Unsupervised Detection of Downward-Entailing Operators By Maximizing Classification Certainty

Jackie CK Cheung and Gerald Penn
Department of Computer Science
University of Toronto
Toronto, ON, M5S 3G4, Canada
{jcheung,gpenn}@cs.toronto.edu

Abstract

We propose an unsupervised, iterative method for detecting downward-entailing operators (DEOs), which are important for deducing entailment relations between sentences. Like the distillation algorithm of Danescu-Niculescu-Mizil et al. (2009), the initialization of our method depends on the correlation between DEOs and negative polarity items (NPIs). However, our method trusts the initialization more and aggressively separates likely DEOs from spurious distractors and other words, unlike distillation, which we show to be equivalent to one iteration of EM prior re-estimation. Our method is also amenable to a bootstrapping method that co-learns DEOs and NPIs, and achieves the best results in identifying DEOs in two corpora.

1 Introduction

Reasoning about text has been a long-standing challenge in NLP, and there has been considerable debate both on what constitutes inference and what techniques should be used to support inference. One task involving inference that has recently received much attention is that of recognizing textual entailment (RTE), in which the goal is to determine whether a hypothesis sentence can be entailed from a piece of source text (Bentivogli et al., 2010, for example).

An important consideration in RTE is whether a sentence or context produces an entailment relation for events that are a superset or subset of the original sentence (MacCartney and Manning, 2008). By default, contexts are upward-entailing, allowing reasoning from a set of events to a superset of events as seen in (1). In the scope of a *downward-entailing operator* (DEO), however, this entailment relation is reversed, such as in the scope of the classical DEO *not* (2). There are also operators which are neither upward- nor downward entailing, such as the expression *exactly three* (3).

(1) *She sang in French.* $\Rightarrow$ *She sang.* (upward-entailing)

(2) *She did not sing in French.* $\Leftarrow$ *She did not sing.* (downward-entailing)

(3) *Exactly three students sang.* $\not\Leftrightarrow$ *Exactly three students sang in French.* (neither upward- nor downward-entailing)

Danescu-Niculescu-Mizil et al. (2009) (henceforth DLD09) proposed the first computational methods for detecting DEOs from a corpus. They proposed two unsupervised algorithms which rely on the correlation between DEOs and *negative polarity items* (NPIs), which by the definition of Ladusaw (1980) must appear in the context of DEOs. An example of an NPI is *yet*, as in the sentence *This project is not complete yet*. The first baseline method proposed by DLD09 simply calculates a ratio of the relative frequencies of a word in NPI contexts versus in a general corpus, and the second is a *distillation* method which appears to refine the baseline ratios using a task-specific heuristic. Danescu-Niculescu-Mizil and Lee (2010) (henceforth DL10) extend this approach to Romanian, where a comprehensive list of NPIs is not available, by proposing a bootstrapping approach to co-learn DEOs and NPIs.

DLD09 are to be commended for having identified a crucial component of inference that nevertheless lends itself to a classification-based approach, as we will show. However, as noted by DL10, the performance of the distillation method is mixed across languages and in the semi-supervised bootstrapping setting, and there is no mathematical grounding of the heuristic to explain why it works and whether the approach can be refined or extended. This paper supplies the missing mathematical basis for distillation and shows that, while its intentions are fundamentally sound, the formulation of distillation neglects an important requirement that the method not be easily distracted by other word co-occurrences in NPI contexts. We call our alternative *certainty*, which uses an unusual posterior classification confidence score (based on the max function) to favour single, definite assignments of DEO-hood within every NPI context. DLD09 actually speculated on the use of max as an alternative, but within the context of an EM-like optimization procedure that throws away its initial parameter settings too willingly. Certainty iteratively and directly boosts the scores of the currently best-ranked DEO candidates relative to the alternatives in a Naïve Bayes model, which thus pays more respect to the initial weights, constructively building on top of what the model already knows. This method proves to perform better on two corpora than distillation, and is more amenable to the co-learning of NPIs and DEOs. In fact, the best results are obtained by co-learning the NPIs and DEOs in conjunction with our method.

2 Related work

There is a large body of literature in linguistic theory on downward entailment and polarity items[1], of which we will only mention the most relevant work here. The connection between downward-entailing contexts and negative polarity items was noticed by Ladusaw (1980), who stated the hypothesis that NPIs must be grammatically licensed by a DEO. However, DEOs are not the sole licensors of NPIs, as NPIs can also be found in the scope of questions, certain numeric expressions (i.e., non-monotone quantifiers), comparatives, and conditionals, among others. Giannakidou (2002) proposes that the property shared by these constructions and downward entailment is *non-veridicality*. If F is a propositional operator for proposition p, then an operator is non-veridical if $Fp \not\Rightarrow p$. Positive operators such as past tense adverbials are veridical (4), whereas questions, negation and other DEOs are non-veridical (5, 6).

