Espresso: Leveraging Generic Patterns for Automatically Harvesting Semantic Relations

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

Marco Pennacchiotti
ART Group - DISP
University of Rome "Tor Vergata"
Viale del Politecnico 1
Rome, Italy
pennacchiotti@info.uniroma2.it

Abstract

In this paper, we present *Espresso*, a weakly-supervised, general-purpose, and accurate algorithm for harvesting semantic relations. The main contributions are: i) a method for exploiting generic patterns by filtering incorrect instances using the Web; and ii) a principled measure of pattern and instance reliability enabling the filtering algorithm. We present an empirical comparison of *Espresso* with various state of the art systems, on different size and genre corpora, on extracting various general and specific relations. Experimental results show that our exploitation of generic patterns substantially increases system recall with small effect on overall precision.

1 Introduction

Recent attention to knowledge-rich problems such as question answering (Pasca and Harabagiu 2001) and textual entailment (Geffet and Dagan 2005) has encouraged natural language processing 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.

To date, researchers have harvested, with varying success, several resources, including concept lists (Lin and Pantel 2002), topic signatures (Lin and Hovy 2000), facts (Etzioni et al. 2005), and word similarity lists (Hindle 1990). Many recent efforts have also focused on extracting semantic relations between entities, such as entailments (Szpektor et al. 2004), *is-a* (Ravichandran and Hovy 2002), *part-of* (Girju et al. 2006), and other relations.

The following desiderata outline the properties of an ideal relation harvesting algorithm:

- *Performance*: it must generate both high precision and high recall relation instances;
- *Minimal supervision*: it must require little or no human annotation;
- *Breadth*: it must be applicable to varying corpus sizes and domains; and
- *Generality*: it must be applicable to a wide variety of relations (i.e., not just *is-a* or *part-of*).

To our knowledge, no previous harvesting algorithm addresses all these properties concurrently.

In this paper, we present *Espresso*, a general-purpose, broad, and accurate corpus harvesting algorithm requiring minimal supervision. The main algorithmic contribution is a novel method for exploiting *generic patterns*, which are broad coverage noisy patterns – i.e., patterns with high recall and low precision. Insofar, difficulties in using these patterns have been a major impediment for minimally supervised algorithms resulting in either very low precision or recall. We propose a method to automatically detect generic patterns and to separate their correct and incorrect instances. The key intuition behind the algorithm is that given a set of *reliable* (high precision) patterns on a corpus, correct instances of a generic pattern will fire more with reliable patterns on a very large corpus, like the Web, than incorrect ones. Below is a summary of the main contributions of this paper:

- ***Algorithm for exploiting generic patterns***: Unlike previous algorithms that require significant manual work to make use of generic patterns, we propose an unsupervised Web-filtering method for using generic patterns; and
- ***Principled reliability measure***: We propose a new measure of pattern and instance reliability which enables the use of generic patterns.

Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the ACL, pages 113–120,
Sydney, July 2006. ©2006 Association for Computational Linguistics

Espresso addresses the desiderata as follows:

- *Performance*: *Espresso* generates balanced precision and recall relation instances by exploiting generic patterns;
- *Minimal supervision*: *Espresso* requires as input only a small number of seed instances;
- *Breadth*: *Espresso* works on both small and large corpora – it uses Web and syntactic expansions to compensate for lacks of redundancy in small corpora;
- *Generality*: *Espresso* is amenable to a wide variety of binary relations, from classical *is-a* and *part-of* to specific ones such as *reaction* and *succession*.

Previous work like (Girju et al. 2006) that has made use of generic patterns through filtering has shown both high precision and high recall, at the expensive cost of much manual semantic annotation. Minimally supervised algorithms, like (Hearst 1992; Pantel et al. 2004), typically ignore generic patterns since system precision dramatically decreases from the introduced noise and bootstrapping quickly spins out of control.

