Semi-Supervised Recognition of Sarcastic Sentences in Twitter and Amazon

Dmitry Davidov
ICNC
The Hebrew University
Jerusalem, Israel
dmitry@alice.nc.huji.ac.il

Oren Tsur
Institute of Computer Science
The Hebrew University
Jerusalem, Israel
oren@cs.huji.ac.il

Ari Rappoport
Institute of Computer Science
The Hebrew University
Jerusalem, Israel
arir@cs.huji.ac.il

Abstract

Sarcasm is a form of speech act in which the speakers convey their message in an implicit way. The inherently ambiguous nature of sarcasm sometimes makes it hard even for humans to decide whether an utterance is sarcastic or not. Recognition of sarcasm can benefit many sentiment analysis NLP applications, such as review summarization, dialogue systems and review ranking systems.

In this paper we experiment with semi-supervised sarcasm identification on two very different data sets: a collection of 5.9 million tweets collected from Twitter, and a collection of 66000 product reviews from Amazon. Using the Mechanical Turk we created a gold standard sample in which each sentence was tagged by 3 annotators, obtaining F-scores of 0.78 on the product reviews dataset and 0.83 on the Twitter dataset. We discuss the differences between the datasets and how the algorithm uses them (e.g., for the Amazon dataset the algorithm makes use of structured information). We also discuss the utility of Twitter #sarcasm hashtags for the task.

1 Introduction

Sarcasm (also known as *verbal irony*) is a sophisticated form of speech act in which the speakers convey their message in an implicit way. One inherent characteristic of the sarcastic speech act is that it is sometimes hard to recognize. The difficulty in recognition of sarcasm causes misunderstanding in everyday communication and poses problems to many NLP systems such as online review summarization systems, dialogue systems or brand monitoring systems due to the failure of state of the art sentiment analysis systems to detect sarcastic comments. In this paper we experiment with a semi-supervised framework for automatic identification of sarcastic sentences.

One definition for sarcasm is: *the activity of saying or writing the opposite of what you mean, or of speaking in a way intended to make someone else feel stupid or show them that you are angry* (Macmillan English Dictionary (2007)). Using the former definition, sarcastic utterances appear in many forms (Brown, 1980; Gibbs and O'Brien, 1991). It is best to present a number of examples which show different facets of the phenomenon, followed by a brief review of different aspects of the sarcastic use. The sentences are all taken from our experimental data sets:

1. *"thank you Janet Jackson for yet another year of Super Bowl classic rock!"* (Twitter)
2. *"He's with his other woman: XBox 360. It's 4:30 fool. Sure I can sleep through the gunfire"* (Twitter)
3. *"Wow GPRS data speeds are blazing fast."* (Twitter)
4. *"[I] Love The Cover"* (book, amazon)
5. *"Defective by design"* (music player, amazon)

Example (1) refers to the supposedly lame music performance in super bowl 2010 and attributes it to the aftermath of the scandalous performance of Janet Jackson in the previous year. Note that the previous year is not mentioned and the reader has to guess the context (use universal knowledge). The words *yet* and *another* might hint at sarcasm.

Example (2) is composed of three short sentences, each of them sarcastic on its own. However, combining them in one tweet brings the sarcasm to its extreme. Example (3) is a factual statement without explicit opinion. However, having a fast connection is a positive thing. A possible sarcasm emerges from the over exaggeration ('wow', '*blazing*-fast').

Example (4) from Amazon, might be a genuine compliment if it appears in the body of the review. However, recalling the expression 'don't judge a book by its cover', choosing it as the title of the review reveals its sarcastic nature. Although the negative sentiment is very explicit in the iPod review (5), the sarcastic effect emerges from the pun that assumes the knowledge that the design is one of the most celebrated features of Apple's products. (None of the above reasoning was directly introduced to our algorithm.)

Modeling the underlying patterns of sarcastic utterances is interesting from the psychological and cognitive perspectives and can benefit various NLP systems such as review summarization (Popescu and Etzioni, 2005; Pang and Lee, 2004; Wiebe et al., 2004; Hu and Liu, 2004) and dialogue systems. Following the 'brilliant-but-cruel' hypothesis (Danescu-Niculescu-Mizil et al., 2009), it can help improve ranking and recommendation systems (Tsur and Rappoport, 2009). All systems currently fail to correctly classify the sentiment of sarcastic sentences.

