Learning Antonyms using Paraphrase Database

Sneha Rajana
University of Pennsylvania
srajana@seas.upenn.edu

Chris Callison-Burch
University of Pennsylvania
ccb@cis.upenn.edu

Abstract

Learning antonyms has widespread applications in natural language processing. This study focuses on antonymy detection using the Paraphrase Database (PPDB). We present a new automatic method of generating antonym pairs using paraphrase pairs obtained using PPDB. We describe a method for discovering these word pairs using a subset of PPDB phrases that indicate an antonym relationship between the target word pairs. This study can be extended to assign antonym relations to entries in PPDB.

1 Introduction

One of the main tasks of language understanding is the ability to identify expressions that have opposite meanings. Recent research on antonym detection has been focused on pattern-based techniques to derive words occurring in the same sentence that are antonyms of each other. A manually created list of antonyms will have limited coverage and may not include all possible antonyms for a given word in different senses. The study aims to handle lexical ambiguity by considering the issue of polysemy and not making the incorrect assumption of only a single sense per word. For example, the opposite of the word **hot** is **cold** in the sentence, *The water was **hot***. However, the word *cold* is not always the opposite of *hot* like in the sentence, *This was a **hot** topic*. Replacing the word hot with cold in this sentence will not give an opposite meaning. So it is important to try and learn antonyms for senses of a word instead of words themselves. WordNet was the resource used to obtain these different senses. WordNet contains synsets or a set of synonyms for each word.

Paraphrasing is the task of identifying different expressions of the same information. If two phrases have opposite meanings then they cannot be paraphrases of each other. Paraphrase resources have a wide applicability to Natural Language Understanding tasks. Different methods of paraphrasing learn paraphrases that are not only meaning equivalent but also learn contradictory pairs. Multilingual paraphrasing methods (Ganitkevitch et al. 2013) also learn hypernym/hyponym pairs due to variation in the discourse structure of translations and unrelated pairs due to to misalignments or polysemy in the foreign language (Pavlick et al. 2015). The study began with examining the relationships between phrase pairs in The Paraphrase Database (PPDB), a resource with 169 million paraphrase transformation rules (Ganitkevitch et al. 2013). The phrase pairs in PPDB not only have a mapping between approximately equivalent phrases (thrown into jail/imprisoned) but also represent a variety of relations, including directed entailment (little girl/girl) and exclusion (nobody/-someone). This study will focus on paraphrases from PPDB that can help in detecting antonyms and relate them to those obtained in the seed set created using WordNet.

2 Related Work

2.1 Degree of Antonymy

Dorr et al. (2008) explored the relationship between what humans consider antonymous and how antonymy manifests itself in utterances. The study talks about 3 degrees of antonymy: strongly antonymous, semantically contrasting and not antonymous. The higher the degree of antonymy between target word pair, the higher the tendency to be considered antonym pairs by native human speakers.

Automatically determining the degree of antonymy between words can be helpful in detecting and generating paraphrases, detecting contradictions, detecting humor (satire and jokes

tend to have contradictions and oxymorons) and in finding words which are semantically contrasting to a target word (probably to filter them out).

Antonymy, Synonymy, Hyponymy etc. are some lexical-semantic relations that apply to two lexical units - A combination of surface form and word sense. The study also explores the paradoxes of antonymy. Why are some pairs better antonyms? (Eg. large-small vs. large-little). Are semantic closeness and antonymy opposites? If two words are associated via synonymy, hyponymy-hypernymy or troponymy relations, they are considered to be semantically close or semantically related. Words that are semantically similar are also semantically related (Eg. plane-glider, doctor-surgeon) but not the other way round (Eg. plane-sky, surgeon-scalpel). Antonymous concepts are semantically related but not semantically similar. The co-occurrence hypothesis states that antonyms occur together in a sentence more often than chance (Charles and Miller 1989). But this is also true for hypernyms, holonyms, meronyms and near-synonyms. Thus, separating antonyms from them has proven to be difficult. Strong co-occurrence is not a sufficient condition for detecting antonyms, but it is useful. The distributional hypothesis of closeness states that words that occur in similar contexts tend to be semantically close.

