Using Linguistic Knowledge to Automatically Learn Monotonicity Properties*

Michelle Renee Morales
The Graduate Center, CUNY
mmorales@gradcenter.cuny.edu

Semanticists have long investigated the linguistic phenomenon of negative polarity items (NPIs) and have demonstrated that there exists a relationship between NPIs and downward entailing (DE) operators. NPI licensing theories aim to explain the exact nature of that relationship. These semantic theories can be leveraged to aid research in other linguistic subfields. The contribution of this paper is to demonstrate how we may successfully use linguistic knowledge and the relationship between NPIs and DEs to build an informed computational system that can successfully detect DE operators from text. This work presents an algorithm that automatically detects a word's monotonicity information.

1. Introduction

Semantic theories can and should be leveraged to aid research in other linguistic subfields. This work is motivated by semantic theory on the relationship between NPIs and DE operators. The contribution of this paper is to demonstrate how we may successfully use linguistic knowledge and the relationship between NPIs and DE operators to build an informed computational system that can successfully detect DE operators from text. This task is a crucial contribution to any inference system and has widespread applications in the field of computational linguistics. Therefore, the goal of this work is to demonstrate how we can integrate knowledge of linguistic theory in designing computational inference systems. Specifically, this paper presents an algorithm that automatically detects a word's monotonicity information. Due to this, we will review some of the leading and most influential semantic accounts of NPI licensing, so that the connection between NPIs and monotonicity can be made clear, and thereby relevant to inference system design. This paper is outlined as follows, Section 2 is dedicated to introducing NPIs

* Thank you to my Advisor Dr. Andrew Rosenberg for his guidance on this project.

Studies in the Linguistic Sciences: Illinois Working Papers **2016**: 36-54

and their distribution, Section 3 provides a short introduction to the licensing of NPIs, Section 4 outlines a novel contribution to the subfield of computational linguistics: a linguistically informed algorithm that automatically learns downward entailing operators from text. Lastly, we conclude our discussion in Section 5.

2. Negative Polarity Items

This section introduces the data on Negative Polarity Items (NPIs) in English. The feature characteristic of NPIs is their restricted distribution. As examples consider (1-2).

(1) a. John didn't know *any* French.
b. * John knew *any* French.

(2) a. I haven't *ever* met Mr. Smith.
b. * I have *ever* met Mr. Smith.

In the preceding examples, the distinction between the (a) and (b) sentences suggests that an NPI is sensitive to its environment, appearing acceptable in the negative sentences but unacceptable in their positive counterparts. Similar findings hold for other NPIs. The class of NPIs in English includes, but is not limited to the lexical items *any*[1], *ever*, *yet*, and *much*. The main question to answer when investigating NPIs, is which contexts license them. This question is extremely difficult due to the variable nature of NPIs. In addition to negation, there exist a number of other expressions that license NPIs in English. The following sentences from Ladusaw 1979 characterize environments in which NPIs can or cannot appear and demonstrate the difficulty of providing a unified account for their distribution. We see that the list of elements includes members of various syntactic categories and extends far beyond what we would consider to be overt negation. In the following examples, each word or phrase in brackets represents an item whose appearance allows or doesn't allow for the subsequent appearance of an NPI, which is italicized.

Adverbial Conjunctions:

[1] This includes any variation of any including anybody, anymore, anyone, anything etc.

(3) a. John will replace the money {before/if} *anyone ever* misses it.
b. * John will replace the money {after/when} *anyone ever* misses it.

Degree Words:

(4) a. John is {too smart} to *ever* do *anything* like that again.
b. * John is {smart enough} to *ever* do *anything* like that again.

Determiners:

(5) a. {No one/At most three people/Few students} who had read *anything* about phrenology attended *any* of the lectures.
b. * {Someone/At least three people/Many students} who had read *anything* about phrenology attended *any* of the lectures.

Prepositions:

(6) a. John finished his homework {without} *any* help.
b. * John finished his homework {with} *any* help.

Quantification Adverbs:

(7) a. I {never/rarely/seldom)} *ever* eat *anything* for breakfast.
b. * I {usually/always/sometimes} *ever* eat *anything* for breakfast.

Verbs and Adjectives:

(8) a. It's {hard/difficult} to find *anyone* who has *ever* read *anything much* about phrenology.
b. * It's {easy/possible} to find *anyone* who has *ever* read *anything much* about phrenology.

