Abusive Language Detection in Online User Content

Chikashi Nobata
Yahoo Labs
Sunnyvale, CA, USA
chikashi@yahoo-inc.com

Joel Tetreault
Yahoo Labs
New York, NY, USA
tetreaul@yahoo-inc.com

Achint Thomas*
Embibe
Bangalore, India
achint@embibe.com

Yashar Mehdad
Yahoo Labs
Sunnyvale, CA, USA
ymehdad@yahoo-inc.com

Yi Chang
Yahoo Labs
Sunnyvale, CA, USA
yichang@yahoo-inc.com

ABSTRACT

Detection of abusive language in user generated online content has become an issue of increasing importance in recent years. Most current commercial methods make use of blacklists and regular expressions, however these measures fall short when contending with more subtle, less ham-fisted examples of hate speech. In this work, we develop a machine learning based method to detect hate speech on online user comments from two domains which outperforms a state-of-the-art deep learning approach. We also develop a corpus of user comments annotated for abusive language, the first of its kind. Finally, we use our detection tool to analyze abusive language over time and in different settings to further enhance our knowledge of this behavior.

General Terms

NLP

Keywords

NLP, Hate Speech, Abusive Language, Stylistic Classification, Discourse Classification

1. INTRODUCTION

Anytime one engages online, whether on message board forums, comments, or social media, there is always a serious risk that he or she may be the target of ridicule and even harassment. Words and sentences such as *kill yrslef a$$hole* or *they should all burn in hell for what they've done* are unfortunately not uncommon online and can have a profound impact on the civility of a community or a user's experience. To combat abusive language, many internet companies have standards and guidelines that users must adhere to and employ human editors, in conjunction with systems which use regular expressions and blacklist, to catch bad language and thus remove a post. As people increasingly communicate online, the need for high quality automated abusive language classifiers becomes much more profound.

Recent cases highlight the impact of hurtful language in online communities, as well as on major corporations. For example, in 2013, Facebook came under fire for hosting pages which were hateful against women such as *Violently raping your friend just for laughs* and *Kicking your girlfriend in the fanny because she won't make you a sandwich.*[1] Within days, a petition was started which amassed over 200,000 supporters, and several major companies either pulled or threatened to pull their ads from Facebook since they were inadvertently places on these pages. Facebook is not the only company that contends with these issues; any company which hosts user generated content will have a moderation issue. This shows the large impact hateful language can have on a community as well as a major company.

At the more individual level, when actor Robin Williams passed away, his daughter Zelda posted a memoriam to her late father and was immediately bullied on Twitter and Instagram and eventually deleted all of her online accounts. This harrassment prompted Twitter to review and revise its hate speech guidelines.[2]

While automatically detecting abusive language online is an important topic and task, the prior art has not been very unified, thus slowing progress. Past research has spanned different fields ranging from Natural Language Processing (NLP) to Web Sciences to Artificial Intelligence, meaning that several similar methods were published in the last three years. Additionally, abusive language can be a bit of a catch-all term. There are some studies, [14] which focus on detecting profanity, and others, such as [18] which focus on hate speech directed to a particular ethnic group. To further complicate matters, to date there has been no de facto testing set with which to compare methods.

*This work was done while the author was at Yahoo Labs.

WWW 2016, April 11–15, 2016, Montréal, Québec, Canada.
ACM 978-1-4503-4143-1/16/04.
http://dx.doi.org/10.1145/2872427.2883062 .

