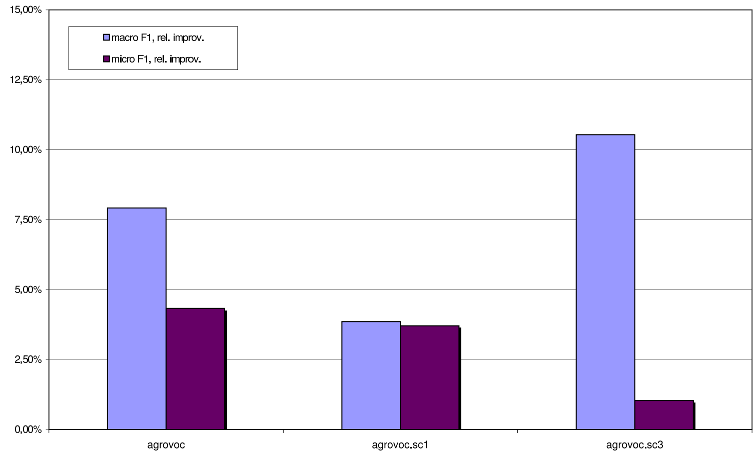


Fig. 2. Bar Chart Illustration of the Relative Improvements of F_1 Scores on all 21 FAODOC Categories for combined Term-Concept Representations vs. 'term'. All numbers are percentages.

The inconsistent results on the FAODOC collection could be attributed to the fact that random effects are much likelier compared to the other experiments as the number of training and test documents is considerably smaller. This is one reason that significance testing has not been conducted for the set of experiments with the FAODOC collection. Another reason is that the smaller number of categories would also lead to a worse reliability of the tests.

7 Related Work

Representing document content through metadata descriptions is a well-known task in the semantic web context, also known as annotation[5]. Typically, however, this is a semi-automatic task that aims at precise metadata descriptions and not at creating features for machine learning algorithms.

To date, the work on integrating semantic background knowledge into text classification or other related tasks is quite scattered. Much of the early work with semantic background knowledge in information retrieval was done in the context of *query expansion* techniques [1].

Feature representations based on concepts from ontological background knowledge were also used in text clustering settings [7] where it could be shown that conceptual representations can significantly improve text cluster purity and reduce the variance among the representations of related documents.

Recent experiments with conceptual feature representations for text classification are presented in [18]. These and other similar published results are, however, still too few to allow insights on whether positive effects can be achieved in general. In some cases, even negative results were reported. For example, a comprehensive comparison of approaches based on different word-sense document representations and different learning algorithms reported in [10] ends with the conclusion of the authors that *"the use of word senses does not result in any significant categorization improvement"*.

Alternative approaches for conceptual representations of text documents that are not based on background knowledge compute kind of "concepts" statistically. Very good results with a probabilistic variant of LSA known as Probabilistic Latent Semantic Analysis (pLSA) were recently reported in [3]. The experiments reported therein are of particular interest as the classification was also based on boosting combined term-concept representation, the latter being however automatically extracted from the document corpus using pLSA.

8 Conclusions

In this paper, we have proposed an approach to incorporate concepts from background knowledge into document representations for text document classification. A very successful ensemble learning algorithm, AdaBoost, was proposed to perform the final classifications based on the classical word vector representations and the conceptual features. Boosting Algorithms, when used with binary feature representations, scale well to a large number of dimensions that typically occur when superconcepts are used as well. At the same time, AdaBoost is capable of integrating heterogenous features that are based on different paradigms without having to adjust any parameters in the feature space representation.

Experiments on three different datasets clearly showed that the integration of concepts into the feature representation clearly improves classification results. The absolute scores achieved on Reuters and OHSUMED are highly competitive with other published results and the reported relative improvements appear to be statistically significant in most cases.

A comparative analysis of the improvements for different concept integration strategies revealed that two separate effects lead to these improvements. A first effect that can be mainly attributed to multi-word expression detection and synonym conflation is achieved through the basic concept integration. A second effect building on this initial improvement is attributed to the use of the ontology structures for generalization through hypernym retrieval and integration.

Outlook The experiments that have been conducted show that the presented approach appears to be promising in most settings. However it has also become obvious that the results depend on the specific constellation of parameters. These include — most importantly — the choice of the appropriate ontology. Further research and experiments should investigate how the specific choice and setup of the used ontologies can lead to even better results and wether other concept extraction strategies lead to a further improvement in classification performance.

It has been mentioned that feature extraction for machine learning and metadata annotation[5] have many things in common. Future work will also analyze, how results for documents that are already enriched with metadata will evolve in the classification context.