(4) *She sang yesterday.* $\Rightarrow$ *She sang.*

(5) *She denied singing.* $\not\Rightarrow$ *She sang.*

(6) *Did she sing?* $\not\Rightarrow$ *She sang.*

While Ladusaw's hypothesis is thus accepted to be insufficient from a linguistic perspective, it is nevertheless a useful starting point for computational methods for detecting NPIs and DEOs, and has inspired successful techniques to detect DEOs, like the work by DLD09, DL10, and also this work. In addition to this hypothesis, we further assume that there should only be one plausible DEO candidate per NPI context. While there are counterexamples, this assumption is in practice very robust, and is a useful constraint for our learning algorithm. An analogy can be drawn to the one sense per discourse assumption in word sense disambiguation (Gale et al., 1992).

The related—and as we will argue, more difficult—problem of detecting NPIs has also been studied, and in fact predates the work on DEO detection. Hoeksema (1997) performed the first corpus-based study of NPIs, predominantly for Dutch, and there has also been work on detecting NPIs in German which assumes linguistic knowledge of licensing contexts for NPIs (Lichte and Soehn, 2007). Richter et al. (2010) make this assumption as well as use syntactic structure to extract NPIs that are multi-word expressions. Parse information is an especially important consideration in freer-word-order languages like German where a MWE may not appear as a contiguous string. In this paper, we explicitly do not assume detailed linguistic knowledge about licensing contexts for NPIs and do not assume that a parser is available, since neither of these are guaranteed when extending this technique to resource-poor languages.

3 Distillation as EM Prior Re-estimation

Let us first review the baseline and distillation methods proposed by DLD09, then show that distillation is equivalent to one iteration of EM prior

[1] See van der Wouden (1997) for a comprehensive reference.

re-estimation in a Naïve Bayes generative probabilistic model up to constant rescaling. The baseline method assigns a score to each word-type based on the ratio of its relative frequency within NPI contexts to its relative frequency within a general corpus. Suppose we are given a corpus $\mathcal{C}$ with extracted NPI contexts $\mathcal{N}$ and they contain $tokens(\mathcal{C})$ and $tokens(\mathcal{N})$ tokens respectively. Let y be a candidate DEO, $count^{\mathcal{C}}(y)$ be the unigram frequency of y in a corpus, and $count^{\mathcal{N}}(y)$ be the unigram frequency of y in $\mathcal{N}$. Then, we define $S(y)$ to be the ratio between the relative frequencies of y within NPI contexts and in the entire corpus[2]:

$$S(y) = \frac{count^{\mathcal{N}}(y)/tokens(\mathcal{N})}{count^{\mathcal{C}}(y)/tokens(\mathcal{C})}. \qquad (7)$$

The scores are then used as a ranking to determine word-types that are likely to be DEOs. This method approximately captures Ladusaw's hypothesis by highly ranking words that appear in NPI contexts more often than would be expected by chance. However, the problem with this approach is that DEOs are not the only words that co-occur with NPIs. In particular, there exist many *piggybackers*, which, as defined by DLD09, collocate with DEOs due to semantic relatedness or chance, and would thus incorrectly receive a high $S(y)$ score.

Examples of piggybackers found by DLD09 include the proper noun *Milken*, and the adverb *vigorously*, which collocate with DEOs like *deny* in the corpus they used. DLD09's solution to the piggybacker problem is a method that they term *distillation*. Let $\mathcal{N}_y$ be the NPI contexts that contain word y; i.e., $\mathcal{N}_y = \{c \in \mathcal{N} | c \ni y\}$. In distillation, each word-type is given a distilled score according to the following equation:

$$S_d(y) = \frac{1}{|\mathcal{N}_y|} \sum_{p \in \mathcal{N}_y} \frac{S(y)}{\sum_{y' \in p} S(y')}. \qquad (8)$$

where p indexes the set of NPI contexts which contain y[3], and the denominator is the number of NPI contexts which contain y.

[2]DLD09 actually use the number of NPI contexts containing y rather than $count^{\mathcal{N}}(y)$, but we find that using the raw count works better in our experiments.