[1] http://www.nytimes.com/2013/05/29/business/media/facebook-says-it-failed-to-stop-misogynous-pages.html?_r=0
[2] https://www.washingtonpost.com/news/the-switch/wp/2014/08/13/twitter-vows-to-improve-our-policies-after-robin-williams-daughter-is-bullied-off-the-network

In this paper we aim to develop a state-of-the-art method for detecting abusive language in user comments, while also addressing the above deficiencies in the field. Specifically, this paper has the following contributions:

- We develop a supervised classification methodology with NLP features to outperform a deep learning approach. We use and adapt several of the features used in prior art in an effort to see how they perform on the same data set. We also extend this feature set with features derived from distributional semantics techniques.
- We make public a new data set of several thousand user comments collected from different domains. This set includes three judgments per comment and for comments which are labeled as abusive, a more fine-grained classification on how each is abusive.
- Prior work has evaluated on a fixed, static data set. However, given the issues with language changing over time and also with users trying to cleverly evade keyword-based approaches, we perform several analyses of how models trained on different types and sizes of data perform over the span of one year, across two different domains. To our knowledge, this is the first longitudinal study of a computational approach to abusive language detection.

In §2 we discuss what is abusive language and what makes it so difficult to process, as well as related work. §3 describes our data sets and crowdsourced annotation efforts. In §4 we discuss our hate speech detection framework and in §5 we discuss a battery of experiments to analyze this tool and hate speech in general. Finally, we want to warn the reader that there are examples of hate speech reproduced in the paper which are here purely for illustrative purposes.

2. BACKGROUND

2.1 Why is this task difficult?

Detecting abusive language is often more difficult than one expects for a variety of reasons. the noisiness of the data in conjunction with a need for world knowledge not only makes this a challenging task to automate but also potentially a difficult task for people as well.

More than simple keyword spotting. The intentional obfuscation of words and phrases to evade manual or automatic checking often makes detection difficult. Obfuscations such as *ni9 9er*, *whoopiuglyniggerratgolberg* and *JOOZ* make it impossible for simple keyword spotting metrics to be successful, especially as there are many permutations to a source word or phrase. Conversely, the use of keyword spotting could lead to false positives.

Difficult to track all racial and minority insults. One can make a reasonably effective abuse or profanity classifier with a blacklist (a collection of words known to be hateful or insulting), however, these lists are not static and are ever changing. So a blacklist would have to be regularly updated to keep up with language change. In addition, some insults which might be unacceptable to one group may be totally fine to another group, and thus the context of the blacklist word is all important (this forms the motivation for the work by [18]).

Abusive language may actually be very fluent and grammatical. While there are many examples on the internet of abusive language being very noisy, such as in *Add anotherJEW fined a bi$$ion for stealing like a lil maggot. Hang thm all.*, which can be a helpful signal for an automated method, there are actually many cases where abusive language, or even more specifically hate speech, is quite fluent and grammatical. For example:*I am surprised they reported on this crap who cares about another dead nigger?*

Abusiveness can be cross sentence boundaries. In the sentence *Chuck Hagel will shield Americans from the desert animals bickering. Let them kill each other, good riddance!*, the second sentence which actually has the most hateful intensity (them kill each other) is dependent on the successful resolution of them to *desert animals* which itself requires world knowledge to resolve. The point here is that abusive language is not limited to just the sentence. In some cases, one has to take the other sentences into account to decide whether the text is abusive or carries incidences of hate speech.

Sarcasm. Finally, we noted cases where some users would post sarcastic comments in the same voice as the people that were producing abusive language. This is a very difficult for humans or machines to get correct as it requires knowledge of the community and potentially even the users themselves: *same thing over and over and over and over day in night and day 'cause i am handicapped and stay home. i hate jews they ran over my legs with their bmw. so i will blast them everyday.. I really hurt them i am so powerful .. If ipost about jews here they all suffer. im sow powerfull bwbwbwbwaaahahahahahah im a cripple but i can destroy them with my posts.. I am super poster. Bwwbwahahahaha noone can find me .. I am chicken so i can post behind yahoos wall of anonymous posters. Bwbwbwbabahahahah i will give him ten thumbs down and slander jews.. Bwbwbwbahahahah..i am adoplh hitler reincarnated.*

2.2 Related Work

Most prior work in the area of abusive language detection has actually been spread across several overlapping fields. This can cause some confusion as different works may tackle specific aspects of abusive language, define the term differently, or apply it to specific online domains only (Twitter, online forums, etc.). To further complicate comparison between approaches, nearly all previous work uses different evaluation sets. One of the contributions of this paper is to provide a public dataset in order to better move the field forward.

