*Computational Intelligence,* Volume 29, Number 3, 2013

# A BOOTSTRAPPING METHOD FOR EXTRACTING PARAPHRASES OF EMOTION EXPRESSIONS FROM TEXTS

FAZEL KESHTKAR AND DIANA INKPEN

*School of Electrical Engineering and Computer Science, University of Ottawa, Ottawa, Ontario, Canada*

Because paraphrasing is one of the crucial tasks in natural language understanding and generation, this paper introduces a novel technique to extract paraphrases for emotion terms, from nonparallel corpora. We present a bootstrapping technique for identifying paraphrases, starting with a small number of seeds. WordNet Affect emotion words are used as seeds. The bootstrapping approach learns extraction patterns for six classes of emotions. We use annotated blogs and other data sets as texts from which to extract paraphrases, based on the highest scoring extraction patterns. The results include lexical and morphosyntactic paraphrases, that we evaluate with human judges.

Received 17 December 2010; Revised 10 March 2012; Accepted 10 March 2012; Published online 4 September 2012

*Key words:* bootstrapping, corpus, emotions, k-window, paraphrases.

## 1. INTRODUCTION

Paraphrasing accrues from the various lexical and grammatical means of expressing meaning accessible in language. The importance of paraphrases has become more recognized recently, attracting researchers in field of computational linguistics (CL). Paraphrases ensure the main aspects of variability in language.

### 1.1. Paraphrase Definition

A recent survey by Madnani and Dorr (2010) in paraphrases extraction and generation provided a definition of the concept of paraphrasing. Based on their survey, paraphrasing is most generally defined on the basis of the principle of semantic equivalence: “A paraphrase is an alternative surface form in the same language expressing the same semantic content as the original form.” However, we looked at different sources such as Wikipedia and Google Dictionary to see how they defined paraphrases. Wikipedia: a paraphrase “is a restatement of a text or passages, using other words.” Google Dictionary: “a rewording of something written or spoken by someone else.” We consider that *paraphrases* are simply different ways to express the same information (the same meaning). Paraphrases may occur at several levels. Therefore, individual lexical items which have the same meaning are usually referred to as lexical paraphrases or, more commonly, synonyms, i.e,: $<hot, warm>$ and $<eat, feed>$. On the other hand, lexical paraphrasing cannot be limited to the concept of synonymy. They can be presented in other forms such as hyperonyms, where one of the words in the paraphrastic relationship is either more general or more specific than the other, i.e.,: $<reply, say>$ and $<landlady, hostess>$.

Also there is the term *phrasal paraphrases* which refers to phrasal fragments sharing the same semantic content. These fragments most commonly take the form of syntactic phrases, such as ($<very\ happy, so\ glad>$ and $<take\ over, assume\ control\ of>$) they may also be patterns with linked variables, for example, $<Y\ was\ happy\ by\ X, X\ made\ Y\ glad>$ (Madnani and Dorr 2010).

Address correspondence to Fazel Keshtkar and Diana Inkpen, 800 King Edward Ave., Ottawa, ON, K1N6N5, Canada; e-mail: akeshtka@eecs.uOttawa.ca; diana@eecs.uOttawa.ca

TABLE 1. Two Sentence Fragments from the Emotion Class *happy*, from the Blog Corpus.

| |
|---|
| his little boy was so happy to see him |
| princess and she were very glad to visit him |

Algorithms to extract and automatically identify paraphrases are interesting from both a linguistic and practical perspective. Many major functions of Natural Language Processing applications, such as multidocument summarization, need to avoid repetitive information from input documents in this case paraphrasing can be useful in such applications. In natural language generation (NLG), paraphrasing is employed to create more varied and natural text. In our research, we extract paraphrases for emotions, with the intent of using them to automatically generate emotional texts (e.g., friendly or hostile) for conversations between intelligent agents and characters in educational games. The paraphrasing is applied to generate text with more variety. To our knowledge, most current applications manually collect paraphrases for specific applications, or they use lexical resources such as WordNet (Miller et al. 1993) to identify paraphrases.

In this paper, we introduce a novel method for extracting paraphrases for emotions from texts. We focus on the six basic emotions proposed by Ekman (1992): *happiness*, *sadness*, *anger*, *disgust*, *surprise*, and *fear*.

We describe the construction of the paraphrase extractor, and we also propose a $k$-window algorithm for selecting the contexts that are used in the paraphrase extraction method. We automatically learn patterns that can extract the emotion paraphrases from corpora, starting with a set of seed words. We make use of data sets, such as blogs and other annotated corpora, in which the emotions are marked. We also use a large collection of nonparallel corpora, which are described in Section 3. These corpora contain many instances of paraphrasing of using different phrases to express the same emotion.

An example of sentence fragments for one emotion class, *happiness*, is shown in Table 1. From them, the paraphrase pair that our method will extract is: "`so happy to see`" "`very glad to visit`."

We provide an overview of related work on paraphrasing in Section 2. In Section 3 we describe the data sets that were used in this research, and we explain the details of our paraphrase extraction method in Section 4. We present results of our evaluation and discuss our results in Section 5 and finally in Section 7 we address some contributions and future related work.

## 2. RELATED WORK

Three main approaches for collecting paraphrases were proposed in the literature: manual collection, utilization of existing lexical resources, and corpus-based extraction of expressions that occur in similar contexts (Barzilay and McKeown 2001). Manually collected paraphrases were used in NLG (Iordanskaja, Kittredget, and Polguere 1991). Langkilde and Knight (1998) used lexical resources in statistical sentence generation, summarization, and question–answering. Barzilay and McKeown (2001) used a corpus-based method to identify paraphrases from a corpus of multiple English translations of the same source text. Our method is similar to this, but it extracts paraphrases only for a particular emotion, and requires a regular corpus, not a parallel corpus of multiple translations.

## 2.1. Applications of Paraphrases Extraction and Generation

In their survey, Madnani and Dorr (2010) categorized applications of paraphrase extraction and generation into following categories.

(1) Query and pattern extension: One of the most common applications of paraphrasing is the automatic generation of query variants for submission to information retrieval systems or of patterns for submission to information extraction systems.
(2) Human reference data for evaluation: To measure the performance of the machine translation and summarization systems. Machine translation and document summarization are two applications that use comparison against human-authored references; as one of their evaluation methods.
(3) Machine translation: Paraphrasing has also been applied to directly improve the translation process. Automatically induced paraphrases can be used to improve a statistical phrase-based machine translation system. Such a system works by dividing the given sentence into phrases and translating each phrase individually, by looking up its translation from a table.

## 2.2. Paraphrasing with Corpora

Madnani and Dorr (2010) also explored in detail the corpus-based paraphrase generation and extraction approaches that have emerged and have become popular in the last decade or so. These corpus-based methods have the potential of covering a much wider range of paraphrasing phenomena and the advantage of the widespread availability of corpora.

Madnani and Dorr (2010) organized the corpus-based paraphrases generation and extraction by the type of corpora used to generate the paraphrases. They believed this form of organization is the most instructive, because most of the algorithmic decisions made for paraphrase generation will depend heavily on the type of corpus used. These categories are the following.

