A Bootstrapping Algorithm for Automatically Harvesting Semantic Relations

Marco Pennacchiotti
Department of Computer Science
University of Rome "Tor Vergata"
Viale del Politecnico 1
Rome, Italy
pennacchiotti@info.uniroma2.it

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

Abstract

In this paper, we present Espresso, a weakly-supervised iterative algorithm combined with a web-based knowledge expansion technique, for extracting binary semantic relations. Given a small set of seed instances for a particular relation, the system learns lexical patterns, applies them to extract new instances, and then uses the Web to filter and expand the instances. Preliminary experiments show that Espresso extracts highly precise lists of a wide variety of semantic relations when compared with two state of the art systems.

1. Introduction

Recent attention to knowledge-rich problems such as question answering [18] and textual entailment [10] has encouraged Natural Language Processing (NLP) researchers to develop algorithms for automatically harvesting shallow semantic resources. With seemingly endless amounts of textual data at our disposal, we have a tremendous opportunity to automatically grow semantic term banks and ontological resources. Methods must be accurate, adaptable and scalable to the varying sizes of domain corpora (e.g., textbooks vs. World Wide Web), and independent or weakly dependent on human supervision.

In this paper we present *Espresso*, a novel bootstrapping algorithm for automatically harvesting semantic relations, aiming at effectively supporting NLP applications, emphasizing two major points that have been partially neglected by previous systems: *generality* and *weak supervision*.

From the one side, *Espresso* is intended as a general-purpose system able to extract a wide variety of binary semantic relations, from the classical *is-a* and *part-of* relations, to more specific and domain oriented ones like *chemical reactants* in a chemistry domain and *position succession* in political texts. The system architecture is designed with generality in mind, avoiding any relation-specific inference technique. Indeed, for each semantic relation, the system builds specific lexical patterns inferred from textual corpora.

From the other side, *Espresso* requires only weak human supervision. In order to start the extraction process, a user provides only a small set of seed instances of a target relation (e.g. *Italy-country* and *Canada-country* for the *is-a* relation.) In our experience, a handful of seed instances, in general, is sufficient for large corpora while for smaller corpora, a slightly larger set is required. To guarantee weakest supervision, *Espresso* combines its bootstrapping approach with a web-based knowledge expansion technique and linguistic analysis, exploiting the seeds as much as possible.

2. Relevant Work

To date, most research on lexical relation harvesting has focused on *is-a* and *part-of* relations. Approaches fall into two main categories: pattern- and clustering-based.

Most common are *pattern-based approaches*. Hearst [12] pioneered using patterns to extract hyponym (*is-a*) relations. Manually building three lexico-syntactic patterns, Hearst sketched a bootstrapping algorithm to learn more patterns from instances, which has served as the model for most subsequent pattern-based algorithms.

Berland and Charniak [1] propose a system for *part-of* relation extraction, based on the Hearst approach [12]. Seed instances are used to infer linguistic patterns that, in turn, are used to extract new instances, ranked according to various statistical measures. While this study introduces statistical measures to evaluate instance reliability, it remains vulnerable to data sparseness and has the limitation of taking into consideration only one-word terms.

Improving upon Berland and Charniak [1], Girju et al. [11] employ machine learning algorithms and WordNet [8] to disambiguate *part-of* generic patterns, like [*whole-NP's part-NP*]. This study is the first extensive attempt to solve the problem of *generic relational patterns*, that is, those expressive patterns that have high recall while suffering low precision, as they subsume a large set of instances. In order to discard incorrect instances, Girju et al. learn WordNet-based selectional restrictions, like [*whole-NP(scene#4)'s part-NP(movie#1)*]. While making huge grounds on improving precision/recall, the system requires heavy supervision through manual semantic annotations.

Ravichandran and Hovy [20] focus on efficiency issues for scaling relation extraction to terabytes of data. A simple and effective algorithm is proposed to infer surface patterns from a small set of instance seeds by extracting all substrings relating seeds in corpus sentences. The frequencies of the substrings in the corpus are then used to retain the best patterns. The approach gives good results on specific relations such as *birthdates*, however it has low precision on generic ones like *is-a* and *part-of*. Pantel et al. [17] proposed a similar, highly scalable approach, based on an edit-distance technique, to learn lexico-POS patterns, showing both good performances and efficiency. *Espresso* uses a similar approach to infer patterns, but we then apply refining techniques to deal with various types of relations.