(9) a. John {doubted/denied} that *anyone* would *ever* discover that the money was missing.

b. * John {believed/hoped} that *anyone* would *ever* discover that the money was missing.

Comparatives:

(10) a. He was {taller than} we *ever* thought he would be.
b. * He was {so tall that} we *ever* thought he would bump his head.

Questions:

(11) a. Have you *ever* met George?
b. * You have *ever* met George.

Taken together all the preceding examples show that NPIs acceptability and distribution is considerably varied and widespread. Ultimately, the goal of this work is to design an algorithm that relies on NPI distribution to learn a word's monotonicity information or entailment direction. Therefore, understanding the distribution of NPIs is crucial. Early breakthroughs by Fauconnier 1975 and Ladusaw 1979 discovered that the relevant property that distinguished acceptable from unacceptable NPI contexts dealt with entailment patterns. The arrows in the example below from Chierchia 2013 represent the direction of entailment.

(12) a. {x: x eats pizza with anchovies} $\subseteq$ {x: x eats pizza}
b. Superset inference:
i. Somebody ate pizza.
$\Uparrow$
ii. Somebody ate pizza with anchovies.
c. Subset inference:
i. Nobody ate pizza.
$\Downarrow$
ii. Nobody ate pizza with anchovies.

Given the two sets in question, shown in (12a), the inference in (12b) goes from a subset to a superset. Contexts that give rise to this pattern are referred

to as Upward Entailing (UE) or upward monotone. The situation is reversed in (12c); the inference made goes from a superset to a subset. Contexts that exhibit this property are referred to as downward entailing (DE) or downward monotone. Chierchia 2013 describes the notion of being DE as the generalized semantic notion of "being negative". The match between downward entailment and licensing of NPIs is quite remarkable and has attracted significant amounts of discussion, which is summarized and described below in Section 3.

3. NPI Licensing

Although the distribution of NPIs has been treated in both the syntactic and semantic literature, this paper will approach NPI licensing from a semantic angle[2].

3.1. Ladusaw's 1979 Licensing Analysis

According to Ladusaw, NPIs are only licensed in the scope of downward entailing operators. A DE operator is defined as follows:

(13) O is a DE operator iff A $\Rightarrow$ B then O(B) $\Rightarrow$ O(A)

This rule then accounts for the distribution of the NPI *any*.

(14) a. I don't have *any* potatoes.
b. * I have *any* potatoes.

The distinction between (14a) and (14b) illustrates the dependence of *any* on the presence of a DE operator, in this case negation. However, downward entailment is not limited to negation; the quantifier *every* exhibits this property on its first argument.

[2] For syntactic accounts of NPI licensing see Klima 1964, Baker 1970 and Progovac 1988; 1993. Also, from an algebraic theory that focuses on NPI types and contexts see Zwarts 1998.

(15) a. Every person who *ever* ate pepperoni pizza from that place got sick.
b. * Every person who eats my pepperoni pizza will *ever* get sick.

We assume quantifiers like *every* represent relations between sets. The semantics of *every* is defined as follows:

(16) $[\![\text{every}]\!] = [\lambda P.\lambda Q$ for all entities x, if P(x)=1, then Q(x)=1]

Given the examples in (15) and the semantics of *every*, consider the inferences that arise. In order for *every* to be DE on its first argument, the inference that arises from the first argument must be a subset inference: an inference from a set to a subset. On the contrary, the second argument should exhibit a superset inference, demonstrating that *every* is UE on its second argument. To simplify, consider an example that does not include an NPI (17).

(17) a. Every pizza makes me sick. ⇒ Every pepperoni pizza makes me sick.
b. Every time at lunch I eat pepperoni pizza. ⇒ Every time at lunch I eat pizza.

Given the first sentence in (17) a speaker can logically infer that the statement holds true for any subset of *pizza*. This subset inference demonstrates that the first argument of *every* is in a DE environment. Contrastingly, a superset inference in (17b) supports that *every* is UE on its second argument. Following Ladusaw's analysis, this DE property correctly accounts for the distinction between the sentences in (15); the NPI *ever* may only appear in the first argument of *every* leading to the ungrammatically of (15b), where the NPI appears in the second argument: a UE environment.

Although Ladusaw's analysis is quite successful, there remain some issues. Specifically, Ladusaw only provides a descriptive generalization of NPIs and does not delve into how the meaning of NPIs trigger a certain distribution. A major step that subsequent licensing theories made was to question why these contexts license NPIs; an overview of licensing theories is outside the scope of this paper. For a detailed description of influential NPI licensing theories see Kadmon and Landman (1993), Krifka (1995), and Chierchia (2013).