(1) Distributional similarity: It is relatively easy to realize the concept of distributional similarity; words or phrases that share the same distribution. The same set of words in the same context in a corpus tend to have similar meanings.
(2) Paraphrasing using a single monolingual corpus: They are paraphrase generation methods that operate on a single monolingual corpus. The idea is almost the same as the Distributional Similarity.
(3) Paraphrasing using monolingual comparable corpora: It is also possible to generate paraphrase pairs from a parallel corpora where each component of the corpus is in the same language.
(4) Paraphrasing using monolingual parallel corpora: Such corpora are usually not very easily available (comparable instead of being truly parallel). Parallelism between sentences is replaced by just partial semantic and topical overlap at the level of. documents, e.g., describing events under the same topics.
(5) Bilingual parallel corpora: Using bilingual parallel corpora for paraphrasing has the inherent advantage that sentences in the other language are exactly semantically equivalent to sentences in the intended paraphrasing language.

Glickman and Dagan (2004) investigated the extraction of lexical paraphrases from a single corpus. They used a syntactic parser to detect structures related to the same event, and from two such structures they extracted the paraphrases for verbs. Their method works only

TABLE 2. The Number of Emotion-Annotated Sentences in Each Data Set.

| Data set | Happiness | Sadness | Anger | Disgust | Surprise | Fear |
|---|---|---|---|---|---|---|
| LiveJournal | 7,705 | 1,698 | 4,758 | 1,191 | 1,191 | 3,996 |
| TextAffect | 334 | 214 | 175 | 28 | 131 | 166 |
| Fairy tales | 445 | 264 | 216 | 217 | 113 | 165 |
| Annotated blog data set | 536 | 173 | 115 | 115 | 172 | 179 |

for verbs, whereas our method allows the extraction of any lexical paraphrases if sample seeds are available.

Some research has been done in paraphrase extraction for natural language processing and generation for different applications. Das and Smith (2009) presented a approach to determine whether two sentences have a paraphrase relationship. They applied a generative model to produce a paraphrase of a given sentence, then used probabilistic inference to asses whether two sentences share the paraphrase relationship. In another research, Wang et al. (2009) studied the problem of extracting technical paraphrases from a parallel software corpus. Their aim was to report duplicate bugs, and to do this they used sentence selection, and global context-based and cooccurrence-based scoring. Studies have also been done in paraphrase generation in NLG (Zhao et al. 2009; Chevelu et al. 2009).

Bootstrapping methods have been applied to various natural language applications, including word sense disambiguation (Yarowsky 1995), lexicon construction for information extraction (Riloff and Jones 1999). Another bootstrapping method was proposed by Riloff and Wiebe (2003) to learn extraction patterns for subjective expressions. They used unannotated data to automatically create a training set, which is given to an extraction pattern learning algorithm. The learned patterns are then used to identify more subjective sentences. The bootstrapping process learns many subjective patterns and increases recall although maintaining high precision. Banea, Mihalcea, and Wiebe (2008) also used a bootstrapping method for building subjectivity lexicons for languages with scarce resources. Their method is able to create a subjectivity lexicon by using a small seed set of subjective words, an online dictionary, and a small raw corpus, coupled with a bootstrapping process that ranks new candidate words based on a similarity measure. Their algorithm, with a rule-based sentence level subjectivity classifier showed an 18% absolute improvement in F-measure as compared to previously proposed semi-supervised methods. In another research that was done by Collins and Singer (1999), a bootstrapping algorithm was used for named entity classification. In our research, we use the bootstrapping approach to learn paraphrases for emotions.

Pantel and Pennacchiotti (2006) developed an algorithm called Espresso, that filters out incorrect patterns by using Web data. Their algorithm detects the similarity between the incorrect patterns and the texts from the Web. They do not extract paraphrases; instead Espresso produces generic patterns that are semantically related.

## 3. DATA SET

The text data from which we extract paraphrases is composed of four concatenated data sets (Table 2). These data sets contain sentences that labeled with the six basic emotions. The number of sentences in each data set is presented in Table 2. A brief description of the data sets follows.

### 3.1. LiveJournal Blog Data Set

We used the blog corpus that was collected by Mishne (2005) for his research. The corpus contains 815,494 blog posts from Livejournal,[1] a free weblog service used by millions of people to create weblogs. In Livejournal, users can optionally specify their current emotion or mood. They choose their emotion/mood from a provided list of 132 emotions/moods, and thus, the data is labeled by the user who created the blog. We assume that user's emotional state matches his/her blog post, even if it might not always be the case. We selected only the texts corresponding to the six basic emotions above (Ekman's emotions).

### 3.2. Text Affect Data Set

This data set,[2] complied by Strapparava and Mihalcea (2007), consists of newspaper headlines that were used in the SemEval 2007-Task 14 Workshop. It includes a development data set of 250 annotated headlines, and a test data set of 1,000 news headlines. We used all of these. The annotations were made with the six basic emotions on intensity scales of $-100$ to $+100$, meaning a threshold is used to choose the main emotion of each sentence.

### 3.3. Fairy Tales Data Set

This data set, collected by Alm, Roth, and Sproat (2005), consists of 1,580 annotated sentences from tales by the Grimm Brothers, Hans Christian Andersen, and Beatrix Potter. The annotations used the extended set of nine basic emotions of Izard (1971). We selected only those marked with the six emotions that we focused on.

### 3.4. Annotated Blog Data Set

We also used the data set provided by Aman and Szpakowicz (2007). Emotion-rich sentences were selected from personal blogs, and annotated with the six emotions (as well as a nonemotion class, which is not considered here). They worked with blog posts, which were collected directly from the Web. They first prepared a list of seed words for each of the six basic emotion categories proposed by Ekman (1992). They then selected words commonly used in the context of a particular emotion. Finally, they used the seed words for each category, and retrieved blog posts containing one or more of these for the annotation process.

## 4. OUR METHOD FOR PARAPHRASE EXTRACTION

For each of the six emotions, we ran our method on the set of sentences marked with the corresponding emotion from the concatenated corpus. We started with a set of seed words from WordNet Affect (Strapparava and Valitutti 2004), for each emotion of interest. The number of seed words was: happiness 395, surprise 68, fear 140, disgust 50, anger 250, and sadness 200. Table 3 shows some of the seeds for each category of emotion.

[1] http://www.livejournalinc.com

[2] In the Text Affect data set, each sentence is annotated with six scores, one for each class of emotion. Because in our corpus each sentence has only one emotion label, we also used one label for the Text Affect sentences, by considering the dominant emotion label (based on the highest score, except the cases when all the scores were low).