In this paper we utilize the semi-supervised sarcasm identification algorithm (SASI) of (Tsur et al., 2010). The algorithm employs two modules: semi supervised pattern acquisition for identifying sarcastic patterns that serve as features for a classifier, and a classification stage that classifies each sentence to a sarcastic class. We experiment with two radically different datasets: 5.9 million tweets collected from Twitter, and 66000 Amazon product reviews. Although for the Amazon dataset the algorithm utilizes structured information, results for the Twitter dataset are higher. We discuss the possible reasons for this, and also the utility of Twitter #sarcasm hashtags for the task. Our algorithm performed well in both domains, substantially outperforming a strong baseline based on semantic gap and user annotations. To further test its robustness we also trained the algorithm in a cross domain manner, achieving good results.

2 Data

The datasets we used are interesting in their own right for many applications. In addition, our algorithm utilizes some aspects that are unique to these datasets. Hence, before describing the algorithm, we describe the datasets in detail.

Twitter Dataset. Since Twitter is a relatively new service, a somewhat lengthy description of the medium and the data is appropriate.

Twitter is a very popular microblogging service. It allows users to publish and read short messages called *tweets* (also used as a verb: to tweet: the act of publishing on Twitter). The tweet length is restricted to 140 characters. A user who publishes a tweet is referred to as a *tweeter* and the readers are casual readers or *followers* if they are registered to get all tweets by this tweeter.

Apart from simple text, tweets may contain references to url addresses, references to other Twitter users (these appear as @<user>) or a content tag (called *hashtags*) assigned by the tweeter (#<tag>). An example of a tweet is: *"listening to Andrew Ridgley by Black Box Recorder on @Grooveshark: http://tinysong.com/cO6i #goodmusic"*, where 'grooveshark' is a Twitter user name and #goodmusic is a tag that allows to search for tweets with the same tag. Though frequently used, these types of meta tags are optional. In order to ignore specific references we substituted such occurrences with special tags: [LINK], [USER] and [HASHTAG] thus we have *"listening to Andrew Ridgley by Black Box Recorder on [USER]: [LINK] [HASHTAG]"*. It is important to mention that hashtags are not formal and each tweeter can define and use new tags as s/he likes.

The number of special tags in a tweet is only subject to the 140 characters constraint. There is no specific grammar that enforces the location of special tags within a tweet.

The informal nature of the medium and the 140 characters length constraint encourages massive use of slang, shortened lingo, ascii emoticons and other tokens absent from formal lexicons.

These characteristics make Twitter a fascinating domain for NLP applications, although posing great challenges due to the length constraint, the complete freedom of style and the out of discourse nature of tweets.

We used 5.9 million unique tweets in our dataset: the average number of words is 14.2

words per tweet, 18.7% contain a url, 35.3% contain reference to another tweeter and 6.9% contain at least one hashtag[1].

The #sarcasm hashtag One of the hashtags used by Twitter users is dedicated to indicate sarcastic tweets. An example of the use of the tag is: '*I guess you should expect a WONDERFUL video tomorrow. #sarcasm*'. The sarcastic hashtag is added by the *tweeter*. This hashtag is used infrequently as most users are not aware of it, hence, the majority of sarcastic tweets are not explicitly tagged by the tweeters. We use tagged tweets as a secondary gold standard. We discuss the use of this tag in Section 5.

Amazon dataset. We used the same dataset used by (Tsur et al., 2010), containing 66000 reviews for 120 products from Amazon.com. The corpus contained reviews for books from different genres and various electronic products. Amazon reviews are much longer than tweets (some reach 2000 words, average length is 953 characters), they are more structured and grammatical (good reviews are very structured) and they come in a known context of a specific product. Reviews are semi-structured as besides the body of the review they all have the following fields: writer, date, star rating (the overall satisfaction of the review writer) and a one line summary.

Reviews refer to a specific product and rarely address each other. Each review sentence is, therefore, part of a context – the specific product, the star rating, the summary and other sentences in that review. In that sense, sentences in the Amazon dataset differ radically from the contextless tweets. It is worth mentioning that the majority of reviews are on the very positive side (star rating average of 4.2 stars).

3 Classification Algorithm

Our algorithm is semi-supervised. The input is a relatively small seed of labeled sentences. The seed is annotated in a discrete range of $1 \dots 5$ where 5 indicates a clearly sarcastic sentence and 1 indicates a clear absence of sarcasm. A $1 \dots 5$ scale was used in order to allow some subjectivity and since some instances of sarcasm are more explicit than others.

[1]The Twitter data was generously provided to us by Brendan O'Connor.

Given the labeled sentences, we extracted a set of features to be used in feature vectors. Two basic feature types are utilized: syntactic and pattern-based features. We constructed feature vectors for each of the labeled examples in the training set and used them to build a classifier model and assign scores to unlabeled examples. We next provide a description of the algorithmic framework of (Tsur et al., 2010).

Data preprocessing A sarcastic utterance usually has a *target*. In the Amazon dataset these targets can be exploited by a computational algorithm, since each review targets a product, its manufacturer or one of its features, and these are explicitly represented or easily recognized. The Twitter dataset is totally unstructured and lacks textual context, so we did not attempt to identify targets.