One of the first works to address abusive language was [21] which used a supervised classification technique in conjunction with n-gram, manually developed regular expression patterns, contextual features which take into account the abusiveness of previous sentences.

As most basic approaches make use of predefined blacklists, [15] noted that some blacklist words might not be abusive in the proper context. In their work they showed an improvement in *profanity* detection by making use of lists as well as an edit distance metric. The latter allowed them to catch such un-normalized terms as *@ss* or *sh1t*. Another contribution of the work was that they were the first to use crowdsourcing to annotate abusive language. In their task, they used Amazon Mechanical Turk workers to label 6,500 internet comments as abusive or not abusive. They only

used comments in which a majority of the turkers agreed on the label. 9% of the comments were deemed as carrying profane words. In our work, we also make use of crowdsourcing to curate a corpus of several thousand internet comments. The main differences are that we do not limit the task to just profanity and also have the workers annotate for other types of hate speech and abusive language. In addition, we are making this dataset public.

[3] was one of the first to use a combination of lexical and parser features to detect offensive language in youtube comments to shield adolescents. While they do note that they do not have a strict definition of offensive language in mind, their tool can be tuned by the use of a threshold which can be set by parents or teachers so online material can be filtered out before it appears on a web browser. The work takes a supervised classification approach using Support Vector Machines (SVMs) with features including n-grams, automatically derived blacklists, manually developed regular expressions and dependency parse features. They achieve a performance on the task of inflammatory sentence detection of precision of 98.24% and recall of 94.34%. One difference between our work and this one is that they attempt to spell-correct and normalize noisy text *before* feature extraction. We believe that this noise is a potentially good signal for abuse detection and thus have features to capture different types of noise. Our work also makes use of dependency features, though with a much broader set of tuples than [3].

[18] provide the most comprehensive investigation of *hate speech* (hateful language directed towards a minority or disadvantaged group) to date, with working definitions and an annotation task. Here their focus was less on abusive language and more specifically on anti-semitic hate. First, they manually annotated a corpus of websites and user comments, with Fleiss kappa interlabeler agreement at 0.63. Next, they adopted a related approach to the aforementioned supervised classification methods by first targeting certain words that could either be hateful or not, and then using Word Sense Disambiguation techniques [20] to determine the polarity of the word. Their method performs at 0.63 F-score. To our knowledge, this is the only work to target hate speech and the only one to have done a rigorous annotation of data, though the set could not be made public. We build on their work by crowdsourcing the annotation of a data set of user comments, categorizing each comment as abuse, profanity, and/or hate speech. This set will be made public.

Finally, [5] use a paragraph2vec approach adopted from [8] to classify language on user comments as abusive or clean. Their approach outperformed a bag-of-words (BOW) implementation (0.8007 to 0.7889 AUC). In our work, we use a more sophisticated algorithm to learn the representation of comments as low-dimensional dense vectors. Moreover, our representation is learned using only unigrams in order to compliment other relevant features. In our work, we aim for a method that is efficient and flexible but also operates at a high accuracy by combining different light-weight features. We include an evaluation using their data to directly compare our system but also experiment with their approach as additional features in our methodology.

3. DATA

All data used for training and testing in this paper was extracted from comments found on Yahoo! Finance and News. These comments were moderated by Yahoo employees whose primary function was providing editorial labels for various annotation/editorial tasks. All subjects had at least an undergraduate degree and were familiar with the concept of judging text passages for different types of annotation tasks and requirements. Before taking on the actual moderation task, they were trained in order to familiarize themselves to with text judgment guidelines.

As mentioned in §2.2, there are many different forms of abusive language, and sometimes the term abusive language is conflated with hate speech. For our work, abusive language encompasses hate speech, profanity and derogatory language. A summary of the guidelines and examples for each category is shown in Table 4.