[3]In DLD09, the corresponding equation does not indicate that p should be the contexts that include y, but it is clear from the surrounding text that our version is the intended meaning. If all the NPI contexts were included in the summation, $S_d(y)$ would reduce to inverse relative frequency.

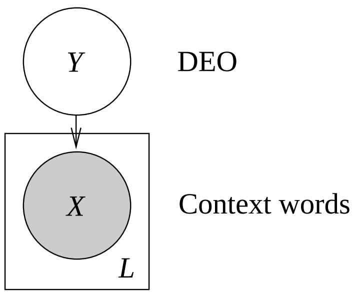


Figure 1: Naïve Bayes formulation of DEO detection.

DLD09 find that distillation seems to improve the performance of DEO detection in BLLIP. Later work by DL10, however, shows that distillation does not seem to improve performance over the baseline method in Romanian, and the authors also note that distillation does not improve performance in their experiments on co-learning NPIs and DEOs via bootstrapping.

A better mathematical grounding of the distillation method's apparent heuristic in terms of existing probabilistic models sheds light on the mixed performance of distillation across languages and experimental settings. In particular, it turns out that the distillation method of DLD09 is equivalent to one iteration of EM prior re-estimation in a Naïve Bayes model. Given a lexicon $\mathcal{L}$ of L words, let each NPI context be one sample generated by the model. One sample consists of a latent categorical (i.e., a multinomial with one trial) variable Y whose values range over $\mathcal{L}$, corresponding to the DEO that licenses the context, and observed Bernoulli variables $\vec{X} = X_{i=1...L}$ which indicate whether a word appears in the NPI context (Figure 1). This method does not attempt to model the order of the observed words, nor the number of times each word appears. Formally, a Naïve Bayes model is given by the following expression:

$$P(\vec{X}, Y) = \prod_{i=1}^{L} P(X_i|Y)P(Y). \qquad (9)$$

The probability of a DEO given a particular NPI context is

$$P(Y|\vec{X}) \propto \prod_{i=1}^{L} P(X_i|Y)P(Y). \qquad (10)$$

The probability of a set of observed NPI contexts $\mathcal{N}$ is the product of the probabilities for each sample:

$$P(\mathcal{N}) = \prod_{\vec{X} \in \mathcal{N}} P(\vec{X}) \tag{11}$$

$$P(\vec{X}) = \sum_{y \in \mathcal{L}} P(\vec{X}, y). \tag{12}$$

We first instantiate the baseline method of DLD09 by initializing the parameters to the model, $P(X_i = 1|y)$ and $P(Y = y)$, such that $P(Y = y)$ is proportional to $S(y)$. Recall that this initialization utilizes domain knowledge about the correlation between NPIs and DEOs, inspired by Ladusaw's hypothesis:

$$P(Y = y) = S(y) / \sum_{y'} S(y') \tag{13}$$

$$P(X_i = 1|y) = \begin{cases} 1 & \text{if } X_i \text{ corresponds to } y \\ 0.5 & \text{otherwise.} \end{cases} \tag{14}$$

This initialization of $P(X_i = 1|y)$ ensures that the the value of y corresponds to one of the words in the NPI context, and the initialization of $P(Y)$ is simply a normalization of $S(y)$.

Since we are working in an unsupervised setting, there are no labels for Y available. A common and reasonable assumption about learning the parameter settings in this case is to find the parameters that maximize the likelihood of the observed training data; i.e., the NPI contexts:

$$\hat{\theta} = \underset{\theta}{\text{argmax}}\ P(\mathcal{N}; \theta). \tag{15}$$

The EM algorithm is a well-known iterative algorithm for performing this optimization. Assuming that the prior $P(Y = y)$ is a categorical distribution, the M-step estimate of these parameters after one iteration through the corpus is as follows:

$$P^{t+1}(Y = y) = \sum_{\vec{X} \in \mathcal{N}} \frac{P^t(y|\vec{X})}{\sum_{y'} P^t(y'|\vec{X})} \tag{16}$$

We do not re-estimate $P(X_i = 1|y)$ because their role is simply to ensure that the DEO responsible for an NPI context exists in the context. Estimating these parameters would exacerbate the problems with EM for this task which we will discuss shortly.