Our algorithmic methodology is based on patterns. We could use patterns that include the targets identified in the Amazon dataset. However, in order to use less specific patterns, we automatically replace each appearance of a product, author, company, book name (Amazon) and user, url and hashtag (Twitter) with the corresponding generalized meta tags '[PRODUCT]','[COMPANY]','[TITLE]' and '[AUTHOR]' tags[2] and '[USER]','[LINK]' and '[HASHTAG]'. We also removed all HTML tags and special symbols from the review text.

Pattern extraction Our main feature type is based on surface patterns. In order to extract such patterns automatically, we followed the algorithm given in (Davidov and Rappoport, 2006). We classified words into high-frequency words (HFWs) and content words (CWs). A word whose corpus frequency is more (less) than F_H (F_C) is considered to be a HFW (CW). Unlike in (Davidov and Rappoport, 2006), we consider all punctuation characters as HFWs. We also consider [product], [company], [title], [author] tags as HFWs for pattern extraction. We define a pattern as an ordered sequence of high frequency words and slots for content words. The F_H and F_C thresholds were set to 1000 words per million (upper bound for F_C) and 100 words per million (lower bound for F_H)[3].

[2]Appropriate names are provided with each review so this replacement can be done automatically.

[3]Note that F_H and F_C set bounds that allow overlap between some HFWs and CWs.

The patterns allow 2-6 HFWs and 1-6 slots for CWs. For each sentence it is possible to generate dozens of patterns that may overlap. For example, given a sentence “Garmin apparently does not care much about product quality or customer support”, we have generated several patterns including “[COMPANY] CW does not CW much”, “does not CW much about CW CW or”, “not CW much” and “about CW CW or CW CW.”. Note that “[COMPANY]” and “.” are treated as high frequency words.

Pattern selection The pattern extraction stage provides us with hundreds of patterns. However, some of them are either too general or too specific. In order to reduce the feature space, we have used two criteria to select useful patterns.

First, we removed all patterns which appear only in sentences originating from a single product/book (Amazon). Such patterns are usually product-specific. Next we removed all patterns which appear in the seed both in some example labeled 5 (clearly sarcastic) and in some other example labeled 1 (obviously not sarcastic). This filters out frequent generic and uninformative patterns. Pattern selection was performed only on the Amazon dataset as it exploits review’s meta data.

Pattern matching Once patterns are selected, we have used each pattern to construct a single entry in the feature vectors. For each sentence we calculated a feature value for each pattern as follows:

$$\begin{cases} 1: & \text{Exact match – all the pattern components appear in the sentence in correct order without any additional words.} \\ \alpha: & \text{Sparse match – same as exact match but additional non-matching words can be inserted between pattern components.} \\ \gamma * n/N: & \text{Incomplete match – only } n > 1 \text{ of } N \text{ pattern components appear in the sentence, while some non-matching words can be inserted in-between. At least one of the appearing components should be a HFW.} \\ 0: & \text{No match – nothing or only a single pattern component appears in the sentence.} \end{cases}$$

$0 \leq \alpha \leq 1$ and $0 \leq \gamma \leq 1$ are parameters we use to assign reduced scores for imperfect matches. Since the patterns we use are relatively long, exact matches are uncommon, and taking advantage of partial matches allows us to significantly reduce the sparsity of the feature vectors. We used

$\alpha \backslash \gamma$	0.05	0.1	0.2
0.05	0.48	0.45	0.39
0.1	0.50	0.51	0.40
0.2	0.40	0.42	0.33

Table 1: Results (F-Score for ”no enrichment” mode) of cross validation with various values for α and γ on Twitter+Amazon data

$\alpha = \gamma = 0.1$ in all experiments. Table 1 demonstrates the results obtained with different values for α and γ.

Thus, for the sentence “Garmin apparently does not care much about product quality or customer support”, the value for “[company] CW does not” would be 1 (exact match); for “[company] CW not” would be 0.1 (sparse match due to insertion of ‘does’); and for “[company] CW CW does not” would be $0.1 * 4/5 = 0.08$ (incomplete match since the second CW is missing).

Punctuation-based features In addition to pattern-based features we used the following generic features: (1) Sentence length in words, (2) Number of “!” characters in the sentence, (3) Number of “?” characters in the sentence, (4) Number of quotes in the sentence, and (5) Number of capitalized/all capitals words in the sentence. All these features were normalized by dividing them by the (maximal observed value · averaged maximal value of the other feature groups), thus the maximal weight of each of these features is equal to the averaged weight of a single pattern/word/n-gram feature.