The paper states that manually-created lexicons have limited coverage and do not include most semantically contrasting word pairs. They presented a new automatic and empirical measure of antonymy that combines corpus statistics with the structure of a published thesaurus. Their approach was as follows. The adjacency heuristic is that adjacent categories in most published thesauri are considered to be contrasting categories. Given a target word pair, the algorithm determined whether they are antonymous or not, and if they are, whether they have a high, medium, or low degree of antonymy. If the target words belong to the same thesaurus paragraphs as any of the seed antonyms linking the two contrasting categories, then the words have a high degree of antonymy. If the target words do not belong to the same thesaurus paragraphs as a seed antonym pair, but occur in contrasting categories, they have a low degree of antonymy if the word-paors have a lower tendency to co-occur and a medium degree of antonymy if the word-pairs have a higher tendency to co-occur. This algorithm when evaluated on a set of closest-opposite questions, obtained a precision of over 80%.

2.2 Semantic Taxonomy Induction

Snow et al. (2006) proposed a novel algorithm for inducing semantic taxonomies. Previous algorithms for taxonomy induction focused on independent classifiers for discovering single relationships based on hand-constructed or automatically generated textual patterns. Whereas their algorithm incorporates evidence from multiple classifiers over heterogenous relationships to optimize the entire structure of the taxonomy. Though wide variety of relationship-specific classifiers like the pattern-based classifiers have achieved some degree of success, they frequently lack the global knowledge necessary to integrate their predictions into a complex taxonomy with multiple relations.

The paper mentions that previous algorithms focused only on inferring small taxonomies over a single relation, or has used evidence for multiple relations independently from one another. Another major shortfall was the inability to handle lexical ambiguity as these previous approaches sidestepped the issue of polysemy by making the assumption of only a single sense per word and inferring taxonomies explicitly over words and not senses. The authors approach simultaneously provides a solution to the problems of jointly considering evidence about multiple relationships as well as lexical ambiguity within a single probabilistic framework. Within their model, they define the goal of taxonomy induction to be to find the taxonomy that maximizes the conditional probability of their observations given the relationships of the taxonomy.

They have also extended their model to manage Lexical Ambiguity. If the objects in the taxonomy are word senses, they extended their model to allow for a many-to-many mapping (eg. word-to-sense mapping) between the the sets of objects. They have presented an algorithm for inducing semantic taxonomies which attempts to globally optimize the entire structure of the taxonomy. The models ability to integrate heterogeneous evidence from different classifiers offers a solution to the key problem of choosing the correct word sense to which to attach a new relation (hypernym, hyponym, antonym etc).

2.3 Pattern based techniques

Marti A Hearst (1992) described a method for the automatic acquisition of the hyponymy lexical relation from unrestricted text. The motivation behind this approach was based on avoidance of the need for pre-encoded knowledge and applicability across a wide range of text. The study identified a set of lexico-syntactic patterns that are easily recognizable, that occur frequently and across text genre boundaries, and that indisputably indicated the lexical relation of interest. They described a method for discovering these patterns and suggested that other lexical relations will also be acquirable in this way.

Hearst patterns can be used to search for specific lexical relations that are expressed in well-known ways instead of interpreting everything in the text in great detail. Surprisingly useful information can be found with only a very simple understanding of a text.

