Detecting Insults in Online Comments

Poorna Krishnamoorthy
Stanford University
poorna@stanford.edu

Rory MacQueen
Stanford University
macqueen@stanford.edu

Sebastian Schuster
Stanford University
sebschu@stanford.edu

Abstract

Abusive or offensive comments on discussion forums irritate users and stifle fruitful discussions. Moderating forums by hand is cumbersome and inefficient, thereby creating a need for an automatic insult detection system. In this paper we propose a system which combines an SVM classifier with hand-written deterministic rules. Features for our classifier are based on N-gram word models and on the syntactic structure common to most insults. On a test set of 2235 hand-labeled comments, we were able to achieve an F1 score of 0.710. We discuss how our error stems from both the irregular structure of insults and from inherent problems with sentiment analysis in the NLU field.

1 Introduction

The high prevalence of abusive and insulting language on the Internet contributes to its reputation of being an "anything goes" environment. While some people may be tolerant or even appreciative of the unfiltered nature of online commentary, studies show that such language can skew people's perception of products or ideas online (Anderson et al., 2013), and can even undermine the trust of certain websites. Since it has become time consuming and expensive for humans to moderate discussion forums manually, a pressing need has arisen for a system which could automatically detect and flag potentially abusive comments.

Since natural language is inherently subjective (and abusive language perhaps even more so), a precise definition of an insult is required. We define an insult to be an intentionally abusive remark directed at someone else on the forum. The phrase "George Bush is an asshole", while it may be considered offensive to some of his ardent supporters, is not here considered an insult since it is directed towards a third-party and not towards someone else on the forum itself. However, a comment such as "You are a complete idiot" would be considered an insult, since the "you" to which this comment refers is likely someone else on the forum.

In addition to the well-known challenges which confront any NLP task (word sense disambiguation, natural language inference), the problem of automatic insult detection has its own difficulties. Abusive language tends to have a disproportionate amount of spelling or grammatical errors, and also tends to use non-standard ASCII characters and symbols in place of alphabetical characters. These quiddities can make words difficult for a machine to recognize, even if those same words can still be interpreted by a human (e.g. "you a$$hole"). Insults can also depend heavily on world knowledge: to understand that the sentence "you should be like all the other lemmings and jump off of a cliff" is insulting requires knowing both that lemmings are generally considered to be stupid animals and that "jumping off of a cliff" entails one's own death. Perhaps the most challenging aspect of insult detection, and one that reveals the shortcomings of NLU, is the use of sarcasm in language. In the context of an online discussion forum, the phrase "your mother must be so proud of you" is much more likely to be a sarcasm-dependent insult than it is to be a genuine statement of praise. Yet NLU machines currently have no way

of drawing this distinction.

1.1 Applications

The most obvious application of the type of classifier discussed here would be its use in automatic moderation of discussion forums. New comments which our classifier flagged as insults could be forwarded to a human moderator, who could then make the final decision about whether to allow the comment to be posted. Presumably, the vast majority of comments in any given forum are *not* insults, and as a result, our classifier is dealing with a dataset that is heavily skewed towards one of the two classes. However, if even one of the insulting messages manages to slip through the cracks, an entire discussion thread can be ruined. For both these reasons, we chose to heavily prioritize recall over precision as an evaluation metric. False positives are not such a bad thing in our scenario, since all comments that have been flagged will still go through a human moderator. But having just a few false negatives can make use of our system entirely pointless, since its purpose is to keep the discussion forum clean. The analogy to airport security helps here: a bomb detection system should be much less tolerant of false negatives than it should be of false positives.

2 Previous Work

Spertus et al. (Spertus, 1997) were among the first to conduct research on the topic of automatic insult detection. They use a syntactic parser and hand-written rules to generate a feature vector for each sentence, which is then given to a decision tree model to make a prediction. The hand-written rules tried to exploit syntactic structures common to insults, e.g. the use of the second person pronoun followed by a noun phrase ("You idiot!"), or the the appearance of certain imperative statements at the start of a sentence ("Go to hell!"). Such an approach can run the risk of being overly aggressive in its classification - the imperative rule, for example, would also pick up on compliments: "Keep up the good work", or "Have a nice day".

2.1 Machine Learning

More recent work on the topic has used machine learning classifiers to make predictions (Amir Hossein Razavi, 2010; Xiang et al., 2012). Features for these classifiers are often based on the appearance and frequency of words drawn from some predefined 'abusive language' dictionary (Amir Hossein Razavi, 2010). Since a dictionary alone is often not exhaustive enough to capture all insults, some papers instead used a bag-of-words model to capture the discriminative power of each word in a corpus (Pang et al., 2002). Xiang et al. (2012) used a bootstrapping approach to identify Twitter users who tended to tweet offensive comments, and then applied an LDA model to learn topics from those offensive tweets.

