Detecting threats of violence in YouTube comments

Anonymous NAACL submission

Abstract

Here comes an abstract for this article

1 Introduction

Threats of violence constitute an increasingly common occurrence in online discussions. It disproportionally affects women and minorities, often to the point of effectively eliminating them from taking part in discussions online. Moderators of social networks operate on such a large scale that manually reading all posts is an insurmountable task. Methods for automatically detecting threats could therefore potentially be very helpful, both to moderators of social networks, and to the members of those networks.

In this article, we evaluate different types of features for the task of detecting threats of violence in YouTube comments. We draw on both lexical, morphosyntactic and lexical semantic information sources and experiment with different machine learning algorithms. Our results indicate that successful detection of threats of violence is largely determined by lexical information.

2 Previous work

There is little previous work specifically devoted to the detection of threats of violence in text, however, there is previous work which examines other types of closely related phenomena, such as cyberbullying and hate-speech.

?) proposes a method for the detection of cyberbullying by targeting combinations of profane or negative words, and words related to several predetermined sensitive topics. Their data set consists of over 50,000 YouTube comments taken from videos about controversial topics. They adopt a two-stage approach, where the first stage consists of using a lexicon of negative words and a list of profane words, as well as part-of-speech tags from the training data that were correlated with bullying. The second stage is category-specific, and makes use of commonly observed uni- and bigrams from each category as features. The experiments reported accuracies from 63 % to 80 %, but did not report precision or recall.

There has been quite a bit of work focused on the detection of threats in a data set of Dutch tweets (**?**; **?**). The data set used for these experiments consists of a collection of 5000 threatening tweets collected by a website over a period of about two years. In addition, a large number of random tweets were collected for development and testing. A set of 2.3 million random tweets was used for development, and a set of 1 million was used for testing. The system relies on manually constructed recognition patterns, in the form of n-grams (uni-, bi- and trigrams, as well as skip bi- and trigrams), but do not go into detail about the methods used to construct these patterns, stating that the researchers relied on their (linguistic) intuition as speakers of Dutch (**?**). In **?**), a manually crafted shallow parser is added to the system. This improves results to a precision of 39% and a recall of 59% [1].

?) present a method for detecting unwanted or illegal comments in user-generated web text from the

[1] Perhaps a footnote on the peculiarities of the evaluation??

internet, which relies on machine learning in combination with template-based features. The data set used in the research came from two sources. The first consists of posts from Yahoo news groups, the second were webpages collected by the American Jewish Congress that had been identified as offensive. The data set was manually annotated, and then the hate speech was assigned to a category, such as anti-Semitic, anti-woman, anti-Asian, etc. The task is then approached as a word-sense disambiguation task, since the same words can be used in both hateful and non-hateful contexts. The features used in the classification were combinations of uni-, bi- and trigrams, part-of-speech-tags and Brown clusters. The best results of the classifications were obtained using only unigrams as features, with a precision of 67 % and a recall of 60 %. The other feature sets garnered much lower results, and the authors suggest that deeper parsing could reveal significant phrase patterns.

?) reports an experiment on the data set that we will be using in our own experiments, although it has been changed slightly since the publication of his initial study. The method used a logistic LASSO regression analysis on bigrams (skip-grams) of important words to classify sentences as threats of violence or not. The method described in the article uses a set of important words that are correlated with threats of violence. The features are bigrams of two of these important words observed in the same sentence. The article does not describe exactly how these important words were selected, stating only that words were chosen that were significantly correlated with the response (violent/non-violent sentence). Results are reported only in terms of proportion of false positives for the two classes and it is not clear how the data was split for training and evaluation.

3 The YouTube threat data set

The YouTube threat data set is comprised of user-written comments from eight different YouTube videos (?). A comment consists of a set of sentences, each of them manually annotated to be either a threat of violence (or support for a threat of violence) or not. The data set furthermore records the username[2] of the user that posted the comment. The eight videos that the comments were posted to cover religious and political topics like halal slaughter, immigration, Anders Behring Breivik, Jihad, etc. (?).

	Commments	Sentences	Users posting
Total	$9,845$	$28,643$	$5,483$
Threats	$1,285$	$1,384$	992

Table 1: Number of comments, sentences and users in the YouTube threat data set

The data set consists of 9,845 comments, comprised of 28,643 sentences, see table ??. In total there are 402,673 tokens in the sentences in the data set. There are 1,285 comments containing threats, and 1,384 sentences containing threats, as seen in table ??. ?) report inter annotator agreement on this data set to be 98 %, as calculated on 120 of the comments, doubly annotated for evaluation.

Figure ?? provides some examples of comments containing threats of violence taken from the data set. The first line is the username or name, and the subsequent lines are the sentences of the comment. An empty line indicates the end of a comment. The sentences are annotated with a number indicating whether they contain a threat of violence (1), or not (0).

4 Experiments

Much of the previous work in Section ?? made use of precompiled lists of correlated words and manually crafted patterns. Whereas these resources can be effective, they are highly task-specific and do not easily lend themselves to replication. The previous work further highlights the effectiveness of lexical features (words-based features), however, several of the authors suggest that parsing may be beneficial for these tasks.