3.1 Primary Data Set

Data for our primary training models is sampled from comments posted on Yahoo! Finance and News during the period between October 2012 and January 2014. The original data is collected as follows. A completely random 10% subset of the comments that are posted each day on Yahoo! Finance and News articles are sent for review by Yahoo's in-house trained raters. Further, all comments which are reported as "abusive" for any reason by visitors to the Finance and News portals are also sent to the raters for review and judgment. Such reports are termed community moderation. To maintain business confidentiality, we cannot divulge the volume of comments in the community moderation bucket.

The breakdown of "Clean" and "Abusive" comments for both domains is shown in Table 1. The percentage of abusive comments for Finance is roughly 7.0% for Finance and 16.4% for News. In our experiments with this set, we train on 80% of the data and test on the remaining 20%.

Table 1: Primary Data Set Statistics

Finance data		News data	
Clean	705,886	Clean	1,162,655
Abusive	53,516	Abusive	228,119
Total	759,402	Total	1,390,774

3.2 Temporal Data Set

The data used in our temporal experiments (§5.4) are also sampled from comments posted on Yahoo! Finance and News; the period is between April 2014 and April 2015. The number of "Clean" and "Abusive" comments on both domains are shown in Table 2. The ratio of abusive comments on each domain is lower than our older main data, around 3.4% on Finance, 10.7% on News. The main reason is our collective efforts including deploying our models in the production environment during that time.

Table 2: Temporal Data Set Statistics

Finance data		News data	
Clean	433,255	Clean	655,732
Abusive	15,181	Abusive	70,311
Total	448,436	Total	726,073

3.3 WWW2015 Data Set

To directly compare against prior work, we also use the data set described in [5]. The set comprises 56,280 comments labeled as "Abusive" and 895,546 comments labeled as "Clean" collected from Yahoo! Finance in the same manner as Data Set 1. The percentage of abusive comments on this data is around 5.9%. To replicate their evaluation methodology, we conduct 5-fold cross-validation on this set.

Table 3: WWW2015 Statistics

Clean	895,456
Abusive	56,280
Total	951,736

3.4 Evaluation Data Set

In addition to the three prior data sets, we wanted a corpus specifically set aside for evaluation in which each comment is labeled by three trained raters. This overlapping labeling allows us to determine the level of human agreement in this task.

We extracted several thousand comments between March and April 2015 for the raters to label. For the "Clean" and "Abuse" binary distinction, the agreement rate is 0.922 and Fleiss's Kappa is 0.843, thus showing that humans can achieve a relatively high agreement for this task. We also had the raters label the subcategory of abuse (hate, derogatory language and profanity), where multiple subcategories can be labeled for a comment. Agreement for this task drops to 0.603 and Fleiss's Kappa is 0.456.

From that labeled set, we used 1,000 marked as "Clean" and 1,000 marked as "Abusive" for a total of 2,000 comments. We used the majority vote of this binary categorization to evaluate our model in Section §5.3. We will be making this data public through the Yahoo Webscope program.[3] To our knowledge this is the first publicly available abusive language data set with multiple annotations and the different subcategories. There does exist a data set for insults which was used in a Kaggle competition.[4]

3.5 Amazon Turk Experiment

As a corollary to the evaluation data set collection and labeling, we also investigated if crowdsourcing the labeling to untrained raters could be used to efficiently label comments as well as the trained raters. We used Amazon's Mechanical Turk (AMT), a marketplace of human workers willing to solve tasks requiring human intelligence. Requesters can post Human Intelligence Tests (HITs), and offer a reward for its completion. Once a worker completes a HIT and submits the results, the requester can review the solution before authorizing payment. Certain performance criteria can be set, which once met, entitles the worker to bonus payments. A number of studies have shown that workers on produce results that are comparable to conducting laboratory studies [2, 7, 12, 17].

We posted text judgment challenges as HITs on AMT and offered financial incentives for workers to participate in the study. Each HIT consisted of a set of 10 text passage instances that had to be attempted. Workers who completed a HIT would be awarded US$ 0.20/HIT (this translates to US$0.02 for each text passage attempted). We set the per-worker HIT limit to 5, and so each worker could only judge at most 50 text passages (5 HITs x 10 text passage instances).

