Learning Arguments and Supertypes of Semantic Relations using Recursive Patterns

Zornitsa Kozareva and Eduard Hovy
USC Information Sciences Institute
4676 Admiralty Way
Marina del Rey, CA 90292-6695
{kozareva,hovy}@isi.edu

Abstract

A challenging problem in open information extraction and text mining is the learning of the selectional restrictions of semantic relations. We propose a minimally supervised bootstrapping algorithm that uses a single seed and a recursive lexico-syntactic pattern to learn the arguments and the supertypes of a diverse set of semantic relations from the Web. We evaluate the performance of our algorithm on multiple semantic relations expressed using "verb", "noun", and "verb prep" lexico-syntactic patterns. Human-based evaluation shows that the accuracy of the harvested information is about 90%. We also compare our results with existing knowledge base to outline the similarities and differences of the granularity and diversity of the harvested knowledge.

1 Introduction

Building and maintaining knowledge-rich resources is of great importance to information extraction, question answering, and textual entailment. Given the endless amount of data we have at our disposal, many efforts have focused on mining knowledge from structured or unstructured text, including ground facts (Etzioni et al., 2005), semantic lexicons (Thelen and Riloff, 2002), encyclopedic knowledge (Suchanek et al., 2007), and concept lists (Katz et al., 2003). Researchers have also successfully harvested relations between entities, such as is-a (Hearst, 1992; Pasca, 2004) and part-of (Girju et al., 2003). The kinds of knowledge learned are generally of two kinds: ground instance facts (*New York is-a city*, *Rome is the capital of Italy*) and general relational types (*city is-a location*, *engines are part-of cars*).

A variety of NLP tasks involving inference or entailment (Zanzotto et al., 2006), including QA (Katz and Lin, 2003) and MT (Mt et al., 1988), require a slightly different form of knowledge, derived from many more relations. This knowledge is usually used to support inference and is expressed as selectional restrictions (Wilks, 1975) (namely, the types of arguments that may fill a given relation, such as *person live-in city* and *airline fly-to location*). Selectional restrictions constrain the possible fillers of a relation, and hence the possible contexts in which the patterns expressing that relation can participate in, thereby enabling sense disambiguation of both the fillers and the expression itself.

To acquire this knowledge two common approaches are employed: clustering and patterns. While clustering has the advantage of being fully unsupervised, it may or may not produce the types and granularity desired by a user. In contrast pattern-based approaches are more precise, but they typically require a handful to dozens of seeds and lexico-syntactic patterns to initiate the learning process. In a closed domain these approaches are both very promising, but when tackling an unbounded number of relations they are unrealistic. The quality of clustering decreases as the domain becomes more continuously varied and diverse, and it has proven difficult to create collections of effective patterns and high-yield seeds manually.

In addition, the output of most harvesting systems is a flat list of lexical semantic expressions such as *"New York is-a city"* and *"virus causes flu"*. However, using this knowledge in inference requires it to be formulated appropriately and organized in a semantic repository. (Pennacchiotti and Pantel, 2006) proposed an algorithm for automatically ontologizing semantic relations into WordNet. However, despite its high precision entries, WordNet's limited coverage makes it impossible for relations whose arguments are not present in WordNet to be incorporated. One would like a procedure that dynamically organizes and extends

Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, pages 1482–1491,
Uppsala, Sweden, 11-16 July 2010. ©2010 Association for Computational Linguistics

its semantic repository in order to be able to accommodate all newly-harvested information, and thereby become a global semantic repository.

Given these considerations, we address in this paper the following question: *How can the selectional restrictions of semantic relations be learned automatically from the Web with minimal effort using lexico-syntactic recursive patterns?*

The contributions of the paper are as follows:

- A novel representation of semantic relations using recursive lexico-syntactic patterns.
- An automatic procedure to learn the selectional restrictions (arguments and supertypes) of semantic relations from Web data.
- An exhaustive human-based evaluation of the harvested knowledge.
- A comparison of the results with some large existing knowledge bases.

The rest of the paper is organized as follows. In the next section, we review related work. Section 3 addresses the representation of semantic relations using recursive patterns. Section 4 describes the bootstrapping mechanism that learns the selectional restrictions of the relations. Section 5 describes data collection. Section 6 discusses the obtained results. Finally, we conclude in Section 7.

2 Related Work

A substantial body of work has been done in attempts to harvest bits of semantic information, including: semantic lexicons (Riloff and Shepherd, 1997), concept lists (Lin and Pantel, 2002), is-a relations (Hearst, 1992; Etzioni et al., 2005; Pasca, 2004; Kozareva et al., 2008), part-of relations (Girju et al., 2003), and others. Knowledge has been harvested with varying success both from structured text such as Wikipedia's infoboxes (Suchanek et al., 2007) or unstructured text such as the Web (Pennacchiotti and Pantel, 2006; Yates et al., 2007). A variety of techniques have been employed, including clustering (Lin and Pantel, 2002), co-occurrence statistics (Roark and Charniak, 1998), syntactic dependencies (Pantel and Ravichandran, 2004), and lexico-syntactic patterns (Riloff and Jones, 1999; Fleischman and Hovy, 2002; Thelen and Riloff, 2002).