Moreover, there exist some problem cases that Ladusaw's theory and other subsequent licensing theories licensing struggle to account for. For example, Linebarger (1987) showed, contra Ladusaw (1979) that merely being in a DE environment is not sufficient for licensing and that NPIs are subject to locality conditions. Locality[3] is especially relevant to our computational system, and we will consider the importance of locality when constructing our system, which will be made clear in Section 4.

3.2. Summary

Ladusaw demonstrated that NPI licensing takes place in DE contexts: a descriptive generalization, although many issues still remained. The goal of this work is to build an informed system that can learn DE operators using linguistic knowledge of NPIs. The main motivation for our computational system comes from the common stance shared by many of licensing theories in the field —NPIs appear in DE environments. We inform our system by constructing linguistic rules which mirror ideas from these theories. Specifically, we adopt Ladusaw's hypothesis and require NPIs to appear in the scope of DE operators. This requirement allows our system to make a DE operator prediction in every NPI context it finds. We also incorporate a linguistic rule meant to capture locality, by imposing some restriction on the distance between the NPI and DE operator. Our system is described in further detail in the following section.

4. Discovering Downward Entailing Operators

Given two sentences (T and H) a computational inference system must determine whether T entails H. Many Natural Language Processing (NLP) applications[4] involve semantic inference as way to recognize that one target meaning can be deduced from different text variants. There are many linguistic phenomena that complicate this task. One property researchers have considered is monotonicity. Since monotonicity is a pervasive feature of natural language, it is an important factor to consider. As we have noted previously, in the scope of a downward entailing operator, such as negation, the entailment direction is reversed as demonstrated in (18).

[3] For a discussion of locality and intervention effects see Linebarger (1987).

[4] Some NLP applications that require semantic inference include Question Answering, Information Extraction, summarization, and machine translation.

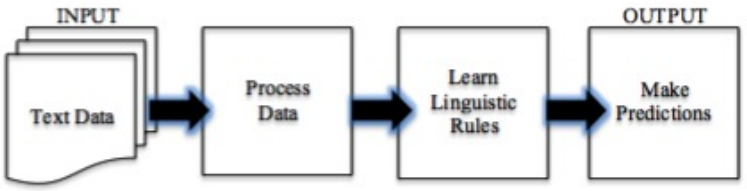
(18) a. T: I will adopt a black and white kitten. ⇒ H: I will adopt a kitten.
b. T: I will not adopt a kitten. ⇒ H: I will not adopt a black and white kitten.

Given *black and white kitten* is a subset of *kitten*, in a UE environment, like (18a), we infer from the subset to the set. However, in a DE environment this entailment direction is reversed, as is shown in (18b). If we could automatically learn DE environments, then a inference system would be able to correctly predict when the entailment direction between two sentences should be reversed, leading to better precision in inference tasks.

Given the importance and prevalence of monotonicity in ordinary reasoning, researchers have incorporated monotonicity components into their inference systems MacCartney (2009); MacCartney & Manning (2009). However, these monotonicity components relied on a manually annotated list that includes words and their monotonicity information: labels for UE or DE. Although these sorts of lists have proven to be very useful, they are quite difficult and time intensive to create. The existence of these lists is also a significant barrier to applying these techniques to new languages. Therefore, some efforts have been made to automatically learn DE operators from text; this task is the main contribution of this work. We build an algorithm that automatically learns DE operators from text using linguistic rules. Our approach is motivated by linguistic knowledge of negative polarity items (NPIs). Many have claimed Ladusaw (1979); Kadmon & Landman (1993); Krifka (1995); Chierchia (2013) that negative polarity items require a downward entailing environment. Therefore, we adopt this hypothesis and use NPIs as clues to discover DE operators from text.

We propose a novel method for detecting DE operators and compare our approach with two previous approaches: Cheung and Penn 2012 and Danescu et al. 2009. Our method is the first to use word vectors and a cosine similarity measure to score and rank possible DE operators. We achieve state-of-the-art results finding our approach to outperform previous approaches in average precision. The figure below gives a simplified breakdown of our system's pipeline.