In order for the Turkers to be able to judge the text passages as accurately as done by the in-house raters, we distilled the annotation guidelines used internally at Yahoo and provided example use-cases as well as how they should be potentially judged. These guidelines are presented in Table 4.

For the binary classification, Turkers had an agreement rate of 0.867 and Fleiss's Kappa of 0.401. With the more fine-grained abuse classification, agreement dropped to 0.405 and Fleiss's Kappa to 0.213. Compared to the in-house raters, the Turkers exhibit much worse agreement, suggesting that they might not be entirely effective for this task, or more extensive training or more judges per comment are required.

how did they calculate the agreement rate

4. METHODOLOGY

For our work we employ a supervised classification method which uses NLP features which measure different aspects of the user comment. Specifically, we use the Vowpal Wabbit's regression model[5] in its standard setting with a bit rate of 28. We base our NLP features on prior work in sentiment [9], text normalization [1] among others. Our features can be divided into four classes: N-grams, Linguistic, Syntactic and Distributional Semantics. For the first three features, we do some mild pre-processing to transform some of the noise found in the data which could impact the number of sparse features in the model. Example transformations include normalizing numbers, replacing very long unknown words with the same token, replacing repeated punctuation with the same token, etc. For the fourth feature class, we did none of the above normalization.

4.1 N-gram Features

We employ character n-grams (from 3 to 5 characters, spaces included) and token unigrams and bigrams. In contrast to prior work in this field which either ignored unnormalized text or used simple edit distance metrics to normalize them, we use character n-grams to model the types of conscious or unconscious bastardizations of offensive words.

4.2 Linguistic Features

To further handle the noisiness of data, we developed specialized features based on work by [1]. These features are intended to explicitly look for inflammatory words (such as the use of pre-existing hate lists) but also elements of non-abusive language such as the use of politeness words or modal verbs. These features include:

- length of comment in tokens
- average length of word
- number of punctuations
- number of periods, question marks, quotes, and repeated punctuation

[3] https://webscope.sandbox.yahoo.com/

[4] https://www.kaggle.com/c/detecting-insults-in-social-commentary

[5] https://github.com/JohnLangford/vowpal_wabbit

Table 4: Annotation Instructions

Given the passage below, first rate the passage on whether the passage is clean or abusive. And if abusive, • whether it contains hate speech • whether it contains derogatory languge • whether it contains profanity	
Hate Speech:	language which attacks or demeans a group based on race, ethnic origin, religion, disability, gender, age, disability, or sexual orientation/gender identity. Some examples: • The JEWZ are Market Manipulators Hate them freaking Jews !! To bad Hitler did not finish them off !!! • I hate the mall because there are too many unemployed apefricans and mudsharks. • 300 missiles are cool! Love to see um launched into TelAviv! Kill all the gays there! • EGYPTIANS ARE DESCENDED FROM APES AND PIGS. NUKE EGYPT NOW Ah me tu se ku sh
Derogatory:	language which attacks an individual or a group, but which is not hate speech. Some examples: • All you perverts (other than me) who posted today, needs to leave the O Board. Dfasdfdasfadfs • yikes...another republiCUNT weighs in....
Profanity:	language which contains sexual remarks or profanity. Some examples: • T.Boone Pickens needs to take a minimum wage job in FL for a week. I guarantee he shuts the f up after that. • Somebody told me that Little Debbie likes to take it up the A.$.$. • So if the pre market is any indication Kind of like the bloody red tampons that you to suck on all day??