When research focuses on a particular relation, careful attention is paid to the pattern(s) that express it in various ways (as in most of the work above, notably (Riloff and Jones, 1999)). But it has proven a difficult task to manually find effectively different variations and alternative patterns for each relation. In contrast, when research focuses on *any* relation, as in TextRunner (Yates et al., 2007), there is no standardized manner for re-using the pattern learned. TextRunner scans sentences to obtain relation-independent lexico-syntactic patterns to extract triples of the form (*John*, *fly to*, *Prague*). The middle string denotes some (unspecified) semantic relation while the first and third denote the learned arguments of this relation. But TextRunner does not seek specific semantic relations, and does not re-use the patterns it harvests with different arguments in order to extend their yields.

Clearly, it is important to be able to specify both the actual semantic relation sought *and* use its textual expression(s) in a controlled manner for maximal benefit.

The objective of our research is to combine the strengths of the two approaches, and, in addition, to provide even richer information by automatically mapping each harvested argument to its supertype(s) (i.e., its semantic concepts). For instance, given the relation *destination* and the pattern *X flies to Y*, automatically determining that *John, Prague*) and (*John, conference*) are two valid filler instance pairs, that (*RyanAir, Prague*) is another, as well as that *person* and *airline* are supertypes of the first argument and *city* and *event* of the second. This information provides the selectional restrictions of the given semantic relation, indicating that living things like people can fly to cities and events, while non-living things like airlines fly mainly to cities. This is a significant improvement over systems that output a flat list of lexical semantic knowledge (Thelen and Riloff, 2002; Yates et al., 2007; Suchanek et al., 2007).

Knowing the sectional restrictions of a semantic relation supports inference in many applications, for example enabling more accurate information extraction. (Igo and Riloff, 2009) report that patterns like "*attack on* $\langle NP \rangle$" can learn undesirable words due to idiomatic expressions and parsing errors. Over time this becomes problematic for the bootstrapping process and leads to significant deterioration in performance. (Thelen and Riloff, 2002) address this problem by learning multiple semantic categories simultaneously, relying on the often unrealistic assumption that a word cannot belong to more than one semantic category. How-

ever, if we have at our disposal a repository of semantic relations with their selectional restrictions, the problem addressed in (Igo and Riloff, 2009) can be alleviated.

In order to obtain selectional restriction classes, (Pennacchiotti and Pantel, 2006) made an attempt to ontologize the harvested arguments of *is-a*, *part-of*, and *cause* relations. They mapped each argument of the relation into WordNet and identified the senses for which the relation holds. Unfortunately, despite its very high precision entries, WordNet is known to have limited coverage, which makes it impossible for algorithms to map the content of a relation whose arguments are not present in WordNet. To surmount this limitation, we do not use WordNet, but employ a different method of obtaining superclasses of a filler term: the inverse doubly-anchored patterns DAP^{-1} (Hovy et al., 2009), which, given two arguments, harvests its supertypes from the source corpus. (Hovy et al., 2009) show that DAP^{-1} is reliable and it enriches WordNet with additional hyponyms and hypernyms.

3 Recursive Patterns

A singly-anchored pattern contains one example of the seed term (the anchor) and one open position for the term to be learned. Most researchers use singly-anchored patterns to harvest semantic relations. Unfortunately, these patterns run out of steam very quickly. To surmount this obstacle, a handful of seeds is generally used, and helps to guarantee diversity in the extraction of new lexico-syntactic patterns (Riloff and Jones, 1999; Snow et al., 2005; Etzioni et al., 2005).

Some algorithms require ten seeds (Riloff and Jones, 1999; Igo and Riloff, 2009), while others use a variation of 5, 10, to even 25 seeds (Talukdar et al., 2008). Seeds may be chosen at random (Davidov et al., 2007; Kozareva et al., 2008), by picking the most frequent terms of the desired class (Igo and Riloff, 2009), or by asking humans (Pantel et al., 2009). As (Pantel et al., 2009) show, picking seeds that yield high numbers of different terms is difficult. Thus, when dealing with unbounded sets of relations (Banko and Etzioni, 2008), providing many seeds becomes unrealistic.

Interestingly, recent work reports a class of patterns that use only one seed to learn as much information with only one seed. (Kozareva et al., 2008; Hovy et al., 2009) introduce the so-called doubly-anchored pattern (DAP) that has two anchor seed positions "⟨*type*⟩ *such as* ⟨*seed*⟩ *and* *", plus one open position for the terms to be learned. Learned terms can then be replaced into the seed position automatically, creating a recursive procedure that is reportedly much more accurate and has much higher final yield. (Kozareva et al., 2008; Hovy et al., 2009) have successfully applied DAP for the learning of hyponyms and hypernyms of *is-a* relations and report improvements over (Etzioni et al., 2005) and (Pasca, 2004).

Surprisingly, this work was limited to the semantic relation *is-a*. No other study has described the use or effect of recursive patterns for different semantic relations. Therefore, going beyond (Kozareva et al., 2008; Hovy et al., 2009), we here introduce recursive patterns other than DAP that use only one seed to harvest the arguments and supertypes of a wide variety of relations.