2 Relevant Work

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

Most common are *pattern-based approaches*. Hearst (1992) 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 (1999) proposed a system for *part-of* relation extraction, based on the (Hearst 1992) approach. Seed instances are used to infer linguistic patterns that are used to extract new instances. While this study introduces statistical measures to evaluate instance quality, it remains vulnerable to data sparseness and has the limitation of considering only one-word terms.

Improving upon (Berland and Charniak 1999), Girju et al. (2006) employ machine learning algorithms and WordNet (Fellbaum 1998) to disambiguate *part-of* generic patterns like "*X's Y*" and "*X of Y*". This study is the first extensive attempt to make use of generic patterns. In order to discard incorrect instances, they learn WordNet-based selectional restrictions, like "*X(scene#4)'s Y(movie#1)*". While making huge grounds on improving precision/recall, heavy supervision is required through manual semantic annotations.

Ravichandran and Hovy (2002) focus on scaling relation extraction to the Web. A simple and effective algorithm is proposed to infer surface patterns from a small set of instance seeds by extracting substrings relating seeds in corpus sentences. 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. (2004) proposed a similar, highly scalable approach, based on an edit-distance technique, to learn lexico-POS patterns, showing both good performance and efficiency. *Espresso* uses a similar approach to infer patterns, but we make use of generic patterns and apply refining techniques to deal with wide variety of relations.

Other pattern-based algorithms include (Riloff and Shepherd 1997), who used a semi-automatic method for discovering similar words using a few seed examples, KnowItAll (Etzioni et al. 2005) that performs large-scale extraction of facts from the Web, Mann (2002) who used part of speech patterns to extract a subset of *is-a* relations involving proper nouns, and (Downey et al. 2005) who formalized the problem of relation extraction in a coherent and effective combinatorial model that is shown to outperform previous probabilistic frameworks.

Clustering approaches have so far been applied only to *is-a* extraction. These methods use 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 (1999) proposed the first attempt, which used conjunction and apposition features to build noun clusters. Recently, Pantel and Ravichandran (2004) 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

Espresso is based on the framework adopted in (Hearst 1992). It is a minimally supervised bootstrapping algorithm that takes as input a few seed instances of a particular relation and iteratively learns surface patterns to extract more instances. The key to *Espresso* lies in its use of *generic patters*, i.e., those broad coverage noisy patterns that

extract both many correct and incorrect relation instances. For example, for *part-of* relations, the pattern "*X of Y*" extracts many correct relation instances like "*wheel of the car*" but also many incorrect ones like "*house of representatives*".

The key assumption behind *Espresso* is that in very large corpora, like the Web, correct instances generated by a generic pattern will be instantiated by some *reliable patterns*, where reliable patterns are patterns that have high precision but often very low recall (e.g., "*X consists of Y*" for *part-of* relations). In this section, we describe the overall architecture of *Espresso*, propose a principled measure of reliability, and give an algorithm for exploiting generic patterns.

3.1 System Architecture

Espresso iterates between the following three phases: *pattern induction*, *pattern ranking/selection*, and *instance extraction*.

The algorithm begins with seed instances of a particular binary relation (e.g., *is-a*) and then iterates through the phases until it extracts τ_1 patterns or the average pattern score decreases by more than τ_2 from the previous iteration. In our experiments, we set $\tau_1 = 5$ and $\tau_2 = 50\%$.

For our tokenization, in order to harvest multiword terms as relation instances, we adopt a slightly modified version of the term definition given in (Justeson 1995), as it is one of the most commonly used in the NLP literature:

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

Pattern Induction

In the *pattern induction* phase, *Espresso* infers a set of surface patterns *P* that connects as many of the seed instances as possible in a given corpus. Any pattern learning algorithm would do. We chose the state of the art algorithm described in (Ravichandran and Hovy 2002) with the following slight modification. For each input instance $\{x, y\}$, we first retrieve all sentences containing the two terms *x* and *y*. The sentences are then generalized into a set of new sentences $S_{x,y}$ 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**"*

Term generalization is useful for small corpora to ease data sparseness. Generalized patterns are naturally less precise, but this is ameliorated by our filtering step described in Section 3.3.

