Identifying Relations for Open Information Extraction

Anthony Fader, Stephen Soderland, and Oren Etzioni
University of Washington, Seattle
{afader,soderlan,etzioni}@cs.washington.edu

Abstract

Open Information Extraction (IE) is the task of extracting assertions from massive corpora without requiring a pre-specified vocabulary. This paper shows that the output of state-of-the-art Open IE systems is rife with uninformative and incoherent extractions. To overcome these problems, we introduce two simple syntactic and lexical constraints on binary relations expressed by verbs. We implemented the constraints in the REVERB Open IE system, which more than doubles the area under the precision-recall curve relative to previous extractors such as TEXTRUNNER and WOE^{pos}. More than 30% of REVERB's extractions are at precision 0.8 or higher—compared to virtually none for earlier systems. The paper concludes with a detailed analysis of REVERB's errors, suggesting directions for future work.[1]

1 Introduction and Motivation

Typically, Information Extraction (IE) systems learn an extractor for each target relation from labeled training examples (Kim and Moldovan, 1993; Riloff, 1996; Soderland, 1999). This approach to IE does not scale to corpora where the number of target relations is very large, or where the target relations cannot be specified in advance. Open IE solves this problem by identifying *relation phrases*—phrases that denote relations in English sentences (Banko et al., 2007). The automatic identification of relation phrases enables the extraction of *arbitrary* relations from sentences, obviating the restriction to a pre-specified vocabulary.

Open IE systems have achieved a notable measure of success on massive, open-domain corpora drawn from the Web, Wikipedia, and elsewhere. (Banko et al., 2007; Wu and Weld, 2010; Zhu et al., 2009). The output of Open IE systems has been used to support tasks like learning selectional preferences (Ritter et al., 2010), acquiring common sense knowledge (Lin et al., 2010), and recognizing entailment (Schoenmackers et al., 2010; Berant et al., 2011). In addition, Open IE extractions have been mapped onto existing ontologies (Soderland et al., 2010).

We have observed that two types of errors are frequent in the output of Open IE systems such as TEXTRUNNER and WOE: *incoherent extractions* and *uninformative extractions*.

Incoherent extractions are cases where the extracted relation phrase has no meaningful interpretation (see Table 1 for examples). Incoherent extractions arise because the learned extractor makes a sequence of decisions about whether to include each word in the relation phrase, often resulting in incomprehensible predictions. To solve this problem, we introduce a syntactic constraint: every multi-word relation phrase must begin with a verb, end with a preposition, and be a contiguous sequence of words in the sentence. Thus, the identification of a relation phrase is made in one fell swoop instead of on the basis of multiple, word-by-word decisions.

Uninformative extractions are extractions that omit critical information. For example, consider the sentence "Faust made a deal with the devil." Previ-

[1] The source code for REVERB is available at http://reverb.cs.washington.edu/

Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, pages 1535–1545,
Edinburgh, Scotland, UK, July 27–31, 2011.

ous Open IE systems return the uninformative

(Faust, made, a deal)

instead of

(Faust, made a deal with, the devil).

This type of error is caused by improper handling of relation phrases that are expressed by a combination of a verb with a noun, such as light verb constructions (LVCs). An LVC is a multi-word expression composed of a verb and a noun, with the noun carrying the semantic content of the predicate (Grefenstette and Teufel, 1995; Stevenson et al., 2004; Allerton, 2002). Table 2 illustrates the wide range of relations expressed this way, which are not captured by existing open extractors. Our syntactic constraint leads the extractor to include nouns in the relation phrase, solving this problem.

Although the syntactic constraint significantly reduces incoherent and uninformative extractions, it allows overly-specific relation phrases such as *is offering only modest greenhouse gas reduction targets at*. To avoid overly-specific relation phrases, we introduce an intuitive lexical constraint: a binary relation phrase ought to appear with at least a minimal number of distinct argument pairs in a large corpus.

In summary, this paper articulates two simple but surprisingly powerful constraints on how binary relationships are expressed via verbs in English sentences, and implements them in the REVERB Open IE system. We release REVERB and the data used in our experiments to the research community.

The rest of the paper is organized as follows. Section 2 analyzes previous work. Section 3 defines our constraints precisely. Section 4 describes REVERB, our implementation of the constraints. Section 5 reports on our experimental results. Section 6 concludes with a summary and discussion of future work.

2 Previous Work

Open IE systems like TEXTRUNNER (Banko et al., 2007), WOE^{pos}, and WOE^{parse} (Wu and Weld, 2010) focus on extracting binary relations of the form (arg1, relation phrase, arg2) from text. These systems all use the following three-step method:

1. **Label:** Sentences are automatically labeled with extractions using heuristics or distant supervision.