Data enrichment Since we start with only a small annotated seed for training (particularly, the number of clearly sarcastic sentences in the seed is modest) and since annotation is noisy and expensive, we would like to find more training examples without requiring additional annotation effort.

To achieve this, we posited that sarcastic sentences frequently co-appear in texts with other sarcastic sentences (i.e. example (2) in Section 1). We performed an automated web search using the Yahoo! BOSS API[4], where for each sentence s in the training set (seed), we composed a search engine query q_s containing this sentence[5]. We collected up to 50 search engine snippets for each example and added the sentences found in these snippets to the training set. The label (level of sar-

[4]http://developer.yahoo.com/search/boss.

[5]If the sentence contained more than 6 words, only the first 6 words were included in the search engine query.

casm) $Label(s_q)$ of a newly extracted sentence s_q is similar to the label $Label(s)$ of the seed sentence s that was used for the query that acquired it. The seed sentences together with newly acquired sentences constitute the (enriched) training set.

Data enrichment was performed only for the Amazon dataset where we have a manually tagged seed and the sentence structure is closer to standard English grammar. We refer the reader to (Tsur et al., 2010) for more details about the enrichment process and for a short discussion about the usefulness of web-based data enrichment in the scope of sarcasm recognition.

Classification In order to assign a score to new examples in the test set we use a k-nearest neighbors (kNN)-like strategy. We construct feature vectors for each example in the training and test sets. We would like to calculate the score for each example in the test set. For each feature vector v in the test set, we compute the Euclidean distance to each of the matching vectors in the extended training set, where matching vectors share at least one pattern feature with v.

Let $t_i, i = 1..k$ be the k vectors with lowest Euclidean distance to v[6]. Then v is classified with a label l as follows:

$$Count(l) = \text{Fraction of training vectors with label } l$$

$$Label(v) = \left[\frac{1}{k}\sum_i \frac{Count(Label(t_i)) \cdot Label(t_i)}{\sum_j Count(label(t_j))}\right]$$

Thus the score is a weighted average of the k closest training set vectors. If there are less than k matching vectors for the given example then fewer vectors are used in the computation. If there are no matching vectors found for v, we assigned the default value $Label(v) = 1$, since sarcastic sentences are fewer in number than non-sarcastic ones (this is a 'most common tag' strategy).

4 Evaluation Setup

Seed and extended training sets (Amazon). As described in the previous section, SASI is semi supervised, hence requires a small seed of annotated data. We used the same seed of 80 positive (sarcastic) examples and 505 negative examples described at (Tsur et al., 2010).

After automatically expanding the training set, our training data now contains 471 positive examples and 5020 negative examples. These ratios are to be expected, since non-sarcastic sentences outnumber sarcastic ones, definitely when most online reviews are positive (Liu et al., 2007). This generally positive tendency is also reflected in our data – the average number of stars is 4.12.

[6]We used $k = 5$ for all experiments.

Seed training set with #sarcasm (Twitter). We used a sample of 1500 tweets marked with the #sarcasm hashtag as a positive set that represents sarcasm styles special to Twitter. However, this set is very noisy (see discussion in Section 5).

Seed training set (cross domain). Results obtained by training on the 1500 #sarcasm hashtagged tweets were not promising. Examination of the #sarcasm tagged tweets shows that the annotation is biased and noisy as we discuss in length in Section 5. A better annotated set was needed in order to properly train the algorithm. Sarcastic tweets are sparse and hard to find and annotate manually. In order to overcome sparsity we used the positive seed annotated on the Amazon dataset. The training set was completed by manually selected negative example from the Twitter dataset. Note that in this setting our training set is thus of mixed domains.

4.1 Star-sentiment baseline

Many studies on sarcasm suggest that sarcasm emerges from the gap between the expected utterance and the actual utterance (see echoic mention, allusion and pretense theories in Related Work Section(6)). We implemented a baseline designed to capture the notion of sarcasm as reflected by these models, trying to meet the definition "saying the opposite of what you mean in a way intended to make someone else feel stupid or show you are angry".

We exploit the meta-data provided by Amazon, namely the star rating each reviewer is obliged to provide, in order to identify unhappy reviewers. From this set of negative reviews, our baseline classifies as sarcastic those sentences that exhibit strong positive sentiment. The list of positive sentiment words is predefined and captures words typically found in reviews (for example, 'great', 'excellent', 'best', 'top', 'exciting', etc).

4.2 Evaluation procedure

We used two experimental frameworks to test SASI's accuracy. In the first experiment we evaluated the pattern acquisition process, how consistent it is and to what extent it contributes to correct classification. We did that by 5-fold cross validation over the seed data.