As in the original algorithm, all substrings linking terms *x* and *y* are then extracted from $S_{x,y}$, and overall frequencies are computed to form *P*.

Pattern Ranking/Selection

In (Ravichandran and Hovy 2002), a frequency threshold on the patterns in *P* is set to select the final patterns. However, low frequency patterns may in fact be very good. In this paper, instead of frequency, we propose a novel measure of pattern reliability, r_π, which is described in detail in Section 3.2.

Espresso ranks all patterns in *P* according to reliability r_π and discards all but the top-*k*, where *k* is set to the number of patterns from the previous iteration plus one. 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.

Instance Extraction

In this phase, *Espresso* retrieves from the corpus the set of instances *I* that match any of the patterns in *P*. In Section 3.2, we propose a principled measure of instance reliability, r_ι, for ranking instances. Next, *Espresso* filters incorrect instances using the algorithm proposed in Section 3.3 and then selects the highest scoring *m* instances, according to r_ι, as input for the subsequent iteration. We experimentally set *m*=200.

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 an *expansion phase*, where instances are expanded as follows:

Web expansion: New instances of the patterns in *P* are retrieved from the Web, using the Google search engine. Specifically, for each instance $\{x, y\} \in I$, the system creates a set of queries, using each pattern in *P* instantiated with *y*. 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*. The noise generated from this expansion is attenuated by the filtering algorithm described in Section 3.3.

Syntactic expansion: New instances are created from each instance $\{x, y\} \in I$ by extracting sub-terminological expressions from *x* corresponding to the syntactic head of terms. For ex-

ample, the relation *"new record of a criminal conviction part-of FBI report"* expands to: *"new record part-of FBI report"*, and *"record part-of FBI report"*.

3.2 Pattern and Instance Reliability

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 that are extracted by *p*. Since it is non-trivial to estimate automatically the precision of a pattern, we are wary of keeping patterns that generate many instances (i.e., patterns that generate high recall but potentially disastrous precision). Hence, we desire patterns that are highly associated with the input instances. Pointwise mutual information (Cover and Thomas 1991) is a commonly used metric for measuring this 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*:

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

where $r_\iota(i)$ is the reliability of instance *i* (defined below) 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 $r_\iota(i) = 1$. 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. We thus multiply $pmi(i, p)$ with the discounting factor suggested in (Pantel and Ravichandran 2004).

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, we define the reliability of an instance *i*, $r_\iota(i)$, as:

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

where $r_\pi(p)$ is the reliability of pattern *p* (defined earlier) and max_{pmi} is as before. Note that $r_\iota(i)$ and $r_\pi(p)$ are recursively defined, where $r_\iota(i) = 1$ for the manually supplied seed instances.

3.3 Exploiting Generic Patterns

Generic patterns are high recall / low precision patterns (e.g, the pattern "*X of Y*" can ambiguously refer to a *part-of*, *is-a* and *possession* relations). Using them blindly increases system recall while dramatically reducing precision. Minimally supervised algorithms have typically ignored them for this reason. Only heavily supervised approaches, like (Girju et al. 2006) have successfully exploited them.

Espresso's recall can be significantly increased by automatically separating correct instances extracted by generic patterns from incorrect ones. The challenge is to harness the expressive power of the generic patterns while remaining minimally supervised.