$P(Y)$ gives a prior probability that a certain word-type y is a DEO in an NPI context, without normalizing for the frequency of y in NPI contexts. Since we are interested in estimating the context-independent probability that y is a DEO, we must calculate the probability that a word is a DEO given that it appears in an NPI context. Let X_y be the observed variable corresponding to y. Then, the expression we are interested in is $P(y|X_y = 1)$. We now show that $P(y|X_y = 1) = P(y)/P(X_y = 1)$, and that this expression is equivalent to (8).

$$P(y|X_y = 1) = \frac{P(y, X_y = 1)}{P(X_y = 1)} \tag{17}$$

Recall that $P(y, X_y = 0) = 0$ because of the assumption that a DEO appears in the NPI context that it generates. Thus,

$$\begin{aligned} P(y, X_y = 1) &= P(y, X_y = 1) + P(y, X_y = 0) \\ &= P(y) \end{aligned} \tag{18}$$

One iteration of EM to calculate this probability is equivalent to the distillation method of DLD09. In particular, the numerator of (17), which we just showed to be equal to the estimate of $P(Y)$ given by (16), is exactly the sum of the responsibilities for a particular y, and is proportional to the summation in (8) modulo normalization, because $P(\vec{X}|y)$ is constant for all y in the context. The denominator $P(X_y = 1)$ is simply the proportion of contexts containing y, which is proportional to $|\mathcal{N}_y|$. Since both the numerator and denominator are equivalent up to a constant factor, an identical ranking is produced by distillation and EM prior re-estimation.

Unfortunately, the EM algorithm does not provide good results on this task. In fact, as more iterations of EM are run, the performance drops drastically, even though the corpus likelihood is increasing. The reason is that unsupervised EM learning is not constrained or biased towards learning a good set of DEOs. Rather, a higher data likelihood can be achieved simply by assigning high prior probabilities to frequent word-types.

This can be seen qualitatively by considering the top-ranking DEOs after several iterations of EM/distillation (Figure 2). The top-ranking words are simply function words or other words common in the corpus, which have nothing to do with downward entailment. In effect,

1 iteration	*2 iterations*	*3 iterations*
denies	the	the
denied	to	to
unaware	denied	that
longest	than	than
hardly	that	and
lacking	if	has
deny	has	if
nobody	denies	of
opposes	and	denied
highest	but	denies

Figure 2: Top 10 DEOs after iterations of EM on BLLIP.

EM/distillation overrides the initialization based on Ladusaw's hypothesis and finds another solution with a higher data likelihood. We will also provide a quantitative analysis of the effects of EM/distillation in Section 5.

4 Alternative to EM: Maximizing the Posterior Classification Certainty

We have seen that in trying to solve the piggybacker problem, EM/distillation too readily abandons the initialization based on Ladusaw's hypothesis, leading to an incorrect solution. Instead of optimizing the data likelihood, what we need is a measure of the number of plausible DEO candidates there are in an NPI context, and a method that refines the scores towards having only one such plausible candidate per context. To this end, we define the *classification certainty* to be the product of the maximum posterior classification probabilities over the DEO candidates. For a set of hidden variables $y^{\mathcal{N}}$ for NPI contexts $\mathcal{N}$, this is the expression:

$$Certainty(y^{\mathcal{N}}|\mathcal{N}) = \prod_{\vec{X}\in\mathcal{N}} \max_{y} P(y|\vec{X}). \quad (19)$$

To increase this certainty score, we propose a novel iterative heuristic method for refining the baseline initializations of $P(Y)$. Unlike EM/distillation, our method biases learning towards trusting the initialization, but refines the scores towards having only one plausible DEO per context in the training corpus. This is accomplished by treating the problem as a DEO classification problem, and then maximizing an objective ratio that favours one DEO per context. Our method is not guaranteed to increase classification certainty between iterations, but we will show that it does increase certainty very quickly in practice.

The key observation that allows us to resolve the tension between trusting the initialization and enforcing one DEO per NPI context is that the distributions of words that co-occur with DEOs and piggybackers are different, and that this difference follows from Ladusaw's hypothesis. In particular, while DEOs may appear with or without piggybackers in NPI contexts, piggybackers do not appear without DEOs in NPI contexts, because Ladusaw's hypothesis stipulates that a DEO is required to license the NPI in the first place. Thus, the presence of a high-scoring DEO candidate among otherwise low-scoring words is strong evidence that the high-scoring word is not a piggybacker and its high score from the initialization is deserved. Conversely, a DEO candidate which always appears in the presence of other strong DEO candidates is likely a piggybacker whose initial high score should be discounted.

We now describe our heuristic method that is based on this intuition. For clarity, we use scores rather than probabilities in the following explanation, though it is equally applicable to either. As in EM/distillation, the method is initialized with the baseline $S(y)$ scores. One iteration of the method proceeds as follows. Let the score of the strongest DEO candidate in an NPI context p be:

$$M(p) = \max_{y\in p} S_h^t(y), \quad (20)$$

where $S_h^t(y)$ is the score of candidate y at the tth iteration according to this heuristic method.