An example of such a pattern described in this paper is
$such\ NP\ as\ \{NP, \} * \{(or|and)\}\ NP$
... works by such authors as Herrick, Goldsmith, and Shakespeare.
This implies hyponym("author", "Herrick"),
hyponym("author", "Goldsmith"),
hyponym("author", "Shakespeare")

Pattern based techniques can also be extended to obtain antonyms from simple text instead of hyponyms. Some of the possible patterns to derive antonyms are as follows:
a) *NP rather than NP*
... He was sad rather than happy.
This implies antonym("sad", "happy")
b) *NP and not* {*} *NP*
... I play a lot of summer sports and not many winter sports.
This implies antonym("summer", "winter")
c) *NP* {*}*, but NP* {*}
... She was optimistic about attending college, but pessimistic about paying for it.
This implies antonym("optimistic", "pessimistic")

2.4 Natural logic

The task of textual inference involves automatically determining whether a natural-language hypothesis can be inferred from a given premise. The NatLog system (MacCartney et al., 2007) which popularized natural logic for Rich Textual Entailment (RTE) tasks presented the first use of a computational model of natural logic - a system of logical inference operating over natural language for textual inference. Most current RTE systems achieve robustness by sacrificing semantic precision and those systems that rely on first-order logic and theorem proving are precise but excessively brittle. Their system found a low-cost edit sequence which transformed the premise into the hypothesis and learned to classify entailment relations across atomic edits. This system uses natural language as a representation and performs natural language inference using a structured algebra model. However, important kinds of inference like temporal reasoning, causal reasoning, paraphrasing and relation extraction are not addresses by natural logic.

2.5 Adding semantics to data-driven paraphrasing

In addition to the equivalence relation between phrase pairs in PPDB, Pavlick et al. (2015) show the existence of other interpretable semantic relations like entailment and exclusion between the phrase pairs and the distribution of these relations in different sizes of PPDB. They also show that some of these pairs have other relations or are unrelated. Automatically assigning semantic entailment relations to entries in PPDB was done with a high accuracy and it was done using features derived from the NatLog system and semantc taxonomy induction. They also developed a statistical model to predict these relations automatically and demonstrated improvements to a proof based RTE system.

They focused on paraphrase pairs from PPDB that occurred in RTE data and used monolingual and bilingual features for this task. Sentence pairs are labeled using a 3-way entailment classification which includes entailment, contradiction and neutral. They combined the seven basic entailment relations described in the NatLog system (MacCartney et al. 2009) into five classes that were distinguished using a logistic regression classifier and computed a variety of lexical features and WordNet features.

The monolingual features were based on observed usage in the Annotated Gigaword corpus and included path and distributional features. We attempt to generate similar paths in our experiments (Section 4.4). Path features used lexico-

syntactic patterns to mine taxonomic relations and distributional features mine inference rules from text by building dependency context vectors for each word in the data and computing symmetric and asymmetric similarity measures for identifying entailment paraphrases. Bilingual features are based on translational probabilities observed in bilingual parallel corpora. By classifying paraphrase entries with entailment relations, they provide them with an interpretable semantics.

3 Resources

3.1 WordNet

WordNet is a large lexical database of English. Nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept. Synsets are interlinked by means of conceptual-semantic and lexical relations. Adjectives are organized in terms of antonymy. WordNet encodes antonymy as a lexical relationship a relation between two words and not concepts (Gross et al., 1989). WordNet antonym pairs comprise of "direct antonyms" (wet/dry, young/old) and "indirect antonyms" (dry/parched). Individual words across synsets are marked as direct antonyms. These pairs reflect the strong semantic contract of their members. Each of these polar adjectives in turn is linked to a number of semantically similar ones. Semantically similar adjectives are indirect antonyms of the contral member of the opposite pole. Even after including the indirect antonyms, WordNets coverage is limited. WordNet or any other manually-created repository of antonyms does not encode the degree of antonymy between words (Saif Mohammad et al., 2008). Neverthess, we use WordNet to create a seed set of antonym pairs using a cross product of the synonyms and antonyms of words from WordNet and this was used as a gold standard in our experiments (Section 4 below).