The input to our system is a large corpus of English text. We then process the data by performing part of speech tagging, parsing, and word vector extraction. Next, our system learns the linguistic rules we constructed that were motivated by NPI licensing theories from the semantic literature. Lastly,



using the processed data, the linguistic rules, and our novel method for making predictions our system outputs a ranked list of possible DE operators. We compare our system with two other existing systems. The following sections explain the details of each step in our pipeline.

4.1. Data

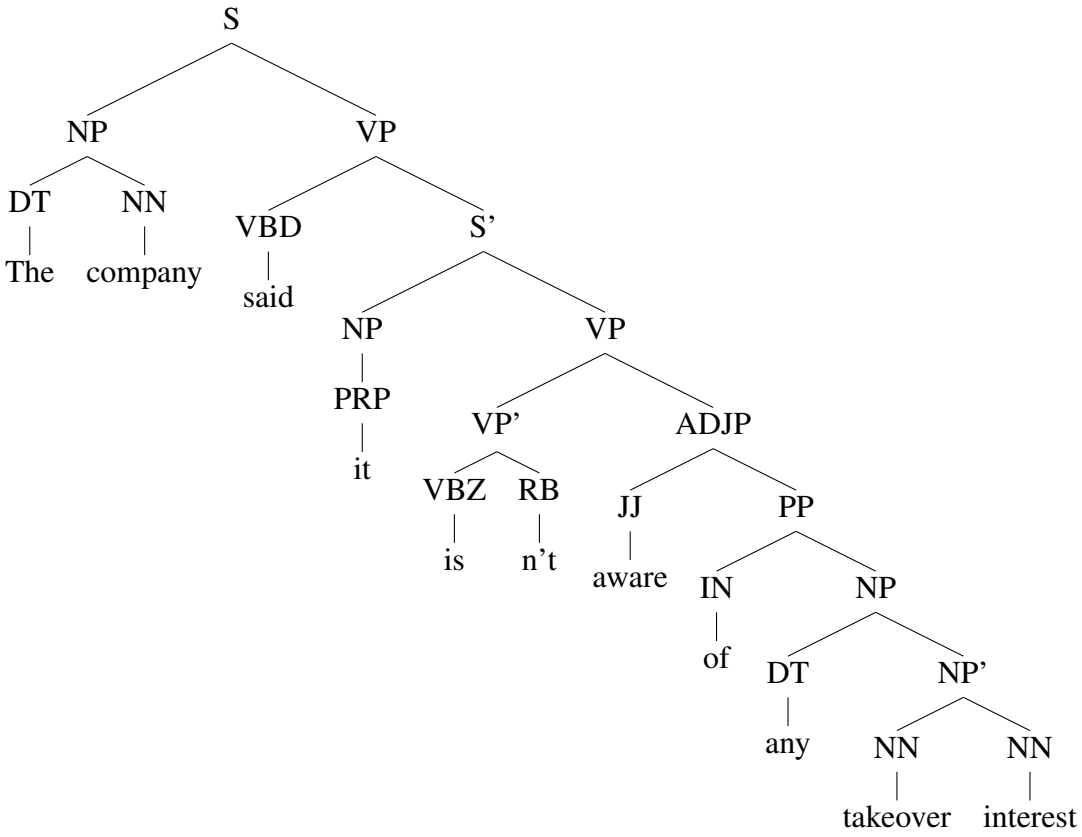
This work uses the Brown Corpus available from the Natural Language Toolkit Bird (2006). The Brown Corpus includes 57,340 sentences, which consist of approximately 1.2 million words. When exploring how often NPIs appear in the corpus, we find 2,562 NPI contexts.

4.2. Processes

This step consists of 3 processing tasks: part-of-speech (POS) tagging, parsing, and word vector extraction. For each sentence in the corpus, we first tag each word with its POS tag using NLTK's Penn Treebank tagger Bird (2006), as shown in (19).

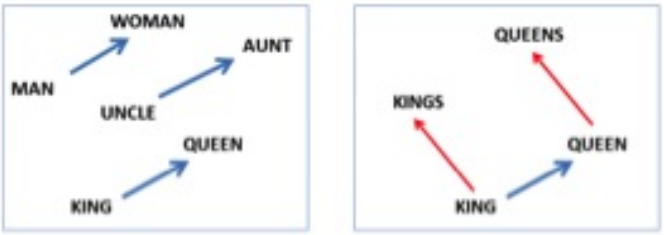
(19) a. The company said it isn't aware of *any* takeover interest.
b. The.DT company.NN said.VBD it.PRP is.VBZ nt.RB aware.JJ of.IN any.DT takeover.NN interest.NN

After each sentence is tagged it is then parsed to derive each sentence's syntax tree using a statistical parser from the NLTK toolkit trained on the Penn Treebank, the parser's output for example (19) is shown below.