Then, for each word-type y in each context p, we compare the current score of y to the scores of the other words in p. If y is currently the strongest DEO candidate in p, then we give y credit equal to the proportional change to $M(p)$ if y were removed (Context p without y is denoted $p \setminus y$). A large change means that y is the only plausible DEO candidate in p, while a small change means that there are other plausible DEO candidates. If y is not currently the strongest DEO candidate, it receives no credit:

$$cred(p,y) = \begin{cases} \frac{M(p)-M(p\setminus y)}{M(p)} & \text{if } S_h^t(y) = M(p) \\ 0 & \text{otherwise.} \end{cases} \quad (21)$$

NPI contexts

$$A\ B\ C, B\ C, B\ C, D\ C$$

Original scores

$$S(A) = 5, S(B) = 4, S(C) = 1, S(D) = 2$$

Updated scores

$$\begin{aligned}
S_h(A) &= 5 \times (5-4)/5 &&= 1\\
S_h(B) &= 4 \times (0 + 2 \times (4-1)/4)/3 &&= 2\\
S_h(C) &= 1 \times (0+0+0) &&= 0\\
S_h(D) &= 2 \times (2-1)/2 &&= 1
\end{aligned}$$

Figure 3: Example of one iteration of the certainty-based heuristic on four NPI contexts with four words in the lexicon.

Then, the average credit received by each y is a measure of how much we should trust the current score for y. The updated score for each DEO candidate is the original score multiplied by this average:

$$S_h^{t+1}(y) = \frac{S_h^t(y)}{|\mathcal{N}_y|} \times \sum_{p \in \mathcal{N}_y} cred(p, y). \qquad (22)$$

The probability $P^{t+1}(Y = y)$ is then simply $S_h^{t+1}(y)$ normalized:

$$P^{t+1}(Y = y) = \frac{S_h^{t+1}(y)}{\sum_{y' \in \mathcal{L}} S_h^{t+1}(y')}. \qquad (23)$$

We iteratively reduce the scores in this fashion to get better estimates of the relative suitability of word-types as DEOs.

An example of this method and how it solves the piggybacker problem is given in Figure 3. In this example, we would like to learn that B and D are DEOs, A is a piggybacker, and C is a frequent word-type, such as a stop word. Using the original scores, piggybacker A would appear to be the most likely word to be a DEO. However, by noticing that it never occurs on its own with words that are unlikely to be DEOs (in the example, word C), our heuristic penalizes A more than B, and ranks B higher after one iteration. EM prior re-estimation would not correctly solve this example, as it would converge on a solution where C receives all of the probability mass because it appears in all of the contexts, even though it is unlikely to be a DEO according to the initialization.

5 Experiments

We evaluate the performance of these methods on the BLLIP corpus (~30M words) and the AFP portion of the Gigaword corpus (~338M words). Following DLD09, we define an NPI context to be all the words to the left of an NPI, up to the closest comma or semi-colon, and removed NPI contexts which contain the most common DEOs like *not*. We further removed all empty NPI contexts or those which only contain other punctuation. After this filtering, there were 26696 NPI contexts in BLLIP and 211041 NPI contexts in AFP, using the same list of 26 NPIs defined by DLD09.

We first define an automatic measure of performance that is common in information retrieval. We use average precision to quantify how well a system separates DEOs from non-DEOs. Given a list of known DEOs, G, and non-DEOs, the average precision of a ranked list of items, X, is defined by the following equation:

$$AP(X) = \frac{\sum_{k=1}^{n} P(X_{1 \ldots k}) \times \mathbf{1}(x_k \in G)}{|G|}, \qquad (24)$$

where $P(X_{1 \ldots k})$ is the precision of the first k items and $\mathbf{1}(x_k \in G)$ is an indicator function which is 1 if x is in the gold standard list of DEOs and 0 otherwise.

DLD09 simply evaluated the top 150 output DEO candidates by their systems, and qualitatively judged the precision of the top-k candidates at various values of k up to 150. Average precision can be seen as a generalization of this evaluation procedure that is sensitive to the ranking of DEOs and non-DEOs. For development purposes, we use the list of 150 annotations by DLD09. Of these, 90 were DEOs, 30 were not, and 30 were classified as "other" (they were either difficult to classify, or were other types of non-veridical operators like comparatives or conditionals). We discarded the 30 "other" items and ignored all items not in the remaining 120 items when evaluating a ranked list of DEO candidates. We call this measure AP_{120}.