Sentence	Incoherent Relation
The guide *contains* dead links and *omits* sites.	contains omits
The Mark 14 *was central* to the *torpedo* scandal of the fleet.	was central torpedo
They *recalled* that Nungesser *began* his career as a precinct leader.	recalled began

Table 1: Examples of incoherent extractions. Incoherent extractions make up approximately 13% of TEXTRUNNER's output, 15% of WOE^{pos}'s output, and 30% of WOE^{parse}'s output.

is	is an album by, is the author of, is a city in
has	has a population of, has a Ph.D. in, has a cameo in
made	made a deal with, made a promise to
took	took place in, took control over, took advantage of
gave	gave birth to, gave a talk at, gave new meaning to
got	got tickets to, got a deal on, got funding from

Table 2: Examples of uninformative relations (left) and their completions (right). Uninformative relations occur in approximately 4% of WOE^{parse}'s output, 6% of WOE^{pos}'s output, and 7% of TEXTRUNNER's output.

2. **Learn:** A relation phrase extractor is learned using a sequence-labeling graphical model (*e.g.*, CRF).

3. **Extract:** the system takes a sentence as input, identifies a candidate pair of NP arguments (arg1, arg2) from the sentence, and then uses the learned extractor to label each word between the two arguments as part of the relation phrase or not.

The extractor is applied to the successive sentences in the corpus, and the resulting extractions are collected.

This method faces several challenges. First, the training phase requires a large number of labeled training examples (*e.g.*, 200,000 heuristically-labeled sentences for TEXTRUNNER and 300,000 for WOE). Heuristic labeling of examples obviates hand labeling but results in noisy labels and distorts the distribution of examples. Second, the extraction step is posed as a sequence-labeling problem, where each word is assigned its own label. Because each assignment is uncertain, the likelihood that the extracted relation phrase is flawed increases with the length of the sequence. Finally, the extractor

chooses an extraction's arguments heuristically, and cannot backtrack over this choice. This is problematic when a word that belongs in the relation phrase is chosen as an argument (for example, *deal* from the "made a deal with" sentence).

Because of the feature sets utilized in previous work, the learned extractors ignore both "holistic" aspects of the relation phrase (*e.g.*, is it contiguous?) as well as lexical aspects (*e.g.*, how many instances of this relation are there?). Thus, as we show in Section 5, systems such as TEXTRUNNER are unable to learn the constraints embedded in REVERB. Of course, a learning system, utilizing a different hypothesis space, and an appropriate set of training examples, *could* potentially learn and refine the constraints in REVERB. This is a topic for future work, which we consider in Section 6.

The first Open IE system was TEXTRUNNER (Banko et al., 2007), which used a Naive Bayes model with unlexicalized POS and NP-chunk features, trained using examples heuristically generated from the Penn Treebank. Subsequent work showed that utilizing a linear-chain CRF (Banko and Etzioni, 2008) or Markov Logic Network (Zhu et al., 2009) can lead to improved extraction. The WOE systems introduced by Wu and Weld make use of Wikipedia as a source of training data for their extractors, which leads to further improvements over TEXTRUNNER (Wu and Weld, 2010). Wu and Weld also show that dependency parse features result in a dramatic increase in precision and recall over shallow linguistic features, but at the cost of extraction speed.

Other approaches to large-scale IE have included Preemptive IE (Shinyama and Sekine, 2006), On-Demand IE (Sekine, 2006), and weak supervision for IE (Mintz et al., 2009; Hoffmann et al., 2010). Preemptive IE and On-Demand IE avoid relation-specific extractors, but rely on document and entity clustering, which is too costly for Web-scale IE. Weakly supervised methods use an existing ontology to generate training data for learning relation-specific extractors. While this allows for learning relation-specific extractors at a larger scale than what was previously possible, the extractions are still restricted to a specific ontology.

Many systems have used syntactic patterns based on verbs to extract relation phrases, usually relying on a full dependency parse of the input sentence (Lin and Pantel, 2001; Stevenson, 2004; Specia and Motta, 2006; Kathrin Eichler and Neumann, 2008). Our work differs from these approaches by focusing on relation phrase patterns expressed in terms of POS tags and NP chunks, instead of full parse trees. Banko and Etzioni (Banko and Etzioni, 2008) showed that a small set of POS-tag patterns cover a large fraction of relationships in English, but never incorporated the patterns into an extractor. This paper reports on a substantially improved model of binary relation phrases, which increases the recall of the Banko-Etzioni model (see Section 3.3). Further, while previous work in Open IE has mainly focused on syntactic patterns for relation extraction, we introduce a lexical constraint that boosts precision and recall.

Finally, Open IE is closely related to semantic role labeling (SRL) (Punyakanok et al., 2008; Toutanova et al., 2008) in that both tasks extract relations and arguments from sentences. However, SRL systems traditionally rely on syntactic parsers, which makes them susceptible to parser errors and substantially slower than Open IE systems such as REVERB. This difference is particularly important when operating on the Web corpus due to its size and heterogeneity. Finally, SRL requires hand-constructed semantic resources like Propbank and Framenet (Martha and Palmer, 2002; Baker et al., 1998) as input. In contrast, Open IE systems require no relation-specific training data. ReVerb, in particular, relies on its explicit lexical and syntactic constraints, which have no correlate in SRL systems. For a more detailed comparison of SRL and Open IE, see (Christensen et al., 2010).

3 Constraints on Relation Phrases

In this section we introduce two constraints on relation phrases: a syntactic constraint and a lexical constraint.