In the second experiment we evaluated SASI on a test set of unseen sentences, comparing its output to a gold standard annotated by a large number of human annotators (using the Mechanical Turk). This way we verify that there is no over-fitting and that the algorithm is not biased by the notion of sarcasm of a single seed annotator.

5-fold cross validation (Amazon). In this experimental setting, the seed data was divided to 5 parts and a 5-fold cross validation test is executed. Each time, we use 4 parts of the seed as the training data and only this part is used for the feature selection and data enrichment. This 5-fold process was repeated ten times. This procedure was repeated with different sets of optional features.

We used 5-fold cross validation and not the standard 10-fold since the number of seed examples (especially positive) is relatively small hence 10-fold is too sensitive to the broad range of possible sarcastic patterns (see the examples in Section 1).

Classifying new sentences (Amazon & Twitter). Evaluation of sarcasm is a hard task due to the elusive nature of sarcasm, as discussed in Section 1. In order to evaluate the quality of our algorithm, we used SASI to classify *all* sentences in both corpora (besides the small seed that was pre-annotated and was used for the evaluation in the 5-fold cross validation experiment). Since it is impossible to created a gold standard classification of each and every sentence in the corpus, we created a small test set by sampling 90 sentences which were classified as sarcastic (labels 3-5) and 90 sentences classified as not sarcastic (labels 1,2). The sampling was performed on the whole corpus leaving out only the seed data.

Again, the meta data available in the Amazon dataset allows us a stricter evaluation. In order to make the evaluation harder for our algorithm and more relevant, we introduced two constraints to the sampling process: *i*) we sampled only sentences containing a named-entity or a reference to a named entity. This constraint was introduced in order to keep the evaluation set relevant, since sentences that refer to the named entity (the target of the review) are more likely to contain an explicit or implicit sentiment. *ii*) we restricted the non-sarcastic sentences to belong to negative reviews (1-3 stars) so that all sentences in the evaluation set are drawn from the same population, increasing the chances they convey various levels of direct or indirect *negative* sentiment[7].

Experimenting with the Twitter dataset, we simply classified each tweet into one of 5 classes (class 1: not sarcastic, class 5: clearly sarcastic) according to the label given by the algorithm. Just like the evaluation of the algorithm on the Amazon dataset, we created a small evaluation set by sampling 90 sentences which were classified as sarcastic (labels 3-5) and 90 sentences classified as not sarcastic (labels 1,2).

Procedure Each evaluation set was randomly divided to 5 batches. Each batch contained 36 sentences from the evaluation set and 4 anchor sentences: two with sarcasm and two sheer neutral. The anchor sentences were not part of the test set and were the same in all five batches. The purpose of the anchor sentences is to control the evaluation procedure and verify that annotation is reasonable. We ignored the anchor sentences when assessing the algorithm's accuracy.

We used Amazon's Mechanical Turk[8] service in order to create a gold standard for the evaluation. We employed 15 annotators for each evaluation set. We used a relatively large number of annotators in order to overcome the possible bias induced by subjectivity (Muecke, 1982). Each annotator was asked to assess the level of sarcasm of each sentence of a set of 40 sentences on a scale of 1-5. In total, each sentence was annotated by three different annotators.

Inter Annotator Agreement. To simplify the assessment of inter-annotator agreement, the scaling was reduced to a binary classification where 1 and 2 were marked as non-sarcastic and 3-5 as sarcastic (recall that 3 indicates a hint of sarcasm and 5 indicates 'clearly sarcastic'). We checked the Fleiss' κ statistic to measure agreement between multiple annotators. The inter-annotator agreement statistic was $\kappa = 0.34$ on the Amazon dataset and $\kappa = 0.41$ on the Twitter dataset.

These agreement statistics indicates a fair agreement. Given the fuzzy nature of the task at

[7]Note that the second constraint makes the problem less easy. If taken from all reviews, many of the sentences would be positive sentences which are clearly non-sarcastic. Doing this would bias selection to positive vs. negative samples instead of sarcastic-nonsarcastic samples.

[8]https://www.mturk.com/mturk/welcome

	Prec.	Recall	Accuracy	F-score
punctuation	0.256	0.312	0.821	0.281
patterns	0.743	**0.788**	0.943	0.765
pat+punct	0.868	0.763	0.945	0.812
enrich punct	0.4	0.390	0.832	0.395
enrich pat	0.762	0.777	0.937	0.769
all: SASI	**0.912**	0.756	**0.947**	**0.827**

Table 2: 5-fold cross validation results on the Amazon gold standard using various feature types. *punctuation*: punctuation mark;, *patterns*: patterns; *enrich*: after data enrichment; *enrich punct*: data enrichment based on punctuation only; *enrich pat*: data enrichment based on patterns only; SASI: all features combined.

hand, this κ value is certainly satisfactory. We attribute the better agreement on the twitter data to the fact that in twitter each sentence (tweet) is context free, hence the sentiment in the sentence is expressed in a way that can be perceived more easily. Sentences from product reviews come as part of a full review, hence the the sarcasm sometimes relies on other sentences in the review. In our evaluation scheme, our annotators were presented with individual sentences, making the agreement lower for those sentences taken out of their original context. The agreement on the control set (anchor sentences) had $\kappa = 0.53$.