2.2 Sub-Problems

Some studies have tried to focus on specific subproblems in the insult detection domain, such as trying to detect just sarcasm (Davidov et al., 2010), or trying to detect insults that are direct towards a specific ethnic minority group (Warner and Hirschberg, 2012). Techniques from these research areas can be aggregated for the general problem of insult detection. To tackle the problem of misspellings, Sood et al. (2012) computed the Levensthein edit distance between each target word and each word in the abusive language dictionary. If a given word had a sufficiently low edit distance, and also did not appear in a dictionary of English words, then it could be tagged as profanity.

2.3 Sentiment Analysis

Since the problem of insult detection can itself be viewed as a special case of sentiment analysis, many techniques from sentiment analysis are directly applicable to this field. Pang and Lee (2002) have tried to amplify the performance of sentiment analysis models by applying them only to the subjective portions of a given document. They extract the subjective portions by arranging all sentences into a graph and then finding (in polynomial time) a minimum-cost cut through the graph which will divide the objective and subjective sentences. By applying a standard sentiment classifier (Naive Bayes) to just the subjective portions of text, accuracy improves from 82% to 86%.

2.4 Previous Work on the Dataset

To the best of our knowledge, no academic work has been done so far on the dataset we use. However,

as the dataset was part of a past machine learning challenge, there are models available that have been used on the dataset. The best-performing system[1] used only N-grams and a small dictionary of profane words as features for an ensemble classifier consisting of an SVM and a MaxEnt classifier.

3 Data

We worked with a dataset from a past Kaggle competition on detecting insults in online commentary[2]. The samples were drawn from a wide variety of conversation streams, such as the comments section of various news sites, magazines, and message boards. The corpus contains 8832 comments labeled with a 0 or 1, with 1 representing an insulting comment and 0 representing a neutral comment. In addition, the dataset also has the time at which the comment was made using a 24 hour clock scale. The comments range in length from just 1 word to a max of 180 words per sentence, with the average being 37.4 words per comment. The number of characters in the comment ranges from as little as 4 characters to a maximum of 6061 characters, with the average of 180 characters per comment. The comments tend to be interspersed with non-standard spelling and grammar - for e.g. "you are an a$$hole", "your full of chit". As previously mentioned, for the purposes of this task, a comment is only marked insulting if it is directed to a participant in the blog or forum; comments directed to non-participants such as celebrities are not considered to be insults. To illustrate the kinds of comments that are posted, below we offer an excerpt from the training data below.

Insulting Comments
- Even as a troll you are a pathetic failure.
- Howe does it feel to be a Freedom Leech parasite?
- You are a child
- No people like you are the problem we are having in this world
- Wow way to tell us how rich you are...stupid yuppie

Neutral Comments
- Conservatives are social leeches and the scum of the earth.
- Somebody owes you a refund.
- Holy shit, you're right!
- Are you in France?
- You are not alone!

We used the 3 pre-defined splits of the dataset - a training corpus to train our classifier, a development set for testing during development and performing error analysis, and a held out test set to perform a final evaluation of our system. The distribution of insults vs. neutral comments in these three sets is shown in Table 1. In order to assess how well our system performs on real-world data and across domains, we also scraped 984 comments from a potentially controversial video on YouTube on gay marriage[3]. We hand-annotated the comments to be either insulting or neutral. 8.53% of the comments in this dataset were labeled as insulting.

Set	% Insults	Dataset Size
train	26.57%	3947
dev-test	26.18%	2647
test	48.18%	2235

Table 1: Corpus Statistics.

4 Methodology

Our algorithm is semi-supervised. Given the labeled comments in our training set, we extract a set of features to be used in feature vectors. We constructed feature vectors for each of the labeled examples in our training set and used them to build a classifier model.