(Banko and Etzioni, 2008) show that semantic relations can be expressed using a handful of relation-independent lexico-syntactic patterns. Practically, we can turn *any* of these patterns into recursive form by giving as input only one of the arguments and leaving the other one as an open slot, allowing the learned arguments to replace the initial seed argument directly. For example, for the relation "*fly to*", the following recursive patterns can be built: "* *and* ⟨*seed*⟩ *fly to* *", "⟨*seed*⟩ *and* * *fly to* *", "* *fly to* ⟨*seed*⟩ *and* *", "* *fly to* * *and* ⟨*seed*⟩", "⟨*seed*⟩ *fly to* *" or "* *fly to* ⟨*seed*⟩", where ⟨*seed*⟩ is an example like *John* or *Ryanair*, and (∗) indicates the position on which the arguments are learned. Conjunctions like *and*, *or* are useful because they express list constructions and extract arguments similar to the *seed*. Potentially, one can explore all recursive pattern variations when learning a relation and compare their yield, however this study is beyond the scope of this paper.

We are particularly interested in the usage of recursive patterns for the learning of semantic relations not only because it is a novel method, but also because recursive patterns of the DAP fashion are known to: (1) learn concepts with high precision compared to singly-anchored patterns (Kozareva et al., 2008), (2) use only one seed instance for the discovery of new previously unknown terms, and (3) harvest knowledge with minimal supervision.

4 Bootstrapping Recursive Patterns

4.1 Problem Formulation

The main goal of our research is:

> **Task Definition**: Given a seed and a semantic relation expressed using a recursive lexico-syntactic pattern, learn in bootstrapping fashion the selectional restrictions (i.e., the arguments and supertypes) of the semantic relation from an unstructured corpus such as the Web.

Figure 1 shows an example of the task and the types of information learned by our algorithm.

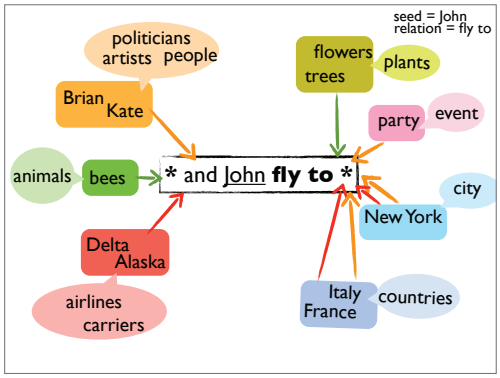

Figure 1: *Bootstrapping Recursive Patterns.*

Given a seed *John* and a semantic relation *fly to* expressed using the recursive pattern "* *and John fly to* *", our algorithm learns the left side arguments {*Brian, Kate, bees, Delta, Alaska*} and the right side arguments {*flowers, trees, party, New York, Italy, France*}. For each argument, the algorithm harvests supertypes such as {*people, artists, politicians, airlines, city, countries, plants, event*} among others. The colored links between the right and left side concepts denote the selectional restrictions of the relation. For instance, *people* fly to *events* and *countries*, but never to *trees* or *flowers*.

4.2 System Architecture

We propose a minimally supervised bootstrapping algorithm based on the framework adopted in (Kozareva et al., 2008; Hovy et al., 2009). The algorithm has two phases: *argument* harvesting and *supertype* harvesting. The final output is a ranked list of interlinked concepts which captures the selectional restrictions of the relation.

4.2.1 Argument Harvesting

In the argument extraction phase, the first bootstrapping iteration is initiated with a seed Y and a recursive pattern "X^* *and* Y *verb+prep|verb|noun* Z^*", where X^* and Z^* are the placeholders for the arguments to be learned. The pattern is submitted to Yahoo! as a web query and all unique snippets matching the query are retrieved. The newly learned and previously unexplored arguments on the X^* position are used as seeds in the subsequent iteration. The arguments on the Z^* position are stored at each iteration, but never used as seeds since the recursivity is created using the terms on X and Y. The bootstrapping process is implemented as an exhaustive breadth-first algorithm which terminates when all arguments are explored.

We noticed that despite the specific lexico-syntactic structure of the patterns, erroneous information can be acquired due to part-of-speech tagging errors or flawed facts on the Web. The challenge is to identify and separate the erroneous from the true arguments. We incorporate the harvested arguments on X and Y positions in a directed graph $G = (V, E)$, where each vertex $v \in V$ is a candidate argument and each edge $(u, v) \in E$ indicates that the argument v is generated by the argument u. An edge has weight w corresponding to the number of times the pair (u, v) is extracted from different snippets. A node u is ranked by $u = \frac{\sum_{\forall(u,v)\in E} w(u,v) + \sum_{\forall(v,u)\in E} w(v,u)}{|V|-1}$ which represents the weighted sum of the outgoing and incoming edges normalized by the total number of nodes in the graph. Intuitively, our confidence in a correct argument u increases when the argument (1) discovers and (2) is discovered by many different arguments.