The intuition behind our method is that in a very large corpus, like the Web, correct instances of a generic pattern will be instantiated by many of *Espresso*'s reliable patterns accepted in *P*. Recall that, by definition, *Espresso*'s reliable patterns extract instances with high precision (yet often low recall). In a very large corpus, like the Web, we assume that a correct instance will occur in at least one of *Espresso*'s reliable pattern even though the patterns' recall is low. Intuitively, our confidence in a correct instance increases when, i) the instance is associated with many reliable patterns; and ii) its association with the reliable patterns is high. At a given *Espresso* iteration, where P_R represents the set of previously selected reliable patterns, this intuition is captured by the following measure of confidence in an instance $i = \{x, y\}$:

$$S(i) = \sum_{p \in P_R} S_p(i) \times \frac{r_\pi(p)}{T}$$

where *T* is the sum of the reliability scores $r_\pi(p)$ for each pattern $p \in P_R$, and

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

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 used as input for the system.

	Is-a (12)	*Part-Of (12)*	*Succession (12)*	*Reaction (13)*	*Production (14)*
Seeds	wheat :: crop George Wendt :: star nitrogen :: element diborane :: substance	leader :: panel city :: region ion :: matter oxygen :: water	Khrushchev :: Stalin Carla Hills :: Yeutter Bush :: Reagan Julio Barbosa :: Mendes	magnesium :: oxygen hydrazine :: water aluminum metal :: oxygen lithium metal :: fluorine gas	bright flame :: flares hydrogen :: metal hydrides ammonia :: nitric oxide copper :: brown gas
Espresso	Picasso :: artist tax :: charge protein :: biopolymer HCl :: strong acid	trees :: land material :: FBI report oxygen :: air atom :: molecule	Ford :: Nixon Setrakian :: John Griesemer Camero Cardiel :: Camacho Susan Weiss :: editor	hydrogen :: oxygen Ni :: HCl carbon dioxide :: methane boron :: fluorine	electron :: ions glycerin :: nitroglycerin kidneys :: kidney stones ions :: charge

where pointwise mutual information between instance *i* and pattern *p* is estimated with Google as follows:

$$S_p(i) \approx \frac{|x, p, y|}{|x| \times |y| \times |p|}$$

An instance *i* is rejected if $S(i)$ is smaller than some threshold τ.

Although this filtering may also be applied to reliable patterns, we found this to be detrimental in our experiments since most instances generated by reliable patterns are correct. In *Espresso*, we classify a pattern as generic when it generates more than 10 times the instances of previously accepted reliable patterns.

4 Experimental Results

In this section, we present an empirical comparison of *Espresso* with three state of the art systems on the task of extracting various semantic relations.

4.1 Experimental Setup

We perform our experiments using the following two datasets:

- **TREC**: 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 (Brown et al. 2003).

Each corpus is pre-processed using the Alembic Workbench POS-tagger (Day et al. 1997).

Below we describe the systems used in our empirical evaluation of *Espresso*.

- **RH02**: The algorithm by Ravichandran and Hovy (2002) described in Section 2.
- **GI03**: The algorithm by Girju et al. (2006) described in Section 2.
- **PR04**: The algorithm by Pantel and Ravichandran (2004) described in Section 2.
- **ESP-**: The *Espresso* algorithm using the pattern and instance reliability measures, but without using generic patterns.
- **ESP+**: The full *Espresso* algorithm described in this paper exploiting generic patterns.

For *ESP+*, we experimentally set τ from Section 3.3 to $\tau = 0.4$ for TREC and $\tau = 0.3$ for CHEM by manually inspecting a small set of instances.

Espresso is designed to extract various semantic relations exemplified by a given small set of seed instances. We consider the standard *is-a* and *part-of* relations as well as the following more specific relations:

- *succession*: This relation indicates that a person 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[1]. For example, *ammonia* produces *nitric oxide*. We evaluate this relation on the CHEM corpus.

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

4.2 Precision and Recall

We implemented the systems outlined in Section 4.1, except for GI03, and applied them to the

[1] *Production* is an ambiguous relation; it is intended to be a *causation* relation in the context of chemical reactions.

Table 2. System performance: TREC/*is-a*.

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
ESP+	69,156	36.2%	**8.26**

Table 3. System performance: CHEM/*is-a*.