Using Twitter #sarcasm hashtag. In addition to the gold standard annotated using the Mechanical Turk, we collected 1500 tweets that were tagged #sarcastic by their tweeters. We call this sample the *hash*-gold standard. It was used to further evaluate recall. This set (along with the negative sample) was used for a 5-fold cross validation in the same manner describe for Amazon.

5 Results and discussion

5-fold cross validation (Amazon). Results are analyzed and discussed in detail in (Tsur et al., 2010), however, we summarize it here (Table 2) in order to facilitate comparison with the results obtained on the Twitter dataset. SASI, including all components, exhibits the best overall performances with 91.2% precision and with F-Score of 0.827. Interestingly, although data enrichment brings SASI to the best performance in both precision and F-score, patterns+punctuations achieves almost comparable results.

Newly introduced sentences (Amazon). In the second experiment we evaluated SASI based on a gold standard annotation created by 15 annotators. Table 3 presents the results of our algorithm as well as results of the heuristic baseline that makes

	Prec.	Recall	**FalsePos**	**FalseNeg**	**F Score**
Star-sent.	0.5	0.16	0.05	0.44	0.242
SASI (AM)	0.766	0.813	0.11	0.12	0.788
SASI (TW)	0.794	0.863	0.094	0.15	0.827

Table 3: Evaluation on the Amazon (AM) and the Twitter (TW) evaluation sets obtained by averaging on 3 human annotations per sentence. TW results were obtained with cross-domain training.

	Prec.	Recall	Accuracy	F-score
punctuation	0.259	0.26	0.788	0.259
patterns	0.765	0.326	0.889	0.457
enrich punct	0.18	0.316	0.76	0.236
enrich pat	0.685	0.356	0.885	0.47
all no enrich	**0.798**	0.37	**0.906**	0.505
all SASI:	0.727	**0.436**	0.896	**0.545**

Table 4: 5-fold cross validation results on the Twitter *hash*-gold standard using various feature types. *punctuation*: punctuation marks; *patterns*: patterns; *enrich*: after data enrichment; *enrich punct*: data enrichment based on punctuation only; *enrich pat*: data enrichment based on patterns only; SASI: all features combined.

use of meta-data, designed to capture the gap between an explicit negative sentiment (reflected by the review's star rating) and explicit positive sentiment words used in the review. Precision of SASI is 0.766, a significant improvement over the baseline with precision of 0.5.

The F-score shows more impressive improvement as the baseline shows decent precision but a very limited recall since it is incapable of recognizing subtle sarcastic sentences. These results fit the works of (Brown, 1980; Gibbs and O'Brien, 1991) claiming many sarcastic utterances do not conform to the popular definition of "saying or writing the opposite of what you mean". Table 3 also presents the false positive and false negative ratios. The low false negative ratio of the baseline confirms that while recognizing a common type of sarcasm, the naive definition of sarcasm cannot capture many other types sarcasm.

Newly introduced sentences (Twitter). Results on the Twitter dataset are even better than those obtained on the Amazon dataset, with accuracy of 0.947 (see Table 3 for precision and recall).

Tweets are less structured and are context free, hence one would expect SASI to perform poorly on tweets. Moreover, the positive part of the seed is taken from the Amazon corpus hence might seem tailored to sarcasm type targeted at products and part of a harsh review. On top of that, the positive seed introduces some patterns with tags that never occur in the Twitter test set ([product/company/title/author]).

Our explanation of the excellent results is threefold: *i*) SASI's robustness is achieved by the sparse match (α) and incomplete match (γ) that tolerate imperfect pattern matching and enable the use of variations of the patterns in the learned feature vector. α and γ allow the introduction of patterns with components that are absent from the positive seed, and can perform even with patterns that contain special tags that are not part of the test set. *ii*) SASI learns a model which spans a feature space with more than 300 dimensions. Only part of the patterns consist of meta tags that are special to product reviews, the rest are strong enough to capture the structure of general sarcastic sentences and not product-specific sarcastic sentences only. *iii*) Finally, in many cases, it might be that the contextless nature of Twitter forces tweeters to express sarcasm in a way that is easy to understand from individual sentence. Amazon sentences co-appear with other sentences (in the same review) thus the sarcastic meaning emerges from the context. Our evaluation scheme presents the annotators with single sentences therefore Amazon sentences might be harder to agree on.