Similarly, to rank the arguments standing on the Z position, we build a bipartite graph $G' = (V', E')$ that has two types of vertices. One set of vertices represents the arguments found on the Y position in the recursive pattern. We will call these V_y. The second set of vertices represents the arguments learned on the Z position. We will call these V_z. We create an edge $e'(u', v') \in E'$ between $u' \in V_y$ and $v' \in V_z$ when the argument on the Z position represented by v' was harvested by the argument on the Y position represented by u'. The weight w' of the edge indicates the number of times an argument on the Y position found Z. Vertex v' is ranked as $v' = \frac{\sum_{\forall(u',v')\in E'} w(u',v')}{|V'|-1}$. In a very large corpus, like the Web, we assume that a correct argument Z is the one that is frequently discovered by various arguments Y.

4.2.2 Supertype Harvesting

In the supertype extraction phase, we take all <X,Y> argument pairs collected during the argument harvesting stage and instantiate them in the inverse DAP^{-1} pattern *"* such as X and Y"*. The query is sent to Yahoo! as a web query and all 1000 snippets matching the pattern are retrieved. For each <X,Y> pair, the terms on the (*) position are extracted and considered as candidate supertypes.

To avoid the inclusion of erroneous supertypes, again we build a bipartite graph $G'' = (V'', E'')$. The set of vertices V_{sup} represents the supertypes, while the set of vertices V_p corresponds to the $\langle$X,Y$\rangle$ pair that produced the supertype. An edge $e''(u'', v'') \in E''$, where $u'' \in V_p$ and $v'' \in V_{sup}$ shows that the pair $\langle$X,Y$\rangle$ denoted as u'' harvested the supertype represented by v''.

For example, imagine that the argument *X**= *Ryanair* was harvested in the previous phase by the recursive pattern *"X* and EasyJet fly to Z*"*. Then the pair $\langle$*Ryanair,EasyJet*$\rangle$ forms a new Web query *"* such as Ryanair and EasyJet"* which learns the supertypes *"airlines"* and *"carriers"*. The bipartite graph has two vertices v''_1 and v''_2 for the supertypes *"airlines"* and *"carriers"*, one vertex u''_3 for the argument pair $\langle$*Ryanair, EasyJet*$\rangle$, and two edges $e''_1(u''_3, v''_1)$ and $e''_2(u''_3, v''_1)$. A vertex $v'' \in V_{sup}$ is ranked by v''=$\frac{\sum_{\forall(u'',v'')\in E''} w(u'',v'')}{|V''|-1}$. Intuitively, a supertype which is discovered multiple times by various argument pairs is ranked highly.

However, it might happen that a highly ranked supertype actually does not satisfy the selectional restrictions of the semantic relation. To avoid such situations, we further instantiate each supertype concept in the original pattern[1]. For example, *"aircompanies fly to *"* and *"carriers fly to *"*. If the candidate supertype produces many web hits for the query, then this suggests that the term is a relevant supertype.

Unfortunately, to learn the supertypes of the Z arguments, currently we have to form all possible combinations among the top 150 highly ranked concepts, because these arguments have not been learned through pairing. For each pair of Z arguments, we repeat the same procedure as described above.

[1]Except for the *"dress"* and *"person"* relations, where the targeted arguments are adjectives, and the supertypes are nouns.

5 Semantic Relations

So far, we have described the mechanism that learns from one seed and a recursive pattern the selectional restrictions of any semantic relation. Now, we are interested in evaluating the performance of our algorithm. A natural question that arises is: *"How many patterns are there?"*. (Banko and Etzioni, 2008) found that 95% of the semantic relations can be expressed using eight lexico-syntactic patterns. Space prevents us from describing all of them, therefore we focus on the three most frequent patterns which capture a large diversity of semantic relations. The relative frequency of these patterns is 37.80% for "verbs", 22.80% for "noun prep", and 16.00% for "verb prep".

5.1 Data Collection

Table 1 shows the lexico-syntactic pattern and the initial seed we used to express each semantic relation. To collect data, we ran our knowledge harvesting algorithm until complete exhaustion. For each query submitted to Yahoo!, we retrieved the top 1000 web snippets and kept only the unique ones. In total, we collected 30GB raw data which was part-of-speech tagged and used for the argument and supertype extraction. Table 1 shows the obtained results.

recursive pattern	seed	X arg	Z arg	#iter
X and Y **work for** Z	Charlie	2949	3396	20
X and Y **fly to** Z	EasyJet	772	1176	19
X and Y **go to** Z	Rita	18406	27721	13
X and Y **work in** Z	John	4142	4918	13
X and Y **work on** Z	Mary	4126	5186	7
X and Y **work at** Z	Scott	1084	1186	14
X and Y **live in** Z	Harry	8886	19698	15
X and Y **live at** Z	Donald	1102	1175	15
X and Y **live with** Z	Peter	1344	834	11
X and Y **cause** Z	virus	12790	52744	19
X and Y **celebrate**	Jim	6033	–	12
X and Y **drink**	Sam	1810	–	13
X and Y **dress**	nice	1838	–	8
X and Y **person**	scared	2984	–	17

Table 1: *Total Number of Harvested Arguments.*

An interesting characteristic of the recursive patterns is the speed of leaning which can be measured in terms of the number of unique arguments acquired during each bootstrapping iteration. Figure 2 shows the bootstrapping process for the *"cause"* and *"dress"* relations. Although both relations differ in terms of the total number of iterations and harvested items, the overall behavior of the learning curves is similar. Learning starts of very slowly and as bootstrapping progresses a

rapid growth is observed until a saturation point is reached.

