Class Extraction from the World Wide Web

Ana-Maria Popescu **Alexander Yates** **Oren Etzioni**
University of Washington
Department of Computer Science and Engineering
Box 352350
{amp, ayates, etzioni}@cs.washington.edu

Abstract

In previous work we introduced KNOWITALL a data-driven, Web-based information extraction system. This paper focuses on the task of automatically extending the system's initial ontology by extracting subclasses of given general classes and by discovering other closely related classes. We first show that the basic KNOWITALL model can be easily extended to accommodate high-precision *subclass extraction (SE)* and that subclass knowledge can lead to increases by a factor of 10 in the number of class instances extracted by KNOWITALL. Second, we introduce two methods that on average increase the number of subclasses extracted by the SE module itself by a factor of 3. Next, we show how the model can be extended to achieve high-quality *related class extraction (RCE)*, which consists of discovering new classes closely related to the set of known classes. Of the classes discovered by KNOWITALL, in the test domains of Geography and Computers, 79.5% were bona-fide related classes.

1. Introduction and Motivation

(Etzioni *et al.* 2004a) introduced KNOWITALL, an open-ended information extraction (IE) system that automatically gathers large quantities of facts from the Web. The basic version of the system contains an Extractor that collects information using *domain-independent extraction patterns* (Hearst 1992). The system's Assessor verifies the truth of the extracted candidate instances using Web-based Pointwise Mutual Information (PMI) statistics (Turney 2001). Currently, KNOWITALL starts with a fixed ontology and extracts instances for the given classes. In this paper, we focus on automatically extending the system's ontology by automatically extracting subclasses of and classes related to those in the original ontology. For example, given the class Scientist, our system extracted the subclasses shown in Table 1.

We expect automatic subclass extraction to be particularly valuable for general, naturally decomposable classes such as Plant, Animal, or Machine where text usually refers to their named subclasses (*e.g.*, Flower, Mammal, Computer). To use the psychological terminology of (Rosch *et al.* 1976), we believe that text on the Web refers to instances as elements of "basic level" categories such as Flower much more frequently than as elements of super-ordinate ones such as Plant.

biologist	zoologist
astronomer	meteorologist
mathematician	economist
geologist	sociologist
anthropologist	pharmacist
psychologist	climatologist
paleontologist	neuropsychologist
engineer	microbiologist

Table 1: **Sample subclasses found for the class Scientist.**

Our main contributions are as follows:

a) We explain how the KNOWITALL architecture can be extended elegantly to accommodate *subclass extraction (SE)* and show that our SE methods have high precision (most of the extracted subclasses are bona-fide subclasses of the appropriate class), find subclasses missing from WordNet, and substantially increase KNOWITALL's recall of class instances.

b) We investigate methods which, on the average, improve the *recall* of the basic subclass extraction module by a factor of 3. The *recall* of a method refers to the percentage of existent correct subclasses for a given class retrieved by that method. Because the correct set of subclasses for a given class on the Web is unknown, it is approximated by the union of the correct subclass sets corresponding to each evaluated method.

c) We show how are methods can be extended to the task of *related class extraction (RCE)*. We use Web-based PMI statistics in order to discover new concepts and measure their degree of relevance to a given domain; we show promising results in two different domains, Geography and Computers.

The paper starts by with a brief description of the KNOWITALL architecture; section 3 describes our basic subclass extraction module and its experimental evaluation. Section 4 describes our techniques for improving the recall of the subclass extraction module and reviews their performance, while section 5 describes on-going work in related class extraction. Section 6 reviews related ontology-extension and semantic lexicon acquisition work; finally, section 7 contains our conclusions and directions for future research.

2. The KNOWITALL System

KNOWITALL is an autonomous, domain-independent system that extracts facts, concepts, and relationships from the Web (Etzioni *et al.* 2004a). The only domain-specific input to KNOWITALL is a set of classes and relations that constitute its *focus*. KNOWITALL is also given a set of generic extraction patterns, but these are domain independent.