4.1 Data Preprocessing

We do a small amount of preprocessing to eliminate url links, html tags such as tags, html character entities such as   and special non-breakable space character sequences such as \xc2 \xa0, which appeared with a fair amount of regularity in the text. We also normalize words that were lengthened by character repetition to their shortened

[1] https://www.kaggle.com/c/detecting-insults-in-social-commentary/forums/t/2744/what-did-you-use/15951#post15951

[2] http://www.kaggle.com/c/detecting-insults-in-social-commentary/data

[3] http://www.youtube.com/watch?v=X-YCdcnf_P8

form, e.g. waaaaaaaahhh! to wah!. Curse words tend to be masked by using non-alphabetic characters ($#) in place of swears, e.g f*** or t@rd. Some commonly interchanged non-alphabetic sequences in words include using @ for 'a' and $ for 's' - which we replace by the letters a and s. Further, we replace user handles starting with '@' symbols, which usually indicate that a user directs his or her comment to another poster by a special tag.

In addition to pre-processing, we tokenize our comments using the sentiment aware tokenization method by Christopher Potts [4]. Compared to using a Tree bank tokenizer, this allows us to preserve as much sentiment information as possible such as smilies or the excessive use of punctuation.

4.2 Insulting and Abusive Words Dictionary

Some of our features use a list containing insulting and abusive words. For this reason we compiled a dictionary of 478 commonly used abusive and profane words which we collected from an online source[5]. We gradually enriched the list based on observations in our training and development data. The list also includes deliberately misspelled or masked curses employed by people to avoid detection.

4.3 Features

4.3.1 N-gram Feature

Since N-gram features are the most commonly used in topic based text classification, we use unigrams and bigrams as word features. Instead of just counting the occurrences of each word, we use the term frequency-inverse document frequency (tf-idf) as the value for each feature. Tf-idf is useful in measuring how important a word is to a comment in a corpus, by measuring the frequency of the word in a comment while at the same time discounting very frequent words in the corpus. The collection of all comments in our training set serves as the corpus for computing the tf-idf. We use the top 10,000 word unigrams and bigrams which we select by performing a χ^2 feature selection test on the training set to pick the most discriminating features. In addition to word unigrams and bigrams, we also use character unigrams and character bigrams.

[4] http://sentiment.christopherpotts.net/

[5] http://urbanoalvarez.es/blog/2008/04/04/bad-words-list/

4.3.2 Word Pair Feature

We hypothesized that close proximity of insulting words from our insulting and abusive words list and pronouns such as "you" are a clear indicator of insult, because insulting words are usually directed at the intended target. This feature assigns a boolean value of 1 if insulting words are used in the comment in conjunction with pronouns such as "you're", "you", "yourself", etc. In the absence of proximity between pronoun and insulting words, the feature is assigned a value of 0. For example, the following examples illustrate cases where this feature would be useful:

"Coolest president ever? Your a complete waste of oxygen and resources. Phucking idiot"

"More grammatical errors, Vicky<2. You're on a roll today. You are so stupid..."

4.3.3 Capitalization

In online discussions, excessive capitalization, such as writing the entire comment in upper-case is considered equivalent to shouting. Some insulting comments tend to have excessive capitalization, and we decided to incorporate the number of words in a comment that have all-caps as a feature value.
Some examples are :

"YOU ARE A RETARD"

"you're so FUCKING DUMB!"

4.4 Classifier

Because this is a binary classification problem, we use a SVM classifier with a linear kernel as our core classifier. The SVM classifier is trained on our preprocessed training set using the features described above. To classify new comments, we preprocess the comment, compute its feature vector and input this vector to our SVM classifier which returns a probabilistic score reflecting how likely it is that the comment is insulting or not.

4.5 Deterministic rules

To further improve recall, we also implemented a hybrid classifier that augments the predictions of the SVM classifier with a few hand crafted deterministic rules that come into play in edge cases, i.e. where the prediction of the SVM is slightly below the threshold, to make a final prediction of whether a comment is insulting or not. In particular, one such

System	Precision	Recall	F1
Kaggle Winner	**0.814**	0.649	0.722
SVM	0.812	0.680	0.740
SVM+Rules	0.780	0.714	**0.745**
SVM+Sentences	0.789	0.693	0.738
SVM+Rules+Sent.	0.759	**0.719**	0.738

Table 2: Results on development dataset.

System	Precision	Recall	F1
Kaggle Winner	**0.854**	0.558	0.675
SVM	0.822	0.561	0.667
SVM+Rules	0.804	0.608	0.692
SVM+Sentences	0.815	0.599	0.691
SVM+Rules+Sent.	0.802	**0.638**	**0.710**

Table 3: Results on test dataset.

rule says that when the SVM returns a prediction below the confidence threshold but greater than 0.3, the comment is classified as insulting if it begins with a second person pronoun variant such as "you are" , "your", "u r", "ur", "you're" or starts with "go" or "get". Similarly, an SVM prediction greater than 0.4, combined with the presence of an abusive word from our dictionary, would also trigger a rule to classify the sentence as an insult.