TABLE 3. Some of the Seeds from WordNet Affect for Each Category of Emotion.

| | |
|---|---|
| Happiness: | avidness, glad, warmheartedness, exalt, enjoy, comforting, joviality, amorous, joyful, like, cheer, adoring, fascinating, happy, impress, great, satisfaction, cheerful, charmed, romantic, joy, pleased, inspire, good, fulfill, gladness, merry |
| Sadness: | poor, sorry, woeful, guilty, miserable, glooming, bad, grim, tearful, glum, mourning, joyless, sadness, blue, rueful, hamed, regret, hapless, regretful, dismay, dismal, misery, godforsaken, oppression, harass, dark, sadly, attrition |
| Anger: | belligerence, envious, aggravate, resentful, abominate, murderously, greedy, hatred, disdain, envy, annoy, mad, jealousy, huffiness, sore, anger, harass, bother, enraged, hateful, irritating, hostile, outrage, devil, irritate, angry |
| Disgust: | nauseous, sicken, foul, disgust, nausea, revolt, hideous, horror, detestable, wicked, repel, offensive, repulse, yucky, repulsive, queasy, obscene, noisome |
| Surprise: | wondrous, amaze, gravel, marvel, fantastic, wonderful, surprising, marvelous, wonderment, astonish, wonder, admiration, terrific, dumfounded, trounce |
| Fear: | fearful, apprehensively, anxiously, presage, horrified, hysterical, timidity, horrible, timid, fright, hesitance, affright, trepid, horrific, unassertive, apprehensiveness, hideous, scary, cruel, panic, scared, terror, awful, dire, fear, dread, crawl, anxious, distrust, diffidence |

Because sentences in our data sets differ and they are not aligned as parallel sentences as in Barzilay and McKeown (2001), our algorithm constructs pairs of similar sentences, based on the local context. We assumed that if the contexts surrounding two seeds look similar, then these contexts are likely to be helpful for extracting new paraphrases. Once we find which contexts are useful for identifying paraphrases, we are able to extract more paraphrase patterns from our data set. Therefore, we simply define a context as a window of $\pm k$ words that surround an emotion seeds. Verb-object relations and noun-modifier relations are examples of such contexts. They have been used from nonparallel corpora in word similarity tasks (Pereira, Tishby, and Lee 1993; Hatzivassiloglou and McKeown 1997). However, in our task, we can use any relations for paraphrasing, because we know which sentences convey the same information and belong to the same category of emotions.

Figure 1 illustrates the high-level architecture of our paraphrase extraction method. The input to this is a text corpus for an emotion category and a manually defined list of seed words. Before starting bootstrapping, we ran the $k$-window algorithm on every sentence in the corpus, to construct *candidate contexts*. In Section 4.6 we explain how the bootstrapping algorithm processes and selects the paraphrases based on strong surrounding contexts. As shown in Figure 1, our method has several stages: extracting candidate contexts, using them to extract patterns, selecting the best patterns, extracting potential paraphrases, and filtering these to get the final paraphrases.

## 4.1. Preprocessing

During preprocessing, HTML and XML tags are eliminated from the blogs data and other data sets, then the text is tokenized and annotated with part-of-speech tags. We use the Stanford part-of-speech tagger and chunker (Toutanova et al. 2003) to identify noun and verb phrases in the sentences. Then we use a sliding window based on the $k$-window approach, to identify candidate contexts that contain the target seeds.

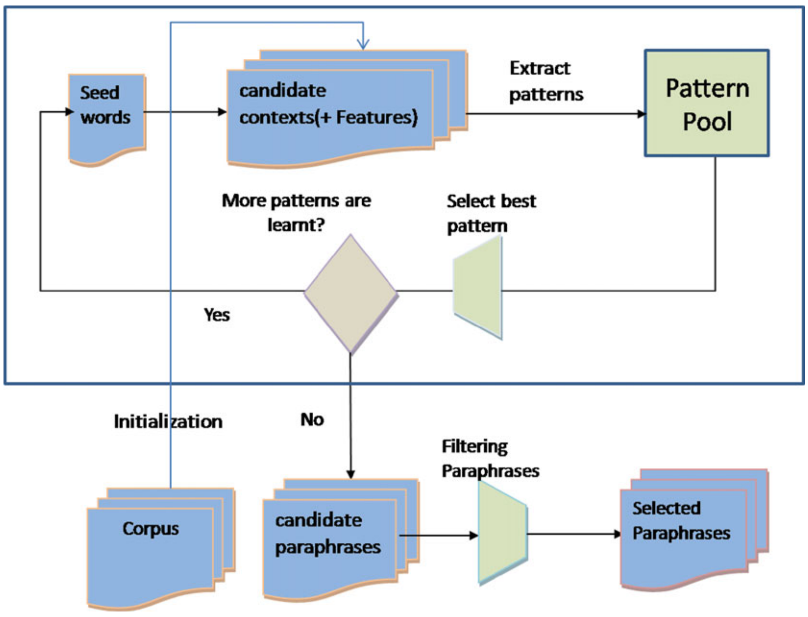

FIGURE 1. High-level view of the paraphrase extraction method (adopted from Banea et al. 2008).

## 4.2. The $k$-Window Algorithm

We use the $k$-window algorithm introduced by Bostad (2003) to identify all the tokens surrounding a specific term in a window with a size of $\pm k$ words. We use this approach to extract candidate patterns for each seed, from the sentences. Starting with one seed, we truncate all surrounding contexts within a window of $k$ words before and $k$ words after the seed, until all the seeds have been processed. Because the value of $k$ was set to 5 due to recommendations from related work (Khan 2007; Barzilay and McKeown 2001), for these experiments the longest candidate contexts will have the form $w_1, w_2, w_3, w_4, w_5, seed, w_6, w_7, w_8, w_9, w_{10}$, where $w_i (i = 1, \dots, 10)$ are word tokens. We explain the features we extract from each candidate context to determine similar contexts in the same subsection.

## 4.3. Feature Extraction

Previous research on word sense disambiguation on contextual analysis has identified several local and topical features that are good indicators of word properties. These include surrounding words and their part-of-speech tags, collocations, and keywords in contexts (Mihalcea 2004). Other features have also been proposed: bigrams, named entities, syntactic features, and semantic relations with other words in the context.

We transfer the candidate phrases extracted by the sliding $k$-window algorithm into a vector space of features. We consider features that include both lexical and syntactic descriptions of the paraphrases for all pairs of two candidates. The lexical features include the sequence of tokens for each phrase in the paraphrase pair. The syntactic features consist of a sequence of part-of-speech (PoS) tags where equivalent words and words with the same root and the same PoS are marked. For example, the value of the syntactic feature for the pair "`so glad to see`" and "`very happy to visit`" is "$RB_1$ $JJ_1$ $TO$ $VB_1$" and "$RB_2$ $JJ_2$ $TO$ $VB_{2,}$' where indices indicate word equalities. However, based on the above

TABLE 4. An Example of Extracted Features (the Features Are Separated by Commas (",")).

| Candidate context: He was further annoyed by the jay bird |
|---|
| 'PRP VBD RB VBN IN DT NN NN','VBD RB',?,was, ?,?,?,He/PRP,was/VBD,further/RB,annoyed,by/IN,the/DT, jay/NN,bird/NN,?,?,jay,?,'IN DT NN' |