3.2 The Paraphrase Database (PPDB)

PPDB is an automatically extracted database containing millions of paraphrases in 16 different languages. The goal of PPBD is to improve language processing by making systems more robust to language variability and unseen words. It is currently the largest available collection of paraphrases. PPDB contains over 150 million paraphrase rules covering three paraphrase types lexical (single word), phrasal (multiword), and syntactic restructuring rules. We focus on lexical and phrasal paraphrases upto two words in length, of which there are over half a million rules. The paraphrase database is released in six sizes (S, M, L, XL, XXL and XXXL) divided based on precision and recall scores. We have used the english PPDB, version 2.0 and XXXL size for all our experiments described later.

Direct antonyms	Indirect antonyms
clean/dirty	clean/foul
rise/fall	clean/grime
rise/set	clean/muddy
sleep/wake	rise/downfall
asleep/awake	rise/spill
above/below	sleep/rouse
	sleeping/awake
	above/under

Table 1: Direct antonyms and Indirect antonyms from WordNet

4 Our Approach

This is the first approach to have used PPDB to obtain antonyms and includes the following steps.

4.1 Creation of a seed set of antonym pairs

We first created a seed set of antonyms generated using WordNet. As mentioned in Section 2, antonyms derived from WordNet are direct antonyms. We extended this list to include indirect antonym word pairs that were derived from a cross product of the synonyms of each word in the antonym pair and its direct antonym. Table 1 shows examples of direct antonym pairs and indirect antonym pairs. This seed set of antonym pairs generated from WordNet was used like a gold standard for further experiments. The pseudocode for extracting antonyms from WordNet is shown below.

```
#Algorithm: Extracting WordNet antonyms
#Input: None
#Output: Set of all antonym pairs

#packages required
from nltk.corpus import wordnet as wn

for i in wn.all_synsets():
   for j in lemmas(i):
      if j has antonyms:
            #direct antonyms
            word1 = j
            word2 = antonym(j)
            antonyms.add((word1,word2))
            #indirect antonyms
```

```
            #antonym(synonyms(word1),word2)
            word1_synonyms =
                wn.synsets(word1)
            for s in word1_synonyms:
               antonyms.add((s,word2))
            #antonym(synonyms(word2),word1)
            word2_synonyms =
                wn.synsets(word2)
            for s in word2_synonyms:
               antonyms.add((s,word1))

print(antonyms)
```

4.2 Selection of paraphrases

In this study we focused on two types of paraphrase pairs from PPDB. The first type is of the form $< p1, p2 >$ where $p1$ contains the word 'not' in it and $p2$ is a single word or lexical phrase (Example: not happy/happy). The second type is of the form $< p1, p2 >$ where either $p1$ or $p2$ contains a negating prefix (Example: unhappy, demotivate, inaccurate etc.). We consider negating prefixes of the form 'in', 'un', 'anti' and 'de'. Table 2 shows examples of pairs in PPDB falling into these two categories. These two types of paraphrase pairs were chosen as we believed these to be most indicative of an antonym relationship between the target words. The pseudocodes for extracting PPDB paraphrases of the two types are shown below.

```
#Algorithm: Extracting PPDB paraphrase
    type 1
#Input: PPDB 2.0 XXXL phrasal file
#Output: Paraphrase pairs of type 1

#packages required
from nltk import word_tokenize

for line in ppdb_file:
   text = line.split("|||")
   source = text[1]
   source1 = word_tokenize(source)
   target = text[2]
   target1 = word_tokenize(target)
   if 'not ' in source and
       len(source1)==2 and
       len(target1)==1:
          phrases.add((source,target))
   if 'not ' in target and
       len(target1)==2 and
       len(source1)==1:
          phrases.add((target,source))

print(phrases)
```

```
#Algorithm: Extracting PPDB paraphrase
    type 2
#Input: PPDB 2.0 XXXL phrasal file
#Output: Paraphrase pairs of type 2

#packages required
from nltk import word_tokenize

for line in ppdb_file:
   text = line.split("|||")
   source = text[1]
   source1 = word_tokenize(source)
   target = text[2]
   target1 = word_tokenize(target)
   if len(source1) == 1:
      if(source starts with 'in' or
          'un' or 'anti' or 'de'):
         phrases.add((source, target))
   if len(target1) == 1:
      if(target starts with 'in' or
          'un' or 'anti' or 'de'):
         phrases.add((target, source))

print(phrases)
```