In this work we experiment with a wide range of linguistic information as features in our system and investigate the introduction of linguistic context through a set of feature templates. We also gen-

[2] In 2013, YouTube changed its commenting system from using unique usernames, to using ”real names”, like Facebook and other sites (?). Some accounts, however, did not provide real names, so they continue to only be identified by their usernames.

```
timpa666
1    and i will kill every fucking muslim and arab!

NimsXdimensions
0    Need a solution?
1    Drop one good ol' nuke on that black toilet in Mecca.

Ammar Alozaibi
1    Funny, We will conquer you all in just few years, U will be
     my slave and your women will be my Sex Toy in Bed.
```

Figure 1: Examples of comments from the data set.

eralize these features through a backoff technique. Throughout, we make use of freely available, reusable resources and tools in order to construct our system.

4.1 Experimental setup

Pre-processing Since threat annotation is performed on the sentence level, the data set has been manually split into sentences as part of the annotation process. We performed tokenization, lemmatization, PoS-tagging and dependency parsing using the SpaCy toolkit[3]. SpaCy assigns both the standard Penn Treebank PoS-tags (**?**), as well as the more coarse-grained Universal PoS tag set of **?**). The dependency parser assigns an analysis in compliance with the ClearNLP converter (**?**), see Figure **??** for an example dependency graph from the data set. The data set was further enriched with the cluster labels described in Turian et al. (**?**), created using the Brown clustering algorithm (**?**) and induced from the RCV1 corpus, a corpus containing Reuters English newswire text, with approximately 63 million words and 3.3 million sentences. We vary the number of clusters to be either 100, 320, 1000 or 3200 clusters and use the full cluster label. We also make use of the WordNet resource (**?**) to determine a word's synset, parent and grandparent synset.

Classifiers Toolkits and references

Tuning Short on tuning

Features Based on the enriched data set, as described above, we experiment with the following lexical, morphosyntactic and semantic feature types:

Lexical word form, lemma

Morphosyntactic Penn Treebank (PTB) or Universal (uPOS) PoS and Dependency Relation

Semantic Brown cluster label (100, 320, 1000 and 3200 clusters) and WordNet synset, parent synset and grandparent synset

These features are structured according to a set of *feature templates*, which introduce varying degrees of linear order and syntactic context: bag-of-features (unordered), bigrams, trigrams and dependency triples. These feature templates are applied to the feature types presented above, in order to yield features like the following for our examples sentence in Figure **??**:

- {i m, m fucking, fucking scared, scared kill, kill them, them d, d :}
- {PRON VERB VERB, VERB VERB ADJ, VERB ADJ VERB, ADJ VERB PRON, VERB PRON X, PRON X PUNCT}
- {<m, nsubj, i>, <root, root, m>, <scared, amod, fucking>, <m, acomp, scared>, <m, conj, kill>, <kill, dobj, them>, <kill, advmod, d>, <m, punct, :> }

Our lexical features are very specific and require the exact combination of two lexical items in order to apply to a new instance. Following **?**), we therefore experiment with generalization of features by backoff to a more general category, e.g. from word form to lemma or PoS. A dependency triple over word

[3] https://spacy.io/

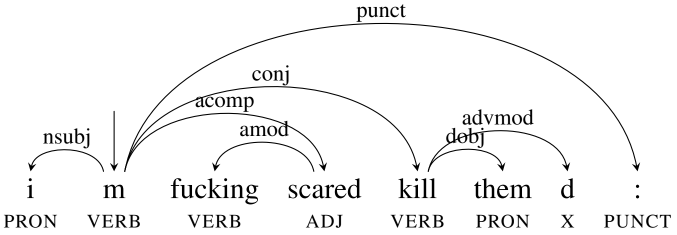

Figure 2: Dependency parse of example sentence from the data set, with assigned uPOS tags.

forms like <kill, dobj, them> would thus be generalized to <VERB, dobj, them> using *head-backoff* and <kill, dobj, PRON> using *modifier-backoff*. We apply backoff to bigram, trigram and dependency triple features.

4.2 Development results

Table **??** shows baseline results in terms of F-score for bag-of, bigram and trigram templates over lexical features (word form and lemma). The presented results constitute the best F-scores after tuning (which was only done on the MaxEnt classifier and the SVM, not on the random forest classifier), as discussed above. The overall best result came from the logistic regression classifier with the bag of word form feature set. Generally we see that feature sets containing bag of lexical features in some form outperform the n-gram feature sets. The feature set we will use as a baseline in the next stage of our experiments is bag of word forms. The feature set recieved an F-score of 0.6123, with a precision of 0.6777 and a recall of 0.5585, using the MaxEnt classifier after tuning. We will also be using only the MaxEnt classifier for the remainder of the feature set experiments.

Morphosyntactic features: - bag of, bigram, trigram of pos, deprel - combine with baseline

Semantic features: - bag of, bigram, trigram of Brown, WordNet - combine with baseline

Combination of best from baseline, morphosyntactic and semantic

4.3 Error analysis

4.4 Held-out results

5 Conclusion

	MaxEnt	SVM	Random Forest
Bag of word forms	**0.6123**	**0.6068**	**0.5918**
Bag of lemmas	0.5902	0.5982	0.5856
Bigrams	0.4856	0.4887	0.4944
Trigrams	0.2776	0.2856	0.2859

Table 2: Results for baseline system; F-score for bag-of, bigram and trigram templates over lexical features (word form and lemma).