KNOWITALL begins with a bootstrap learning phase where it automatically instantiates its set of generic extraction patterns into class-specific extraction rules for each of the classes in its *focus*. KNOWITALL uses these rules to find a set of seed instances and uses the seeds to estimate conditional probabilities that are used by its Assessor module. After this bootstrap phase, the Extractor module begins to extract candidate instances from the Web, and the Assessor assigns a probability to each candidate.

KNOWITALL uses statistics computed by querying search engines to assess the likelihood that the Extractor's conjectures are correct. Specifically, the Assessor uses a form of *point-wise mutual information* (PMI) between words and phrases that is estimated from Web search engine hit counts in a manner similar to Turney's PMI-IR algorithm (Turney 2001). The Assessor computes the PMI between each extracted instance and multiple, *automatically generated discriminator phrases* associated with the class (such as "city of" for the class `City`)

KNOWITALL's domain-independent extraction rules have limited recall; in (Etzioni *et al.* 2004b) we compared several methods of increasing the recall of the system, one of which is *subclass extraction (SE)*. (Etzioni *et al.* 2004b) contained a one-page sketch of an early version of SE and reported on its positive impact on the recall of instances in three classes. This paper describes key enhancements to SE that substantially improve its performance, and analyzes its precision and recall on a wide range of classes; it also introduces our work on related class extraction (RCE).

3. Subclass Extraction (SE)

As it turns out, subclass extraction can be achieved by a recursive application of KNOWITALL's main loop (with some important extensions). In the following, we describe the basic subclass extraction method (SE_{base}), discuss two variations (SE_{self} and SE_{iter}) aimed at increasing SE's recall and present encouraging results for a number of different classes.

Extracting Candidate Subclasses

In general, the SE_{base} extraction module has the same design as the original KNOWITALL extraction module. Its input consists of domain-independent extraction rules for generating candidate terms, for which matches are found on the Web. The generic rules that extract instances of a class will also extract subclasses, with some modifications. To begin with, the rules need to distinguish between instances and subclasses of a class. Rules for instances already contain a proper noun test (using a part-of-speech tagger and a capitalization test). Rules for extracting subclasses instead check that the extracted noun is a common noun (i.e., not capitalized). While these tests are heuristic, they work reasonably well in practice, and KNOWITALL also falls back on its Assessor module to weed out erroneous extractions.The patterns for our subclass extraction rules appear in Table 2. Most of our patterns were simple variations of well-known ones in the information-extraction literature (Hearst 1992). C_1 and C_2 denote known classes and "CN" denotes a common noun or common noun phrase. Note that the last two rules can only be used once two subclasses of the class have already been found. Also, when we perform subclass extraction in a given context, the search engine queries contain a relevant keyword together with the instantiated extraction rule (for instance, "pharmaceutical" in the case of the Pharmaceutical domain).

Pattern	Extraction
C_1 {",") "such as" CN	$isA(CN, C_1)$
"such" C_1 "as" CN	$isA(CN, C_1)$
CN {",") "and other" C_1	$isA(CN, C_1)$
CN {",") "or other" C_1	$isA(CN, C_1)$
C_1 {",") "including" CN	$isA(CN, C_1)$
C_1 {",") "especially" CN	$isA(CN, C_1)$
C_1 "and" CN	$isA(CN, class(C_1))$
C_1 {",") C_2 {",") "and" CN	$isA(CN, class(C_1))$

Table 2: **Rules for Subclass Extraction**.

Assessing Candidate Subclasses

The SE_{base} Assessor decides which of the candidate subclasses from the SE_{base} Extractor are correct. First, the Assessor checks the morphology of the candidate term, since some subclass names are formed by attaching a prefix to the name of the class (e.g., "microbiologist" is a subclass of "biologist"). Then, if the subclass extraction is done in a context-independent manner, the Assessor checks whether a subclass is a hyponym of the class in WordNet and if so, it assigns it a very high probability.