3.1 Syntactic Constraint

The syntactic constraint serves two purposes. First, it eliminates incoherent extractions, and second, it reduces uninformative extractions by capturing relation phrases expressed by a verb-noun combination, including light verb constructions.

$$V \mid VP \mid VW^*P$$
V = verb particle? adv?
W = (noun | adj | adv | pron | det)
P = (prep | particle | inf. marker)

Figure 1: A simple part-of-speech-based regular expression reduces the number of incoherent extractions like *was central torpedo* and covers relations expressed via light verb constructions like *gave a talk at*.

The syntactic constraint requires the relation phrase to match the POS tag pattern shown in Figure 1. The pattern limits relation phrases to be either a verb (*e.g.*, *invented*), a verb followed immediately by a preposition (*e.g.*, *located in*), or a verb followed by nouns, adjectives, or adverbs ending in a preposition (*e.g.*, *has atomic weight of*). If there are multiple possible matches in a sentence for a single verb, the longest possible match is chosen. Finally, if the pattern matches multiple adjacent sequences, we merge them into a single relation phrase (*e.g.*, *wants to extend*). This refinement enables the model to readily handle relation phrases containing multiple verbs. A consequence of this pattern is that the relation phrase must be a contiguous span of words in the sentence.

The syntactic constraint eliminates the incoherent relation phrases returned by existing systems. For example, given the sentence

> Extendicare agreed to buy Arbor Health Care for about US $432 million in cash and assumed debt.

TEXTRUNNER returns the extraction

(Arbor Health Care, for assumed, debt).

The phrase *for assumed* is clearly not a valid relation phrase: it begins with a preposition and splices together two distant words in the sentence. The syntactic constraint prevents this type of error by simply restricting relation phrases to match the pattern in Figure 1.

The syntactic constraint reduces uninformative extractions by capturing relation phrases expressed via LVCs. For example, the POS pattern matched against the sentence "Faust made a deal with the Devil," would result in the relation phrase *made a deal with*, instead of the uninformative *made*.

Finally, we require the relation phrase to appear between its two arguments in the sentence. This is a common constraint that has been implicitly enforced in other open extractors.

3.2 Lexical Constraint

While the syntactic constraint greatly reduces uninformative extractions, it can sometimes match relation phrases that are so specific that they have only a few possible instances, even in a Web-scale corpus. Consider the sentence:

> The Obama administration is offering only modest greenhouse gas reduction targets at the conference.

The POS pattern will match the phrase:

is offering only modest greenhouse gas reduction targets at (1)

Thus, there are phrases that satisfy the syntactic constraint, but are not relational.

To overcome this limitation, we introduce a lexical constraint that is used to separate valid relation phrases from overspecified relation phrases, like the example in (1). The constraint is based on the intuition that a valid relation phrase should take many distinct arguments in a large corpus. The phrase in (1) is specific to the argument pair *(Obama administration, conference)*, so it is unlikely to represent a *bona fide* relation. We describe the implementation details of the lexical constraint in Section 4.

3.3 Limitations

Our constraints represent an idealized model of relation phrases in English. This raises the question: How much recall is lost due to the constraints?

To address this question, we analyzed Wu and Weld's set of 300 sentences from a set of random Web pages, manually identifying all verb-based relationships between noun phrase pairs. This resulted in a set of 327 relation phrases. For each relation phrase, we checked whether it satisfies our constraints. We found that 85% of the relation phrases do satisfy the constraints. Of the remaining 15%, we identified some of the common cases where the constraints were violated, summarized in Table 3.

Many of the example relation phrases shown in Table 3 involve long-range dependencies between words in the sentence. These types of dependencies are not easily representable using a pattern over POS tags. A deeper syntactic analysis of the input sentence would provide a much more general language for modeling relation phrases. For example, one could create a model of relations expressed in

	Binary Verbal Relation Phrases
85%	Satisfy Constraints
8%	Non-Contiguous Phrase Structure Coordination: X is produced and maintained by Y Multiple Args: X was founded in 1995 by Y Phrasal Verbs: X turned Y off
4%	Relation Phrase Not Between Arguments Intro. Phrases: Discovered by Y, X ... Relative Clauses: ... the Y that X discovered
3%	Do Not Match POS Pattern Interrupting Modifiers: X has a lot of faith in Y Infinitives: X to attack Y

Table 3: Approximately 85% of the binary verbal relation phrases in a sample of Web sentences satisfy our constraints.

terms of dependency parse features that would capture the non-contiguous relation phrases in Table 3. Previous work has shown that dependency paths do indeed boost the recall of relation extraction systems (Wu and Weld, 2010; Mintz et al., 2009). While using dependency path features allows for a more flexible model of relations, it significantly increases processing time, which is problematic for Web-scale extraction. Further, we have found that this increased recall comes at the cost of lower precision on Web text (see Section 5).

The results in Table 3 are similar to Banko and Etzioni's findings that a set of eight POS patterns cover a large fraction of binary verbal relation phrases. However, their analysis was based on a set of sentences known to contain either a company acquisition or birthplace relationship, while our results are on a random sample of Web sentences. We applied Banko and Etzioni's verbal patterns to our random sample of 300 Web sentences, and found that they cover approximately 69% of the relation phrases in the corpus. The gap in recall between this and the 85% shown in Table 3 is largely due to LVC relation phrases (*made a deal with*) and phrases containing multiple verbs (*refuses to return to*), which their patterns do not cover.