The last part of the processing stage is to extract word vector representations for each word in the corpus. The use of these types of representations dates back to the 1980s Williams & Hinton (1986), and has subsequently been used to group similar words in many computational linguistic tasks. The idea behind word vector representations stems from a Distributional Semantics approach and there is an increasing body of research supporting the argument that distributional information plays a larger role in language processing than previously thought Saffran et al. (1996); Landauer & Dumais (1997); Redington et al. (1998); McDonald (2000). Semantic vector space models of language represent each word with a real-valued vector, which is learned using context and co-occurrence values. In the model, a word is located in space according to the degree to which it co-occurs with each of the others in space. Therefore, two words that tend to occur in similar linguistic contexts will be positioned closer in semantic space. Co-occurrence frequency information is extracted from a large corpus of natural language that acts as a record of language experience.

Researchers have noted many interesting properties of word vectors suggesting that they are able to capture semantic regularities. Interestingly, words with similar meanings appear together in space, even across languages; for example, given the vector for *frog* the model is able to predict that *rana*[5] is semantically similar Pennington et al. (2014). In addition, using vector arithmetic the model can capture semantic relations between words. As the diagram below shows, given the angle and distance between the man/woman pair, a system can determine that the king/queen relation is semantically similar. Similarly the model is able to capture plurality as is shown with king/kings and queen/queens Mikolov et al. (2013).



(Mikolov et al., NAACL HLT, 2013)

Using a vector representation is extremely advantageous, since it can be applied to different domains and different languages without supervised learning or manual pattern construction. The motivation for using this approach in this work comes from a preliminary analysis of the word vectors generated for NPIs. After training a word vector model, given a word vector we are able to search for its closest neighbor. If we search for the closest vector to an NPI vector, what we find closest tends to be a vector for a DE operator. For example, if we look for the closest vector to *any* the model returns the vector for *without*.

NPI Vector	Closest Vector	Cosine Similarity
any	without	.62
anything	nothing	.70
ever	never	.70
yet	though	.58

Table 1. NPIs and Their Neighbors

We hypothesize that the word vector model is capturing the correlation between NPIs and DE operators. Additionally, we assume the NPI and

[5] *Rana* is the Spanish word for frog.

DE operator will represent the closest relationship in semantic space. We incorporate word vector representations into our approach by training a word vector model to generate word vector representations for each word in our corpus. We use the Mikolov et al.'s Word2vec implementation in Python using the Gensim package. We learn using the distributed Skip-gram neural network model because it has been found to give good word representations when the monolingual data is small Mikolov et al. (2013). These word vectors are then used as a key component in our scoring algorithm.

4.3. Linguistic Rules

In this stage, we borrow ideas from the semantic literature on negative polarity items and construct three rules that are integrated into our system:

(20) (Rule 1) The words any, any+[6] , ever, yet are negative polarity items
(Rule 2) If a negative polarity item appears in a sentence so must a DE operator
(Rule 3) Locality constraints:
   a. The NPI and DE operator must be clausemates
   b. The NPI must appear in the scope of the DE operator

Given the known relationship between NPIs and DE environments our system uses NPIs as clues that a DE operator may be present. Rule 1 provides our system with a list of NPIs. Given this list, our system can then search the corpus for sentences that include NPIs. We assume Ladusaw's Hypothesis 1979 to be true and capture it with Rule 2, which requires an NPI to appear with a DE operator. Since NPIs are subject to locality conditions Linebarger (1987); Kadmon & Landman (1993); Chierchia (2013) we use Rule 3a to help ensure that the NPI and DE operator in question appear in the same clause. We determine clausehood using the sentence's parsed tree structure. Rule 3b is also included to help eliminate contexts in which the NPI appears higher in the structure then the DE operator. Since NPIs at times will be licensed at a distance we explore multiple models both with and without Rule 3. In addition to the model with Rule 3, we have two other models that consider different contexts. The first is a naively constrained context where a sentence is narrowed down to a NPI context using punctuation by restricting

[6] any+ represents all the variations of any, such as anyone, anybody, anywhere, and so on.

the context to any words appearing to the left of the NPI up until a comma, semi-colon or end of sentence. The last context is the sentence in its entirety. After the linguistic rules are learned, we proceed to the final step of making predictions using our algorithms.