The rest of the extractions are evaluated in a manner similar to the instance assessment in KNOWITALL (with some modifications). The Assessor computes co-occurrence statistics of candidate terms with a set of class discriminators. Such statistics represent features combined a naive Bayesian probability update (Etzioni *et al.* 2004b). The SE_{base} Assessor uses a training set in order to estimate the conditional probabilities of observing a feature if an example is or is not a correct subclass. The Assessor ranks the set of proposed subclasses based on the number of matching extraction rules and then re-ranks the top n (n = 40) according to their average PMI score with respect to the set of discriminators for the class. The top $k = 10$ subclass candidates that also co-occur with every discriminator in the set represent the set of positive examples for the class. The negative examples for each class are collected from among the positive examples for the other classes in the ontology.

Experimental Results

Before presenting our experimental results, we need to introduce two key distinctions. We distinguish between finding subclasses in a *context-independent* manner versus finding

Method	**Scientist**				**Film**			
	Precision	Recall	NW	Total	Precision	Recall	NW	Total
SE_{base}	0.91	0.28	0.08	11	1.0	0.36	0.5	8
SE_{self}	0.87	0.69	0.15	27	0.94	0.77	0.82	**17**
SE_{iter}	0.84	0.74	0.17	**29**	0.93	0.68	0.8	16

Table 3: **Results of the 3 Subclass Extraction methods (SE_{base}, SE_{self} and SE_{iter}) for the Scientist and Film classes. For each method, we report on the precision and recall numbers. We also include Total, the number of *correct* subclas ses, and NW, the proportion of the correctly identified subclasses missing from WordNet.**

Method	**People**				**Organization**				**Product**			
	Precision	Recall	NW	Total	Precision	Recall	NW	Total	Precision	Recall	NW	Total
SE_{base}	1.0	0.28	0.07	14	0.92	0.20	0.09	11	0.88	0.44	1.0	31
SE_{self}	1.0	0.86	0.02	42	0.87	0.84	0.36	47	0.86	0.74	1.0	51
SE_{iter}	0.95	0. 94	0.02	**46**	0.89	0.95	0.22	**52**	0.84	0.88	1.0	**62**

Table 4: **Results of the 3 Subclass Extraction methods (SE_{base}, SE_{self} and SE_{iter}) for the People, Product and Organization classes in the context of the Pharmaceutical domain. For each method, we report on the precision and recall numbers. We also include Total, the number of *correct* subclasses and NW, the proportion of the correctly identified subclasses missing from WordNet.**

subclasses in a *context-dependent* manner. The term *context* refers to a set of keywords provided by the user that suggest a knowledge domain of interest (*e.g.*, the pharmaceutical domain, the political domain, etc.). In the absence of a domain description, KNOWITALL finds subclasses in a context-independent manner and they can differ from context-dependent subclasses. For instance, if we are looking for any subclasses of Person (or People), Priest would be a good candidate. However, if we are looking for subclasses of Person (or People) in a Pharmaceutical context, Priest is probably not a good candidate, whereas Pharmacist is. We also distinguish between *named subclasses* and *derived subclasses*. *Named subclasses* are represented by novel terms, whereas *derived subclasses* are phrases whose head noun is the same with the name of the superclass. For instance, *Capital* is a named subclass of *City*, whereas *European City* is a derived subclass of *City*. While derived subclasses are interesting in themselves, we focus on the extraction of named subclasses, as they are more useful in increasing KNOWITALL's instance recall. The reason is that extraction rules that use derived subclasses tend to extract a lot of the same instances as the rules using the name of the superclass (see Table 2).

We now turn to our experimental results. We have evaluated our basic subclass extraction method in two different settings.

a) **Context-independent SE** First, we chose three classes, Scientist, City and Film and looked for context-independent subclasses using the SE_{base} approach described above. SE_{base} found only one named subclass for City, "capital", which is also the only one listed in the WordNet hyponym hierarchy for this class. SE_{base} found 8 correct subclasses for Film and 11 for Scientist - this confirmed our intuition that subclass extraction would be most successful on general classes, such as Scientist and least successful on specific classes such as City. As shown in Table 3, we have evaluated the output of SE_{base} along four metrics: precision, recall, total number of correct subclassses and proportion of (correct) found subclasses that do not ap pear in WordNet. As we can see, SE_{base} has high-precision but relatively low recall, reflecting the low recall of our domain-independent patterns.