- number of one letter tokens
- number of capitalized letters
- number of URLS
- number of tokens with non-alpha characters in the middle
- number of discourse connectives, based on [13]
- number of politeness words
- number of modal words (to measure hedging and confidence by speaker)
- number of unknown words as compared to a dictionary of English words (meant to measure uniqueness and any misspellings)
- number of insult and hate blacklist words[6]

4.3 Syntactic Features

The use of natural language parsing is common for tasks ranging from sentiment analysis [9] to best answer prediction in CQA analysis [16]. We derive features from the ClearNLP v2.0 dependency parser[7]. The features are essentially different types of tuples making use of the words, POS tags and dependency relations. These include:

- parent of node
- grandparent of node
- POS of parent
- POS of grandparent
- tuple consisting of the word, parent and grandparent
- children of node[8]
- tuples consisting of the permutations of the word or its POS, the dependency label connecting the word to its parent, and the parent or its POS

The motivation behind these features is to capture long-range dependencies between words which n-grams may not be able to do (such as in the example: *Jews are lower class pigs*, where an n-gram model would not be able to connect Jews and pigs, however using a dependency parser would generate the tuple - are-Jews-pigs where *Jews* and *pigs* are the children of *are*.

4.4 Distributional Semantics Features

The ideas of distributed and distributional word and text representations has supported many applications in natural language processing successfully. The related work is largely focused on the notion of word and text representations (as in [10],[8] and [4]), which improve previous efforts on modeling lexical semantics using vector space models [10]. To date, only [5] has made use of these in their approach to abusive language detection.

We use three types of embedding-derived features. The first two are based on averaging the word embeddings of all words in the comment, in essence a shallow method meant to approximate an embedding for a larger piece of text that has had some success in tasks such as sentiment analysis [6]. In one we use a pre-trained embeddings[9] derived a large corpus of news text (henceforth *pretrained*. In the second we use word2vec[10] to train embeddings from our large corpora of news and finance comments respectively (henceforth *word2vec*). For both features we use a 200 dimensional embedding vector.

More recently, the concept of embeddings has been extended beyond words to a number of text segments, including phrases [11], sentences and paragraphs [8], entities [19] and documents. For our third embeddings features, we develop a comment embeddings approach akin to [8].

In order to obtain the embeddings of comments we learn distributed representations for our comments dataset. The

[6]For example, we used lists scraped from `http://www.hatebase.org/`

[7]`http://clearnlp.wikispaces.com/`

[8]Though we only use ones that are NN, V, J, CD, PRP.

[9]`https://github.com/turian/crfchunking-with-wordrepresentations`

[10]`https://code.google.com/p/word2vec/`

comments are represented as low-dimensional vectors and are jointly learned with distributed vector representations of tokens using a distributed memory model explained in [8]. In particular, we take advantage of the content of comments to model word sequences within them. While the word vectors contribute to predict the next word in comments, comment vectors also contribute to predict the next word given many contexts sampled from the comment.

In our comment embeddings model (henceforth *comment2vec*, every comment is mapped to a unique vector in a matrix representing comments and every word is mapped to a unique vector in a matrix representing words. Then comment vectors and word vectors are concatenated [8] to predict the next word in a context. More precisely, the probability distribution of observing a word depends not only on the fixed number of surrounding words, but also depends on the specific comment. In this way we represent each comment by a dense low-dimensional vector which is trained to predict words in the comment and overcomes the weaknesses of word embeddings solely.

We train the embeddings of the words in comments using skip-bigram model [10] with window size of 10 using hierarchical softmax training. For the embedding of comments we exploit the distributed memory model since it usually performs well for most tasks [8]. We train a low-dimensional model (100 dimensions) because we intend to add this representations to other features such as n-gram distributions. We also limit the number of iterations to 10 to increase the efficiency.

A great advantage of learning distributed representation vectors for comments in this way is that the algorithm is not sensitive to comment length and it does not require specific tuning for word weights. As a disadvantage, this algorithm needs the constant retraining when new comments are added, which makes the model less efficient for the online applications. This challenge can be addressed in various ways: *i*) a scalable vector tuning and updating for new comments, *ii*) inferring low-dimentional vector for new comments using gradient descent using the parameters, the word vectors and the softmax weights from the trained model, and *iii*) approximating the new vector by estimating the distance of the new comment to the previous comments using the words and their representations. We plan to investigate these methods in future work.