4.4. Scoring Algorithms

We compare our novel Word Similarity Algorithm to two other scoring algorithms: Danescu et al.'s 2009 Distillation Algorithm and Cheung and Penn's 2012 Certainty algorithm. Since these two systems currently represent the state of the art in this task.

4.4.1. Novel Word Similarity Alogrithm

Given the NPI contexts we established using linguistic rules, we use the extracted word vectors to make predictions. For each NPI context, we compare the NPI vector, $\vec{i}$, with each other word vector, $\vec{j}$ in its context. For each pair $(\vec{i},\vec{j})$, we calculate the cosine similarity between the two.

$$sim(i,j) = cos(\vec{i},\vec{j}) = \frac{\vec{i}\cdot\vec{j}}{\|\vec{i}\| * \|\vec{j}\|}$$

After computing the cosine similarity between each pair of vectors, we pick the closest word vector to the NPI vector as the possible DE candidate for that context. The similarity measure is then used as that candidate's score. Lastly, we rank all the candidates by their scores. We hypothesize that the DE operator and NPI vectors will represent the closest relationship in each context.

4.4.2. Distillation Alogrithm

In this approach Danescu et al. first rank DE operator candidates by a score. Given a corpus C and NPI contexts N, *tokens(C)* and *tokens(N)* represent respectively the number of words in C and N. Let y be a DE candidate, then *CountC(y)* represents the frequency of y in the corpus and *CountN(y)* represents the frequency of y in NPI contexts. The score given

to each candidate *y* is *S(y)*. The score captures the correlation between DE candidates and NPIs by highly ranking words that appear in NPI contexts more, as shown below.

$$S(y) = \frac{countN(y)/tokens(N)}{countC(y)/tokens(C)}$$

For example, given the DE candidate *but*, this approach counts its frequency in NPI contexts and in the corpus. Say the frequency values for *but* are as follows: *CountC(y)* = 100, *CountN(y)*= 20, *tokens(N)* = 10,000, and *tokens(C)* = 1,000,000. As a DE candidate *but* would then receive a score of:

$$S(but) = \frac{20/10,000}{100/1,000,000} = 20$$

An issue with this approach is that there exist words that are not DE operators that co-occur with NPIs, which will receive a high score. Danescu et al. refer to these interlopers as 'piggybackers' and deal with the issue by adding a distillation step. In this step, each NPI context is distributed a budget of a total score of 1 among its candidates. In other words, each score is normalized so that the total score for that NPI context sums to 1. In so doing, apparently plausible candidates that often appear in contexts with multiple candidates receive a low distilled score, despite a high initial score. As it is expected to find true DE candidates alone (without piggybackers) throughout NPI contexts, plausible candidates should still receive a high score. However, as piggybackers should only appear in contexts with multiple candidates, they will in turn receive a low distilled score.

4.4.3. Certainty Alogrithm

Cheung and Penn build off of Danescu et al.'s approach by beginning with the same initial score *S(y)*. Their heuristic method then adjusts each score by assigning credit to candidates within each NPI context. The strongest candidate in each NPI context, the one with the highest score, is represented by *M(p)*. Each NPI context is represented by *p*. If *y* is currently the strongest DE operator candidate in *p* then it is given credit equal to the proportional change to the highest score if *y* was removed. If *y* is not the highest scoring candidate it receives a credit of zero.

$$cred(p,y) = \begin{cases} \frac{M(p)-M(p/y)}{M(y)} & \text{if } S(y) = M(p) \\ 0 & \text{otherwise} \end{cases}$$

After credit is distributed across contexts, each score is updated so that the new score for each DE operator candidate represents the original score multiplied by its average credit received. All updated scores are then normalized across NPI contexts. So for example, given just one NPI context, this approach updates the original scores as follows:

NPI Context:

but in *any*

Original Scores:

S(but) = 20, *S(in)* = 5

Updated Scores:

S_U *(but)* = 20 x (20 - 5)/20 = 15

S_U *(in)* = 5 x 0 = 0

Since the DE candidate *but* represents the strongest DE candidate in the context it receives credit, according to the above formula, leading to its updated score. Contrarily, the other DE candidate *in* receives a credit of 0 leading to an updated score of 0. Similar to Danescu et al., Cheung and Penn use the credit heuristic to help distinguish DE operators from possible piggybackers. We test our novel algorithm, the Word Similarity algorithm, against the pre-existing scoring algorithms from Danescu et al. and Cheung and Penn. We report our findings in the next section.