b) **Context-dependent SE** A second evaluation of SE_{base} (detailed in Table 4) was done for a context-dependent subclass extraction task, using as input three categories that were shown to be productive in previous semantic lexicon acquisition work (Phillips & Riloff 2002): People, Products and Organizations in the Pharmaceutical domain. SE_{base} exhibits the same high-precision/low-recall behavior we noticed in the context-independent case. We also notice that most of the subclasses of People and Organizations are in fact in WordNet, whereas none of the found subclasses for Products in the Pharmaceutical domain appears in WordNet.

Next, we investigate two methods for increasing the recall of the subclass extraction module.

4. Improving Subclass Extraction Recall

Generic extraction rules have low recall and do not generate all of the subclasses we would expect. In order to improve our subclass recall, we add another extraction-and-verification step. After a set of subclasses for the given class is obtained in the manner of SE_{base}, the high-recall enumeration rules in Table 2 are seeded with known subclasses and extract additional subclass candidates. For instance, given the sentence "Biologists, physicists and chemists have convened at this inter-disciplinary conference.", such rules identify "chemists" as a possible sibling of "biologists" and "physicists". The candidate subclass sets extracted in this fashion contain reliable seed subclasses (whose probability was already determined by the Naive Bayes Assessor), terms previously classified as negative examples and novel terms. We experiment with two methods, SE_{self} and SE_{iter} in order to assess the extractions obtained at this step.

a) SE_{self} is a simple assessment method based on the empirical observation that an extraction matching a large number of different enumeration rules is likely to be a good subclass candidate. We have tried to use the enumeration rules directly as features for a Naive Bayes classifier, but the very nature of the enumeration rule instantiations ensures that positive examples don't have to occur in any specific instantiation, as long they occur frequently enough. We simply convert the number of different enumeration rules matched by each example and the average number of times an example matches its corresponding rules into boolean features (using a learned threshold). Since we have a large quantity of unlabeled data at our disposal, we estimate the thresholds and train a simple Naive-Bayes classifier using the *self-training* paradigm (Nigam & Ghani 2000), chosen as it has been shown to outperform EM in a variety of situations. At each iteration, we label the unlabeled data and retain the example labeled with highest confidence as part of the training set. The procedure is repeated until all the unlabeled data is exhausted. The extractions whose probabilities are greater than 0.8 represent the final set of subclasses (since subclasses are generally used by KNOWITALL for instance extraction, bad subclasses translate into time wasted by the system and as such, we retain only candidate subclasses whose probability is relatively high).

b) SE_{iter} is a heuristic assessment method that seeks to adjust the probabilities assigned to the extractions based on *confidence scores* assigned to the enumeration rules in a recursive fashion. The *confidence score* of a rule is given by the average probability of extractions matched by that rule. After rule confidence scores have been determined, the extraction matching the most rules is assigned a probability $p = \frac{c(R1)+c(R2)}{2}$, where $R1$ and $R2$ are the two matching rules with highest confidence scores. The rule confidence scores are then re-evaluated and the process ends when all extractions have been assigned a probability. This scheme has the effect of clustering the extractions based on the rules they match and it works to the advantage of good subclasses that match a small set of good extraction rules. However, as we will later see, this method it is sensitive to noise. As in the case of SE_{self}, we only retain the extractions whose probability is greater than 0.8.

Experimental Results

We evaluated the methods introduced above on two of the three context-independent classes (Scientist and Film) in Table 4.[1] We also evaluated the methods and on all three Pharmaceutical domain classes (People, Product, Organization) in Table 3. We found that both SE_{self} and SE_{iter} significantly improved upon the recall of the baseline method: for both, this increase in recall is traded for a loss in precision. SE_{iter} has the highest recall, at the price of an average 2.3% precision loss with respect to SE_{base}. In the future, we will perform additional experiments to assess which one of the two methods is less sensitive to noise, but based upon inspection of the test set and the behavior of both methods, SE_{self} appears more robust to noise than SE_{iter}.