4.3 Generating antonym pairs

Paraphrase type 1: We used PPDB to retrieve paraphrase mappings that are useful in learning antonyms (Eg. not happy/happy, unhappy/happy, deactivate/activate etc.). For each word-pair obtained using PPDB, we found synonyms for both words first using WordNet synsets and then with PPDB. We then proceeded to map all synonyms of word1 in the antonym pair to word2 of the antonym pair and all synonyms of word2 in the antonym pair to word1 of the antonym pair. We used WordNet seed set as a gold standard to compare the newly generated antonym list using PPDB.

Paraphrase type 2: A list of words obtained from PPDB containing negating prefixes were compiled and a morphological analysis was performed on these words to remove the negation markers. Antonyms were then generated from the base non-negated forms and all paraphrases of the negated form. Table 3 shows examples of antonym pairs derived from the two type of paraphrases from PPDB. The pseudocodes for generating antonyms from PPDB paraphrases of the two types are shown below.

```
#Algorithm: Generating antonym pairs of
    the form <x,y>
#Input: File containing PPDB
    paraphrases of the form <not x,y>
#Output: Antonym pairs of the form <x,y>

for line in input_file:
   line = line.replace('not ', '')
   x,y = line.split('\t')
   antonyms.add((x,y))

print(antonyms)
```

Category 1	Category 2
not satisfactory/unsatisfactory	inadequate/ quite inadequate
not appropriate/inappropriate	unacceptable/ wholly unacceptable
not insignificant/significant	intolerable/ quite intolerable
not acceptable/objectionable	anti-discrimination/ non discrimination
not perfect/imperfect	incredible/absolutely incredible
not true/untrue	inappropriate/totally inappropriate
not infrequently/frequently	unusual/rather unusual
not compatible/incompatible	deactivate/turned off
not correct/incorrect	deforestation/destruction
not identical/different	unjustifiable/completely unreasonable

Table 2: PPDB paraphrases of 2 categories: Category 1 (source phrase contains the word 'not') and Category 2 (source phrase has a negating prefix)

Paraphrase type 1	Paraphrase type 2
sufficient/insufficient	dependence/independent
adequate/not enough	significant/negligible
happy/unhappy	effective/no good
happy/enraged	conventional/irregular
alright/not good	usual/pretty strange
fair/unacceptable	considerable/insignificant
necessary/not useful	frequent/quite rare
available/run low	resolved/unresolved
good/cold	gradation/deterioration
stop/hold on	believable/unreal