4.5. Precision Results

We examined the top 150 items that were ranked by each system. Previous work Danescu-Niculescu-Mizil et al. (2009); Cheung & Penn (2012) provided public lists of annotated DE operators. We use these lists to help evaluate the output of all the systems. We implement our novel approach as well as re-implement Danescu et al.'s and Cheung and Penn's algorithms. We score our systems' outputs by first checking to see if the ranked DE candidates appear on one of the lists. If the item does not appear on any of the lists, we then hand annotate the item. The guideline criteria we use for judging items is taken from the examples given in the semantic literature

on NPIs. If an item creates a context in which an NPI can be licensed we mark that item as correct. Therefore, we are not strictly reporting precision for predicting DE operators but more so DE/negative environment triggers. We are allowing items like comparatives, the antecedent of the conditional, question words, and comparatives to be counted as correct. We choose a relaxed criterion because we do not want to exclude these important items. We believe these triggers are important to discover and provide useful information about the context in which they appear. We evaluate our performance using the same evaluation measure used in previous work; Danescu et al. and Cheung and Penn evaluated the top 150 operators outputted and judged the precision *k* at various values of *k* beginning at 10 and increasing by 10 up until 150. They then reported average precision. We report precision to allow for a direct comparison with previous work. Table 2 reports the average precision *k* for the top ranked 150 items of each approach.

	Rule-governed	Punctuation-based	Full Sentence
Word Similarity	11.1%	**55.3%**	9.0%
Distillation	14.6%	38.0%	10.7%
Certainty	18.4%	22.8%	16.4%

Table 2. Average Precision Reported

We report results for each of the scoring algorithms. With each algorithm we consider three different NPI contexts: (1) the rule-governed context (following Rule 3), (2) the punctuation-based context (any words to the left of the NPI up until the first punctuation mark), and (3) the entire sentence. We find using the punctuation-based context works best across all scoring approaches. In addition, we find our novel scoring algorithm, the Word Similarity algorithm to perform the highest across all models. The top 20 ranked DE operators from our model are given in Table 3.

Word Similarity with the Punctuation-based Context
nobody, how, only, hadn't, whether, too, isn't, nor none, without, difficult, neither, doubted, but, didn't, wasn't, if, hardly, no, lest

Table 3. System Output

Although we see lower results for our approach in the other two contexts we actually find our scoring approach to discover the highest number of DE operators even though the overall average precision reported is lower. Total

number of DE operators discovered for each scoring approach is shown in Table 4.

	Rule-governed	Punctuation-based	Full Sentence
Word Similarity	**17**	**49**	**20**
Distillation	13	37	15
Certainty	12	27	11

Table 4. Total DE Operators Discovered

The precision results for the entire sentence reflect the difficulty in using the entire sentence, as it represents the largest context with the most candidates to choose from; this is reflected in Table 5, which shows that the average word count for the full sentence is substantially larger then the other two contexts.

Rule-governed	Punctuation-based	Full Sentence
8.1	6.8	29

Table 5. Average Word Count per Context

However, differences in average word count alone can not account for the disparity in precision results for the rule-governed context. Since we find that the rule-governed context and punctuation-based context only differ on average by about 1-2 words, it is unlikely that this is main the cause of the low precision results. The rule-governed context was based on Rule 3 which was largely motivated by locality theories from the NPI literature Linebarger (1987); Kadmon & Landman (1993); Chierchia (2013); as this context represented the most informed way to restrict the context we hypothesized it would perform the best. Determining the rule-governed context relies on the implementation of Rule 3: (a) the NPI and DE operator must be clausemates and (b) the NPI must appear in the scope of the DE operator. There are many factors within Rule 3's implementation that could have ultimately affected the precision results. When we analyze the contexts generated using Rule 3, we find some cases where a given sentence's context is empty. Rule 3 relies on a generated parse structure, and if the parser is unable to return a parse for the sentence it could result in an empty context. Also if the NPI in the sentence is within an embedded clause, the NPI may represent the highest word in that clause. If so, the NPI would not be in the scope of a DE operator, and following Rule 3b the context would be empty. In addition to empty contexts, when checking Rule 3a it is possible that the clauses being

chosen do not contain the NPI and its licenser; this could be due to errors by the parser or by how we specified clauses in our code. It is likely that a combination of all the preceding reasons contributed to the low precision results of the rule-governed context. However, if implemented correctly, we do believe a rule-governed approach, motivated by locality, could perform well. In future work, we will focus on locality and intervention effects aiming to refine how we determine the appropriate context to consider.