Other pattern-based algorithms include Riloff and Shepherd [21], who used a semi-automatic method for discovering similar words using a few seed examples by using pattern-based techniques and human supervision, KnowItAll [7] that performs large-scale extraction of facts from the Web, Mann [15] and Fleischman et al. [9] who used part of speech patterns to extract a subset of *is-a* relations involving proper nouns, and Downey et al. [6] who formalized the problem of relation extraction in a coherent and effective combinatorial model that is shown to outperform previous probabilistic frameworks.

Clustering approaches to relation extraction are less common and have insofar been applied only to *is-a* extraction. These methods employ clustering algorithms to group words according to their meanings in text, label the clusters using its members' lexical or syntactic dependencies, and then extract an *is-a* relation between each cluster member and the cluster label. Caraballo [3] proposed the first attempt, which used conjunction and apposition features to build noun clusters. Recently, Pantel and Ravichandran [16] extended this approach by making use of all syntactic dependency features for each noun. The advantage of clustering approaches is that they permit algorithms to identify *is-a* relations that do not explicitly appear in text, however they generally fail to produce coherent clusters from fewer than 100 million words; hence they are unreliable for small corpora.

3. The *Espresso* Algorithm

The *Espresso* algorithm is based on a similar framework to the one adopted in [12]. For a specific semantic binary relation (e.g., *is-a*), the algorithm requires as input a small set of seed instances I_s and a corpus C. An instance is a pair of terms x and y governed by the relation at hand (e.g., *Pablo Picasso is-a artist*). Starting from these seeds, the algorithm begins a four-phase loop. In the first phase, the algorithm infers a set of patterns P that captures as many of the seed instances as possible in C. In the second phase, we define a reliability measure to select the best set of patterns $P' \subseteq P$. In phase three, the patterns in P' are used to extract a set of instances I. Finally, in phase four, *Espresso* scores each instance and then selects the best instances I' as input seeds for the next iteration. The algorithm terminates when a predefined stopping condition is met (for our preliminary experiments, the stopping condition is set according to the size of the corpus). For each induced pattern p and instance i, the information theoretic scores, $r_\pi(p)$ and $r_\iota(i)$ respectively, aim to express their reliability.

Below, Sections 3.2–3.5 describe in detail these different phases of *Espresso*.

3.1. Term definition

Before one can extract relation instances from a corpus, it is necessary to define a tokenization procedure for extracting terms. Terms are commonly defined as *surface representations of stable and key domain concepts* [19]. Defining regular expressions over POS-tagged corpora is the most commonly used technique to both define and extract terms. We adopt a slightly modified version of the term definition given in [13], as it is one of the most commonly used in the literature:

$$((Adj|Noun)+|((Adj|Noun)*(NounPrep)?)(Adj|Noun)*)Noun$$

We operationally extend the definition of *Adj* to include present and past participles as most noun phrases composed of them are usually intended as terms (e.g., *boiling point*). Thus, unlike many approaches for automatic relation extraction, we allow complex multi-word terms as anchor points. Hence, we can capture relations between complex terms, such as *"record of a criminal conviction" part-of "FBI report"*.

3.2. Phase 1: Pattern discovery

The pattern discovery phase takes as input a set of instances I' and produces as output a set of lexical patterns P. For the first iteration $I' = I_s$, the set of initial seeds. In order to induce P, we apply a slight modification to the approach presented in [20]. For each input instance $i = \{x, y\}$, we first retrieve all sentences $S_{x,y}$ containing the two terms x and y. Sentences are then generalized into a set of new sentences $SG_{x,y}$ by replacing all terminological expressions by a terminological label (*TR*). For example:

"Because/IN HF/NNP is/VBZ a/DT weak/JJ acid/NN and/CC **x** is/VBZ a/DT **y**"

is generalized as:

"Because/IN ***TR*** is/VBZ a/DT ***TR*** and/CC **x** is/VBZ a/DT **y**"

All substrings linking terms x and y are then extracted from the set $SG_{x,y}$, and overall frequencies are computed. The most frequent substrings then represent the set of new patterns P, where the frequency cutoff is experimentally set. Term generalization is particularly useful for small corpora, where generalization is vital to ease the data sparseness. However, the generalized patterns are naturally less precise. Hence, when dealing with bigger corpora, the

system allows the use of $S_{x,y} \cup SG_{x,y}$ in order to extract substrings. For our experiments, we used the set $SG_{x,y}$.