System	*Instances*	*Precision*[*]	*Rel Recall*[†]
RH02	2556	25.0%	3.76
PR04	108	40.0%	0.25
ESP-	200	**85.0%**	1.00
ESP+	1490	76.0%	**6.66**

Table 4. System performance: TREC/*part-of*.

System	*Instances*	*Precision*[*]	*Rel Recall*[†]
RH02	12,828	35.0%	42.52
ESP-	132	**80.0%**	1.00
ESP+	87,203	69.9%	**577.22**

Table 5. System performance: CHEM/*part-of*.

System	*Instances*	*Precision*[*]	*Rel Recall*[†]
RH02	11,582	33.8%	**58.78**
ESP-	111	**60.0%**	1.00
ESP+	5973	50.7%	45.47

Table 6. System performance: TREC/*succession*.

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

Table 7. System performance: CHEM/*reaction*.

System	*Instances*	*Precision*[*]	*Rel Recall*[†]
RH02	6,083	30%	53.67
ESP-	40	85%	1.00
ESP+	3102	**91.4%**	**89.39**

Table 8. System performance: CHEM/*production*.

System	*Instances*	*Precision*[*]	*Rel Recall*[†]
RH02	197	57.5%	0.80
ESP-	196	**72.5%**	1.00
ESP+	1676	55.8%	**6.58**

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 CHEM corpus) and evaluating their quality manually using two human judges (a total of 680 instances were annotated per judge). For each instance, judges 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 kappa statistic (Siegel and Castellan Jr. 1988) on this task was K = 0.69[2]. The precision for a given set of instances is the sum of the judges' scores divided by the total instances.

Although knowing the total number of correct 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. Following (Pantel et al. 2004), we define the relative recall of system *A* given system *B*, $R_{A|B}$, as:

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

where R_A is the recall of *A*, C_A is the number of correct instances extracted by *A*, *C* is the (unknown) total number of correct instances in the corpus, P_A is *A*'s precision in our experiments, and $|A|$ is the total number of instances discovered by *A*.

Tables 2 – 8 report 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 *ESP-* system. For example, in Table 2, *RH02* has a relative recall of 5.31 with *ESP-*, which means that the *RH02* system outputs 5.31 times more correct relations than *ESP-* (at a cost of much lower precision). Similarly, *PR04* has a relative recall of 0.23 with *ESP-*, which means that *PR04* outputs 4.35 fewer correct relations than *ESP-* (also with a smaller precision). We did not include the results from GI03 in the tables since the system is only applicable to *part-of* relations and we did not reproduce it. However, the authors evaluated their system on a sample of the TREC-9 dataset and reported 83% precision and 72% recall (this algorithm is heavily supervised.)

[2] The kappa statistic jumps to K = 0.79 if we treat partially correct classifications as correct.

[*] Because of the small evaluation sets, we estimate the 95% confidence intervals using bootstrap resampling to be in the order of ± 10-15% (absolute numbers).

[†] Relative recall is given in relation to ESP-.