TABLE 5. The Features that We Used for Paraphrase Extraction.

| Features | Description |
|---|---|
| F1 | Sequence of part-of-speech |
| F2 | Sequence of PoS between the seed and the first verb before the seed |
| F3 | Sequence of PoS between the seed and the first noun before the seed |
| F4 | First verb before the seed |
| F5 | First noun before the seed |
| F6 | Token before the seed |
| F7 | Seed |
| F8 | Token after the seed |
| F9 | First verb after the seed |
| F10 | First noun after the seed |
| F11 | Sequence of PoS between the seed and the first verb after the seed |
| F12 | Sequence of PoS between the seed and the first noun after the seed |

TABLE 6. Two Sentence Fragments (Candidate Contexts) from the Emotion Class *happy*, from the Blog Corpus.

| |
|---|
| his little boy was so "*seed*" to see him |
| princess and she were very "*seed*" to visit him |

evidence and our previous research, we also investigate other features that could help to achieve our goal. Table 5 lists the features that we used for paraphrase extraction, which also include some term frequency features. For example, in Table 4 we show extracted features from a relevant context. Our results prove that these features work and they are useful for our algorithm.

## 4.4. Analyzing the Context Surrounding the Seeds

Considering the differences between sentences from different corpora, our method builds on the similarity of the local contexts within *k*-windows, rather than on global contexts. For example, consider the two sentences in Table 6. Analyzing the contexts surrounding the "seed" placeholder (in italics and bold in both sentences), it is expected that the two contexts would have the same meaning, because they have the similar premodifiers ("so" and "very"), and a postmodifier related to the same preposition, "to." In fact, the first seed is "glad," and the second seed is "happy." We hypothesize that if the contexts surrounding two phrases look similar enough, these two phrases are likely to be paraphrases. Once we know which contexts are good paraphrase predictors, we can extract paraphrase patterns from our corpus.

**Algorithm 1: Bootstrapping Algorithm.**

```
For each seed for an emotion
  Loop until no more paraphrases or no more contexts are learned.
    1- Locate the seeds in each sentence
    2- Find similar contexts surrounding a pair of two seeds
    3- Analyze all contexts surrounding the two seeds to extract
       the strongest patterns
    4- Use the new patterns to learn more paraphrases
```

FIGURE 2. Our bootstrapping algorithm for extracting paraphrases.

We should note here that the algorithm was run separately for the each six classes of emotions. Because the *seeds* are disjoint, the extracted paraphrases will not overlap when the algorithm runs for specific emotion. We keep the classes separate.

Can we conclude that identical modifiers of a subject imply verb similarity? To address this question, we need to identify contexts that are good predictors for paraphrasing in a corpus. Because the $k$-windows have fixed size for all candidate contexts, it is obvious that all the windows can be aligned. To find relevant contexts, we analyze all the contexts surrounding equivalent words in the pairs of aligned $k$-window, and use these to learn new paraphrases. This provides the basis for a bootstrapping mechanism. Starting with equivalent seed words in aligned sentences, we can incrementally learn the relevant contexts, then use these to learn new paraphrases. Equivalent seed words play two roles in this process; they are used to learn context rules, and they are used in the application of these rules, because the rules contain information about the equality of the words in context. This method has been previously applied to a variety of natural language tasks, including word sense disambiguation (Yarowsky 1995), lexicon construction for information extraction (Riloff and Jones 1999), and named entity classification (Collins and Singer 1999). In our case, the bootstrapping process creates a binary classifier, which predicts whether a given pair of phrases creates a paraphrase or not.

## 4.5. Predicting Pairs of Paraphrases from Contextual Features

We define the contextual features as combinations of the left and right syntactic contexts surrounding actual known paraphrases. There are a number of context representations that are possible candidates, including lexical $n$-grams, POS $n$-grams and parse tree fragments. Part-of-Speech tags provide the required level of abstraction, and can be accurately computed for our texts. Suppose we have two candidate $k$-windows $w1$ and $w2$. Then we define $pair_1$ = ($left_1$, "seed", $right_1$) and $pair_2$ = ($left_2$, "seed", $right_2$). As mentioned in Section 4.2, the left or right context is a sequence of part-of-speech tags of $k$ words, occurring on the left or right of the paraphrase. We mark tags of equivalent words in each pair as syntactic paraphrase features. For example, when $k = 5$, the contextual feature for the paraphrase pair ("glad," "happy") from the sentences in Table 6 is: "$RB_1\ JJ_1\ TO\ VB_1$" and "$RB_2\ JJ_2\ TO\ VB_2$." In the next section, we describe how we learned to extract paraphrases, using these contextual features.

## 4.6. Bootstrapping Algorithm for Paraphrase Extraction

Our bootstrapping algorithm is summarized in Figure 2. It begins with a set of seeds, which are considered initial paraphrases. The set of extraction patterns is initially empty. The algorithm generates candidate contexts from the aligned similar contexts (how to obtain contexts is explained in Section 4.4). The candidate patterns are scored by the number of paraphrases they can extract. Those with higher scores are added to the set of extraction patterns. Using the extended set of extraction patterns, more paraphrase pairs are extracted and added to the set of paraphrases. Using the enlarged set of paraphrases, more extraction patterns are extracted. The process repeats until no new patterns or paraphrases are learned. Our bootstrapping algorithm uses two loops, one for each seed and one for each paraphrase pattern; therefore, the complexity of our bootstrapping algorithm is $O(n^2)$, where $n$ is the number of seeds.

Our method can accumulate a large lexicon of emotion phrases, by bootstrapping from the manually initialized list of seed words. In each iteration, the paraphrase set is expanded with related phrases found in the corpus, which are filtered using a measure of surrounding context similarity. This measure is based on the number of pair of paraphrase that extracted by each pair. On the other hand, pair that extracts more paraphrases is stronger. The bootstrapping process starts by selecting a subset of the extraction patterns that are intended to extract the paraphrases. We call this set the "pattern pool." The phrases extracted by these patterns become candidate paraphrases. They are filtered based on how many patterns select them, to produce the final paraphrases from the set of candidate paraphrases.

*4.6.1. Initialization.* Words which appear in both sentences of an aligned pair are used to create the initial seed patterns. Using seed words (or equivalent words) for the emotion class, we create a set of positive paraphrasing examples, such as $word_1 = glad$, and $word_2 = happy$. The words that are not equivalent in aligned pairs are used to produce negative examples as well, so these words are not paraphrases of each other. To find negative examples, we match the equivalent words against all the different words in the pairs, and we stipulate that equivalent words can match only each other, and no other word in the aligned pairs.

*4.6.2. Extracting Patterns.* Using these initial seed, we record contexts around positive and negative paraphrase examples. From all the extracted contexts, we need to identify those ones which are strong predictors of their category. We extracted the features from each candidate context, as described above. Then we learn extraction patterns, in which some words might be substituted by their part-of-speech. We use the seeds to build initial patterns. Two candidate contexts that contain the same seed create one positive example. By using each initial seed, we can extract all contexts surrounding these positive examples. Then we select the stronger ones. We used the method of Collins and Singer (1999) to compute the strength of each example. If we consider $x$ as a context, the strength of $x$ as a positive example is defined as:

$$Strength(x) = count(x+)/count(x), \tag{1}$$

where $count(x+)$ is the number of times context $x$ surrounded a seed in a positive example, and $count(x)$ is frequency of the context $x$. This allows us to score the potential pattern.