Table 3: Antonyms derived from PPDB paraphrases of two types

```
#Algorithm: Generating antonym pairs of
    the form <synonyms(x),y> and
    <x,synonyms(y)> using WordNet to
    extract synonyms
#Input: File containing antonym pairs
    of the form <x,y>
#Output: Antonym pairs of the form
    <synonyms(x),y> and <x,synonyms(y)

for line in input_file:
   x,y = line.split('\t')
   x_synonyms = get_wordnet_synsets(x)
   for s in x_synonyms:
      antonyms.add((s,y))
   y_synonyms = get_wordnet_synsets(y)
   for s in y_synonyms:
      antonyms.add((s,x))

print(antonyms)
```

```
#Algorithm: Generating antonym pairs of
    the form <paraphrases(x),y> and
    <x,paraphrases(y)> using PPDB to
    extract paraphrases
#Input: File containing antonym pairs
    of the form <x,y>
#Output: Antonym pairs of the form
    <paraphrases(x),y> and
    <x,paraphrases(y)>

for line in input_file:
   x,y = line.split('\t')
   x_synonyms =
       get_paraphrases_from_ppdb(x)
   for s in x_synonyms:
      antonyms.add((s,y))
   y_synonyms =
       get_paraphrases_from_ppdb(y)
   for s in y_synonyms:
      antonyms.add((s,x))

print(antonyms)
```

```
#Algorithm: Performing morphological
    analysis on input list of words to
    remove the negating prefix
#Input: List of words containing a
    negating prefix from PPDB
#Output: Mapping between the word with
    negating prefix and its non-negated
    base form

#package required - MORSEL

#ppdb_morph_file: File containing words
    to be morphologically analysed with
    their frequencies
for word in input_list:
   ppdb_morph_file.write(word+
   get_frequency_from_ppdb(word))

#Run the following command on the PPDB
    corpus wordlist, write the analysis
    to `out.txt`, write the log to
    `log.txt`, and use the conservative
    parameter set do the following:
`java -jar morsel.jar <ppdb_morph_file>
    out.txt log.txt
    params/conservative.txt`

#out.txt contains a mapping between the
    word and its non-negated base form
```

```
#Algorithm: Generating antonym pairs of
```

```
    the form
    <non_negated(x),paraphrases(negated(x))>
#Input: File containing a mapping
    between the word with negating
    prefix and its non-negated base form
#Output: Antonym pairs of the form
    <non_negated(x),paraphrases(negated(x))>

morph_words =
    get_list_words_with_negated_prefix()

morph_mapping =
    get_mapping_from_inputfile()

for phrase_pair in ppdb_phrases:
   #source = source_phrase in
       phrase_pair
   #target = target_phrase in
       phrase_pair
   if(source in morph_words):
      antonyms.add((morph_mapping[source],
          target))

print(antonyms)
```

4.4 Hearst/Snow patterns for antonym pairs

In section 2.3 we described pattern based techniques to derive lexical relations from unrestricted text. Snow et al. (2004) and Hearst (1992) used easily recognizable and frequently occurring lexico-syntactic patterns to automatically derive lexical relations (hypernymy, hyponymy etc.) between noun pairs. Pavlick et al. (2015) used monolingual path features to learn new patterns to differentiate between more subtle relations like equivalence, negation and entailment. We used similar path features generated from dependency parses and Linguistic Data Consortium (LDC) data to learn new patterns that are indicative of an antonym relationship. Table 4 gives examples of some of these paths.

5 Results and Analysis

The number of unique antonym/paraphrase pairs generated in each step above is shown in Table 5. From the results shown in Table 5, we can see that the coverage of WordNet is limited. PPDB has a much larger vocabulary than WordNet (Approximately 10 times more) and so using PPDB along with WordNet for the antonym learning task will be more beneficial. As expected the coverage increased by a large bound when using PPDB along with WordNet to generate antonym pairs.

Comparing entries 2-b-i and 2-b-ii in Table 6, we see that around 27 million word pairs were generated when the WordNet was used to generate synonyms of a word vs. around one million word pairs generated when PPDB was used to generate synonyms/lexical paraphrases of the target word. This is because WordNet synsets contain a larger list of synonyms for a given word than the number of lexical paraphrases for a given word in PPDB. However, the quality of the final antonym list (Human evaluation) generated using WordNet synsets to find synonyms for a given word was better than the antonym list generated using PPDB lexical paraphrases to find synonyms for a given word. PPDB paraphrase rules are extracted by pivoting through foreign translations. So the data in PPDB is noisier possibly due to polysemy in the foreign language. Also, synonyms of a word generated using WordNet synsets are closer to those found in a thesaurus compared to those generated using PPDB lexical paraphrases as the paraphrase rules might consider other unrelated external factors while generating paraphrases and is not dependent on the given word or its senses alone.