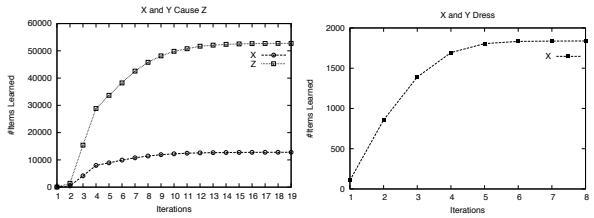

Figure 2: Items extracted in 10 iterations.

The speed of leaning is related to the connectivity behavior of the arguments of the relation. Intuitively, a densely connected graph takes shorter time (i.e., fewer iterations) to be learned, as in the *"work on"* relation, while a weakly connected network takes longer time to harvest the same amount of information, as in the *"work for"* relation.

6 Results

In this section, we evaluate the results of our knowledge harvesting algorithm. Initially, we decided to conduct an automatic evaluation comparing our results to knowledge bases that have been extracted in a similar way (i.e., through pattern application over unstructured text). However, it is not always possible to perform a complete comparison, because either researchers have not fully explored the same relations we have studied, or for those relations that overlap, the gold standard data was not available.

The online demo of TextRunner[2] (Yates et al., 2007) actually allowed us to collect the arguments for all our semantic relations. However, due to Web based query limitations, TextRunner returns only the first 1000 snippets. Since we do not have the complete and ranked output of TextRunner, comparing results in terms of recall and precision is impossible.

Turning instead to results obtained from structured sources (which one expects to have high correctness), we found that two of our relations overlap with those of the freely available ontology Yago (Suchanek et al., 2007), which was harvested from the Infoboxes tables in Wikipedia. In addition, we also had two human annotators judge as many results as we could afford, to obtain Precision. We conducted two evaluations, one for the arguments and one for the supertypes.

[2]http://www.cs.washington.edu/research/textrunner/

6.1 Human-Based Argument Evaluation

In this section, we discuss the results of the harvested arguments. For each relation, we selected the top 200 highly ranked arguments. We hired two annotators to judge their correctness. We created detailed annotation guidelines that define the labels for the arguments of the relations, as shown in Table 2. (Previously, for the same task, researchers have not conducted such an exhaustive and detailed human-based evaluation.) The annotation was conducted using the CAT system[3].

TYPE	LABEL	EXAMPLES
Correct	Person	*John, Mary*
	Role	*mother, president*
	Group	*team, Japanese*
	Physical	*yellow, shabby*
	NonPhysical	*ugly, thought*
	NonLiving	*airplane*
	Organization	*IBM, parliament*
	Location	*village, New York, in the house*
	Time	*at 5 o'clock*
	Event	*party, prom, earthquake*
	State	*sick, anrgy*
	Manner	*live in happiness*
	Medium	*work on Linux, Word*
	Fixed phrase	*go to war*
Incorrect	Error	*wrong part-of-speech tag*
	Other	*none of the above*

Table 2: *Annotation Labels.*

We allow multiple labels to be assigned to the same concept, because sometimes the concept can appear in different contexts that carry various conceptual representations. Although the labels can be easily collapsed to judge correct and incorrect terms, the fine-grained annotation shown here provides a better overview of the information learned by our algorithm.

We measured the inter-annotator agreement for all labels and relations considering that a single entry can be tagged with multiple labels. The Kappa score is around 0.80. This judgement is good enough to warrant using these human judgements to estimate the accuracy of the algorithm. We compute *Accuracy* as the number of examples tagged as $Correct$ divided by the total number of examples.

Table 4 shows the obtained results. The overall accuracy of the argument harvesting phase is 91%. The majority of the occurred errors are due to part-of-speech tagging. Table 3 shows a sample of 10 randomly selected examples from the top 200 ranked and manually annotated arguments.

[3]http://cat.ucsur.pitt.edu/default.aspx

Relation	*Arguments*
(X) Dress:	stylish, comfortable, expensive, shabby, gorgeous silver, clean, casual, Indian, black
(X) Person:	honest, caring, happy, intelligent, gifted friendly, responsible, mature, wise, outgoing
(X) Cause:	pressure, stress, fire, bacteria, cholesterol flood, ice, cocaine, injuries, wars
GoTo (Z):	school, bed, New York, the movies, the park, a bar the hospital, the church, the mall, the beach
LiveIn (Z):	peace, close proximity, harmony, Chicago, town New York, London, California, a house, Australia
WorkFor (Z):	a company, the local prison, a gangster, the show a boss, children, UNICEF, a living, Hispanics

Table 3: *Examples of Harvested Arguments.*

6.2 Comparison against Existing Resources

In this section, we compare the performance of our approach with the semantic knowledge base Yago[4] that contains 2 million entities[5], 95% of which were manually confirmed to be correct. In this study, we compare only the unique arguments of the *"live in"* and *"work at"* relations. We provide Precision scores using the following measures:

$$Pr_{Yago} = \frac{\#terms\ found\ in\ Yago}{\#terms\ harvested\ by\ system}$$

$$Pr_{Human} = \frac{\#terms\ judged\ correct\ by\ human}{\#terms\ harvested\ by\ system}$$

$$NotInYago = \#terms\ judged\ correct\ by\ human\ but\ not\ in\ Yago$$

Table 5 shows the obtained results.