3.3. Phase 2: Pattern filtering

In this phase, *Espresso* selects among the patterns P those that are most reliable. Intuitively, a reliable pattern is one that is both highly precise and one that extracts many instances. The recall of a pattern p can be approximated by the fraction of input instances in I' that are extracted by p. Since it is difficult at run-time to estimate the precision of a pattern, we are weary of keeping patterns that generate many instances (i.e., patterns that generate high recall but potentially disastrous precision). We thus prefer patterns that are highly associated with the input patterns I'. Pointwise mutual information [4] is a commonly used metric for measuring the strength of association between two events x and y:

$$pmi(x, y) = \log \frac{P(x, y)}{P(x)P(y)}$$

We define the reliability of a pattern p, $r_\pi(p)$, as its average strength of association across each input instance i in I', weighted by the reliability of each instance i:

$$r_\pi(p) = \frac{\sum_{i \in I'} \left(\frac{pmi(i, p)}{\max_{pmi}} * r_t(i) \right)}{|I'|}$$

where $r_t(i)$ is the reliability of instance i (defined in Section 3.5) and max_{pmi} is the maximum pointwise mutual information between all patterns and all instances. $r_\pi(p)$ ranges from [0,1]. The reliability of the manually supplied seed instances are $r_t(i) = 1$. The pointwise mutual information between instance $i = \{x, y\}$ and pattern p is estimated using the following formula:

$$pmi(i, p) = \log \frac{|x, p, y|}{|x,*, y||*, p,*|}$$

where $|x, p, y|$ is the frequency of pattern p instantiated with terms x and y and where the asterisk (*) represents a wildcard. A well-known problem is that pointwise mutual information is biased towards infrequent events. To address this, we multiply $pmi(i, p)$ with the discounting factor suggested in [16].

The set of highest n scoring patterns P', according to $r_\pi(p)$, are then selected and retained for the next phase, where n is the number of patterns of the previous iteration incremented by 1. In general, we expect that the set of patterns is formed by those of the previous iteration plus a new one. Yet, new statistical evidence can lead the algorithm to discard a pattern that was previously discovered.

Moreover, to further discourage too generic patterns that might have low precision, a threshold t is set for the number of instances that a pattern retrieves. Patterns firing more than t instances are then discarded, no matter what their score is. In this paper, we experimentally set t to a value dependent on the size of the corpus. In future work, this parameter can be learned using a development corpus.

Our reliability measure ensures that overly generic patterns, which may potentially have very low precision, are discarded. However, we are currently exploring a web-expansion algorithm that could both help detect generic patterns and also filter out their incorrect instances. We estimate the precision of the instance set generated by a new pattern p by looking at the number of these instances that are instantiated on the Web by previously accepted patterns.

Generic patterns will generate instances with higher Web counts than incorrect patterns. Then, the Web counts can also be used to filter out incorrect instances from the generic patterns' instantiations. More details are discussed in Section 4.3.

3.4. Phase 3: Instance discovery

In this phase, *Espresso* retrieves from the corpus the set of instances *I* that match any of the lexical patterns in *P'*.

In small corpora, the number of extracted instances can be too low to guarantee sufficient statistical evidence for the pattern discovery phase of the next iteration. In such cases, the system enters a *web expansion* phase, in which new instances for the given patterns are retrieved from the Web, using the Google search engine. Specifically, for each instance $i \in I$, the system creates a set of queries, using each pattern in *P'* with its *y* term instantiated with *i*'s *y* term. For example, given the instance *"Italy ; country"* and the pattern *[Y such as X]* , the resulting Google query will be *"country such as *"*. New instances are then created from the retrieved Web results (e.g. *"Canada ; country"*) and added to *I*. We are currently exploring filtering mechanisms to avoid retrieving too much noise.

Moreover, to cope with data sparsity, a *syntactic expansion* phase is also carried out. A set of new instances is created for each instance $i \in I$ by extracting sub-terminological expressions from *x* corresponding to the syntactic head of terms. For example, expanding the relation *"new record of a criminal conviction" part-of "FBI report"*, the following new instances are obtained: *"new record" part-of "FBI report"*, and *"record" part-of "FBI report"*.