In addition, we annotated DEO candidates from the top-150 rankings produced by our certaintyabsolve, abstain, banish, bereft, boycott, caution, clear, coy, delay, denial, desist, devoid, disavow, discount, dispel, disqualify, downplay, exempt, exonerate, foil, forbid, forego, impossible, inconceivable, irrespective, limit, mitigate, nip, noone, omit, outweigh, precondition, pre-empt, prerequisite, refute, remove[5], repel, repulse, scarcely, scotch, scuttle, seldom, sensitive, shy, sidestep, snuff, thwart, waive, zero-tolerance

Figure 4: Lemmata of DEOs identified in this work not found by DLD09.

based heuristic on BLLIP and also by the distillation and heuristic methods on AFP, in order to better evaluate the final output of the methods. This produced an additional 68 DEOs (narrowly defined) (Figure 4), 58 non-DEOs, and 31 “other” items[4]. Adding the DEOs and non-DEOs we found to the 120 items from above, we have an expanded list of 246 items to rank, and a corresponding average precision which we call AP_{246}.

We employ the frequency cut-offs used by DLD09 for sparsity reasons. A word-type must appear at least 10 times in an NPI context and 150 times in the corpus overall to be considered. We treat BLLIP as a development corpus and use AP_{120} on AFP to determine the number of iterations to run our heuristic (5 iterations for BLLIP and 13 iterations for AFP). We run EM/distillation for one iteration in development and testing, because more iterations hurt performance, as explained in Section 3.

We first report the AP_{120} results of our experiments on the BLLIP corpus (Table 1 second column). Our method outperforms both EM/distillation and the baseline method. These results are replicated on the final test set from AFP using the full set of annotations AP_{246} (Table 1 third column). Note that the scores are lower when using all the annotations because there are more non-DEOs relative to DEOs in this list, making the ranking task more challenging.

A better understanding of the algorithms can be obtained by examining the data likelihood and the classification certainty at each iteration of the algorithms (Figure 5). Whereas EM/distillation maximizes the former expression, the certainty-based heuristic method actually decreases data likelihood for the first couple of iterations before increasing it again. In terms of classification certainty, EM/distillation converges to a lower classification certainty score compared to our heuristic method. Thus, our method better captures the assumption of one DEO per NPI context.

Method	BLLIP AP_{120}	AFP AP_{246}
Baseline	.879	.734
Distillation	.946	.785
This work	**.955**	**.809**

Table 1: Average precision results on the BLLIP and AFP corpora.

6 Bootstrapping to Co-Learn NPIs and DEOs

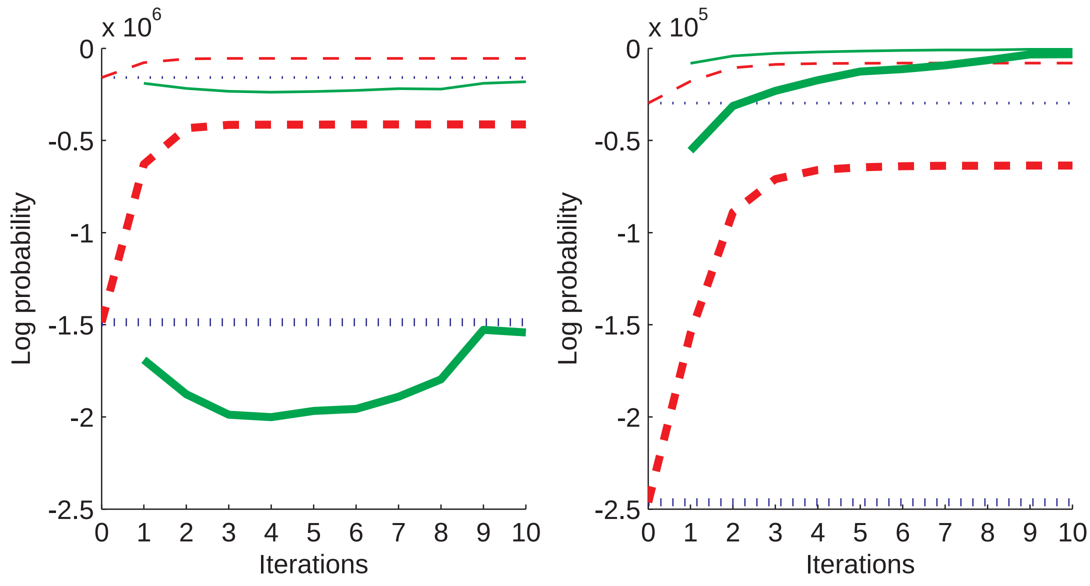
The above experiments show that the heuristic method outperforms the EM/distillation method given a list of NPIs. We would like to extend this result to novel domains, corpora, and languages. DLD09 and DL10 proposed the following bootstrapping algorithm for co-learning NPIs and DEOs given a much smaller list of NPIs as a seed set.