4.6 Sentence-Level Classification

In general, a comment can be considered insulting if even one sentence in a long comment composed of several sentences has an abusive intent. For this reason we implemented the following sentence-level classifier. Instead of computing the feature vector for the complete comment, we split the comments at a sentence-level, compute the features for each sentence and classify each sentence separately. Our classifier then assigns each sentence a probability for being insulting or not. In order to classify the comment, we then take the maximum probability of all the sentences within a comment. However, this introduces a new problem. Just by chance, it is likely that one sentence within a comment of many sentences has an insult probability above our threshold, despite the fact that the comment as a whole is not insulting. To counter this bias, we also compute the mean probability over all sentences in a comment. In case this mean is very low (below 0.2) we assign the mean probability to the comment instead of the maximum probability. Thus the comment will be classified as neutral.

5 Results and Discussion

We evaluated our system using the precision, recall and F1 measures on our development and test dataset. We compared our results to those of the winning system in the Kaggle competition. In total we evaluated four different systems:

- A classifier solely based on the features described above and the prediction of a SVM (SVM)
- A classifier using the SVM predictions and the hand-written rules (SVM+Rules)
- A classifier using the SVM predictions of each sentence in the comment (SVM+Sentence)
- A classifier using the SVM predictions of each sentence in the comment and the hand-written rules (SVM+Rules+Sentence).

The results for each dataset are presented in Tables 2, 3 and 4. As we can see our system that uses the prediction from the SVM on the sentences combined with the deterministic rules has on all datasets the highest recall. While the winning system in the Kaggle competition has a higher precision on the Kaggle datasets, our system seems to generalize better as we have a statistically significant (using a McNemar test) higher precision and recall on the YouTube dataset. Thus in terms of maximizing recall our system using rules and classifying on a sentence level performs better. Otherwise the SVM+Sentences system seems to give the best trade-off between precision and recall, especially on more real-world data such as the YouTube comments.

As already discussed in the introduction, a system with a very high recall and a reasonably high precision could take away a lot of work from human moderators as they would only have to check comments that are potential insults. Further, one has to keep in mind that the majority of comments are usually not insulting. Only around 8% of the YouTube

System	Precision	Recall	F1
Kaggle Winner	0.513	0.464	0.488
SVM	0.512	0.500	0.506
SVM+Rules	0.475	0.560	0.514
SVM+Sentences	**0.543**	0.679	**0.603**
SVM+Rules+Sent.	0.487	**0.690**	0.571

Table 4: Results on YouTube dataset.

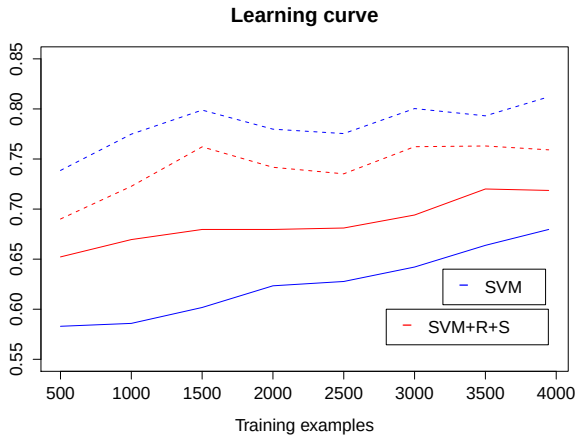

Figure 1: Learning curves for SVM (blue) and SVM+Rules+Sentences (red) systems. The dotted lines show precision and the solid lines show recall. Precision and recall were measured on the development set.

comments on a video discussing a very controversial topic were insults. Thus even with a relatively low precision of around 50 % a human moderator would only need to check 16% of all comments.

The learning curve in Figure 1 also indicates that our system using rules and operating on the sentence-level depends less on the amount of training data compared to using only the SVM predictions. However, as none of the lines seem to have reached a plateau, we could most likely improve our system further by training it on more data.

5.1 Feature Analysis

We verified that all the features we are using improved our classifier by doing a leave-one-out feature analysis on the development set. In order to rule out the possibility that certain outliers in our data might make a feature seem useful, we ran the model on 10 bootstrapped data sets. Each of these data sets was 80% the size of the development set and generated by sampling with replacement from the development set. We then computed the recall and precision by taking the mean over the 10 runs. A feature was considered useful if the recall increased while the F1 score did not decrease. The mean precision and recall values and the corresponding F1 scores obtained by using the SVM system are presented in Table 5.