***hash* gold standard (Twitter).** In order to further test out algorithm we built a model consisting of the positive sample of the Amazon training, the #sarcasm hash-tagged tweets and a sample of non sarcastic tweets as the negative training set. We evaluated it in a 5-fold cross validation manner (only against the *hash*-gold standard). While precision is still high with 0.727, recall drops to 0.436 and the F-Score is 0.545.

Looking at the *hash*-gold standard set, we observed three main uses for the #sarcasm hashtag. Differences between the various uses can explain the relatively low recall. *i*) The tag is used as a search anchor. Tweeters add the hashtag to tweets in order to make them retrievable when searching for the tag. *ii*) The tag is often abused and added to non sarcastic tweets, typically to clarify that a previous tweet should have been read sarcastically, e.g.: *"@wrightfan05 it was #Sarcasm "*. *iii*) The tag serves as a sarcasm marker in cases of a very subtle sarcasm where the lack of context, the 140 length constraint and the sentence structure make it impossible to get the sarcasm without the explicit marker. Typical examples are: *"#sarcasm not at all."* or *"can't wait to get home tonite #sarcasm."*, which cannot be decided sarcastic without the full context or the #sarcasm marker.

These three observations suggest that the hash-gold standard is noisy (containing non-sarcastic tweets) and is biased toward the hardest (inseparable) forms of sarcasm where even humans get it wrong without an explicit indication. Given the noise and the bias, the recall is not as bad as the raw numbers suggest and is actually in synch with the results obtained on the Mechanical Turk human-annotated gold standard. Table 4 presents detailed results and the contribution of each type of feature to the classification.

We note that the relative sparseness of sarcastic utterances in everyday communication as well as in these two datasets make it hard to accurately estimate the recall value over these huge unannotated data sets. Our experiment, however, indicates that we achieve reasonable recall rates.

Punctuation Surprisingly, punctuation marks serve as the weakest predictors, in contrast to Teppermann et al. (2006). An exception is three consecutive dots, which when combine with other features constitute a strong predictor. Interestingly though, while in the cross validation experiments SASI performance varies greatly (due to the problematic use of the *#sarcasm* hashtag, described previously), performance based only on punctuation are similar (Table 2 and Table 4).

Tsur et al. (2010) presents some additional examples for the contribution of each type of feature and their combinations.

6 Related Work

While the use of irony and sarcasm is well studied from its linguistic and psychologic aspects (Muecke, 1982; Stingfellow, 1994; Gibbs and Colston, 2007), automatic recognition of sarcasm is a novel task, addressed only by few works. In the context of opinion mining, sarcasm is mentioned briefly as a hard nut that is yet to be cracked, see comprehensive overview by (Pang and Lee, 2008).

Tepperman et al. (2006) identify sarcasm in spoken dialogue systems, their work is restricted to sarcastic utterances that contain the expression 'yeah-right' and it depends heavily on cues in the spoken dialogue such as laughter, pauses within the speech stream, the gender (recognized by voice) of the speaker and prosodic features.

Burfoot and Baldwin (2009) use SVM to determine whether newswire articles are true or satirical. They introduce the notion of *validity* which models absurdity via a measure somewhat close to

PMI. Validity is relatively lower when a sentence includes a made-up entity or when a sentence contains unusual combinations of named entities such as, for example, those in the satirical article beginning "Missing Brazilian balloonist Padre spotted straddling Pink Floyd flying pig". We note that while sarcasm can be based on exaggeration or unusual collocations, this model covers only a limited subset of the sarcastic utterances.

Tsur et al. (2010) propose a semi supervised framework for recognition of sarcasm. The proposed algorithm utilizes some features specific to (Amazon) product reviews. This paper continues this line, proposing SASI a robust algorithm that successfully captures sarcastic sentences in other, radically different, domains such as twitter.

Utsumi (1996; 2000) introduces the *implicit display* theory, a cognitive computational framework that models the *ironic environment*. The complex axiomatic system depends heavily on complex formalism representing world knowledge. While comprehensive, it is currently impractical to implement on a large scale or for an open domain.

Mihalcea and Strapparava (2005) and Mihalcea and Pulman (2007) present a system that identifies humorous one-liners. They classify sentences using naive Bayes and SVM. They conclude that the most frequently observed semantic features are negative polarity and human-centeredness. These features are also observed in some sarcastic utterances.

Some philosophical, psychological and linguistic theories of irony and sarcasm are worth referencing as a theoretical framework: the *constraints satisfaction* theory (Utsumi, 1996; Katz, 2005), the *role playing* theory (Clark and Gerrig, 1984), the *echoic mention* framework (Wilson and Sperber, 1992) and the *pretence* framework (Gibbs, 1986). These are all based on violation of the maxims proposed by Grice (1975).