3.5. Phase 4: Instance filtering

Estimating the reliability of an instance is similar to estimating the reliability of a pattern. Intuitively, a reliable instance is one that is highly associated with as many reliable patterns as possible (i.e., we have more confidence in an instance when multiple reliable patterns instantiate it.) Hence, analogous to our pattern reliability measure in Section 3.3, we define the reliability of an instance *i*, $r_l(i)$, as:

$$r_l(i) = \frac{\sum_{p \in P'} \frac{pmi(i,p)}{\max_{pmi}} * r_\pi(p)}{|P'|}$$

where $r_\pi(p)$ is the reliability of pattern *p* (defined in Section 3.3) and max_{pmi} is the maximum pointwise mutual information between all patterns and all instances, as in Section 3.3.

Espresso finally selects the highest scoring *m* instances, *I'*, and retains them as input for the subsequent iteration. In this paper, we experimentally set *m* = 200.

4. Experimental Results

4.1. Experimental Setup

In this section, we present a preliminary comparison of *Espresso* with two state of the art systems on the task of extracting various semantic relations.

4.1.1. Datasets

We perform our experiments using the following two datasets:

- **TREC-9**: This dataset consists of a sample of articles from the Aquaint (TREC-9) newswire text collection. The sample consists of 5,951,432 words extracted from the following data files: AP890101 – AP890131, AP890201 – AP890228, and AP890310 – AP890319.
- **CHEM**: This small dataset of 313,590 words consists of a college level textbook of introductory chemistry [2].

We preprocess the corpora using the Alembic Workbench POS-tagger [5].

4.1.2. Systems

We compare the results of *Espresso* with the following two state of the art extraction systems:

- **RH02**: This algorithm by Ravichandran and Hovy [20] learns lexical extraction patterns from a set of seed instances of a particular relation (see Section 2.)
- **PR04**: This *is-a* extraction algorithm from Pantel and Ravichandran [16] first automatically induces concepts (clusters) from a raw corpus, names the concepts, and then extracts an is-a relation between each cluster member and its cluster label. For each cluster member, the system may generate multiple possible *is-a* relations, but in this evaluation we only keep the highest scoring one. To apply this algorithm, both datasets were first analyzed using the Minipar parser [14].
- **ESP**: This is the algorithm described in this paper (details in Section 3).

4.1.3. Semantic Relations

Espresso is designed to extract various semantic relations exemplified by a given small set of seed instances. For our preliminary evaluation, we consider the standard *is-a* and *part-of* relations as well as three novel relations:

- *succession*: This relation indicates that one proper noun succeeds another in a position or title. For example, *George Bush* succeeded *Bill Clinton* and *Pope Benedict XVI* succeeded *Pope John Paul II*. We evaluate this relation on the TREC-9 corpus.
- *reaction*: This relation occurs between chemical elements/molecules that can be combined in a chemical reaction. For example, *hydrogen gas* reacts-with *oxygen gas* and *zinc* reacts-with *hydrochloric acid*. We evaluate this relation on the CHEM corpus.
- *production*: This relation occurs when a process or element/object produces a result. For example, *ammonia* produces *nitric oxide*. We evaluate this relation on the CHEM corpus.

For each semantic relation, we manually extracted a set of seed examples. The seeds were used for both *Espresso* as well as RH02[1]. Table 1 lists a sample of the seeds as well as sample outputs from *Espresso*.

4.2. Precision and Recall

We implemented each of the three systems outlined in Section 4.1.2 and applied them to the TREC and CHEM datasets. For each output set, per relation, we evaluate the precision of the system by extracting a random sample of instances (50 for the TREC corpus and 20 for the

[1] PR04 does not require any seeds.

Table 1. Sample seeds used for each semantic relation and sample outputs from *Espresso*. The number in the parentheses for each relation denotes the total number of seeds.