Examples of antonym pairs generated using the two types of PPDB paraphrases are listed in table 3. Both the antonym lists contained good results and covered different senses of a given word For example, consider the antonym pair good/-cold. This pair is probably indicative of characteristics of a person - 'he was good' vs. 'he was cold'. The list also contained the pair hot/cold. This pair contains the same word 'cold' but in a different sense and this antonym pair is a direct antonym pair - 'hot water' vs. 'cold water'. Unsurprisingly, the quality of the antonym pairs generated using PPDB paraphrase type 2 that contain words with a negating prefix was better than those generated using PPDB paraphrase type 1 that contains phrases containing the word 'not'. For example, consider the paraphrases 'not acceptable/objectionable' under category 1 in table 2 and 'unacceptable/wholly unacceptable' under category 2 in table 2. These phrases will return the antonym pairs 'acceptable/objectionable' and 'acceptable/wholly unacceptable' respectively using the method described in section 4.3. Pairing the word acceptable with paraphrases of unacceptable will indicate a better antonym relationship than pairing acceptable with paraphrases of not acceptable as unacceptable/acceptable indicates a stronger and more obvious antonym relationship than not acceptable/acceptable.

Pattern	Example sentence
compared with X, Y	*compared with the older generation, the new generation*
to X or to Y	*to fight or to surrender*
X rather than Y	*maximizing it rather than minimizing it*
either X or Y	*either low or high doses*
neither X nor Y	*neither women nor men*

Table 4: Paths indicative of an antonym relationship between phrases

Sl no.	Method	No. of unique pairs
1	Antonyms from WordNet	18,306
a	Direct antonyms	3,337
b	Indirect antonyms	14,969
2	PPDB phrases of the form <Not X, Y>	7,878
a	Antonyms of the form <X,Y> from PPDB phrase <Not X,Y>	7,878
b	Antonyms of the form <X, synonyms(Y)> and <synonyms(X), Y>	
i	Synonyms generated using WordNet	113,481
ii	Synonyms generated using PPDB	4,582,854
3	PPDB Phrases of the form <X with negating prefix, Y>	183,159
a	Antonyms of the form <Base non-negated form of X,Y> from PPDB Phrase: <X with negating prefix, Y>	183,159
b	Lexical words in PPDB containing a negating prefix	1,966
c	Antonyms of the form <Non-negated form of X,Paraphrases(negated form of X)>	19,732

Table 5: Number of unique pairs generated in each step involved in generating the resulting antonym list

6 Evaluation

We performed a manual evaluation of the quality of the antonyms that were extracted. A combined list of randomly selected antonyms from all of the methods listed above had about 44% accuracy. We also evaluated the percentage of antonyms yielded by each method. Table 6 shows the accuracy of the baseline and the individual methods. WordNet antonyms expanded with WordNet synsets achieved the highest accuracy of about 70%. Antonyms of the form <X,Y> from PPDB phrase <Not X,Y> had the second best accuracy followed by PPDB antonym pairs from the morphological negation pattern. The errors in these methods mostly consisted of phrases and words that do not have a contradictory meaning after the removal of the negation pattern. For example, the antonym pair *till/until* was derived from the PPDB phrase, *not till/until*. But 'till' and 'until' have an equivalent meaning and are not actually antonyms of each other. Other non-antonyms generated from the above methods can be classified into unrelated pairs (background/figure), paraphrases or pairs that have an equivalent meaning (admissible/permissible), words that belong to a category (Africa/Asia), pairs that have an entailment relation (valid/equally valid) and pairs that are related but not with an antonym relationship (habitants/general public). Table 7 gives some examples of categories of non-antonyms. PPDB pairs directly classified as 'Exclusion' contained a lot of pairs that belong to a categories like colors (blue/red), countries (India/Indonesia) etc. Unrelated pairs and pairs with entailment and other relationships were present in the antonym list derived from PPDB paraphrases.