Last but not least attention should also be paid to the setup of the classification algorithm as the general nature of AdaBoost would allow to integrate more advanced weak learners. Such weak learners might also exploit background knowledge even more directly.

Acknowledgements

This research was partially supported by the European Commission under contract FP6-001765 aceMedia. The expressed content is the view of the authors but not necessarily the view of the aceMedia project as a whole.

References

1. R. C. Bodner and F. Song. Knowledge-Based Approaches to Query Expansion in Information Retrieval. In *Advances in Artificial Intelligence*. Springer, New York, NY, USA, 1996.
2. E. Bozsak et al. KAON – Towards a Large Scale Semantic Web. In *Proc. of the 3rd International Conference on E-Commerce and Web Technologies (EC-Web 2002)*, pages 304–313, Aix-en-Provence, France, 2002. LNCS 2455 Springer.
3. L. Cai and T. Hofmann. Text Categorization by Boosting Automatically Extracted Concepts. In *Proc. of the 26th Annual Int. ACM SIGIR Conference on Research and Development in Informaion Retrieval*, Toronto, Canada, 2003. ACM Press.
4. Y. Freund and R. E. Schapire. A Decision Theoretic Generalization of On-Line Learning and an Application to Boosting. In *Second European Conference on Computational Learning Theory (EuroCOLT-95)*, pages 23–37, 1995.
5. S. Handschuh and S. Staab, editors. *Annotation for the Semantic Web*. IOS Press, 2003.
6. W. R. Hersh, C. Buckley, T. J. Leone, and D. H. Hickam. Ohsumed: An Interactive Retrieval Evealuation and new large Test Collection for Research. In *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval*. ACM Press, 1994.
7. A. Hotho, S. Staab, and G. Stumme. Wordnet improves Text Document Clustering. In *Proc. of the Semantic Web Workshop of the 26th Annual International ACM SIGIR Conference*, Toronto, Canada, 2003.
8. N. Ide and J. Véronis. Introduction to the Special Issue on Word Sense Disambiguation: The State of the Art. *Computational Linguistics*, 24(1):1–40, 1998.
9. T. Joachims. Text Categorization with Support Vector Machines: Learning With Many Relevant Features. In *Proceedings of ECML-98*, 1998.
10. A. Kehagias, V. Petridis, V. G. Kaburlasos, and P. Fragkou. A Comparison of Word- and Sense-Based Text Categorization Using Several Classification Algorithms. *Journal of Intelligent Information Systems*, 21(3):227–247, 2000.
11. B. Lauser. Semi-Automatic Ontology Engineering and Ontology Supported Document Indexing in a Multilingual Environment. Master's thesis, University of Karlsruhe, 2003.
12. R. Meir and G. Rätsch. An Introduction to Boosting and Leveraging. In *Advanced Lectures on Machine Learning*, LNCS. Springer, Heidelberg, DE, 2003.
13. G. A. Miller, C. Fellbaum, D. Gross, and K. J. Miller. Introduction to WordNet: an On-Line Lexical Database. *International Journal of Lexicography*, 3(4):235–244, 1990.
14. M. F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.
15. G. Salton. *Automatic Text Processing*. Addison-Wesley Publishing Inc, Boston, MA, USA, 1989.
16. R. E. Schapire and Y. Singer. BoosTexter: A Boosting-based System for Text Categorization. *Machine Learning*, 39(2/3):135–168, 2000.

17. F. Sebastiani. Machine Learning in Automated Text Categorization. *ACM Computing Surveys*, 34(1):1–47, 2002.
18. B. B. Wang, R. I. Mckay, H. A. Abbass, and M. Barlow. A comparative study for domain ontology guided feature extraction. In *Proceedings of the 26th Australian Computer Science Conference (ACSC-2003)*, pages 69–78. Australian Computer Society, 2003.
19. Y. Yang. An Evaluation of Statistical Approaches to Text Categorization. *Information Retrieval*, 1(1-2):69–90, 1999.
20. Y. Yang and X. Liu. A re-examination of text categorization methods. In *Proceedings of SIGIR-99, 22nd ACM International Conference on Research and Development in Information Retrieval*, Berkeley, CA, 1999.

Mutual Enhancement of Schema Mapping and Data Mapping

Mingchuan Guo, Yong Yu

Department of Computer Science and Engineering
Shanghai JiaoTong University. Shanghai, China. 200030
{gmc,yyu}@apex.sjtu.edu.cn

Abstract. Schema mapping and data mapping have been two topics widely studied in traditional database related communities. Recently, with the rising interest in the semantical web, the tasks of integrating heterogeneous information sources, both in a schema and a data instance level, are becoming of more practical importance.
Current efforts at resolving these two mapping tasks have been carried out separately. In this paper, we propose a new method that simultaneously attacks these two tasks and achieves a kind of mutual enhancement between them. By applying our method to a *movie-hunting* scenario, we show that precision and recall are both quite high. Our method works well when dealing with numerical-valued attributes. We also show that this method is especially appealing from a semantic web perspective, given its assets of being relatively lightweight and easy to extend.