1. Begin with a small set of seed NPIs
2. Iterate:
   (a) Use the current list of NPIs to learn a list of DEOs
   (b) Use the current list of DEOs to learn a list of NPIs

Interestingly, DL10 report that while this method works in Romanian data, it does not work in the English BLLIP corpus. They speculate that the reason might be due to the nature of the English DEO *any*, which can occur in all classes of DE contexts according to an analysis by Haspelmath (1997). Further, they find that in Romanian, distillation does not perform better than the baseline method during Step (2a). While this linguistic explanation may certainly be a factor, we raise

[4]The complete list will be made publicly available.

[5]We disagree with DLD09 that *remove* is not downward-entailing; e.g., *The detergent removed stains from his clothing.* ⇒ *The detergent removed stains from his shirts.*



(a) Data log likelihood. (b) Log classification certainty probabilities.

Figure 5: Log likelihood and classification certainty probabilities of NPI contexts in two corpora. Thinner lines near the top are for BLLIP; thicker lines for AFP. Blue dotted: baseline; red dashed: distillation; green solid: our certainty-based heuristic method. $P(\vec{X}|y)$ probabilities are not included since they would only result in a constant offset in the log domain.

a second possibility that the distillation algorithm itself may be responsible for these results. As evidence, we show that the heuristic algorithm is able to work in English with just the single seed NPI *any*, and in fact the bootstrapping approach in conjunction with our heuristic even outperforms the above approaches when using a static list of NPIs.

In particular, we use the methods described in the previous sections for Step (2a), and the following ratio to rank NPI candidates in Step (2b), corresponding to the baseline method to detect DEOs in reverse:

$$T(x) = \frac{count^{\mathcal{D}}(x)/tokens(\mathcal{D})}{count^{\mathcal{C}}(x)/tokens(\mathcal{C})}. \qquad (25)$$

Here, $count^{\mathcal{D}}(x)$ refers to the number of occurrences of NPI candidate x in DEO contexts $\mathcal{D}$, defined to be the words to the right of a DEO operator up to a comma or semi-colon. We do not use the EM/distillation or heuristic methods in Step (2b). Learning NPIs from DEOs is a much harder problem than learning DEOs from NPIs. Because DEOs (and other non-veridical operators) license NPIs, the majority of occurrences of NPIs will be in the context of a DEO, modulo ambiguity of DEOs such as the free-choice *any* and other spurious correlations such as piggybackers as discussed earlier. In the other direction, it is not the case that DEOs always or nearly always appear in the context of an NPI. Rather, the most common collocations of DEOs are the selectional preferences of the DEO, such as common arguments to verbal DEOs, prepositions that are part of the subcategorization of the DEO, and words that together with the surface form of the DEO comprise an idiomatic expression or multi-word expression. Further, NPIs are more likely to be composed of multiple words, while many DEOs are single words, possibly with PP subcategorization requirements which can be filled in post hoc. Because of these issues, we cannot trust the initialization to learn NPIs nearly as much as with DEOs, and cannot use the distillation or certainty methods for this step. Rather, the hope is that learning a noisy list of "pseudo-NPIs", which often occur in negative contexts but may not actually be NPIs, can still improve the performance of DEO detection.

There are a number of parameters to the method which we tuned to the BLLIP corpus using AP_{120}. At the end of Step (2a), we use the current top 25 DEOs plus 5 per iteration as the DEO list for the next step. To the initial seed NPI of

Method	BLLIP AP_{120}	AFP AP_{246}
Baseline	.889 (+.010)	.739 (−.005)
Distillation	.930 (−.016)	.804 (+.019)
This work	**.962** (+.007)	**.821** (+.012)

Table 2: Average precision results with bootstrapping on the BLLIP and AFP corpora. Absolute gain in average precision compared to using a fixed list of NPIs given in brackets.

anymore, anything, anytime, avail, bother, bothered, budge, budged, countenance, faze, fazed, inkling, iota, jibe, mince, nor, whatsoever, whit

Figure 6: Probable NPIs found by bootstrapping using the certainty-based heuristic method.

any, we add the top 5 ranking NPI candidates at the end of Step (2b) in each subsequent iteration. We ran the bootstrapping algorithm for 11 iterations for all three algorithms. The final evaluation was done on AFP using AP_{246}.