5. EXPERIMENTS

In this section, we describe a battery of experiments meant to evaluate our classifier, compare it to prior work and then use it as a tool to analyze trends of hate speech in user comments.

In §5.1 we show the overall performance of our model on the Primary Finance and News data sets. We evaluate the impact of each feature and discuss which are best for this task. In §5.2 we then compare our model to the prior work of [5] on the WWW2015 set. Next, we evaluate on our curated Evaluation data set (§5.3) and in §5.4 we investigate the question: How does performance vary over time? One could hypothesize that language and use of hate speech changes rapidly and this will thus impact a classifier's performance if the model is not updated.

5.1 Evaluation on Primary Data Set

In this set of experiments, we train and test our model using the Primary Data Set for both domains (Finance and News). For each domain, we use 80% for training and 20% for testing. Table 5 shows the results for each domain when a model trained with a single feature type as well as with all features combined. For both domains, combining all features yields the best performance (0.795 for Finance and 0.817 for News). News has a slight performance edge though that may be easily accounted for by the fact that there is a larger training corpus available for that domain.

In terms of individual features, for both sets, character n-grams have the largest contribution. The two sets do exhibit different behavior in terms of other features. In the Finance set, the syntactic and distributional semantics features do not perform as well as they fare in the News domain. We believe that the Finance domain is slightly noisier than News and thus these more complex features do not fare as well.

Table 5: Primary Data Set Results (by F-score)

Features	Finance	News
Lexicon	0.539	0.522
Trained Lexicon	0.656	0.669
Linguistic	0.558	0.601
Token N-grams	0.722	0.740
Character N-grams	0.726	0.769
Syntactic	0.689	0.748
word2vec	0.653	0.698
pretrained	0.602	0.649
comment2vec	0.680	0.758
All Features	**0.795**	**0.817**

5.2 Evaluation on WWW2015 Set

We next conducted an experiment on the data used in [5] to directly compare our work. As in their experimental setup, we use 5-fold cross validation. Table 6 shows that our model outperforms the prior art by 10 points AUC (0.9055 to 0.8007). We also report precision, recall and F-score for our model with all features and several other baselines. Simply using our blacklist lexicon (lexicon) as a lookup table produces an F-score of 0.537. Training a model on this lexicon such that we have weights on each word produces a slightly better F-score of 0.595. As in §5.1, the token and character n-grams by themselves are extremely predictive and outperform [5]. They are also extremely close in performance to our model with all features. The distributional features do not perform as well as their n-gram counterparts but easily outperform the lexicon-based baselines. They also help improve the overall results by boosting the recall over precision. Among the distributional features, comment2vec outperforms word2vec mainly because the comment2vec algorithm preserves the semantic aspect of comments taking advantage of co-training words and comments. In contrast, simply using the average embeddings of the words reduces the context and word order sensitivity and eventually the semantics of the comment embeddings.

Table 6: WWW2015 Results

Method	Rec.	Prec.	F-score	AUC
[5]	-	-	-	0.8007
Lexicon	0.557	0.519	0.537	-
Trained Lexicon	0.540	0.662	0.595	0.7597
Linguistic	0.501	0.523	0.512	0.6463
Token Ngrams	0.771	0.713	0.741	0.8532
Character Ngrams	0.821	0.732	0.774	0.9037
Syntactic	0.723	0.593	0.651	0.7902
word2vec	0.766	0.597	0.671	0.8409
pretrained	0.714	0.566	0.631	0.7851
comment2vec	0.780	0.590	0.672	0.8521
All Features	**0.794**	**0.773**	**0.783**	**0.9055**

5.3 Evaluation Data Set Experiment

We applied our model to the more highly curated evaluation data set discussed in §3.4. The ground truth labels are obtained as the majority vote of the manually assigned judgments. The evaluation results of our models on this data are shown in Table 7. The performance is comparable to our results in §5.1 and §5.2. Additionally, we experimented with different gold standard references: we evaluated the system when all three raters agreed (unanimous agreement) and where exactly two agreed (and we use their judgment). Although the number of "All Agreed" comments is dominant in this data (1,766 of 2,000), and the difference between labels by the majority vote and those by "All Agreed" are small, the results on the cases where *all* graders agreed have higher results compared to those with exactly 2 of 3 raters agreed (0.839 to 0.826).