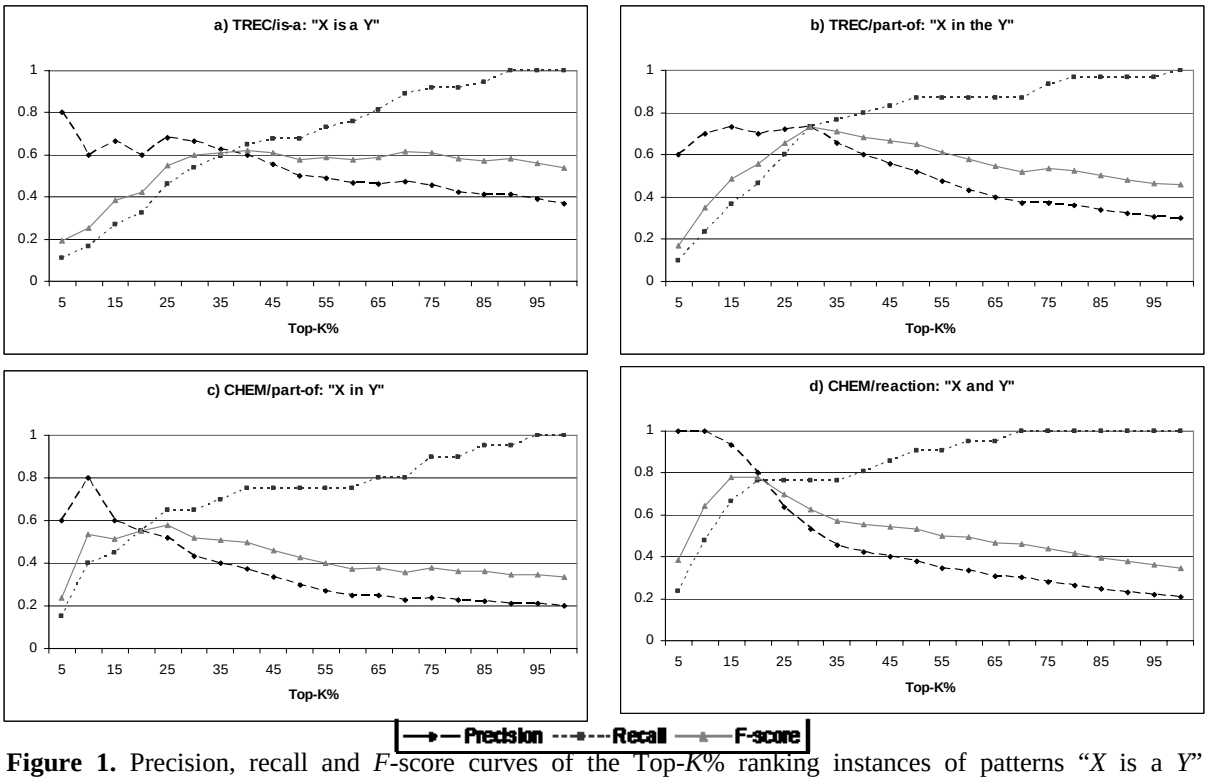

Figure 1. Precision, recall and *F*-score curves of the Top-*K*% ranking instances of patterns "*X* is a *Y*" (TREC/*is-a*), "*X* in *Y*" (TREC/*part-of*), "*X* in the *Y*" (CHEM/*part-of*), and "*X* and *Y*" (CHEM/*reaction*).

In all tables, RH02 extracts many more relations than ESP-, but with a much lower precision, because it uses generic patterns without filtering. The high precision of ESP- is due to the effective reliability measures presented in Section 3.2.

4.3 Effect of Generic Patterns

Experimental results, for all relations and the two different corpus sizes, show that *ESP*- greatly outperforms the other methods on precision. However, without the use of generic patterns, the *ESP*- system shows lower recall in all but the *production* relation.

As hypothesized, exploiting generic patterns using the algorithm from Section 3.3 substantially improves recall without much deterioration in precision. *ESP*+ shows one to two orders of magnitude improvement on recall while losing on average below 10% precision. The *succession* relation in Table 6 was the only relation where *Espresso* found no generic pattern. For other relations, *Espresso* found from one to five generic patterns. Table 4 shows the power of generic patterns where system recall increases by 577 times with only a 10% drop in precision. In Table 7, we see a case where the combination of filtering with a large increase in retrieved instances resulted in both higher precision and recall.

In order to better analyze our use of generic patterns, we performed the following experiment. For each relation, we randomly sampled 100 instances for each generic pattern and built a gold standard for them (by manually tagging each instance as correct or incorrect). We then sorted the 100 instances according to the scoring formula $S(i)$ derived in Section 3.3 and computed the average precision, recall, and *F*-score of each top-*K* ranked instances for each pattern[5]. Due to lack of space, we only present the graphs for four of the 22 generic patterns: "*X* is a *Y*" for the *is-a* relation of Table 2, "*X* in the *Y*" for the *part-of* relation of Table 4, "*X* in *Y*" for the *part-of* relation of Table 5, and "*X* and *Y*" for the *reaction* relation of Table 7. Figure 1 illustrates the results.