The results show that bootstrapping can indeed improve performance, even in English (Table 2). Using bootstrapping to co-learn NPIs and DEOs actually results in better performance than specifying a static list of NPIs. The certainty-based heuristic in particular achieves gains with bootstrapping in both corpora, in contrast to the baseline and distillation methods. Another factor that we found to be important is to add a sufficient number of NPIs to the NPI list each iteration, as adding too few NPIs results in only a small change in the NPI contexts available for DEO detection. DL10 only added one NPI per iteration, which may explain why they did not find any improvement with bootstrapping in English. It also appears that learning the pseudo-NPIs does not hurt performance in detecting DEO, and further, that a number of true NPIs are learned by our method (Figure 6).

7 Conclusion

We have proposed a novel unsupervised method for discovering downward-entailing operators from raw text based on their co-occurrence with negative polarity items. Unlike the distillation method of DLD09, which we show to be an instance of EM prior re-estimation, our method directly addresses the issue of piggybackers which spuriously correlate with NPIs but are not downward-entailing. This is achieved by maximizing the posterior classification certainty of the corpus in a way that respects the initialization, rather than maximizing the data likelihood as in EM/distillation. Our method outperforms distillation and a baseline method on two corpora as well as in a bootstrapping setting where NPIs and DEOs are jointly learned. It achieves the best performance in the bootstrapping setting, rather than when using a fixed list of NPIs. The performance of our algorithm suggests that it is suitable for other corpora and languages.

Interesting future research directions include detecting DEOs of more than one word as well as distinguishing the particular word sense and subcategorization that is downward-entailing. Another problem that should be addressed is the scope of the downward entailment, generalizing work being done in detecting the scope of negation (Councill et al., 2010, for example).

Acknowledgments

We would like to thank Cristian Danescu-Niculescu-Mizil for his help with replicating his results on the BLLIP corpus. This project was supported by the Natural Sciences and Engineering Research Council of Canada.

References

Luisa Bentivogli, Peter Clark, Ido Dagan, Hoa T. Dang, and Danilo Giampiccolo. 2010. The sixth pascal recognizing textual entailment challenge. In *The Text Analysis Conference (TAC 2010)*.

Isaac G. Councill, Ryan McDonald, and Leonid Velikovich. 2010. What's great and what's not: Learning to classify the scope of negation for improved sentiment analysis. In *Proceedings of the Workshop on Negation and Speculation in Natural Language Processing*, pages 51–59. Association for Computational Linguistics.

Cristian Danescu-Niculescu-Mizil and Lillian Lee. 2010. Don't 'have a clue'?: Unsupervised co-learning of downward-entailing operators. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 247–252. Association for Computational Linguistics.

Cristian Danescu-Niculescu-Mizil, Lillian Lee, and Richard Ducott. 2009. Without a 'doubt'?: Unsupervised discovery of downward-entailing operators. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*.

William A. Gale, Kenneth W. Church, and David Yarowsky. 1992. One sense per discourse. In *Proceedings of the Workshop on Speech and Natural Language*, pages 233–237. Association for Computational Linguistics.

Anastasia Giannakidou. 2002. Licensing and sensitivity in polarity items: from downward entailment to nonveridicality. *CLS*, 38:29–53.

Martin Haspelmath. 1997. *Indefinite pronouns*. Oxford University Press.

Jack Hoeksema. 1997. Corpus study of negative polarity items. *IV-V Jornades de corpus linguistics 1996–1997*.

William A. Ladusaw. 1980. On the notion 'affective' in the analysis of negative-polarity items. *Journal of Linguistic Research*, 1(2):1–16.

Timm Lichte and Jan-Philipp Soehn. 2007. The retrieval and classification of negative polarity items using statistical profiles. *Roots: Linguistics in Search of Its Evidential Base*, pages 249–266.

Bill MacCartney and Christopher D. Manning. 2008. Modeling semantic containment and exclusion in natural language inference. In *Proceedings of the 22nd International Conference on Computational Linguistics*.

Frank Richter, Fabienne Fritzinger, and Marion Weller. 2010. Who can see the forest for the trees? extracting multiword negative polarity items from dependency-parsed text. *Journal for Language Technology and Computational Linguistics*, 25:83–110.

Ton van der Wouden. 1997. *Negative Contexts: Collocation, Polarity and Multiple Negation*. Routledge.