7 Future work

Although the quality of the antonym pairs generated from the experiments described above was good in general, the lists contain a lot of unrelated pairs of words and phrases. A mechanism to automatically score the resulting antonym pairs and eliminating the low-score antonym pairs will improve the quality of the antonym list. We focused on only two types of phrase pairs from PPDB for our experiments. We can extend this by identifying other types of paraphrase pairs from PPDB that can help us in generating more antonyms and attempt to derive a larger and more accurate

Method	% of antonyms
Combination	51.65
Antonyms from WordNet (including Indirect antonyms)	70.00
PPDB pairs classified as 'Exclusion'	44.00
Antonyms of the form <X,Y> from PPDB phrase <Not X,Y>	52.22
PPDB antonym pairs expanded using WordNet synsets	45.00
PPDB antonym pairs expanded using PPDB paraphrases	24.00
PPDB antonym pairs from the morphological negation patterns	42.00

Table 6: Percentage of antonyms yielded by each method

Unrelated	Paraphrases	Categories	Entailment	Other relation
much/worthless	correct/that's right	Japan/Korea	investing/increased investment	twinkle/dark
disability/present	simply/merely	black/red	efficiency/operational efficiency	naw/not gonna
equality/gap	till/until	Jan/Feb	valid/equally valid	access/available
long/rare	no man/nobody	blonde/brunette	significant/statistically significant	valuable/premium
allow/right	right/alright	Africa/Asia		don't do/not right

Table 7: Examples of different types of non-antonyms

antonym list from those pairs.

Future work could include incorporating these results into PPDB by adding the feature of antonym phrases to the paraphrase database. Given a search query, in addition to returning paraphrases of the given search query, PPDB could also return antonym phrases (phrases that have an opposite meaning to the given phrase).

8 Conclusion

Previous approaches for antonym detection were focused on pattern-based techniques to derive words occurring in the same sentence that are antonyms of each other. But it is impossible to derive such patterns for all possible senses of a word. Those studies that have actually handled lexical ambiguity were focused on hyponym-hypernym detection but not on deriving antonym relationships. Our method not only automatically derived antonym relationships between single word/lexical phrases and covered different senses of that word but also derived antonym relationships between multi-word phrases. We also automatically generated many more Hearst/Snow patterns to detect paths between antonym pairs compared to the previous approaches for the same task.

Acknowledgments

This work was done as part of an Independent study with Prof. Chris Callison-Burch whose invaluable advice and guidance led to the successful completion of this project. We thank the Department of Computer and Information Science at the University of Pennsylvania for providing us with all the required resources for carrying out this study.

References

Derek Gross, Ute Fischer, and George A. Miller. 1989. *Antonymy and the representation of adjectival meanings*. Memory and Language, 28(1):92106.

Juri Ganitkevitch, Chris Callison-Burch. 2014. *The Multilingual Paraphrase Database*. LREC-2014.

Marti A. Hearst. 1992. *Automatic Acquisition of Hyponyms from Large Text Corpora*. Association for Computational Linguistics.

Bill MacCartney and Christopher D. Manning 2007. *Natural Logic for Textual Inference* In RTE '07 Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing, pages 193-200.

Saif Mohammad, Bonnie Dorr and Graeme Hirst. 2008. *Computing Word-Pair Antonymy* In Proceedings of the Conference on Empirical Methods in Natural Language Processing, pages 982-991, Honolulu, October 2008.

Ellie Pavlick, Johan Bos, Malvina Nissim, Charley Beller, Ben Van Durme, Chris Callison-Burch. 2015. *Adding Semantics to Data-Driven Paraphrasing* Association for Computational Linguistics 2015.

Rion Snow, Daniel Jurafsky and Andrew Y. Ng. 2004. *Learning syntactic patterns for automatic hypernym discovery*. In NIPS, volume 17, pages 12971304.

Rion Snow, Daniel Jurafsky and Andrew Y. Ng. 2006. *Semantic Taxonomy Induction from Heterogenous Evidence*. Association for Computational Linguistics 2006.