We carefully analyzed those arguments that were found by one of the systems but were missing in the other. The recursive patterns learn information about non-famous entities like Peter and famous entities like Michael Jordan. In contrast, Yago contains entries mostly about famous entities, because this is the predominant knowledge in Wikipedia. For the *"live in"* relation, both repositories contain the same city and country names. However, the recursive pattern learned arguments like *pain*, *effort* which express a manner of living, and locations like *slums*, *box*. This information is missing from Yago. Similarly for the *"work at"* relation, both systems learned that people work at universities. In addition, the recursive pattern learned a diversity of company names absent from Yago.

While it is expected that our algorithm finds many terms not contained in Yago—specifically, the information not deemed worthy of inclusion in Wikipedia—we are interested in the relatively large number of terms contained in Yago but not found by our algorithm. To our knowledge, no

[4]http://www.mpi-inf.mpg.de/yago-naga/yago/

[5]Names of cities, people, organizations among others.

X WorkFor	A1	A2	WorkFor **Z**	A1	A2
Person	148	152	Organization	111	110
Role	5	7	Person	60	60
Group	12	14	Event	4	2
Organization	8	7	Time	4	5
NonPhysical	22	23	NonPhysical	18	19
Other	5	5	Other	3	4
Acc.	.98	.98	Acc.	.99	.98

X Cause	A1	A2	Cause **Z**	A1	A2
PhysicalObj	82	75	PhysicalObj	15	20
NonPhysicalObj	69	66	NonPhysicalObj	89	91
Event	21	24	Event	72	72
State	29	31	State	50	50
Other	3	4	Other	5	4
Acc.	.99	.98	Acc.	.98	.98

X GoTo	A1	A2	GoTo **Z**	A1	A2
Person	190	188	Location	163	155
Role	4	4	Event	21	30
Group	3	3	Person	11	13
NonPhysical	1	3	NonPhysical	2	1
Other	2	2	Other	3	1
Acc.	.99	.99	Acc.	.99	.99

X FlyTo	A1	A2	FlyTo **Z**	A1	A2
Person	140	139	Location	199	198
Organization	54	57	Event	1	2
NonPhysical	2	2	Person	0	0
Other	4	2	Other	0	0
Acc.	.98	.99	Acc.	1	1

X WorkOn	A1	A2	WorkOn **Z**	A1	A2
Person	173	172	Location	110	108
Role	2	3	Organization	27	25
Group	4	5	Manner	38	40
Organization	6	6	Time	4	4
NonPhysical	15	14	NonPhysical	18	21
Error	1	1	Medium	8	8
Other	1	1	Other	13	15
Acc.	.99	.99	Acc.	.94	.93

X WorkIn	A1	A2	WorkIn **Z**	A1	A2
Person	117	118	Location	104	111
Group	10	9	Organization	10	25
Organization	3	3	Manner	39	40
Fixed	3	1	Time	4	4
NonPhysical	55	59	NonPhysical	22	21
Error	12	10	Medium	8	8
Other	0	0	Error	13	15
Acc.	.94	.95	Acc.	.94	.93

X WorkAt	A1	A2	WorkAt **Z**	A1	A2
Person	193	192	Organization	189	190
Role	1	1	Manner	5	4
Group	1	1	Time	3	3
Organization	0	0	Error	3	2
Other	5	6	Other	0	1
Acc.	.98	.97	Acc.	.99	.99

X LiveIn	A1	A2	LiveIn **Z**	A1	A2
Person	185	185	Location	182	186
Role	3	4	Manner	6	8
Group	9	8	Time	1	2
NonPhysical	1	2	Fixed	5	2
Other	2	1	Other	6	2
Acc.	.99	.99	Acc.	.97	.99

X LiveAt	A1	A2	LiveAt **Z**	A1	A2
Person	196	195	Location	158	157
Role	1	1	Person	5	7
NonPhysical	0	1	Manner	1	2
Other	3	3	Error	36	34
Acc.	.99	.99	Acc.	.82	.83

X LiveWith	A1	A2	LiveWith **Z**	A1	A2
Person	188	187	Person	165	163
Role	6	6	Animal	2	4
Group	2	2	Manner	15	15
NonPhysical	2	3	NonPhysical	15	15
Other	2	2	Other	3	3
Acc.	.99	.99	Acc.	.99	.99

X Dress	A1	A2	**X** Person	A1	A2
Physical	72	59	Physical	8	2
NonPhysical	120	136	NonPhysical	188	194
Other	8	5	Other	4	4
Acc	.96	.98	Acc.	.98	.98