In sum, our model is by no means complete. However, we have empirically shown that the majority of binary verbal relation phrases in a sample of Web sentences are captured by our model. By focusing on this subset of language, our model can be used to perform Open IE at significantly higher precision than before.

4 REVERB

This section introduces REVERB, a novel open extractor based on the constraints defined in the previous section. REVERB first identifies relation phrases that satisfy the syntactic and lexical constraints, and then finds a pair of NP arguments for each identified relation phrase. The resulting extractions are then assigned a confidence score using a logistic regression classifier.

This algorithm differs in three important ways from previous methods (Section 2). First, the relation phrase is identified "holistically" rather than word-by-word. Second, potential phrases are filtered based on statistics over a large corpus (the implementation of our lexical constraint). Finally, REVERB is "relation first" rather than "arguments first", which enables it to avoid a common error made by previous methods—confusing a noun in the relation phrase for an argument, *e.g.* the noun *deal* in *made a deal with*.

4.1 Extraction Algorithm

REVERB takes as input a POS-tagged and NP-chunked sentence and returns a set of (x, r, y) extraction triples.[2] Given an input sentence s, REVERB uses the following extraction algorithm:

1. **Relation Extraction:** For each verb v in s, find the longest sequence of words r_v such that (1) r_v starts at v, (2) r_v satisfies the syntactic constraint, and (3) r_v satisfies the lexical constraint. If any pair of matches are adjacent or overlap in s, merge them into a single match.
2. **Argument Extraction:** For each relation phrase r identified in Step 1, find the nearest noun phrase x to the left of r in s such that x is not a relative pronoun, WHO-adverb, or existential "there". Find the nearest noun phrase y to the right of r in s. If such an (x, y) pair could be found, return (x, r, y) as an extraction.

We check whether a candidate relation phrase r_v satisfies the syntactic constraint by matching it against the regular expression in Figure 1.

[2] REVERB uses OpenNLP for POS tagging and NP chunking: `http://opennlp.sourceforge.net/`

To determine whether r_v satisfies the lexical constraint, we use a large dictionary D of relation phrases that are known to take many distinct arguments. In an offline step, we construct D by finding all matches of the POS pattern in a corpus of 500 million Web sentences. For each matching relation phrase, we heuristically identify its arguments (as in Step 2 above). We set D to be the set of all relation phrases that take at least k distinct argument pairs in the set of extractions. In order to allow for minor variations in relation phrases, we normalize each relation phrase by removing inflection, auxiliary verbs, adjectives, and adverbs. Based on experiments on a held-out set of sentences, we found that a value of $k = 20$ works well for filtering out overspecified relations. This results in a set of approximately 1.7 million distinct normalized relation phrases, which are stored in memory at extraction time.

As an example of the extraction algorithm in action, consider the following input sentence:

> Hudson was born in Hampstead, which is a suburb of London.

Step 1 of the algorithm identifies three relation phrases that satisfy the syntactic and lexical constraints: *was*, *born in*, and *is a suburb of*. The first two phrases are adjacent in the sentence, so they are merged into the single relation phrase *was born in*. Step 2 then finds an argument pair for each relation phrase. For *was born in*, the nearest NPs are *(Hudson, Hampstead)*. For *is a suburb of*, the extractor skips over the NP *which* and chooses the argument pair *(Hampstead, London)*. The final output is

e_1: *(Hudson, was born in, Hampstead)*
e_2: *(Hampstead, is a suburb of, London)*.

4.2 Confidence Function

The extraction algorithm in the previous section has high recall, but low precision. Like with previous open extractors, we want way to trade recall for precision by tuning a confidence threshold. We use a logistic regression classifier to assign a confidence score to each extraction, which uses the features shown in Table 4. All of these features are efficiently computable and relation independent. We trained the confidence function by manually labeling the extractions from a set of $1,000$ sentences from the Web and Wikipedia as correct or incorrect.

Weight	Feature
1.16	(x, r, y) covers all words in s
0.50	The last preposition in r is *for*
0.49	The last preposition in r is *on*
0.46	The last preposition in r is *of*
0.43	$len(s) \leq 10$ words
0.43	There is a WH-word to the left of r
0.42	r matches VW*P from Figure 1
0.39	The last preposition in r is *to*
0.25	The last preposition in r is *in*
0.23	10 words $< len(s) \leq 20$ words
0.21	s begins with x
0.16	y is a proper noun
0.01	x is a proper noun
-0.30	There is an NP to the left of x in s
-0.43	20 words $< len(s)$
-0.61	r matches V from Figure 1
-0.65	There is a preposition to the left of x in s
-0.81	There is an NP to the right of y in s
-0.93	Coord. conjunction to the left of r in s

Table 4: REVERB uses these features to assign a confidence score to an extraction (x, r, y) from a sentence s using a logistic regression classifier.

Previous open extractors require labeled training data to learn a model of relations, which is then used to extract relation phrases from text. In contrast, REVERB uses a specified model of relations for extraction, and requires labeled data only for assigning confidence scores to its extractions. Learning a confidence function is a much simpler task than learning a full model of relations, using two orders of magnitude fewer training examples than TEXTRUNNER or WOE.