In each figure, notice that recall climbs at a much faster rate than precision decreases. This indicates that the scoring function of Section 3.3 effectively separates correct and incorrect instances. In Figure 1a), there is a big initial drop in precision that accounts for the poor precision reported in Table 1.

Recall that the cutoff points on $S(i)$ were set to $\tau = 0.4$ for TREC and $\tau = 0.3$ for CHEM. The figures show that this cutoff is far from the maximum *F*-score. An interesting avenue of future work would be to automatically determine the proper threshold for each individual generic pattern instead of setting a uniform threshold.

[5] We can directly compute recall here since we built a gold standard for each set of 100 samples.

5 Conclusions

We proposed a weakly-supervised, general-purpose, and accurate algorithm, called *Espresso*, for harvesting binary semantic relations from raw text. The main contributions are: i) a method for exploiting generic patterns by filtering incorrect instances using the Web; and ii) a principled measure of pattern and instance reliability enabling the filtering algorithm.

We have empirically compared *Espresso*'s precision and recall with other systems on both a small domain-specific textbook and on a larger corpus of general news, and have extracted several standard and specific semantic relations: *is-a*, *part-of*, *succession*, *reaction*, and *production*. *Espresso* achieves higher and more balanced performance than other state of the art systems. By exploiting generic patterns, system recall substantially increases with little effect on precision.

There are many avenues of future work both in improving system performance and making use of the relations in applications like question answering. For the former, we plan to investigate the use of WordNet to automatically learn selectional constraints on generic patterns, as proposed by (Girju et al. 2006). We expect here that negative instances will play a key role in determining the selectional restrictions.

Espresso is the first system, to our knowledge, to emphasize concurrently *performance*, *minimal supervision*, *breadth*, and *generality*. It remains to be seen whether one could enrich existing ontologies with relations harvested by *Espresso*, and it is our hope that these relations will benefit NLP applications.

References

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

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

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

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

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-97*. Washington D.C.

Downey, D.; Etzioni, O.; and Soderland, S. 2005. A Probabilistic model of redundancy in information extraction. In *Proceedings of IJCAI-05*. pp. 1034-1041. Edinburgh, Scotland.

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.

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

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

Girju, R.; Badulescu, A.; and Moldovan, D. 2006. Automatic Discovery of Part-Whole Relations. *Computational Linguistics*, 32(1): 83-135.

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

Hindle, D. 1990. Noun classification from predicate-argument structures. In *Proceedings of ACL-90*. pp. 268–275. Pittsburgh, PA.

Justeson J.S. and Katz S.M. 1995. Technical Terminology: some linguistic properties and algorithms for identification in text. In *Proceedings of ICCL-95*. pp.539-545. Nantes, France.

Lin, C.-Y. and Hovy, E.H.. 2000. The Automated acquisition of topic signatures for text summarization. In *Proceedings of COLING-00*. pp. 495-501. Saarbrücken, Germany.

Lin, D. and Pantel, P. 2002. Concept discovery from text. In *Proceedings of COLING-02*. pp. 577-583. Taipei, Taiwan.

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

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

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

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.

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.

Riloff, E. and Shepherd, J. 1997. A corpus-based approach for building semantic lexicons. In *Proceedings of EMNLP-97*.

Siegel, S. and Castellan Jr., N. J. 1988. Nonparametric Statistics for the Behavioral Sciences. McGraw-Hill.

Szpektor, I.; Tanev, H.; Dagan, I.; and Coppola, B. 2004. Scaling web-based acquisition of entailment relations. In *Proceedings of EMNLP-04*. Barcelona, Spain.