1 INTRODUCTION

In 2001, T.Berners-Lee proposed the idea of the semantic web [1], in which data have structures, and ontologies(schemas) describe the semantics of the structured data, thus facilitating computers to better understand and process those data.

The idea of building a semantically-rich web raises enormous interest. Yet to put the semantic web into a reality, several major problems have to be solved, with interoperability between heterogeneous information sources being one of them. Given the de-centralized nature of the web, it is certain that there will be large numbers of different information providers, each using their own ontologies(schemas), and there might be overlaps between the instance-level information they provide. It is of practical importance to enable the interoperability between different sources so as to leverage the benefits of the semantic web.

The twin sub-tasks associated with interoperability are:

1. *Schema-level* mapping: The task of discovering the correspondences between different schemas.
2. *Instance-level* mapping: The task of identifying whether two or more instances from different sources actually refer to a single entity in the real world.

For the sake of brevity, hereafter, we simply say *schema mapping* and *data mapping* when referring to the above two sub-tasks respectively.

While the ideas of schema mapping and data mapping are certainly not novelties(as will be briefly discussed in Section 2), it is the semantic web scenario, which puts special emphasis on the importance of data interoperability, that justifies the amount of attention and efforts on these two tasks nowadays. Consider the following real-life scenario:

> *Mike is interested in a movie, say, Finding Nemo. There are many movie web sites and on-line DVD dealers. Mike intends develop a more comprehensive idea of what a movie Finding Nemo is, and at the same time comparing services and prices offered by different cinemas and DVD vendors. With semantic web providing information in a machine-processable way, Mike is alleviated of the somewhat formidable task of manually searching for online movie and DVD dealers/cinemas information, and hopefully, the integration of information and comparison between DVD vendors/cinemas are also automated.*

While such a *movie-hunting* scenario may be hailed as a paradigm application of the semantic web, its feasibility hinges directly upon the two kinds of mapping tasks we just mentioned:

- *Schema mapping.* In order to make comparison meaningful, we must know the correspondences between the schema elements used by different information sources. For example, it makes no sense to compare DVD prices from source A with movie production years from source B.
- *Data mapping.* Mike should always be sure that the information he gets from any specific sources do map to his interested movie, *Finding Nemo*. In the case of a data-level mismatch, the results might be meaningless and even misleading, such as telling Mike that source A's offered price for *Finding Nemo* is lower than source B's offered price for *Finding Fish*, which is a totally different movie.

While both schema mapping and data mapping have been under research for quite some time, these efforts are being carried out somewhat independently.

In this paper,we propose a new method that performs schema mapping by utilizing data mapping information, and at the same time promoting data mapping with the aid of the (partial-)result of schema mapping. We believe, by simultaneously attacking these two tasks, this method will achieve a kind of mutual enhancement between schema mapping and data mapping. Specifically, this paper makes the following contributions:

- Proposes the idea that schema mapping and data mapping might be carried out simultaneously in a mutually-enhancing way.
  To our best knowledge, ours is the first such attempt.
- Lists some desirable characteristics that make our method especially appropriate in the semantic web context.

– Shows how some otherwise intractable mappings can be performed using our method.

The rest of the paper is arranged as follows. In Section 2, we give a brief review of related work. Section 3 explores our intuitions and rationales. It is in Section 4 that we present the mechanism of our method. Preliminary experimental results based on real-world data are given in Section 5. Section 6 provides the list of future work. We conclude this paper in Section 7.

2 RELATED WORK

2.1 Schema Mapping

AnchorPrompt[2], Cupid[3] and SimilarityFlooding[4] are well-known schema mapping methods that rely on schema information alone, such as attribute names and structural information, when performing the mapping task.

Many methods also utilize instance information. LSD[5] uses machine-learning to train a set of base learners and a meta learner. When performing the mapping task, base learners' mapping predictions are coordinated by the meta-learner to get at the final result. In [6], by borrowing ideas such as mutual information and conditional entropy from information theory, mapping is made possible when faced with opaque schema attribute names or opaque data values.

While the above methods utilize instances as a whole(serving as training data, etc), several other methods rely on single specific data instances. In [7], ontology mapping is inspired by language games[8]. Data instances known by both ontologies serve as joint attentions that form the basis of the discovery of shared concepts. The ILA system[9], which also resorts to overlapping items to derive schema mapping, discusses issues such as instance selection criteria and a mapping hypothesis evaluation mechanism.