7 Conclusion

We used SASI, the first robust algorithm for recognition of sarcasm, to experiment with a novel Twitter dataset and compare performance with an Amazon product reviews dataset. Evaluating in various ways and with different parameters configurations, we achieved high precision, recall and F-Score on both datasets even for cross-domain training and with no need for domain adaptation.

In the future we will test the contribution of sarcasm recognition for review ranking and summarization systems and for brand monitoring systems.

References

R. L. Brown. 1980. The pragmatics of verbal irony. In R. W. Shuy and A. Snukal, editors, *Language use and the uses of language*, pages 111–127. Georgetown University Press.

Clint Burfoot and Timothy Baldwin. 2009. Automatic satire detection: Are you having a laugh? In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 161–164, Suntec, Singapore, August. Association for Computational Linguistics.

H. Clark and R. Gerrig. 1984. On the pretence theory of irony. *Journal of Experimental Psychology: General*, 113:121–126.

Cristian Danescu-Niculescu-Mizil, Gueorgi Kossinets, Jon Kleinberg, and Lillian Lee. 2009. How opinions are received by online communities: A case study on amazon.com helpfulness votes. Jun.

D. Davidov and A. Rappoport. 2006. Efficient unsupervised discovery of word categories using symmetric patterns and high frequency words. In *COLING-ACL*.

Macmillan English Dictionary. 2007. *Macmillan English Dictionary*. Macmillan Education, 2 edition.

Raymond W Gibbs and Herbert L. Colston, editors. 2007. *Irony in Language and Thought*. Routledge (Taylor and Francis), New York.

R. W. Gibbs and J. E. O'Brien. 1991. Psychological aspects of irony understanding. *Journal of Pragmatics*, 16:523–530.

R. Gibbs. 1986. On the psycholinguistics of sarcasm. *Journal of Experimental Psychology: General*, 105:3–15.

H. P. Grice. 1975. Logic and conversation. In Peter Cole and Jerry L. Morgan, editors, *Syntax and semantics*, volume 3. New York: Academic Press.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *KDD '04: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177, New York, NY, USA. ACM.

A. Katz. 2005. Discourse and social-cultural factors in understanding non literal language. In Colston H. and Katz A., editors, *Figurative language comprehension: Social and cultural influences*, pages 183–208. Lawrence Erlbaum Associates.

Jingjing Liu, Yunbo Cao, Chin-Yew Lin, Yalou Huang, and Ming Zhou. 2007. Low-quality product review detection in opinion summarization. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 334–342.

Rada Mihalcea and Stephen G. Pulman. 2007. Characterizing humour: An exploration of features in humorous texts. In *CICLing*, pages 337–347.

Rada Mihalcea and Carlo Strapparava. 2005. Making computers laugh: Investigations in automatic humor recognition. pages 531–538, Vancouver, Canada.

D.C. Muecke. 1982. *Irony and the ironic*. Methuen, London, New York.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the ACL*, pages 271–278.

Bo Pang and Lillian Lee. 2008. *Opinion Mining and Sentiment Analysis*. Now Publishers Inc, July.

Ana-Maria Popescu and Oren Etzioni. 2005. Extracting product features and opinions from reviews. In *HLT '05: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 339–346, Morristown, NJ, USA. Association for Computational Linguistics.

Frank Jr. Stingfellow. 1994. *The Meaning of Irony*. State University of NY, New York.

J. Tepperman, D. Traum, and S. Narayanan. 2006. Yeah right: Sarcasm recognition for spoken dialogue systems. In *InterSpeech ICSLP*, Pittsburgh, PA.

Oren Tsur and Ari Rappoport. 2009. Revrank: A fully unsupervised algorithm for selecting the most helpful book reviews. In *International AAAI Conference on Weblogs and Social Media*.

Oren Tsur, Dmitry Davidiv, and Ari Rappoport. 2010. Icwsm – a great catchy name: Semi-supervised recognition of sarcastic sentences in product reviews. In *International AAAI Conference on Weblogs and Social Media*.

Akira Utsumi. 1996. A unified theory of irony and its computational formalization. In *COLING*, pages 962–967.

Akira Utsumi. 2000. Verbal irony as implicit display of ironic environment: Distinguishing ironic utterances from nonirony. *Journal of Pragmatics*, 32(12):1777–1806.

Janyce Wiebe, Theresa Wilson, Rebecca Bruce, Matthew Bell, and Melanie Martin. 2004. Learning subjective language. *Computational Linguistics*, 30(3):277– 308, January.

D. Wilson and D. Sperber. 1992. On verbal irony. *Lingua*, 87:53–76.