Another potential benefit of subclass extraction is an increase in the number of class instances that KNOWITALL is able to extract from the Web. In the case of the Scientist class, for example, the number of scientists extracted by KNOWITALL at precision 0.9 increased by a factor of 10. SE_{iter} was used to extract subclasses and add them to the ontology. We do not see this benefit for classes such as City, where most of the extracted subclasses are derived subclasses (e.g., '"European City"). The reason is that extraction rules that use derived subclasses tend to extract a lot of the same instances as the rules using the name of the superclass (see Table 2).

Discussion

It is somewhat surprising that simple features such as the number of rules matching a given extraction are such good predictors of a candidate representing a subclass. We attribute this to the redundancy of Web data (we were able to find matches for a large number of our instantiated candidate rules) and to the semantics of the enumeration patterns. The subclass sets from SE_{self} and SE_{iter} contain many of the same candidates, although SE_{iter} typically picks up a few more.

Another interesting observation is that the different sets of extracted subclasses have widely varying degrees of overlap with the hyponym information available in WordNet. In fact, all but one of the subclasses identified for People are in WordNet whereas none of those Products appear there (*e.g.*, Antibiotics, Antihistamines, Compounds, etc.). In the case of Organizations, there is a partial overlap with WordNet and it is interesting that terms that can refer both to a Person and an Organization ("Supplier", "Exporter" etc.) tend to appear only as subclasses of Person in WordNet, although they are usually found as subclasses of Organizations by KNOWITALL's subclass extraction methods.

5. Related Class Extraction (RCE)

In the related class extraction task, KNOWITALL starts with a set of predicates (and optionally, a set of instances and/or keywords) relevant to a specific topic and then automatically identifies and explores new *related concepts*, that is, concepts pertaining to the same topic. Using the KNOWITALL terminology, the initial set of predicates/instances/keywords represents the system *information focus* and the task is identifying other predicates relevant to the current focus.

For instance, in the Computer domain, the information focus might contain four classes: "Computer", "Chip", "Monitor" and "Disk", together with a few instances of each. We would like KNOWITALL to automatically identify "Modem", "Drive", and other such conceptually related classes. The challenge is to remain in focus; for instance, in the case of the Geography domain it is easy enough to drift into a related area, such as Economy or Politics. The RCE (related class extraction) module proceeds in an iterative fashion: at each iteration, it uses a sample of the known domain-specific classes to instantiate a set of extraction patterns and then assesses the relevance of the extractions obtained in this man-

[1] We didn't have enough subclasses to instantiate enumeration patterns for City as SE_{base} only identified one named City subclass.

ner with respect to the information focus. The iterative procedure ends when its signal-to-noise ratio drops below a set threshold.

Extracting Candidate Classes

In general, the RCE extraction module has the same design as the original KNOWITALL extraction module. Unlike the rules for subclass extraction, the rules for related class extraction also make use of instance information. For example, given the class City and a few seed instances (Paris, London, etc.), we can find terms potentially related to City using rules instantiated with either the name of the class ("city and other C ", where C is a candidate term) or the name of a seed instance ("Paris and other C", where C is a candidate term).

Assessing Candidate Classes

The RCE Assessor takes as input the set of candidate concepts found in the previous step and computes the *relevance* of each class name with respect to the given information focus IF. The *relevance* of a new class name is measured on the basis of web-scale PMI statistics of the candidate name with IF terms. The relevance measure attempts to eliminate candidate terms that are not domain-specific in an inexpensive manner (more advanced relevance measures, based on identifying probable relationships between a candidate term and known domain-specific terms, are also more expensive).

Given a domain specific term T and a candidate class C, we compute C's semantic similarity to T using the PMI score below:

$PMI(C,T) = \frac{|Hits(C,T)|}{|Hits(C)|*|Hits(T)|}$.

Given a set of terms T_s describing a domain-specific class T (class name, names of class instances) and a new candidate class C, we first compute the similarity between C and each term describing T; we then average the resulting scores to get a measure of the similarity between C and T_s:

$PMI_{avg}(C,T_s) = \frac{\sum PMI(C,e(T_s))}{|T_s|}$,

where $e(T_s)$ is some element in T_s.