Emphasis is also put on leveraging different kinds of information. Apart from LSD, COMA[10] also uses many matchers, each exploring different properties of schema attributes. Domain specific knowledge [5, 10–12] and historical mapping information[10–12] might also contribute to new mapping tasks.

Among all current methods, iMap[12] is distinguished in that it could also find many kinds of complex mappings. By means of deploying specific searchers, it can search and verify candidate complex mappings. iMap also has features such as the ability to explain predicted mappings. Overlapping instances are also used in iMap to discover equation-like mappings.

2.2 Data Mapping

Data mapping is studied in the database community as data cleaning and de-duplication problems. Virtually all incumbent efforts aim to find identical data instances that are in a same table. Common practices([13, 14])are to apply textual similarity functions, and compare the result with a threshold to determine whether two tuples actually refer to a single real world entity.

In the semantic web context it is more probable that we have to map data instances cross different sources. Record linkage[15] is the methodology of bringing together corresponding records from two or more files. In Doan's recent work[16], the PROM solution, a profiler-based approach that performs data mapping cross tables utilizing disjoint attributes, is proposed. In [17], R.Guha innovates by introducing the concepts of Discriminant Descriptions and a bootstrapping process.

Common purpose search engines such as google[1] provide another kind of data mapping, with users specifying the data to be mapped by providing several keywords. It is partially due to the fact that current web infrastructure doesn't support well-structured information presentation that incumbent search engineer's mapping results are usually not satisfying. However, new semantic web-inspired techniques, such as XQuery[2] and XSEarch[18], have already provided us with an inkling of how data mapping might be carried out in a more structured and semantic way.

While some don't insist on data instances to be mapped coming from a single source, all current data mapping methods pre-assume that there must be at least a partial mapping between the schemas of the involved data sources.

To our best knowledge, there has been no attempt to simultaneously consider these two mapping tasks of schema mapping and data mapping.

3 INTUITIONS AND RATIONALES

3.1 Two Practical Requirements

Our research is carried out in a semantic web-oriented way.That is, we would like our method to be especially applicable in (yet not confined to) the semantic web scenario. We discern that two practical requirements, namely, the need to be lightweight and the need to be extensible and self-improving, should be given adequate consideration for this end.

The Need to be Lightweight Given the online, decentralized nature of the semantic web in which most mapping tasks take place, mapping methods being lightweight should be considered as a necessity, rather than a feature.

Firstly, online applications demand online responses. With mapping being a frequently-required applications, users should not be kept waiting for too long for the result to come back.

Secondly, data-intensive methods might incur extremely heavy burden on the underlying networks. The perspective of the network being inundated with data transfered/exchanged for miscellaneous mapping tasks is awful. It is desirable that mapping be accomplished with just a few transfers/exchanges of information. Apart from the network overload considerations, this relaxation in

[1] http://www.google.com
[2] http://www.w3.org/TR/xquery/

information demand has the additional benefit of making the method applicable in more scenarios, where more data-intensive methods might fail simply because of the unavailability of large amounts of data.

The Need to be Extensible and Self-Improving By *extensible*, we are referring to method's ability of incorporating many/new information sources to be mapped without significantly impairing the performances[3]. When carried out in the Internet scale, mapping might well be conducted between a huge number of sources. In addition, due to the openness of the Internet environment, it is highly possible that new information sources will come into the scene continually. Semantic web-oriented mapping methods should have the extensibility to incorporate these newly-arrived sources.

Self-improving refers to the mapping methods' ability to improve its performance over time. On the one hand, *self-improving* is the natural requirement of *extensible*–it is through the improvement over time that a method is truly extensible. On the other hand, the method's being *extensible* means that it can learn from extended mapping tasks, thus making *self-improving* possible.

3.2 The Interplay of Schema Mapping and Data Mapping

To begin with, our method is based on the observation that in real-life applications such as the *movie-hunting* scenario, schemas to be mapped *do* have overlaps of data instances(See Section 5.1 for an empirical proof). In fact, so far as searching tasks are concerned, this is an ex-ante requirement.

From *Schema Mapping* To *Data Mapping* This side of the interplay is quite clear: without schema mapping, it is difficult, if not impossible, to achieve data mapping.

First of all, without schema mapping, we would run the risk of mapping instances on essentially different attributes. Consider the following situation:

source_A:

title	pro_year	dvd_year
Hero	*2000*	*2001*
Hero	*2001*	*2002*

source_B:

name	shoot_year
Hero	*2001*

Not knowing that *pro_year*, instead of *dvd_year*, actually maps to *shoot_ year*, we can't tell which movie in source_A should map to the movie in source_B.