Context length is an important parameter for learning new patterns. Based on our experiments, we found a context length of three words leads to the best results. Moreover, we noticed that for some patterns, less lengthy contexts perform better. For these reasons, when we record contexts around positive examples, in most cases we keep all the contexts with lengths less than or equal to three.

*4.6.3. Extracting the Paraphrasing.* After we extracted the context patterns in the previous step, we applied them to the corpus to detect a new set of pairs of paraphrases. The results of the patterns that were determined by searching for pairs of paraphrases that match the *left* and the *right* parts of a *seed* are strings of up to *k* tokens. Then, for each extracted pair, new paraphrases are recorded and filtered in the same manner as the contextual patterns. This iterative process is terminated when no new paraphrases are detected.

## 5. RESULTS AND EVALUATION

Our algorithm generates a set of extraction patterns, and a set of pairs of paraphrases: some of the extracted paraphrase are shown in Table 7. The paraphrases that are considered correct are shown under *Correct paraphrases*. As explained in the next section, two human judges agreed that these paraphrases are acceptable. The results considered incorrect by the two judges are shown under *Incorrect paraphrases*.

Our algorithm learned 196 extraction patterns, and produced 5,926 pairs of paraphrases. Table 8 shows the number of extraction patterns and the number of paraphrase pairs produced by the algorithm for each class of emotions. We used two techniques to evaluate our algorithm: First, human judges determined if the paraphrases extracted from the sample are correct, and we calculated the degree of agreement between the judges (see Section 5.1). Second, we assessed the recall and precision of our method (see Section 5.2). We describe these evaluations in the following subsections.

### 5.1. Evaluating Correctness with Human Judges

We used the same method as Barzilay and McKeown (2001) to evaluate the correctness of the extracted paraphrase pairs. We randomly selected 600 paraphrase pairs from the lexical paraphrases produced by our algorithm, and 100 paraphrase pairs for each class of emotion. Two human judges evaluated the correctness of each paraphrase, to determine whether or not the two expressions are good paraphrases.

We provided guidelines for the judges, in which we defined paraphrase as “approximate conceptual equivalence,” the same definition used by Barzilay and McKeown (2001). Each judge could choose “Yes” or “No” for each pair of paraphrases being tested. We did not include example sentences which contained these paraphrases. A similar Machine Translation evaluation task for word-to-word translation was done by Melamed (2001).

Figure 3 presents the results of the evaluation (i.e., the correctness for each class of emotion according to judges A and B). The judges were graduate students in CL, and native speakers of English.

We also measured the agreement between the two judges and the Kappa coefficient (Siegel and Castellan 1988). If there was complete agreement between the judges $Kappa = 1$, and if there is no agreement $Kappa = 0$. The Kappa values and agreement values for our judges are presented in Figure 4.

The interjudge agreement for all the paraphrases of the six classes of emotions is 81.72% (490 out of the 600 paraphrase pairs in our sample). The judges agreed that some pairs are good paraphrases or some pairs are not good paraphrases, which is why the values in Figure 3 are higher than the correctness values in Figure 4. The $Kappa$ coefficient compensates for chance agreement. The $Kappa$ value over all the paraphrase pairs is 74.41%, which is significant agreement.

### 5.2. Estimating Recall

Evaluating the *Recall* of our algorithm is difficult, because the algorithm does not include all English words; it can only detect paraphrasing relationships with the words in our corpus. Moreover, a direct comparison using an electronic thesaurus such as WordNet is not feasible,

TABLE 7. Examples of Paraphrases Extracted by Our Algorithm (Correctly and Incorrectly).

| Disgust |
|---|
| *Correct paraphrases*: |
| being a wicked::getting of evil; been rather sick::feeling rather nauseated; |
| feels somewhat queasy::felt kind of sick; damn being sick::am getting sick |
| *Incorrect paraphrases*: |
| disgusting and vile::appealing and nauseated; get so sick::some truly disgusting |
| Fear |
| *Correct paraphrases*: |
| was freaking scared::was quite frightened; just very afraid::just so scared; |
| tears of fright::full of terror; freaking scary::intense fear; |
| *Incorrect paraphrases*: |
| serious panic attack::easily scared; not necessarily fear::despite your fear |
| Anger |
| *Correct paraphrases*: |
| upset and angry::angry and pissed; am royally pissed::feeling pretty angry; |
| made me mad::see me angry; do to torment::just to spite |
| *Incorrect paraphrases*: |
| very pretty annoying::very very angry; bitter and spite::tired and angry |
| Happiness |
| *Correct paraphrases*: |
| the love of::the joy of; in great mood::in good condition; |
| the joy of::the glad of; good feeling::good mood |
| *Incorrect paraphrases*: |
| as much eagerness::as many gladness; feeling smart::feel happy |
| Sadness |
| *Correct paraphrases*: |
| too depressing::so sad; quite miserable::quite sorrowful; |
| strangely unhappy::so misery; been really down::feel really sad |
| *Incorrect paraphrases*: |
| out of pity::out of misery; akward and depressing::terrible and gloomy |
| Surprise |
| *Correct paraphrases*: |
| amazement at::surprised by; always wonder::always surprised; |
| still astounded::still amazed; unexpected surprise::got shocked |
| *Incorrect paraphrases*: |
| passion and tremendous::serious and amazing; tremendous stress::huge shock |

TABLE 8. The Number of Lexical and Extraction Patterns Produced by the Algorithm.

| Class of Emotion | # Paraphrases Pairs | # Extraction Patterns |
|---|---|---|
| Disgust | 1,125 | 12 |
| Fear | 1,004 | 31 |
| Anger | 670 | 47 |
| Happiness | 1,095 | 68 |
| Sadness | 1,308 | 25 |
| Surprise | 724 | 13 |
| Total | 5,926 | 196 |

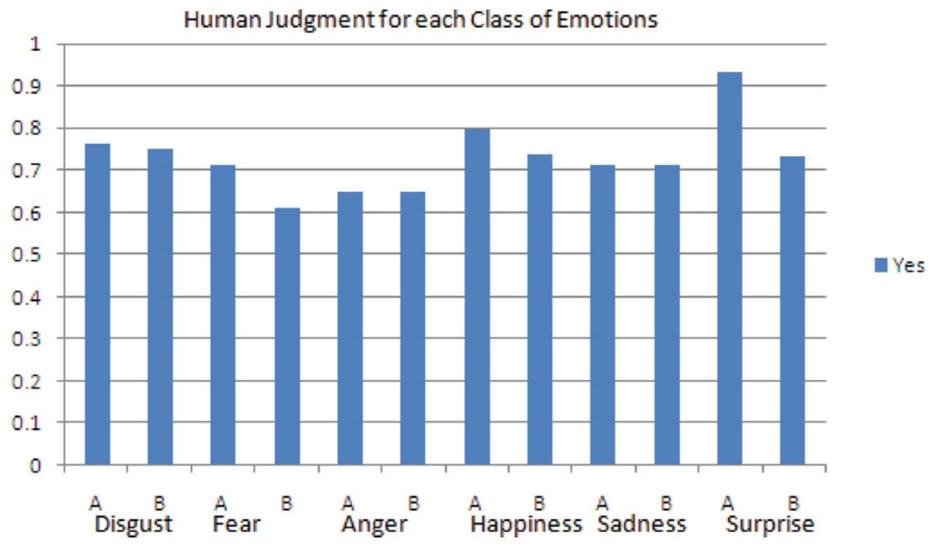

FIGURE 3. The correctness results according the judge A and judge B, for each class of emotion.