$PMI_{avg}(C,T_s)$ measures the average relevance of C to the core class T using all the available information about T. These final scores are used to determine whether a given class C is relevant or not to the given information focus. (Etzioni *et al.* 2004a) evaluated a number of methods for converting PMI scores into suitable classification features for a similar task. Inspired by these evaluation results, we use a set of learned thresholds in order to convert the relevance scores into boolean features. The features are then combined using a Naive Bayes classifier to predict whether the candidate name is relevant with respect to the given IF. The verifier considers all candidates whose relevance probability is greater than 0.8 to be relevant - we refer to this basic RCE extraction and assessment procedure as RCE_{base}.

Increasing Related Class Extraction Recall

In order to increase the recall of the basic RCE module, we experimented with two modified RCE methods, RCE_{self} and RCE_{iter}, with the same structure as the ones used to increase the recall of the SE component. These are versions of RCE_{base} in which the set of good extractions computed at the end of each iteration in the manner described above is augmented using enumeration rules. Once new extractions are obtained in this manner, they are not evaluated with respect to the information focus IF (since this is expensive in terms of search-engine queries and a lot of good class names do not co-occur enough times with IF terms), but they are evaluated with respect to the enumeration rules that extracted them (as described in section 4).

keyboard	drive
modem	mouse
circuit	hardware
battery	laptop
memory	microprocessor
peripheral	semiconductor
bus	server

Table 6: **Sample classes discovered by the baseline RCE method (RCE_{base}) in the Computer domain.**

Experimental Results

We tested our methods for extracting classes relevant to a given domain description on two domains: Geography and Computers. All RCE methods started with basic initial knowledge of each domain in the form of four seed classes: Cities, States, Countries, and Rivers for the Geography domain, and Computers, Chips, Monitors, and Disks for the Computer domain. Due to time constraints, we first performed two iterations of the RCE_{base} method in order to obtain a non-trivial set of good extractions (18 for the Geography domain and 32 for the Computer domain). We then performed a third iteration, this time comparing the simple RCE_{base} method with its modified versions, RCE_{self} and RCE_{iter}. Table 5 reflects the results after the third iteration and Table 6 shows some examples of newly identified classes for the Computer domain. We report on the precision and recall metrics for our three methods— their performance was evaluated by hand-labeling the extracted terms. We also include the total number of relevant classes extracted by each method.

As we had hoped, RCE_{self} and RCE_{iter} led to an increase in RCE recall. We noticed that the RCE_{iter} led to the highest increase in recall, but in the context of a precision drop of 3.5% on average, while RCE_{self} managed to increase the recall, while maintaining (and in the case of the Geography domain, increasing) precision. In the computer domain, where we had started with a large seed set, a lot of the extractions had already been found by our initial set of domain-independent patterns and as such, the recall increase was not as dramatic as in the Geography domain, where the number of relevant new terms almost doubled for RCE_{iter}. Overall, the only statistically significant differences are the jumps in recall from RCE_{base} to the more sophisticated RCE_{self} and RCE_{iter}.

In future work, we plan to analyze the behavior of these methods as we execute longer runs for all of them. It will also be interesting to see how the description of the information focus should adjust over time due to the exhaustion

Method	**Geography**			**Computer**		
	Precision	Recall	Total	Precision	Recall	Total
RCE_{base}	0.74	0.48	25	0.93	0.65	53
RCE_{self}	0.76	0.82	43	0.91	0.81	66
RCE_{iter}	0.71	0.86	**45**	0.88	0.84	**69**

Table 5: **Results of the 3 methods for Related Class Extraction on the Geography and Computer domains. We report on the precision and recall metrics for each method, together with the number of relevant terms extracted.**

of immediately relevant terms. We want to broadly explore a given domain but not drift into a different domain without realizing it.