4.3 TEXTRUNNER-R

The model of relation phrases used by REVERB is specified, but could a TEXTRUNNER-like system learn this model from training data? While it is difficult to answer such a question for all possible permutations of features sets, training examples, and learning biases, we demonstrate that TEXTRUNNER itself cannot learn REVERB's model even when re-trained using the output of REVERB as labeled training data. The resulting system, TEXTRUNNER-R, uses the same feature representation as TEXTRUNNER, but different parameters, and a different set of training examples.

To generate positive instances, we ran REVERB

on the Penn Treebank, which is the same dataset that TEXTRUNNER is trained on. To generate negative instances from a sentence, we took each noun phrase pair in the sentence that does not appear as arguments in a REVERB extraction. This process resulted in a set of $67,562$ positive instances, and $356,834$ negative instances. We then passed these labeled examples to TEXTRUNNER's training procedure, which learns a linear-chain CRF using closed-class features like POS tags, capitalization, punctuation, *etc.* TEXTRUNNER-R uses the argument-first extraction algorithm described in Section 2.

5 Experiments

We compare REVERB to the following systems:

- REVERB$^{\neg lex}$ - The REVERB system described in the previous section, but without the lexical constraint. REVERB$^{\neg lex}$ uses the same confidence function as REVERB.
- TEXTRUNNER - Banko and Etzioni's 2008 extractor, which uses a second order linear-chain CRF trained on extractions heuristically generated from the Penn Treebank. TEXTRUNNER uses shallow linguistic features in its CRF, which come from the same POS tagger and NP-chunker that REVERB uses.
- TEXTRUNNER-R - Our modification to TEXTRUNNER, which uses the same extraction code, but with a model of relations trained on REVERB extractions.
- WOE^{pos} - Wu and Weld's modification to TEXTRUNNER, which uses a model of relations learned from extractions heuristically generated from Wikipedia.
- WOE^{parse} - Wu and Weld's parser-based extractor, which uses a large dictionary of dependency path patterns learned from heuristic extractions generated from Wikipedia.

Each system is given a set of sentences as input, and returns a set of binary extractions as output. We created a test set of 500 sentences sampled from the Web, using Yahoo's random link service.[3] After running each extractor over the input sentences, two human judges independently evaluated each extraction as correct or incorrect. The judges reached agreement on 86% of the extractions, with an agreement score of $\kappa = 0.68$. We report results on the subset of the data where the two judges concur.

[3] http://random.yahoo.com/bin/ryl

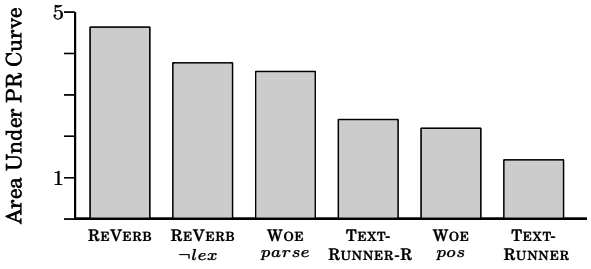

Figure 2: REVERB outperforms state-of-the-art open extractors, with an AUC more than twice that of TEXTRUNNER or WOE^{pos}, and 38% higher than WOE^{parse}.

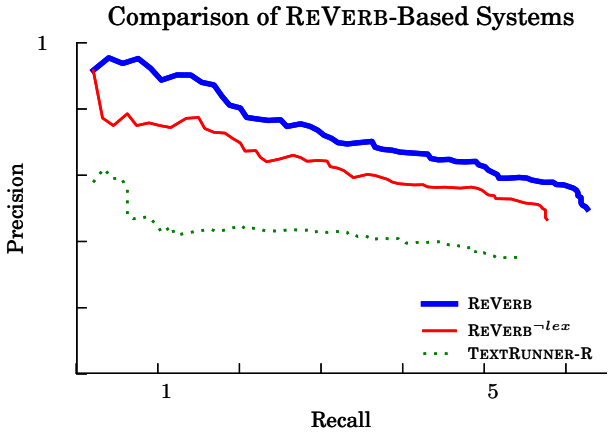

Figure 3: The lexical constraint gives REVERB a boost in precision and recall over REVERB$^{\neg lex}$. TEXTRUNNER-R is unable to learn the model used by REVERB, which results in lower precision and recall.

The judges labeled uninformative extractions conservatively. That is, if critical information was dropped from the relation phrase but included in the second argument, it is labeled correct. For example, both the extractions *(Ackerman, is a professor of, biology)* and *(Ackerman, is, a professor of biology)* are considered correct.