Held-out feature	Precision	Recall	F1
Word Pair	**0.815**	0.650	0.723
Word N-gram	0.772	0.621	0.689
Character N-gram	0.772	0.668	0.716
Capitalization	0.805	0.684	0.740
All	0.806	**0.686**	**0.741**

Table 5: Results of leave-one-out feature analysis.

N	N-grams
1	bitch, dumb, idiot, loser, moron, 're, shut, stupid, you, your
2	an idiot, are an, ass nigga, go back, shut up, you are, you dumb, you idiot, you're, you really

Table 6: Most discriminative N-grams.

We also looked at the most discriminative N-grams according to a χ^2-test. The top 8 discriminative unigrams and bigrams are presented in Table 6. Interestingly, all of them are positive features, i.e. if they occur in the comment, they increase the probability of that comment being an insult. We can also see that the most discriminative feature seems to be the use of profanity and phrases involving "you", such as "you are", which shows that our hand-written rules can also be inferred from the data.

5.2 Error Analysis

We examined the errors our system was making when classifying comments in the development and the YouTube dataset and we noticed that most errors fit in one of the following categories.

5.2.1 Negative Adjectives

Many insults that our system was not able to detect contained negative adjectives rather than profane words. Thus if the adjective was not observed

in the training data our system had no information on whether the content was insulting or not. Examples include "Is that why Obama has borrowed more money as president in 3 years than any other sitting president in US history? You are delusional." or "You seem very unintelligent...".

5.2.2 World Knowledge

Other insults that our system failed to detect were those contingent upon world knowledge. In order to understand that the comment "Take your meds and go back in your cave." is insulting, one needs to know that asking another person to take their medication is a way of accusing another person of being mentally ill and that asking another person to go back to their cave has the connotation that they are simple-minded like a caveman. Our system failed to detect these sorts of comments because they do not contain any profanity and a lot of the words used were not observed in the training date. Further they do not contain any lexical peculiarities.

5.2.3 Sarcasm

Another class of problematic comments are sarcastic comments such as "Your mother must be so proud of you!" or "Actually the opposite. But keep assuming stuff about people you dont know, makes you look smart.". These comments contain a lot of positive words like "proud" or "smart" that normally occur in neutral comments but, as they are meant sarcastically, the positive denotation of the words becomes negative. Our classifier, however, is only able to determine that the comment contains a lot of positive words and thus assumes the more likely case that the comments are actually neutral.

5.2.4 Use of Profane Language

Many false positives were caused by comments that contain profane words, such as "a mother fucking quiche". As profane words are a very good indicator for insults and we explicitly added a handwritten rule that when in doubt marks comments containing profanity as being insulting, our system classifies almost all comments containing profanity as being insulting as long as they don't also contain a lot of positive words.

5.2.5 Insults Directed at Other People

Many other false positives come from comments that are insulting towards a certain group of people, who are not necessarily participating in the discussion. The comment "gay guys are sick" for example is clearly offensive to homosexuals but it is not directed to other discussion participants. Our classifier potentially marks this comment as insulting because of the phrase "are sick", which directed at another participant would be an actual insult. Another example is "f u ron paul sell out schill bastard. rand, you're no dif than any sob on the hill! romney.. burn in hell! id sooner vote for satan". In this case the commenter is attacking Ron Paul (who is probably not a participant in the discussion) but our system is only able to detect that a specific person is attacked and assumes that this person is another poster.

5.2.6 Incorrect Labels

A large number of incorrectly classified comments are also based on the fact that we believe that many comments in the Kaggle dataset are actually mislabeled. The comment "Because too many idiots sue to make money." for example does not seem to be insulting to any specific poster but is labeled as being an insult. On the other hand, the comment "Get you head out of your a z z!!!!!!!!!!!" is in our opinion clearly attacking another poster, yet it is not considered an insult according to the annotators.