6. Previous Work

There has been previous NLP work focused on acquiring domain-specific lexicons from corpora. The bulk of this work consists of weakly supervised learning algorithms for acquiring members of one or several semantic categories using a bootstrapping approach in conjunction with lexical co-occurrence statistics, specific syntactic structures (Roark & Charniak 1998), generic lexico-syntactic patterns (Hearst 1992) and detailed extraction patterns, designed to capture role relationships (Phillips & Riloff 2002). KNOWITALL is a Web-based IE system, instead of a corpus-based one, and so we use search engines to compute co-occurrence statistics and take advantage of the large number of instantiated enumeration patterns available on the Web.

The ontology development field has used exhaustive conceptual clustering techniques to acquire taxonomic relationships (more specifically, the IS-A relationship) from text. Typically an initial taxonomy is built using a combination of statistical and lexical information and the taxonomy is then refined (using automatic classification methods) as new concepts are identified in the text (Maedche & Staab 2001). Due to KNOWITALL's Web-based character, KNOWITALL's subclass acquisition module uses different types of information (Web-scale co-occurrence statistics, a large number of enumeration extraction rules) and it evaluates the acquired information using a different methodology.

The subclass extraction approach closest to ours is described in (Sombatsrisomboon, Matsuo, & Ishizuka 2003). The authors use a single lexical pattern (search engine queries) that yield sentences containing potential hypernyms/hyponyms for a given term. After detecting synonymous words using WordNet, the method uses frequency information to identify the most likely hypernym/hyponym candidates. The paper doesn't give an actual precision figure, just examples of extractions, and reports on tests in just one domain (the computer domain).

7. Conclusion

This paper describes our work in the area of ontology-extension in the context of KNOWITALL, a Web-based information extraction system whose goal is to automatically expand its knowledge over time. We have seen that subclass extraction can significantly increase KNOWITALL's instance recall and we described high-precision, data-driven methods that on average triple the number of correct subclasses found by our module. We have also shown that Web-based co-occurrence statistics can be successfully used to find concepts relevant to a given information focus and reported on our experiments in two domains—approximately 80% of our extractions were relevant new concepts.

There are many directions for future work, such as extracting non-taxonomic relationships between concepts in the same domain and using them to improve subclass recall, using automatically-generated domain-specific extraction patterns for the extraction of both taxonomic and non-taxonomic relationships and integrating other semi-supervised learning methods into our system.

References

Etzioni, O.; Cafarella, M.; Downey, D.; Kok, S.; Popescu, A.; Shaked, T.; Soderland, S.; Weld, D.; and Yates, A. 2004a. Web-scale information extraction in knowitall. In *Proceedings of WWW-04*.

Etzioni, O.; Cafarella, M.; Downey, D.; Popescu, A.; Shaked, T.; Soderland, S.; Weld, D.; and Yates, A. 2004b. Methods for domain-independent information extraction from the web: An experimental comparison. In *Proceedings of AAAI-2004*.

Hearst, M. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th International Conference on Computational Linguistics*, 539–545.

Maedche, A., and Staab, S. 2001. Learning ontologies for the semantic web. In *Proceedings of the Second International Workshop on the Semantic Web*.

Nigam, K., and Ghani, R. 2000. Understanding the behavior of co-training. In *In Proceedings of KDD*.

Phillips, W., and Riloff, E. 2002. Exploiting strong syntactic heuristics and co-training to learn semantic lexicons. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing*.

Roark, B., and Charniak, E. 1998. Noun-phrase co-occurence statistics for semi-automatic semantic lexicon construction. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics*.

Rosch, E.; Mervis, C. B.; Gray, W.; Johnson, D.; and Boyes-Bream, P. 1976. Basic objects in natural categories. *Cognitive Psychology* 3:382–439.

Sombatsrisomboon, R.; Matsuo, Y.; and Ishizuka, M. 2003. Acquisition of hypernyms and hyponyms from the www. In *Proceedings of 2nd International Workshop on Active Mining*.

Turney, P. D. 2001. Mining the Web for synonyms: PMI-IR versus LSA on TOEFL. In *Proceedings of the Twelfth European Conference on Machine Learning*.