Each system returns confidence scores for its extractions. For a given threshold, we can measure the precision and recall of the output. Precision is the fraction of returned extractions that are correct. Recall is the fraction of correct extractions in

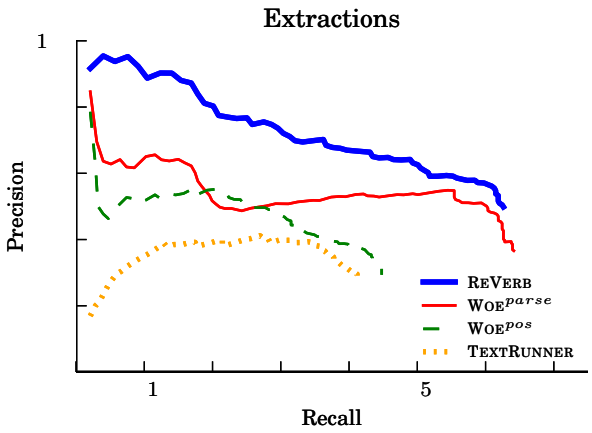

Figure 4: REVERB achieves significantly higher precision than state-of-the-art Open IE systems, and comparable recall to WOE^{parse}.

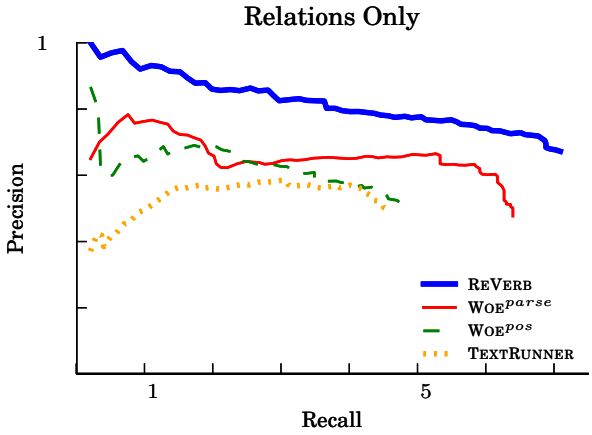

Figure 5: On the subtask of identifying relations phrases, REVERB is able to achieve even higher precision and recall than other systems.

the corpus that are returned. We use the total number of extractions labeled as correct by the judges as our measure of recall for the corpus. In order to avoid double-counting, we treat extractions that differ superficially (*e.g.*, different punctuation or dropping inessential modifiers) as a single extraction. We compute a precision-recall curve by varying the confidence threshold, and then compute the area under the curve (AUC).

5.1 Results

Figure 2 shows the AUC of each system. REVERB achieves an AUC that is 30% higher than WOE^{parse} and is more than double the AUC of WOE^{pos} or TEXTRUNNER. The lexical constraint provides a significant boost in performance, with REVERB achieving an AUC 23% higher than REVERB$^{\neg lex}$. REVERB proves to be a useful source of training data, with TEXTRUNNER-R having an AUC 71% higher than TEXTRUNNER and performing on par with WOE^{pos}. From the training data, TEXTRUNNER-R was able to learn a model that predicts contiguous relation phrases, but still returned incoherent relation phrases (*e.g.*, starting with a preposition) and overspecified relation phrases. These errors are due to TEXTRUNNER-R overfitting the training data and not having access to the lexical constraint.

Figure 3 shows the precision-recall curves of the systems introduced in this paper. TEXTRUNNER-R has much lower precision than REVERB and REVERB$^{\neg lex}$ at all levels of recall. The lexical constraint gives REVERB a boost in precision over REVERB$^{\neg lex}$, reducing overspecified extractions from 20% of REVERB$^{\neg lex}$'s output to 1% of REVERB's. The lexical constraint also boosts recall over REVERB$^{\neg lex}$, since REVERB is able to find a correct relation phrase where REVERB$^{\neg lex}$ finds an overspecified one.

Figure 4 shows the precision-recall curves of REVERB and the external systems. REVERB has much higher precision than the other systems at nearly all levels of recall. In particular, more than 30% of REVERB's extractions are at precision 0.8 or higher, compared to virtually none for the other systems. WOE^{parse} achieves a slightly higher recall than REVERB (0.62 versus 0.64), but at the cost of lower precision.

In order to highlight the role of the relational model of each system, we also evaluate their performance on the subtask of extracting just the relation phrases from the input text. Figure 5 shows the precision-recall curves for each system on the relation phrase-only evaluation. In this case, REVERB has both higher precision and recall than the other systems.

REVERB's biggest improvement came from the elimination of incoherent extractions. Incoherent extractions were a large fraction of the errors made by previous systems, accounting for approximately 13% of TEXTRUNNER's extractions, 15% of WOE^{pos}'s, and 30% of WOE^{parse}'s. Uninformative

REVERB - Incorrect Extractions	
65%	Correct relation phrase, incorrect arguments
16%	N-ary relation
8%	Non-contiguous relation phrase
2%	Imperative verb
2%	Overspecified relation phrase
7%	Other, including POS/chunking errors

Table 5: The majority of the incorrect extractions returned by REVERB are due to errors in argument extraction.

REVERB - Missed Extractions	
52%	Could not identify correct arguments
23%	Relation filtered out by lexical constraint
17%	Identified a more specific relation
8%	POS/chunking error

Table 6: The majority of extractions that were missed by REVERB were cases where the correct relation phrase was found, but the arguments were not correctly identified.

extractions had a smaller effect on other systems' precision, accounting for 4% of WOE^{parse}'s extractions, 5% of WOE^{pos}'s, and 7% of TEXTRUNNER's, while only appearing in 1% of REVERB's extractions. REVERB's reduction in uninformative extractions resulted in a boost in recall, capturing many LVC relation phrases missed by other systems (like those shown in Table 2).