5.3 Ideas that did not work

In order to tackle some of the errors described in the previous section, we implemented several other features. One seemingly promising approach was to take syntactic features into account. For this reason we used the Stanford Parser (De Marneffe et al., 2006; Klein and Manning, 2003) to get the constituency and dependency parses for each sentence. We noticed that imperatives such as "Go to hell!" or "Get a life!" often characterize insults. Thus we added a feature that checked for the presence of a verb phrase in the beginning of the sentence. Further we added a feature that checked which POS tag occurred after "you" as we observed that "you" followed by a noun phrase such as "you idiot" often characterizes insults. Additionally, we also tried to use the dependency parse to try to infer whether profane words actually modify a second person pronoun

or something else such as the "quiche" in one of the examples above. However, all these features made the performance of our system worse. We suspect that the main explanation for this is that many of the parses are simply wrong, perhaps due to the fact that lots of the sentences are grammatically malformed. For example, in the sentence "You retarded idiot!" the parser annotates "retarded" as a verb phrase, when it is in fact part of the noun phrase "retarded idiot". Consequently, the dependency parse is also wrong. Had the author structured the sentence correctly - "You are a retarded idiot" - the parser would have been correct.

We also noticed that many words were (often intentionally) misspelled, so we used a simple spelling correction algorithm[6] to correct all misspelled words using minimal edit distance. However, this algorithm is based on the assumption that the misspelling is caused by an accidental typo, rather than by an conscious intent to mask a word known to be rude. As a result, many corrections changed the meaning of the comment. Further, our classifier worked better when it was able to pick up lexical peculiarities such as "f**k" instead of replacing these obfuscations with the correct word.

To try to the capture the negative sentiment in insults that do not contain profanity, we implemented a feature that calculated the semantic orientation of a comment. This was done by calculating the PMI between each word in the comment and each word in two sets of pre-defined positive and negative words used by Turney et al. (2003). However this feature also lowered both precision and recall which is why we did not include it in our final system. We suspect that our training corpus was too small to learn meaningful semantic orientations.

6 Conclusion and Future Work

We presented a system that combines an SVM classifier and hand-written rules for detecting insults in online comments. Further, we showed that by performing the classification on a sentence-specific level instead of classifying the complete comment, we were able to increase precision and recall over other state-of-the-art insult detection systems on real-world data. However, our analysis showed that our system primarily detects insults containing profanity, very common insults and insults that follow simple patterns such as "You are..." or "Go ..." and still misses more complicated and subtle insults. Despite the fact that simple syntactic rules did not work on our data and we had problems obtaining reliable parse trees, we still assume that by doing better preprocessing and presumably some post-processing of the constituency parses, one could extract common syntactical patterns that are very characteristic of insults, so future work should be done on this topic.

Our analysis also showed that the existing data seems to be very noisy and does not cover the variety of insults that exist. Thus future work on this topic should also focus on compiling a more reliable and more exhaustive dataset. Further, as the sentence-level classification approach has proven to be very effective, one should also consider doing a sentence-level annotation of the training data as so far we are still using the features of entire comments to train our model.

[6] http://norvig.com/spell-correct.html

7 Acknowledgements

We would like to thank Bill MacCartney and Chris Potts for their valuable advice and guidance.

References

Diana Inkpen Amir Hossein Razavi. 2010. Offensive language detection using multi-level classification. pages 16–27.

Ashley A. Anderson, Dominique Brossard, Dietram A. Scheufele, Michael A. Xenos, and Peter Ladwig. 2013. Crude comments and concern: Online incivility's effect on risk perceptions of emerging technologies. *Journal of Computer-Mediated Communication*, page n/a–n/a.

Dmitry Davidov, Oren Tsur, and Ari Rappoport. 2010. Enhanced sentiment learning using twitter hashtags and smileys. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 241–249. Association for Computational Linguistics.

Marie-Catherine De Marneffe, Bill MacCartney, and Christopher D Manning. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of LREC*, volume 6, pages 449–454.

Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting on Association for Computational Lin-*

guistics - Volume 1, ACL '03, page 423–430, Stroudsburg, PA, USA. Association for Computational Linguistics.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pages 79–86. Association for Computational Linguistics.

Sara Owsley Sood, Judd Antin, and Elizabeth F Churchill. 2012. Using crowdsourcing to improve profanity detection. In *2012 AAAI Spring Symposium Series*.

Ellen Spertus. 1997. Smokey: Automatic recognition of hostile messages. In *In Proc. IAAI*, pages 1058–1065.

Peter D. Turney and Michael L. Littman. 2003. Measuring praise and criticism: Inference of semantic orientation from association. *ACM Trans. Inf. Syst.*, 21(4):315–346, October.

William Warner and Julia Hirschberg. 2012. Detecting hate speech on the world wide web. *NAACL-HLT 2012*, page 19.

Guang Xiang, Bin Fan, Ling Wang, Jason Hong, and Carolyn Rose. 2012. Detecting offensive tweets via topical feature discovery over a large scale twitter corpus. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, pages 1980–1984. ACM.