X Drink	A1	A2	**X** Celebrate	A1	A2
Living	165	174	Living	157	164
NonLiving	8	2	NonLiving	42	35
Error	27	24	Error	1	1
Acc	.87	.88	Acc.	.99	.99

Table 4: *Harvested Arguments.*

	Pr_{Yago}	Pr_{Human}	NotInYago
X LiveIn	.19 (2863/14705)	.58 (5165)/8886	2302
LiveIn **Z**	.10 (495/4754)	.72 (14248)/19698	13753
X WorkAt	.12(167/1399)	.88 (959)/1084	792
WorkAt **Z**	.3(15/525)	.95 (1128)/1186	1113

Table 5: *Comparison against Yago.*

other automated harvesting algorithm has ever been compared to Yago, and our results here form a baseline that we aim to improve upon. And in the future, one can build an extensive knowledge harvesting system combining the wisdom of the crowd and Wikipedia.

6.3 Human-Based Supertype Evaluation

In this section, we discuss the results of harvesting the supertypes of the learned arguments. Figure 3 shows the top 100 ranked supertypes for the *"cause"* and *"work on"* relations. The x-axis indicates a supertype, the y-axis denotes the number of different argument pairs that lead to the discovery of the supertype.

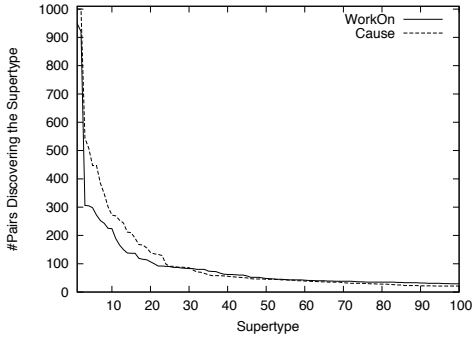

Figure 3: Ranked Supertypes.

The decline of the curve indicates that certain supertypes are preferred and shared among different argument pairs. It is interesting to note that the text on the Web prefers a small set of supertypes, and to see what they are. These most-popular harvested types tend to be the more descriptive terms. The results indicate that one does not need an elaborate supertype hierarchy to handle the selectional restrictions of semantic relations.

Since our problem definition differs from available related work, and WordNet does not contain all harvested arguments as shown in (Hovy et al., 2009), it is not possible to make a direct comparison. Instead, we conduct a manual evaluation of the most highly ranked supertypes which normally are the top 20. The overall accuracy of the supertypes for all relations is 92%. Table 6 shows the top 10 highly ranked supertypes for six of our relations.

Relation	*Arguments*
(Sup_x) **Celebrate**:	men, people, nations, angels, workers, children countries, teams, parents, teachers
(Sup_x) **Dress**:	colors, effects, color tones, activities, patterns styles, materials, size, languages, aspects
(Sup_x) **FlyTo**:	airlines, carriers, companies, giants, people competitors, political figures, stars, celebs
Cause (Sup_z):	diseases, abnormalities, disasters, processes, isses disorders, discomforts, emotions, defects, symptoms
WorkFor (Sup_z)	organizations, industries, people, markets, men automakers, countries, departments, artists, media
GoTo (Sup_z) :	countries, locations, cities, people, events men, activities, games, organizations,
FlyTo (Sup_z)	places, countries, regions, airports, destinations locations, cities, area, events

Table 6: *Examples of Harvested Supertypes.*

7 Conclusion

We propose a minimally supervised algorithm that uses only one seed example and a recursive lexico-syntactic pattern to learn in bootstrapping fashion the selectional restrictions of a large class of semantic relations. The principal contribution of the paper is to demonstrate that this kind of pattern can be applied to almost any kind of semantic relation, as long as it is expressible in a concise surface pattern, and that the recursive mechanism that allows each newly acquired term to restart harvesting automatically is a significant advance over patterns that require a handful of seeds to initiate the learning process. It also shows how one can combine free-form but undirected pattern-learning approaches like TextRunner with more-controlled but effort-intensive approaches like commonly used.

In our evaluation, we show that our algorithm is capable of extracting high quality non-trivial information from unstructured text given very restricted input (one seed). To measure the performance of our approach, we use various semantic relations expressed with three lexico-syntactic patterns. For two of the relations, we compare results with the freely available ontology Yago, and conduct a manual evaluation of the harvested terms.

We will release the annotated and the harvested data to the public to be used for comparison by other knowledge harvesting algorithms.

The success of the proposed framework opens many challenging directions. We plan to use the algorithm described in this paper to learn the selectional restrictions of numerous other relations, in order to build a rich knowledge repository

that can support a variety of applications, including textual entailment, information extraction, and question answering.

Acknowledgments

This research was supported by DARPA contract number FA8750-09-C-3705.

References

Michele Banko and Oren Etzioni. 2008. The tradeoffs between open and traditional relation extraction. In *Proceedings of ACL-08: HLT*, pages 28–36, June.