Even if we could somewhat overcome the adverse effects of mapping instances on essentially different attributes, the prior knowledge of schema mapping will greatly reduce the computational costs—we can then just focus on the comparisons of mapped attributes, avoiding trying all pairwise combinations.

From *Data Mapping* To *Schema Mapping* If we know beforehand that certain instances make presences in both sources of the two schemas to be mapped,

[3] This is different from most current related literatures, where *extensible* means the easiness of attaching new mapping subroutines in a mapping system

then these instances could serve as the joint attention around which schema attributes relationships could be inferred. This comes in several forms:

- Direct string comparison
  - Shared instances usually have identical values for shared concepts of multiple schemas, offering clues to scheme mappings—attributes on which a single instance takes same(or highly similar) values tend to be a map.
  - String comparison also makes complex mappings such as concatenation possible and somewhat more straightforward.
- Numerical attributes mapping
  - Numerical-valued attributes often participate in equation-like complex mappings, due to different scales used(distances in meters *vs.* in kilometers), currency exchange rates(prices denoted in dollars *vs.* in pounds), etc. Such mappings can be discovered provided that we have overlapping instances from which we can suggest equations.

Consider the following situation:

source_A:

title	pro_year	dvd_year	run_time	MPAA
Hero	*2000*	*2001*	111	R
Hero	*2001*	*2002*	128	R
Shrek	*2001*	*2002*	89	PG
Matrix	*1999*	*2000*	100	G

source_B:

name	shoot_year	rate	hours	mins
Hero	*2001*	MPAA G	2	8
Shrek	*2001*	MPAA PG	1	29
Matrix	*1999*	MPAA G	2	8

Applying the above mentioned rationales, we can arrive at the following schema mapping results:

- $title = name$
- $pro\_year = shoot\_year$
- strcat("MPAA",$MPAA$) = $rate$
- $run\_time = 60{*}hours + minutes$

4 The MUTUAL ENHANCEMENT MECHANISM

The mutual enhancement process composes of 5 sub-routines:*Sel_Query_Ins()* selects query data instances from sources to be mapped; *Pro_Mapped_Ins()* proposes potential mapped instances for an incoming query instance; *Pro_Att_Mappings()* proposes attributes' mapping relationships; *GoOn()* decides whether the mapping process should go on; finally, *Decide_Schema_Mapping()* leverages different proposals to arrive at the final mapping result(s).

The *query-propose-decide* mechanism is as follows:

The input parameter *s1,s2* and *iSet1,iSet2* are the two sources' schemas and instance sets respectively.

There are many options as to the implementations of each of the 5 subroutines. Following is a brief description of our current implementations:

- *Sel_Query_Ins()*: Query instances are always sent from the source having the fewer instances[4], they are randomly selected and sent in an exponential way—starting from 5 instance, then 10,20,40... No instance is selected twice as a query instance.

[4] Information such as the size of the instance repository is usually easy to obtain

Algorithm 1 Mutual Enhancement

```
procedure MUTUAL_ENHANCEMENT(s1, s2, iSet1, iSet2)
    while GoOn() do
        qIns ← Sel_Query_Ins()
        for all ins ∈ qIns do
            mIns ← Pro_Mapped_Ins(ins)
            pMappings+ = Pro_Att_Mappings(ins, mIns)
        end for
    end while
    schemaMapping ← Decide_Schema_Mapping(pMappings)
    return schemaMapping
end procedure
```

– *Pro_Mapped_Ins()*: To be accepted as a potential map, an instance should meet the following two requirements:
  1. Having the largest number of identical values as appearing in the query instance; and
  2. Agreeing in value with the query instance on previously-proposed mapped attributes as much as possible.
– *Pro_Att_Mappings()*: All potential attribute mappings, implied by value correspondences, are proposed. A single mapping can thus be proposed many times by different (proposed)mapped instance pairs. Equation discovery begins when we have more than 3 mapped instance pairs, and is fine-tuned with newly-accepted mapped instances.
– *GoOn()*: The *query-and-propose* process stops when all instances have been sent as query instances, or when there is unlikely to be any more attribute mapping. In our preliminary implementation, we think there is unlikely to be more attribute mapping when no attribute mapping is proposed by 3 consecutive proposed mapped instance pairs.
– *Decide_Schema_Mapping()*: All proposed attribute mappings are retained as a part of the final schema mapping if they don't conflict with others. In the case of conflicts, such as *production year* being proposed to map to both *shooting year* and *release year*, the one supported by more proposals is retained.