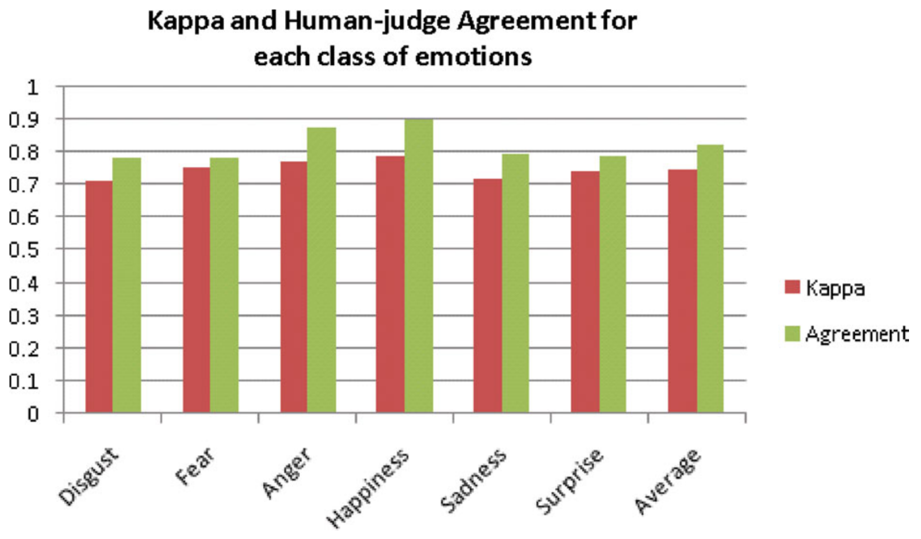

FIGURE 4. The *Kappa* coefficients and the agreement between the two human judges.

TABLE 9. Precision and Recall for a Sample of Texts, for Each Category of Emotion, and Their Average.

| Category of Emotions | Precision | Recall |
|---|---|---|
| Disgust | 82.33% | 92.91% |
| Fear | 82.64% | 88.20% |
| Anger | 93.67% | 80.57% |
| Happiness | 82.00% | 90.89% |
| Sadness | 82.00% | 89.88% |
| Surprise | 79.78% | 89.50% |
| Average | 84.23% | 88.66% |

because WordNet contains mostly synonym sets between words, and very few multiword expressions. Thus we decided to estimate recall manually, by having a human judge extract paraphrases from a sample of text. We randomly selected 60 texts (10 for each emotion class) and asked a judge to extract paraphrases from them. For each emotion class, the judge extracted expressions that reflected the emotion, and then combined pairs that were conceptually equivalent. Because this process was very time-consuming and tedious, it was not practical to ask a second judge to do the same task.

With respect to information retrieval, *Precision* and *Recall* are defined in terms of a set of retrieved documents and a set of relevant documents.[3] In the following, we describe how we computed the *Precision* and *Recall* for our algorithm, compared to the manually extracted paraphrases.

For precision, we compared the number of paraphrases that were extracted by the algorithm from the same texts, with the number that were also extracted by the human judge (see Equation 2). On average, from 89 paraphrases extracted by the algorithm, 74 were identified as paraphrases by the human judge (84.23%). See Table 9 for the values for all the classes.

$$P = \frac{\#\ Correctly\ Retrieved\ Paraphrases\ by\ the\ Algorithm}{All\ Paraphrases\ Retrieved\ by\ the\ Algorithm} \tag{2}$$

For computing the *Recall* we count how many of the paraphrases extracted by the human judge were correctly extracted by the algorithm (Equation 3).

$$R = \frac{\#\ Correctly\ Retrieved\ Paraphrases\ by\ the\ Algorithm}{All\ Paraphrases\ Retrieved\ by\ the\ Human\ Judge} \tag{3}$$

As mentioned, evaluating *Recall* is a difficult task for a human judge, so we provide an explanation of how a judge extracts the paraphrases from texts: We gave the judge several randomly selected texts for each emotion class, from which to extract paraphrases. In the following example, the random text we gave the judge is from the *Happiness* class:

```
''Then the maiden looked at them and recognized her broth-
ers, was glad and crept forth from beneath the bed. When the
cloth was laid, the Lord sat down with the man and his wife,
and he enjoyed their coarse food, for there were happy faces
at the table. But the moment he kissed her she opened her
eyes and awoke, and smiled upon him; and they went out to-
gether; and soon the king and queen also awoke, and all the
```

[3] http://en.wikipedia.org/wiki/

```
court, and gazed on each other with great wonder. Then the
children went home together, and were heartily delighted,
and if they have not died, they are living still. Hans was
delighted as he sat on the horse, drew himself up, squared
his elbows, turned out his toes, cracked his whip, and rode
merrily off, one minute whistling a merry tune, and another
singing, ''How lighthearted I feel,'' said the father, ''so
pleased and cheerful.'' The giant was pleased with the good
cheer, and ate and drank to his heart's content. So Master
Thumb stayed at home with his father and mother, in peace;
for though he had been so great a traveller, and had done
and seen so many fine things, and was fond enough of telling
the whole story, he always agreed that, after all, there's no
place like HOME!''
```

For extracting the paraphrases, the judge performed the following procedure. First, emotion *expressions* were extracted from the whole text, then pairs of two *expressions* were formed in such a way that they can be considered paraphrases. The *expressions* extracted from the above text are: "was glad," "enjoyed," "happy," "smiled," "gazed," "great wonder," "heartily delighted," "delighted," "merrily," "whistling a merry tune," "singing," "lighthearted," "pleased and cheerful," "pleased with the good cheer," "to his heart"s content," "in peace," "fond enough."

The judge decided that the following seven pairs were paraphrases: "whistling a merry tune :: singing," "pleased and cheerful :: pleased with the good cheer," "was glad :: delighted," "lighthearted :: happy," "glad :: happy," "lighthearted :: to his heart's content," "happy :: delighted."

From the seven paraphrases extracted by the judge, our system identified five as paraphrases. The two not detected by the system were:"whistling a merry tune :: singing" and "lighthearted :: to his heart's content." The system also detected two paraphrases which were not detected by the judge: "glad :: like" and "happy :: like." These are considered wrong by the human judge.