		SEEDS	*ESP*
TREC 9	*Is-a (12)*	wheat :: crop George Wendt :: star Miami :: city shark :: predator	Picasso :: artist tax :: charge drug dealers :: felons Italy :: country
	Part-Of (12)	leader :: panel city :: region plastic :: explosive United States :: alliance	shield :: nuclear missile biblical quotations :: book trees :: land material :: FBI report
	Succession (12)	Khrushchev :: Stalin Carla Hills :: Yeutter George Bush :: Ronald Reagan Julio Barbosa de Aquino :: Mendes	Ford :: Nixon Setrakian :: John Griesemer Camero Cardiel :: Camacho Susan Weiss :: editor
CHEM	*Is-a (12)*	NaCl :: ionic compounds diborane :: substance nitrogen :: element gold :: precious metal	Na :: element protein :: biopolymer HCl :: strong acid electromagnetic radiation :: energy
	Part-Of (12)	ion :: matter oxygen :: water light particle :: gas element :: substance	oxygen :: air powdered zinc metal :: battery atom :: molecule ethylene glycol :: automotive antifreeze
	Reaction (13)	magnesium :: oxygen hydrazine :: water aluminum metal :: oxygen lithium metal :: fluorine gas	hydrogen :: oxygen Ni :: HCl carbon dioxide :: methane boron :: fluorine
	Production (14)	bright flame :: flares hydrogen :: solid metal hydrides ammonia :: nitric oxide copper :: brown gas	electron :: ions glycerin :: nitroglycerin kidneys :: kidney stones ions :: charge

CHEM corpus) and evaluating their quality manually using one human judge[2]. For each instance, the judge may assign a score of 1 for correct, 0 for incorrect, and ½ for partially correct. Example instances that were judged partially correct include "*analyst is-a manager*" and "*pilot is-a teacher*". The precision for a given set of relation instances is the sum of the judge's scores divided by the number of instances.

Although knowing the total number of instances of a particular relation in any non-trivial corpus is impossible, it is possible to compute the recall of a system relative to another system's recall. The recall of a system *A*, R_A, is given by the following formula:

$$R_A = \frac{C_A}{C}$$

where C_A is the number of correct instances of a particular relation extracted by *A* and *C* is the total number of correct instances in the corpus. Following [17], we define the relative recall of system *A* given system *B*, $R_{A|B}$, as:

$$R_{A|B} = \frac{R_A}{R_B} = \frac{C_A}{C_B} = \frac{P_A \times |A|}{P_B \times |B|}$$

Using the precision estimates, P_A, from our precision experiments, we can estimate $C_A \approx P_A \times |A|$, where *A* is the total number of instances of a particular relation discovered by system *A*.

[2] In future work, we will perform this evaluation using multiple judges in order to obtain confidence bounds and agreement scores.

Table 2. System performance on the *is-a* relation on the TREC-9 dataset.

System	*Instances*	*Precision**	*Rel Recall*†
RH02	57,525	28.0%	**5.31**
PR04	1,504	47.0%	0.23
ESP	4,154	**73.0%**	1.00

* Precision estimated from 50 randomly sampled instances.
† Relative recall is given in relation to ESP.

Table 3. System performance on the *is-a* relation on the CHEM dataset.

System	*Instances*	*Precision**	*Rel Recall*†
RH02	2556	25.0%	**3.76**
PR04	108	40.0%	0.25
ESP	200	**85.0%**	1.00

* Precision estimated from 20 randomly sampled instances.
† Relative recall is given in relation to ESP.

Table 4. System performance on the *part-of* relation on the TREC-9 dataset.

System	*Instances*	*Precision**	*Rel Recall*†
RH02	12,828	35.0%	**42.52**
ESP	132	**80.0%**	1.00

* Precision estimated from 50 randomly sampled instances.
† Relative recall is given in relation to ESP.

Table 5. System performance on the *part-of* relation on the CHEM dataset.

System	*Instances*	*Precision**	*Rel Recall*†
RH02	11,582	33.8%	**58.78**
ESP	111	**60.0%**	1.00

* Precision estimated from 20 randomly sampled instances.
† Relative recall is given in relation to ESP.

Table 6. System performance on the *succession* relation on the TREC-9 dataset.

System	*Instances*	*Precision**	*Rel Recall*†
RH02	49,798	2.0%	**36.96**
ESP	55	**49.0%**	1.00

* Precision estimated from 50 randomly sampled instances.
† Relative recall is given in relation to ESP.

Table 7. System performance on the *reaction* relation on the CHEM dataset.

System	*Instances*	*Precision**	*Rel Recall*†
RH02	6,083	30%	**53.67**
ESP	40	**85%**	1.00

* Precision estimated from 20 randomly sampled instances.
† Relative recall is given in relation to ESP.

Table 8. System performance on the *production* relation on the CHEM dataset.