To test the systems' speed, we ran each extractor on a set of 100,000 sentences using a Pentium 4 machine with 4GB of RAM. The processing times were 16 minutes for REVERB, 21 minutes for TEXTRUNNER, 21 minutes for WOE^{pos}, and 11 hours for WOE^{parse}. The times for REVERB, TEXTRUNNER, and WOE^{pos} are all approximately the same, since they all use the same POS-tagging and NP-chunking software. WOE^{parse} processes each sentence with a dependency parser, resulting in much longer processing time.

5.2 REVERB Error Analysis

To better understand the limitations of REVERB, we performed a detailed analysis of its errors in precision (incorrect extractions returned by REVERB) and its errors in recall (correct extractions that REVERB missed).

Table 5 summarizes the types of incorrect extractions that REVERB returns. We found that 65% of the incorrect extractions returned by REVERB were cases where a relation phrase was correctly identified, but the argument-finding heuristics failed. The remaining errors were cases where REVERB extracted an incorrect relation phrase. One common mistake that REVERB made was extracting a relation phrase that expresses an n-ary relationship via a ditransitive verb. For example, given the sentence "I gave him 15 photographs," REVERB extracts *(I, gave, him)*. These errors are due to the fact that REVERB only models binary relations.

Table 6 summarizes the correct extractions that were extracted by other systems and were not extracted by REVERB. As with the false positive extractions, the majority of false negatives (52%) were due to the argument-finding heuristics choosing the wrong arguments, or failing to extract all possible arguments (in the case of coordinating conjunctions). Other sources of failure were due to the lexical constraint either failing to filter out an overspecified relation phrase or filtering out a valid relation phrase. These errors hurt both precision and recall, since each case results in the extractor overlooking a correct relation phrase and choosing another.

5.3 Evaluation At Scale

Section 5.1 shows that REVERB outperforms existing Open IE systems when evaluated on a sample of sentences. Previous work has shown that the frequency of an extraction in a large corpus is useful for assessing the correctness of extractions (Downey et al., 2005). Thus, it is possible *a priori* that REVERB's gains over previous systems will diminish when extraction frequency is taken into account.

In fact, we found that REVERB's advantage over TEXTRUNNER when run at scale is qualitatively similar to its advantage on single sentences. We ran both REVERB and TEXTRUNNER on Banko and Etzioni's corpus of 500 million Web sentences and examined the effect of redundancy on precision.

As Downey's work predicts, precision increased in both systems for extractions found multiple times, compared with extractions found only once. However, REVERB had higher precision than

TEXTRUNNER at all frequency thresholds. In fact, REVERB's frequency 1 extractions had a precision of 0.75, which TEXTRUNNER could not approach even with frequency 10 extractions, which had a precision of 0.34. Thus, REVERB is able to return more correct extractions at a higher precision than TEXTRUNNER, even when redundancy is taken into account.

6 Conclusions and Future Work

The paper's contributions are as follows:

- We have identified and analyzed the problems of incoherent and uninformative extractions for Open IE systems, and shown their prevalence for systems such as TEXTRUNNER and WOE.
- We articulated general, easy-to-enforce constraints on binary, verb-based relation phrases in English that ameliorate these problems and yield richer and more informative relations (see, for example, Table 2).
- Based on these constraints, we designed, implemented, and evaluated the REVERB extractor, which substantially outperforms previous Open IE systems in both recall and precision.
- We make REVERB and the data used in our experiments available to the research community.[4]

In future work, we plan to explore utilizing our constraints to improve the performance of learned CRF models. Roth *et al.* have shown how to incorporate constraints into CRF learners (Roth and Yih, 2005). It is natural, then, to consider whether the combination of heuristically labeled training examples, CRF learning, and our constraints will result in superior performance. The error analysis in Section 5.2 also suggests natural directions for future work. For instance, since many of REVERB's errors are due to incorrect arguments, improved methods for argument extraction are in order.

Acknowledgments

We would like to thank Mausam, Dan Weld, Yoav Artzi, Luke Zettlemoyer, members of the KnowItAll group, and the anonymous reviewers for their helpful comments. This research was supported in part by NSF grant IIS-0803481, ONR grant N00014-08-1-0431, and DARPA contract FA8750-09-C-0179, and carried out at the University of Washington's Turing Center.

[4] http://reverb.cs.washington.edu

References

David J. Allerton. 2002. *Stretched Verb Constructions in English*. Routledge Studies in Germanic Linguistics. Routledge (Taylor and Francis), New York.

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The berkeley framenet project. In *Proceedings of the 17th international conference on Computational linguistics*, pages 86–90.

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

Michele Banko, Michael J. Cafarella, Stephen Soderland, Matt Broadhead, and Oren Etzioni. 2007. Open information extraction from the web. In *In the Proceedings of the 20th International Joint Conference on Artificial Intelligence*, pages 2670–2676, January.

Jonathan Berant, Ido Dagan, and Jacob Goldberger. 2011. Global learning of typed entailment rules. In *Proceedings of ACL*, Portland, OR.