## 5.3. Error Analysis

We looked at some of the extracted paraphrases, particularly the samples which the human judges considered correct. A few examples are listed in Table 7.

There are several comments to be made. Some paraphrases are not grammatically correct, but they have the right semantic content from the point of view of the expressed emotions. In such cases, the human judges considered the paraphrases correct. Due to their informal nature, the blog texts contain a lot of ungrammatical sentences, causing our paraphrase extraction algorithm to extract some awkward paraphrases (e.g., *damn being sick::am getting sick*). We chose to keep these paraphrases in our result database. Later on, when we use the paraphrases in NLG applications, we will employ an additional module, such as a language model built on a very large corpus, to ensure that we do not generate awkward sentences.

Some paraphrases contain offensive language, such as the last word in the pair: *upset and angry::angry and pissed*. This is again due to the informal aspect of the blog texts. These paraphrases can be filtered out, when they are used in an application that prohibits such wording in the generated language. We do the filtering of the offensive expressions using a system from our previous work (Razavi et al. 2010).

Among the paraphrases considered incorrect by the human judges, some have the expected parallel structure but the meanings of the words are too different. For example, the conjunctions of adjectives *bitter and spite::tired and angry* were not considered correct paraphrases because the second also implies "tiredness," whereas the first does not.

Other errors could be caused by expressions with the exact same seed, but surrounded by contexts that are not parallel; for example, *not necessarily fear::despite your fear*. In general, these contexts will not have high strength, but some of them could be strong enough to be used by the algorithm.

## 6. DISCUSSION AND COMPARISON TO RELATED WORK

To the best of our knowledge, no similar research has been done involving extracting paraphrases for emotion terms from corpora. However, Barzilay and McKeown (2001) did comparable work for corpus-based identification of general paraphrases from multiple English translations of the same source text. We examined the pros and cons of our method versus their method. The advantages of our method are:

(1) The corpus does not need to be parallel. Our algorithm uses the entire corpus together to construct its bootstrapping method, whereas in Barzilay and McKeown (2001) the parallel corpus is needed to detect positive contexts. Because we construct the candidate contexts based on the $k$-window approach, there is no need for sentences to be aligned. In Barzilay and McKeown (2001) sentence alignment is essential, to recognize equivalent words and positive contexts. The Barzilay and McKeown (2001) algorithm must find positive contexts before it can look for appropriate patterns to extract paraphrases. Thus, if equivalent words do not occur in the aligned sentences, the algorithm fails to find positive contexts. Our algorithm starts with given seeds, which allows us to detect positive context with the $k$-window method.
(2) The experiments and evaluation indicate that our bootstrapping algorithm and paraphrasing method achieve well and produces reasonable outcomes.

Glickman and Dagan (2004) investigated the extraction of lexical paraphrases for verbs from a single corpus. Their method identifies isolated paraphrase instances in a single corpus, without relying on any a priori structure and information. Their algorithm used a syntactic parser to identify the syntactic structure of the corpus sentences, and to identify verb instances. They treated the corpus uniformly as a set of distinct sentences, regardless of the document or paragraph the sentences belonged to. For each verb instance, they extracted the syntactic components that were directly related to the verb, in the parse tree. They conducted their experiments on the first 15 million word/token subset of the Reuters Corpus. They also used human judges for evaluation. Their correctness was evaluated at 61.40%, the agreement between the judges was 0.63, and the $Kappa$ value was 0.61.

Their approach has some limitations and disadvantages, such as it can only extract paraphrases for verbs. The algorithm depends on verifying whether two verb instances are likely to be paraphrases that describe the same event. Although it uses a single corpus (as opposed to parallel), to allow the extraction of verb paraphrases the syntactic structure of the sentences must be about the same events. Our approach is broader, because it works for all types of lexical paraphrases.

A limitation of our method is the need for initial seed words. However, obtaining these is not a problem, as they can be found in online dictionaries, WordNet Affect and other lexical recourses.

### 6.1. Negation and Metaphor

Negation can be considered as a fundamental cognitive operation, and it can help to shape the sentiment expressed at a number of levels. Negation primarily interacts with sentiment or emotion by changing the meaning expressed by a sentence.

Negation and metaphors mostly have been used in sentiment and emotion classification systems to determine whether an author is praising or criticising the primary subject in the document (Pang and Lee 2008; Turney 2002). However, Polanyi and Zaenen (2004) argue that these techniques work with lexical information only, which is insufficient to determine sentiment. Lexical-only techniques do not take into consideration syntactic and discourse structures which modify the opinion polarity of lexical terms. However, Khan (2007) show in his study that lexical approaches to sentiment classification can benefit from syntactic negation, one of the contextual valence shifters suggested by Polanyi and Zaenen (2004). Khan (2007) showed that a sentiment classification system that accurately takes into account negation outperforms a similar system that operates at the lexical level alone.

In English, negation acts in the syntactic, lexical, and semantic level. Some examples provided by Lawler (2007) include: syntactic constructions (*I did not read the book*), morphology (*-n't, -free, un-*), phonology (*do/don't*), intonation (*'Riiight'*), incorporated (*doubt, lack*), calculated (*none, few*), entailed (*prohibit*), and presupposed (*only*).

As we discussed above, negation and metaphors are important in sentiment and emotion detection. In our algorithm, for paraphrase extraction, we do not determine the emotion. We assume that negation, sarcasm, or irony has been expressed in contexts already. Moreover, our algorithm uses negation words implicitly, when aligning two contexts; so hopefully the extracted paraphrases pair will contain expressions with the same meaning. Metaphors and sarcasms are not currently treated in our system.

## 7. CONCLUSION AND FUTURE WORK

In this paper, we introduced a method for corpus-based extraction of paraphrases for emotion terms. We developed a procedure that uses a bootstrapping technique based on contextual and lexical features, that can successfully extract paraphrases using a nonparallel corpus. We showed that a bootstrapping algorithm based on the contexts of the paraphrases achieves good performance results on our data set. The evaluation suggests that our approach, based on algorithms that are adapted to extract many of the paraphrases that are typical for a multiple corpus, is clearly achievable when measured against similar methods. Moreover, a preliminary comparison suggests that the paraphrase extraction algorithm used in our bootstrapping method is more reliable than other methods based on global alignment, instance-based or vector similarity approaches.

In future work, we will extend this technique to extract paraphrases from more corpora and for additional types of emotions. We plan to test different sizes of the context window, to see if they make a difference. In terms of evaluation, we will use the extracted paraphrases as features in machine learning classifiers that sort candidate sentences into classes of emotions. If the results of the classification are good, the extracted paraphrases are of high quality. Future research is planned to extend the approach to handle more complex paraphrase structures, and to increase performance by relying on additional sources of evidence.

## REFERENCES

ALM, C. O., D. ROTH, and R. SPROAT. 2005. Emotions from text: Machine learning for text-based emotion prediction. *In* Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP 2005), Vancouver, Canada, pp. 579–586.