5 EXPERIMENTATION AND DISCUSSION

5.1 Experiment Background

Our experiments were carried out with schemas and instances extracted from 6 movie web sites[5]. Since the web sites do not provide ready schemas, schemas

[5] *www.imdb.com, www.allmovie.com, www.hollywood.com, www.eonline.com, www.movies.com, and www.movieweb.com*

are manually constructed by extracting whatever might be meaningful in describing a movie from these web sites. Schema sizes vary from 12 attributes for *MOVIEWEB* to 25 attributes for *ALLMOVIES*, averaging at 20 attributes.

To assess the extent of instance overlap, we got random movies from each of the 6 web sites, and then queried the remaining 5 web sites with that randomly-chosen movie. Results show that there is a significant level of overlap between different sources, averaged at 44.3%, ranging from the low of 27.8% between *HOLLYWOOD* and *MOVIES*, to the high of 83.3% between *IMDB* and *ALLMOVIE*[6]. This is a testimony to our observation in Section 3.2 that real-life scenarios might have high levels of instance overlaps.

We tested with the 6 sources populated with beforehand-downloaded data instances, instead of using all instances available on the web sites. Thus we were able to evaluate performances under different levels of instance overlaps.

To evaluate the precision and recall of our method, we had to decide on what the correct mappings are. Volunteers were asked to list the mapping relationships, and their opinions were then leveraged to arrive at what we thought ought to be the correct schema mapping. About one half of all attributes appearing in the 6 sources participate in schema mappings, leaving attributes particular to a specific source, such as *sound mix*, unmapped. Data mappings are, however, reasonably decided by the two attributes that appear in all of the 6 movie schemas, *title* and *production year*. We assert that the combination of these two attributes serves as a primary key, and suffices for data mapping purposes.

5.2 Experiment Result

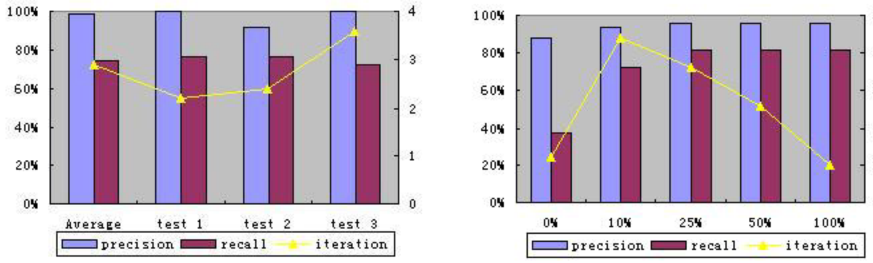
Overall Performance Fig.1 (a) presents our method's schema mapping performances, in terms of precision, recall and iteration numbers. The *Average* columns denote the average performances of all pair-wise combinations of the 6 sources. We list 3 of such combinations(see Table 1). Here we use all the instances we download from the 6 web sites, ranging from 786 to 2419 movies for each web sites respectively, and their levels of overlap roughly reflect the true situation.

It is worth noting that since results might be affected by the querying instances used, for each pair-wise combination, our method is run 5 times, and the results shown here are the mean of the 5 individual runs' results.

Precision is unanimously high, averaging at 98.3%. This is a natural outcome, in that schema mapping is discovered by comparing overlapping instances' attribute values, and it is rarely the case that many instances all take on same values for two or more different attributes so as to mislead the mapping. The ability to discover equation-like mappings further strengths the precision.

[6] Since query instances are always selected from the smaller repository, the overlap ratio we use here is defined as:

$$\frac{\text{number of overlapping instances}}{\text{size of the smaller repository}}$$



(a) (b)

Fig. 1. (a)Overall Performances (b)Stability Analysis

Table 1. Selected Results

	sourc1	size1	source2	size2	overlap	precision	recall	iteration
test 1	IMDB	2419	ALLMOVIE	896	83.3%	100%	76.4%	2.2
test 2	EONLINE	1740	MOVIES	1320	60.8%	94%	81.8%	2.4
test 3	MOVIEWEB	1139	HOLLYWOOD	786	41.4%	100%	72.3%	3.6

While precisions of test 1 and test 3 are both 100%, it is only 94% for test 2. This is because that while both *EONLINE* and *MOVIES* have a *Release Date* attribute(which forms a map), the first schema also has an *In Theater Date* attribute that usually takes on the same value as *Release Date*. Thus this attribute is often proposed, wrongly, to be mapped to *Release Date* attribute of *MOVIES* by the *Pro_-Att_Mappings()* sub-routine. If proposed the same number of times as with the correct mapping, the *Decide_Schema_Mapping()* sub-routine will just retain this false mapping, resulting in the lower precision.