Janara Christensen, Mausam, Stephen Soderland, and Oren Etzioni. 2010. Semantic role labeling for open information extraction. In *Proceedings of the NAACL HLT 2010 First International Workshop on Formalisms and Methodology for Learning by Reading*, FAM-LbR '10, pages 52–60, Stroudsburg, PA, USA. Association for Computational Linguistics.

Doug Downey, Oren Etzioni, and Stephen Soderland. 2005. A probabilistic model of redundancy in information extraction. In *IJCAI*, pages 1034–1041.

Gregory Grefenstette and Simone Teufel. 1995. Corpus-based method for automatic identification of support verbs for nominalizations. In *Proceedings of the seventh conference on European chapter of the Association for Computational Linguistics*, pages 98–103, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Raphael Hoffmann, Congle Zhang, and Daniel S. Weld. 2010. Learning 5000 relational extractors. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 286–295, Stroudsburg, PA, USA. Association for Computational Linguistics.

Holmer Hemsen Kathrin Eichler and Gnter Neumann. 2008. Unsupervised relation extraction from web documents. In *LREC*. http://www.lrec-conf.org/proceedings/lrec2008/.

J. Kim and D. Moldovan. 1993. Acquisition of semantic patterns for information extraction from corpora. In *Procs. of Ninth IEEE Conference on Artificial Intelligence for Applications*, pages 171–176.

Dekang Lin and Patrick Pantel. 2001. DIRT-Discovery of Inference Rules from Text. In *Proceedings of ACM Conference on Knowledge Discovery and Data Mining(KDD-01)*, pages pp. 323–328.

Thomas Lin, Mausam, and Oren Etzioni. 2010. Identifying Functional Relations in Web Text. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1266–1276, Cambridge, MA, October. Association for Computational Linguistics.

Paul Kingsbury Martha and Martha Palmer. 2002. From treebank to propbank. In *In Proceedings of LREC-2002*.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *ACL-IJCNLP '09: Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2*, pages 1003–1011, Morristown, NJ, USA. Association for Computational Linguistics.

V. Punyakanok, D. Roth, and W. Yih. 2008. The importance of syntactic parsing and inference in semantic role labeling. *Computational Linguistics*, 34(2).

E. Riloff. 1996. Automatically constructing extraction patterns from untagged text. In *Procs. of the Thirteenth National Conference on Artificial Intelligence (AAAI-96)*, pages 1044–1049.

Alan Ritter, Mausam, and Oren Etzioni. 2010. A Latent Dirichlet Allocation Method for Selectional Preferences. In *ACL*.

Dan Roth and Wen-tau Yih. 2005. Integer linear programming inference for conditional random fields. In *Proceedings of the 22nd international conference on Machine learning*, ICML '05, pages 736–743, New York, NY, USA. ACM.

Stefan Schoenmackers, Oren Etzioni, Daniel S. Weld, and Jesse Davis. 2010. Learning first-order horn clauses from web text. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 1088–1098, Stroudsburg, PA, USA. Association for Computational Linguistics.

Satoshi Sekine. 2006. On-demand information extraction. In *Proceedings of the COLING/ACL on Main conference poster sessions*, pages 731–738, Morristown, NJ, USA. Association for Computational Linguistics.

Yusuke Shinyama and Satoshi Sekine. 2006. Preemptive Information Extraction using Unrestricted Relation Discovery. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 304–311, New York City, USA, June. Association for Computational Linguistics.

Stephen Soderland, Brendan Roof, Bo Qin, Shi Xu, Mausam, and Oren Etzioni. 2010. Adapting open information extraction to domain-specific relations. *AI Magazine*, 31(3):93–102.

S. Soderland. 1999. Learning Information Extraction Rules for Semi-Structured and Free Text. *Machine Learning*, 34(1-3):233–272.

Lucia Specia and Enrico Motta. 2006. M.: A hybrid approach for extracting semantic relations from texts. In *In. Proceedings of the 2 nd Workshop on Ontology Learning and Population*, pages 57–64.

Suzanne Stevenson, Afsaneh Fazly, and Ryan North. 2004. Statistical measures of the semi-productivity of light verb constructions. In *2nd ACL Workshop on Multiword Expressions*, pages 1–8.

M. Stevenson. 2004. An unsupervised WordNet-based algorithm for relation extraction. In *Proceedings of the "Beyond Named Entity" workshop at the Fourth International Conference on Language Resources and Evalutaion (LREC'04)*.

Kristina Toutanova, Aria Haghighi, and Christopher D. Manning. 2008. A global joint model for semantic role labeling. *Computational Linguistics*, 34(2):161–191.

Fei Wu and Daniel S. Weld. 2010. Open information extraction using Wikipedia. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 118–127, Morristown, NJ, USA. Association for Computational Linguistics.

Jun Zhu, Zaiqing Nie, Xiaojiang Liu, Bo Zhang, and Ji-Rong Wen. 2009. StatSnowball: a statistical approach to extracting entity relationships. In *WWW '09: Proceedings of the 18th international conference on World wide web*, pages 101–110, New York, NY, USA. ACM.