Dmitry Davidov, Ari Rappoport, and Moshel Koppel. 2007. Fully unsupervised discovery of concept-specific relationships by web mining. In *Proc. of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 232–239, June.

Oren Etzioni, Michael Cafarella, Doug Downey, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S. Weld, and Alexander Yates. 2005. Unsupervised named-entity extraction from the web: an experimental study. *Artificial Intelligence*, 165(1):91–134, June.

Michael Fleischman and Eduard Hovy. 2002. Fine grained classification of named entities. In *Proceedings of the 19th international conference on Computational linguistics*, pages 1–7.

Roxana Girju, Adriana Badulescu, and Dan Moldovan. 2003. Learning semantic constraints for the automatic discovery of part-whole relations. In *Proc. of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 1–8.

Marti Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proc. of the 14th conference on Computational linguistics*, pages 539–545.

Eduard Hovy, Zornitsa Kozareva, and Ellen Riloff. 2009. Toward completeness in concept extraction and classification. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 948–957.

Sean Igo and Ellen Riloff. 2009. Corpus-based semantic lexicon induction with web-based corroboration. In *Proceedings of the Workshop on Unsupervised and Minimally Supervised Learning of Lexical Semantics*.

Boris Katz and Jimmy Lin. 2003. Selectively using relations to improve precision in question answering. In *In Proceedings of the EACL-2003 Workshop on Natural Language Processing for Question Answering*, pages 43–50.

Boris Katz, Jimmy Lin, Daniel Loreto, Wesley Hildebrandt, Matthew Bilotti, Sue Felshin, Aaron Fernandes, Gregory Marton, and Federico Mora. 2003. Integrating web-based and corpus-based techniques for question answering. In *Proceedings of the twelfth text retrieval conference (TREC)*, pages 426–435.

Zornitsa Kozareva, Ellen Riloff, and Eduard Hovy. 2008. Semantic class learning from the web with hyponym pattern linkage graphs. In *Proceedings of ACL-08: HLT*, pages 1048–1056.

Dekang Lin and Patrick Pantel. 2002. Concept discovery from text. In *Proc. of the 19th international conference on Computational linguistics*, pages 1–7.

Characteristics Of Mt, John Lehrberger, Laurent Bourbeau, Philadelphia John Benjamins, and Rita Mccardell. 1988. *Machine Translation: Linguistic Characteristics of Mt Systems and General Methodology of Evaluation*. John Benjamins Publishing Co(1988-03).

Patrick Pantel and Deepak Ravichandran. 2004. Automatically labeling semantic classes. In *Proc. of Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 321–328.

Patrick Pantel, Eric Crestan, Arkady Borkovsky, Ana-Maria Popescu, and Vishnu Vyas. 2009. Web-scale distributional similarity and entity set expansion. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 938–947, August.

Marius Pasca. 2004. Acquisition of categorized named entities for web search. In *Proc. of the thirteenth ACM international conference on Information and knowledge management*, pages 137–145.

Marco Pennacchiotti and Patrick Pantel. 2006. Ontologizing semantic relations. In *ACL-44: Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 793–800.

Ellen Riloff and Rosie Jones. 1999. Learning dictionaries for information extraction by multi-level bootstrapping. In *AAAI '99/IAAI '99: Proceedings of the Sixteenth National Conference on Artificial intelligence.*

Ellen Riloff and Jessica Shepherd. 1997. A Corpus-Based Approach for Building Semantic Lexicons. In *Proc. of the Second Conference on Empirical Methods in Natural Language Processing*, pages 117–124.

Brian Roark and Eugene Charniak. 1998. Noun-phrase co-occurrence statistics for semiautomatic semantic lexicon construction. In *Proceedings of the 17th international conference on Computational linguistics*, pages 1110–1116.

Rion Snow, Daniel Jurafsky, and Andrew Y. Ng. 2005. Learning syntactic patterns for automatic hypernym discovery. In *Advances in Neural Information Processing Systems 17*, pages 1297–1304. MIT Press.

Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. In *WWW '07: Proceedings of the 16th international conference on World Wide Web*, pages 697–706.

Partha Pratim Talukdar, Joseph Reisinger, Marius Pasca, Deepak Ravichandran, Rahul Bhagat, and Fernando Pereira. 2008. Weakly-supervised acquisition of labeled class instances using graph random walks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP 2008*, pages 582–590.

Michael Thelen and Ellen Riloff. 2002. A Bootstrapping Method for Learning Semantic Lexicons Using Extraction Pattern Contexts. In *Proc. of the 2002 Conference on Empirical Methods in Natural Language Processing*, pages 214–221.

Yorick Wilks. 1975. A preferential pattern-seeking, semantics for natural language inference. *Artificial Intelligence*, 6(1):53–74.

Alexander Yates, Michael Cafarella, Michele Banko, Oren Etzioni, Matthew Broadhead, and Stephen Soderland. 2007. Textrunner: open information extraction on the web. In *NAACL '07: Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations on XX*, pages 25–26.

Fabio Massimo Zanzotto, Marco Pennacchiotti, and Maria Teresa Pazienza. 2006. Discovering asymmetric entailment relations between verbs using selectional preferences. In *ACL-44: Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 849–856.