Compared with precision, recall for schema mapping is relatively low, averaging at 74.6%. This can be explained by two major causes:

1. Some shared attributes, such as *review* and *movie plot*, are unlikely to take on same values for an identical movie in different sources. Our current implementation just can't find mappings of these attributes.
2. Information on some films might be incomplete. Querying instances all having no value on a particular attribute might probably miss the mapping of that attribute(Following we will give a detailed illustration).

It is somewhat difficult to evaluate the data mapping part of our method. For one thing, so far as the *movie-hunting* scenario is concerned, there are only negligible increases in data mapping in terms of the ratio of the actual mapped instance appearing in the first 3 proposed mapped instances. This is due to the fact that our testing data offers few settings, like the one discriminating between *pro_year* and *rel_year* elaborated in 3.2, that could vividly show the enhancement schema mapping brings to data mapping. For another thing, a major benefit that schema mapping result brings to data mapping is the computational efforts saved, which is hard to quantify.

Stability of the method To assess how our method is affected by the instance overlap level, we tested with instance overlap ratios being 10%, 25%, 50% and 100%. We manipulated the two sides (*EONLINE* and *MOVIES*) to get the desired overlapping ratio, while the absolute size of the two sources were kept unchanged(500 instances for each source). The result, shown in Fig.1(b), is the mean of 5 individual runs.

It can been seen that so far as there *is* instance overlap, schema mapping results in terms of precision(94% $\sim$ 96%) and recall(72.3% $\sim$ 81.9%) are rather stabilized.

The iteration numbers are, understandably, inversely related to the instance overlap level. When there is an 100% overlap, the 5 query instances of the first round suffices the schema mapping task. When overlap ratio is low, it takes more rounds so that enough mapped instances have been identified and these pair-wise instances no longer suggest new schema mappings. Our implementation of sending query instances in an exponential way, while avoiding blindly sending unnecessary large numbers of instances, ensures that the iteration won't be too prolonged. Results show that it works quite well. When the overlap ratio is only 10%, an average of 4.4 iterations is all it takes.

This verifies that our method is quite lightweight. Even when there is relatively low instance overlaps, our method is quick at arriving at the final results, with high precisions and recalls.

An interesting discovery is that, even when there is no instance overlap, some schema mapping still could be found. Some attributes, such as *rating* and *genre*, can only take a quite limited number of different values, so it is highly possible that different instances from two sources take same values on such attributes. Such instances are then proposed as being identical, resulting in the mapping of these attributes. We call such mapping *twisted* mapping. Results also show that when there is no instance overlap, the iteration ends rather promptly(1.2 rounds in our test). This is explained by the fact that counting on attributes such as *rating* as instance mapping criteria usually turns out many *twistedly-mapped* instances, yet these instances could rarely come up with further attribute mappings. So the iteration ends rather promptly.

Table 2 lists each run's specific results between *EONLINE* and *MOVIES* so as to assess our method's sensitivity to different query instances used.

Table 2. Results of Individual Runs

	precision	recall	iteration
1st run	90%	63.6%	2
2nd run	90%	81.8%	3
3rd run	90%	72.7%	2
4th run	100%	81.8%	3
5th run	90%	81.8%	2

The 1st and 3rd runs end up missing schema mapping relationships that other runs do find. The missed mappings are for attributes *running time* and *release date*. A closer look at the instances suggests that many instances of *EONLINE* don't have values on these 2 attributes. If none of the instances involved in the mapping process has values on such attributes, there is no way to figure out these attribute mappings. The number of iteration is varied somewhat for the same reason: If, repeated, incomplete instances are involved, then the iteration ends sooner.

Each of the 4 runs that has precision of 90% turns out 10 attribute mappings, one of which is the false mapping of *Release Date* and *In Theater Date*.

Extendable and Self-Improving While currently, experiments to test the extendable and self-improving characteristics of our method is still under way, here we'd like to show the major ideas of achieving extendability and self-improvement.

As we have shown, our method's performances, especially in terms of the iteration numbers, is somewhat decided by how we select query instances. The more overlapped instances in the query instances, the better. Instead of selecting query instances randomly, we think it is possible to learn from previous mapping tasks about which instances tend to be shared among different sources. For example, previously successfully mapped instances could be recorded so that they could be used as query instances in future mapping tasks, with great chances of reducing the number of iterations.

Another kind of self-improvement stems from our observation that a pair of mapped instances may have different yet similar values on mapped attributes. A movie may be have *"comedy"* as value for *genre* attribute in one web site, yet in another web site, it may be labelled as *"humor"* on attribute *style*. Data mapping may be hampered by such different albeit same-meaning values. If we know that the two moives in fact refer to a single movie, and that attributes *genre* and *style* are mapped, then we can conclude that value *"comedy"* and *"humor"* probably has the same meaning. This information in term may be helpful for future data mapping tasks.