System	*Instances*	*Precision**	*Rel Recall*†
RH02	197	57.5%	0.80
ESP	196	**72.5%**	**1.00**

* Precision estimated from 20 randomly sampled instances.
† Relative recall is given in relation to ESP.

Tables 2 – 8 reports the total number of instances, precision, and relative recall of each system on the TREC-9 and CHEM corpora. The relative recall is always given in relation to the *Espresso* system. For example, in Table 2, RH02 has a relative recall of 5.31 with *Espresso*, which means that the RH02 system output 5.31 times more correct relations than *Espresso* (at a cost of much lower precision). Similarly, PR04 has a relative recall of 0.23 with *Espresso*, which means that PR04 outputs 4.35 fewer correct relations than *Espresso* (also with a smaller precision).

4.3. Discussion

Experimental results, for all relations and the two different corpus sizes, show that *Espresso* greatly outperforms the other two methods on precision. However, *Espresso* fails to match the recall level of RH02 in all but the experiment on the *production* relation. Indeed, the filtering of unreliable patterns and instances during the bootstrapping algorithm not only discards the patterns that are unrelated to the actual relation, but also patterns that are too generic and ambiguous – hence resulting in a loss of recall.

As underlined in Section 3.2, the ambiguity of generic patterns often introduces much noise in the system (e.g, the pattern *[X of Y]* can ambiguously refer to a *part-of*, *is-a* or *possession*

relation). However, generic patterns, while having low precision, yield a high recall, as also reported by [11]. We ran an experiment on the *reaction* relation, retaining the generic patterns produced during *Espresso*'s selection process. As expected, we obtained 1923 instances instead of the 40 reported in Table 7, but precision dropped from 85% to 30%.

The challenge, then, is to harness the expressive power of the generic patterns whilst maintaining the precision of *Espresso*. We propose the following solution that helps both in distinguishing generic patterns from incorrect patterns and also in filtering incorrect instances produced by generic patterns. Unlike Girju et al. [11] that propose a highly supervised machine learning approach based on selectional restriction, ours is an unsupervised method based on statistical evidence obtained from the Web. At a given iteration in *Espresso*, the intuition behind our solution is that the Web is large enough that correct instances will be instantiated by many of the currently accepted patterns *P*. Hence, we can distinguish between generic patterns and incorrect patterns by inspecting the relative frequency distribution of their instances using the patterns in *P*. More formally, given an instance *i* produced by a generic or incorrect pattern, we count how many times *i* instantiates on the Web with every pattern in *P*, using Google. The instance *i* is then considered correct if its web count surpasses a given threshold. The pattern in question is accepted as a generic pattern if a sufficient number of its instances are considered correct, otherwise it is rejected as an incorrect pattern.

Although our results in Section 4.2 do not include this algorithm, we performed a small experiment by adding an a-posteriori *generic pattern recovery* phase to *Espresso*. We tested the 7,634 instances extracted by the generic pattern *[X of Y]* on the CHEM corpus for the *part-of* relation. We randomly sample 200 of these instances and then queried Google for these instances using the pattern [*X consists of Y*]. Manual evaluation of the 25 instances that occurred at least once on Google showed 50% precision. Adding these instances to the results from Table 5 decreases the system precision from 60% to 51%, but dramatically increases *Espresso*'s recall by a factor of 8.16. Furthermore, it is important to note that there are several other generic patterns, like [*X's Y*], from which we expect a similar precision of 50% with a continual increase of recall. This is a very exciting avenue of further investigation.

5. Conclusions

We proposed a weakly supervised bootstrapping algorithm, called *Espresso*, for automatically extracting a wide variety of binary semantic relations from raw text. Given a small set of seed instances for a particular relation, the system learns reliable lexical patterns, applies them to extract new instances ranked by an information theoretic definition of reliability, and then uses the Web to filter and expand the instances.

There are many avenues of future work. Preliminary results show that *Espresso* generates highly precise relations, but at the expense of lower recall. As mentioned above in Section 4.3, we are working on improving system recall with a web-based method to identify generic patterns and filter their instances. Early results appear very promising. We also plan to investigate the use of WordNet selectional constraints, as proposed by [11]. We expect here that negative instances will play a key role in determining the selectional restriction on generic patterns.

Espresso is the first system, to our knowledge, to emphasize both minimal supervision and generality, both in identification of a wide variety of relations and in extensibility to various corpus sizes. It remains to be seen whether one could enrich existing ontologies with relations harvested by *Espresso*, and if these relations can benefit NLP applications such as QA.