In summary, our findings suggest that negative polarity items can be very useful clues to automatically learn DE operators from text. Given a list of NPIs and linguistic rules, which capture the relationship between NPIs and DE operators, a scoring algorithm can successfully predict DE operators. This work further supports the important relationship that exists between NPIs and DE operators and highlights how it may be leveraged to aid an inference system. Also, to our knowledge this is the first attempt to use a word vector representation to learn words' monotonicity information and we find our novel scoring approach outperforms all existing approaches in this task.

5. Concluding Remarks

This work further supports the important relationship that exists between NPIs and DE operators and highlights how it may be leveraged to aid an inference system. To our knowledge this represents the first attempt to use a word vector representation to learn words' monotonicity information and we find our novel scoring approach outperforms all existing approaches in this task. We show that by using linguistic rules we can capture the relationship between NPIs and DE operators, which can then be used by a scoring algorithm to successfully predict DE operators from text.

REFERENCES

Baker, C Lee. 1970. Double negatives. *Linguistic inquiry* 1(2). 169–186.

Bird, Steven. 2006. Nltk: the natural language toolkit. In *Proceedings of the coling/acl on interactive presentation sessions*, 69–72. Association for Computational Linguistics.

Cheung, Jackie CK & Gerald Penn. 2012. Unsupervised detection of downward-entailing operators by maximizing classification certainty. In *Proceedings of the 13th conference of the european chapter of the association for computational linguistics*, 696–705. Association for Computational Linguistics.

Chierchia, Gennaro. 2013. *Logic in grammar: Polarity, free choice, and intervention*, vol. 2. Oxford University Press.

Danescu-Niculescu-Mizil, Cristian, Lillian Lee & 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*, 137–145. Association for Computational Linguistics.

Fauconnier, Gilles. 1975. Polarity and the scale principle. *Proceedings of Chicago* .

Kadmon, Nirit & Fred Landman. 1993. Any. *Linguistics and philosophy* 16(4). 353–422.

Klima, Edward S. 1964. *Negation in english*. na.

Krifka, Manfred. 1995. The semantics and pragmatics of polarity items. *Linguistic analysis* 25(3-4). 209–257.

Ladusaw, William. 1979. Negative polarity items as inherent scope relations. *Unpublished Ph. D. Dissertation, University of Texas at Austin* .

Landauer, Thomas K & Susan T Dumais. 1997. A solution to plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological review* 104(2). 211.

Linebarger, Marcia C. 1987. Negative polarity and grammatical representation. *Linguistics and philosophy* 10(3). 325–387.

MacCartney, Bill. 2009. *Natural language inference*. Ph.D. thesis, Citeseer.

MacCartney, Bill & Christopher D Manning. 2009. An extended model of natural logic. In *Proceedings of the eighth international conference on computational semantics*, 140–156. Association for Computational Linguistics.

McDonald, Scott. 2000. Environmental determinants of lexical processing effort .

Mikolov, Tomas, Wen-tau Yih & Geoffrey Zweig. 2013. Linguistic regularities in continuous space word representations. In *Hlt-naacl*, 746–751.

Pennington, Jeffrey, Richard Socher & Christopher D Manning. 2014. Glove: Global vectors for word representation. *Proceedings of the Empiricial Methods in Natural Language Processing (EMNLP 2014)* 12. 1532–1543.

Progovac, Ljiljana. 1988. *A binding approach to polarity sensitivity*.

Progovac, Ljiljana. 1993. Negative polarity: Entailment and binding. *Linguistics and Philosophy* 16(2). 149–180.

Redington, Martin, Nick Chater & Steven Finch. 1998. Distributional information: A powerful cue for acquiring syntactic categories. *Cognitive Science* 22(4). 425–469.

Saffran, Jenny R, Richard N Aslin & Elissa L Newport. 1996. Statistical learning by 8-month-old infants. *Science* 274(5294). 1926–1928.

Williams, DE Rumelhart GE Hinton RJ & GE Hinton. 1986. Learning representations by back-propagating errors. *Nature* 323–533.

Zwarts, Frans. 1998. Three types of polarity. In *Plurality and quantification*, 177–238. Springer.