6 FUTURE WORK

At the time of this writing, we have only conducted some preliminary experiments. Following are several experiments we are contemplating to carry out:

- To analyze how incorrect/imcomplete information might affect the performances. Our goal is to alleviate their adverse effects, and to further study how *twisted mapping* could be of help to suggest correct mappings;
- To reduce the number of exchanges of instances so as to make our method more lightweight.
- To further test the extendability and self-improvement of our method.

Currently, when proposing attribute mappings, previously-proposed mapping information is not taken into account. Utilizing such information in an earlier stage, in stead of at the final stage of deciding which mappings to retain, might contribute to performances.

Our method somewhat precludes the discovery of non-leaf attribute mappings, in that non-leaf attributes don't directly take on values. Future efforts are needed to work around this drawback.

While we have shown that schema mapping and data mapping can be carried out in a mutually-enhancing way, we admit that our current implementation is biased toward schema mapping. In future work, we will pay more attention to how data mapping could benefit from schema mapping.

7 CONCLUSIONS

In this paper, we proposed a method that, by simultaneously attacking the twin problems of schema mapping and data mapping, achieves a kind of mutual enhancement between them. Our method is based on the observation that in real-life scenarios, instance-level overlapping level tend to be high. We have shown that this method is well-suited for the semantic web scenario in that it is relatively lightweight and extensible. Preliminary experimental results turned out to rather inspiring. Ongoing efforts are being made to achieve better performances.

References

1. Berners-Lee, T., Hendler, J., Lassila, O.: The semantic web. Scientific American (2001) 279
2. Musen, M.A., Noy, N.F.: Anchor-prompt: Using non-local context for semantic matching. (2001)
3. Madhavan, J., Bernstein, P.A., Rahm, E.: Generic schema matching with cupid. In: Proc. 27th VLDB. (2001) 49–58
4. Melnik, S., Garcia-Molina, H., Rahm, E.: Similarity Flooding: A Versatile Graph Matching Algorithm and its Application to Schema Matching. In: Proc. 18th ICDE, San Jose, CA (2002)
5. Doan, A., Domingos, P., Halevy, A.Y.: Reconciling schemas of disparate data sources: a machine-learning approach. SIGMOD (2001) 509–520
6. Kang, J., Naughton, J.F.: On schema matching with opaque column names and data values. In: Proc. of the SIGMOD 2003. (2003) 205–216
7. Wiesman, F., Roos, N., Vogt, P.: Automatic ontology mapping for agent communication. In: Proceedings of the 1st International Joint Conference on Autonomous Agents and Multiagent Systems. (2002) 563–564
8. Ma, C., Steels, L., Vogt, P., Amyot, R., Press, T.M.: Grounding adaptive language games in robotic agents. (2001)
9. Etzioni, O.: Category translation: Learning to understand information on the internet. (2000)
10. Do, H.H., Rahm, E.: Coma–a system for flexible combination of schema matching approaches. In: Proc. of the 28th VLDB. (2002) 610–621
11. Madhavan, J., Bernstein1, P., Chen, K., Halevy, A., Shenoy, P.: Matching schemas by learning from a schema corpus. In: Proceedings of of the IJCAI-03 Workshop on Information Integration. (2003)
12. Dhamankar, R., Lee, Y., Doan, A., Halevy, A., Domingos, P.: imap: Discovering complex semantic matches between database schemas. In: Proceedings of the ACM SIGMOD Conference on Management of Data. (2004)
13. Cohen, W.W.: Integration of heterogeneous databases without common domains using queries based on textual similarity. In: Proceedings of ACM SIGMOD 1998. (1998) 021–212
14. Sarawagi, S., Bhamidipaty, A.: Interactive deduplication using active learning. In: Proceedings of the 8th ACM SIGKDD Conference. (2002)
15. E.Winkler, W.: The state of record linkage and current research problems. In: Proceedings of of the Survey Methods Section. (1999) 73–79
16. Doan, A., Lu, Y., Lee, Y., Han, J.: Object matching for information integration: A profiler-based approach. In: Proceedings of the IJCAI-03 Workshop on Information Integration on the Web. (2003)
17. R.Guha: Object co-identification on the semantic web. In: Proceedings of the 8th ACM SIGKDD Conference). (2002) 350–359
18. Cohen, S., Mamou, J., Kanza, Y., Sagiv, Y.: Xsearch: A semantic search engine for xml. In: Proceedings of of the 29th VLDB Conference, Berlin, Germany. (2003)