Acknowledgements

The authors wish to thank the reviewers for their helpful comments and Andrew Philpot for evaluating the outputs of the systems.

References

[1] Berland, M. and E. Charniak, 1999. Finding parts in very large corpora. In *Proceedings of ACL-1999*. pp. 57-64. College Park, MD.

[2] Brown, T.L.; LeMay, H.E.; Bursten, B.E.; and Burdge, J.R. 2003. *Chemistry: The Central Science, Ninth Edition*. Prentice Hall.

[3] Caraballo, S. 1999. Automatic acquisition of a hypernym-labeled noun hierarchy from text. In *Proceedings of ACL-99*. pp 120-126, Baltimore, MD.

[4] Cover, T.M. and Thomas, J.A. 1991. *Elements of Information Theory*. John Wiley & Sons.

[5] Day, D.; Aberdeen, J.; Hirschman, L.; Kozierok, R.; Robinson, P.; and Vilain, M. 1997. Mixed-initiative development of language processing systems. In *Proceedings of ANLP-1997*. Washington D.C.

[6] Downey, D.; Etzioni, O.; and Soderland, S. 2005. A Probabilistic model of redundancy in information extraction. In *Proceedings of IJCAI-2005*. pp. 1034-1041. Edinburgh, Scotland.

[7] Etzioni, O.; Cafarella, M.J.; Downey, D.; Popescu, A.-M.; Shaked, T.; Soderland, S.; Weld, D.S.; and Yates, A. 2005. Unsupervised named-entity extraction from the Web: An experimental study. *Artificial Intelligence*, 165(1): 91-134.

[8] Fellbaum, C. 1998. *WordNet: An Electronic Lexical Database*. MIT Press.

[9] Fleischman, M.; Hovy, E.; and Echihabi, A. 2003. Offline strategies for online question answering: Answering questions before they are asked. In *Proceedings of ACL-03*. pp. 1-7. Sapporo, Japan.

[10] Geffet, M. and Dagan, I. 2005. The Distributional Inclusion Hypotheses and Lexical Entailment. In *Proceedings of ACL-2005*. Ann Arbor, MI.

[11] Girju, R.; Badulescu, A.; and Moldovan, D. 2003. Learning semantic constraints for the automatic discovery of part-whole relations. In *Proceedings of HLT/NAACL-03*. pp. 80-87. Edmonton, Canada.

[12] Hearst, M. 1992. Automatic acquisition of hyponyms from large text corpora. In *COLING-92*. pp. 539-545. Nantes, France.

[13] Justeson J.S. and Katz S.M. 1995. Technical Terminology: some linguistic properties and algorithms for identification in text. In *Proceedings of ICCL-1995*. pp.539-545. Nantes, France.

[14] Lin, D. 1994. Principar - an efficient, broad-coverage, principle-based parser. In *Proceedings of COLING-94*. pp. 42-48. Kyoto, Japan.

[15] Mann, G. S. 2002. Fine-Grained Proper Noun Ontologies for Question Answering. In *Proceedings of SemaNet' 02: Building and Using Semantic Networks*, Taipei, Taiwan.

[16] Pantel, P. and Ravichandran, D. 2004. Automatically labeling semantic classes. In *Proceedings of HLT/NAACL-04*. pp. 321-328. Boston, MA.

[17] Pantel, P.; Ravichandran, D.; Hovy, E.H. 2004. Towards terascale knowledge acquisition. In *Proceedings of COLING-04*. pp. 771-777. Geneva, Switzerland.

[18] Pasca, M. and Harabagiu, S. 2001. The informative role of WordNet in Open-Domain Question Answering. In *Proceedings of NAACL-01 Workshop on WordNet and Other Lexical Resources*. pp. 138-143. Pittsburgh, PA.

[19] Pazienza M.T. 2000. A domain-specific terminology-extraction system. In *Terminology*, 5:2.

[20] Ravichandran, D. and Hovy, E.H. 2002. Learning surface text patterns for a question answering system. In *Proceedings of ACL-2002*. pp. 41-47. Philadelphia, PA.

[21] Riloff, E. and Shepherd, J. 1997. A corpus-based approach for building semantic lexicons. In *Proceedings of EMNLP-1997*.