AMAN, S., and S. SZPAKOWICZ. 2007. Identifying expressions of emotion in text. *In* Proceedings of TSD 2007. Springer: Berlin Heidelberg, pp. 196–205.

BANEA, C., R. MIHALCEA, and J. WIEBE. 2008. A bootstrapping method for building subjectivity lexicons for languages with scarce resources. *In* Proceedings of the Sixth International Language Resources and Evaluation (LREC'08), Marrakech, Morocco.

BARZILAY, R., and K. R. MCKEOWN. 2001. Extracting paraphrases from a parallel corpus. *In* Proceedings of 39th Annual Meeting of the Association for Computational Linguistics (EACL 2001), Toulouse, France, pp. 50–57.

BOSTAD, T. 2003. Sentence Based Automatic Sentiment Classification. Ph. D. Thesis, Computer Speech Text and Internet Technologies (CSTIT), Computer Laboratory, University of Cambridge, Cambridge, UK.

CHEVELU, J., T. LAVERGNE, Y. LEPAGE, and T. MOUDENC. 2009. Introduction of a new paraphrase generation tool based on Monte-Carlo sampling. *In* Proceedings of ACL-IJCNLP 2009, Singapore, pp. 249–325.

COLLINS, M., and Y. SINGER. 1999. Unsupervised models for named entity classification. *In* Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora, University of Maryland, College Park, MD, pp. 100–110.

DAS, D., and N. A. SMITH. 2009. Paraphrase identification as probabilistic quasi-synchronous recognition. *In* Proceedings of ACL-IJCNLP 2009, Singapore, pp. 468–476.

EKMAN, P. 1992. An argument for basic emotions. Cognition and Emotion, **6**:169–200.

GLICKMAN, O., and I. DAGAN. 2004. Acquiring lexical paraphrases from a single corpus. *In* Recent Advances in Natural Language Processing III. *Edited by* Nicolov, Nicolas, K Bontcheva, G. Angelova, and R. Mitkov. (selected papers from RANLP-2003). John Benjamins: Amsterdam, the Netherlands, pp. 81–90.

HATZIVASSILOGLOU, V., and K. MCKEOWN. 1997. Predicting the semantic orientation of adjectives. *In* Proceedings of 35th Association for Computational Linguistics (ACL) and 8th European Association for Computational Linguistics (EACL), Madrid, Spain.

IORDANSKAJA, L., R. KITTREDGET, and A. POLGUERE. 1991. Natural Language Generation in Artificial Intelligence and Computational Linguistics. Kluwer Academic: Nantes, France.

IZARD, C. E. 1971. The Face of Emotion. Appleton-Century-Crofts: New York.

KHAN, S. 2007. Negation and Antonymy in Sentiment Classification. Ph.D. Thesis, Trinity Hall College, University of Cambridge, Cambridge, UK.

LANGKILDE, I., and K. KNIGHT. 1998. Generation that exploits corpus-based statistical knowledge. *In* Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 1, Montreal, Canada, pp. 704–710.

LAWLER, J. 2007. Negation and negative polarity. *In* Encyclopedia of the Language Sciences. *Edited by* P. C. Hogan. Cambridge University Press: Cambridge, UK.

MADNANI, N., and B. J. DORR. 2010. Generating phrasal and sentential paraphrases: A survey of data-driven methods. computational linguistics. Computational Linguistics, **36**(3):341–387.

MELAMED, I. D. 2001. Empirical Methods for Exploiting Parallel Texts. MIT Press: Cambridge, MA.

MIHALCEA, R. 2004. Co-training and self-training for word sense disambiguation. *In* Proceedings of Natural Language Learning (CoNLL-2004), Boston, MA, pp. 33–40.

MILLER, G., R. BECKWITH, C. FELLBAUM, D. GROSS, and K. MILLER. 1993. Introduction to Wordnet: An On-Line Lexical Database. Cognitive Science Laboratory, Princeton University: Princeton, NJ.

MISHNE, G. 2005. Experiments with mood classification in blog posts. *In* Proceeding of ACM SIGIR 2005.

PANG, B., and L. LEE. 2008. Opinion minding and sentiment analysis. Foundation and Trend in Information Retrieval, **2**:1–2.

PANTEL, P., and M. PENNACCHIOTTI. 2006. Espresso: Leveraging generic patterns for automatically harvesting semantic relations. *In* Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the ACL, Sydney, Australia, pp. 113–120.

PEREIRA, F., N. TISHBY, and L. LEE. 1993. Distributional clustering of english words. *In* Proceedings of the 30th Annual Meeting of the ACL, Columbus, OH, pp. 183–190.

POLANYI, L., and A. ZAENEN. 2004. Contextual valence shifters. *In* Proceedings of AAAI Spring Symposium on Exploring Attitude and Affect in Text. AAAI Press: Menlo Park, CA.

RAZAVI, A. H., D. INKPEN, S. MATWIN, and S. URITSKY. 2010. Offensive language detection using multi-level classification. *In* Proceedings of the 23rd Canadian Conference on Artificial Intelligence (AI 2010), Ottawa, Canada, May 2010, pp. 16–27.

RILOFF, E., and R. JONES. 1999. Learning dictionaries for information extraction by multi-level boot-strapping. *In* Proceedings of the Sixteenth National Conference on Artificial Intelligence. The AAAI Press/MIT Press: Cambridge, MA, pp. 1044–1049.

RILOFF, E., and J. WIEBE. 2003. Learning extraction patterns for subjective expressions. *In* Natural Language Processing (EMNLP-2003), East Stroudsburg, PA, pp. 105–112.

SIEGEL, S., and J. CASTELLAN. 1988. Non Parametric Statistics for Behavioral Sciences. McGraw-Hill: Columbus, OH.

STRAPPARAVA, C., and R. MIHALCEA. 2007. Semeval-2007 task 14: Affective text. *In* Proceedings of the 4th International Workshop on the Semantic Evaluations (SemEval 2007), Prague, Czech Republic.

STRAPPARAVA, C., and A. VALITUTTI. 2004. Wordnet-affect: an affective extension of wordnet. *In* Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC 2004), Lisbon, Portugal, pp. 1083–1086.

TOUTANOVA, K., D. KLEIN, C. MANNING, and Y. SINGER. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. *In* Proceedings of HLT-NAACL 2003, Edmonton, Canada, pp. 252–259.

TURNEY, P. 2002. Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews. *In* Proceedings of ACL-2002, Marrakech, Morocco.

WANG, X., D. LO, J. JIANG, L. ZHANG, and H. MEI. 2009. Extracting paraphrases of technical terms from noisy parallel software corpora. *In* Proceedings of ACL IJCNLP 2009, Singapore, pp. 197–200.

YAROWSKY, D. 1995. Unsupervised word sense disambiguation rivaling supervised methods. *In* Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics, Cambridge, MA, pp. 189–196.

ZHAO, S., X. LAN, T. LIU, and S. LI. 2009. Application-driven statistical paraphrase generation. *In* Proceedings of ACL-IJCNLP 2009, Singapore, pp. 834–842.