Some of the false positive cases are dubious, e.g. our model labels "Abusive" for the comments such as *"Gays in Indiana pooping their pants over this law."*, *"Bug is a lying shill for the mm warmie cult alarmist hoaxers."*, *"Please stop the black on white crimes!"*(sic), which are labeled as "Clean" by 2 of 3 raters. Some comments are also inherently ambiguous, e.g. *"Soak their clothes in gasoline and set them on fire."*, *"or you could...you know...shoot them"* could be "Abusive", but without the context it is difficult to judge. Part of our future work is to extract the thread of comments and use them as context to judge each comment.

Table 7: Evaluation Data Set Evaluation

Experiment	n	Recall	Precision	F-score
Majority	2,000	0.825	0.827	0.826
All Agreed	1,766	0.842	0.837	0.839
2 of 3 Agreed	234	0.378	0.500	0.431

5.4 Evaluation on Temporal Data Set

For our final set of experiments, we seek to answer the following questions: 1) how much training data is actually necessary for a high performance? and 2) does performance degrade over time if a model is not updated? To answer these questions we ran three experiments using the Temporal Set (Data Set 2 in §5.2) which is divided into consecutive slices of 20k comment each.

1. **Original** We use the model developed using Primary Data Set and used in the evaluation §5.1, and evaluate it over the consecutive slices of data in the Temporal Set. Our hypothesis is that if there is significant language change in user comments, performance should degrade. This would mean that any anti-abuse method would need to be updated regularly.

2. **Each Slice** We train a model with the data at each slice (t) and apply the model to the next slice ($t+1$). So each training set consists of only 20k comments and is markedly smaller than the other two evaluations.

3. **Accumulated** We train a model by accumulating data available until that the time ($1..t$) and apply the model to the next slice ($t+1$). Our hypothesis is that this model should outperform the **Each Slice** model since it consists of more data, but the data is smaller than the set used in **Original**.

The results are shown in Figure 1. There are several trends of note. First, the best model in both News and Finance was **Accumulated**. This suggests that having more recent data than a larger data set is preferable by about 5% F-score in the final slice. In both Finance and News data, the improvement by accumulating data seems saturated pretty quickly in the first few slices. Interestingly, in the Finance domain, training by slice is 10-15% below the other models, but in news, it is roughly only 5% behind the **Original** model which is trained on a set several orders of magnitude larger. This once again suggests that one can build a reasonable model off of a much smaller set than used in our initial experiments. Finally, we also note the jagged nature of the trends over time for both data sets. These would seem to indicate that there is always some amount of unseen words and noise in the data, but that performance in the News domain seems to stabilize as more data is added.

whether they could try dimensional reduction in the original data?

6. CONCLUSIONS AND FUTURE WORK

As the amount of online user generated content quickly grows, it is necessary to use accurate, automated methods to flag abusive language is of paramount of importance. Not addressing the problem can lead to users abandoning an online community due to harassment or companies pulling advertisements which are featured next to abusive comments. While there has been much work in this area in several different related fields, to date, there has not been a standard evaluation set with which researchers could compare their methods. Additionally, there have been several NLP methods used in prior work but these features have never been combined or evaluated against each other. In our work we take a major step forward in the field by first providing a curated public dataset and also performing several evaluations of a range of NLP features.

We experimented with several new features for this task: different syntactic features as well as different types of embeddings features, and find them to be very powerful when combined with the standard NLP features. Character n-grams alone fare very well in these noisy data sets. Our model also outperforms a deep learning based model while avoiding the problem of having to retrain embeddings on

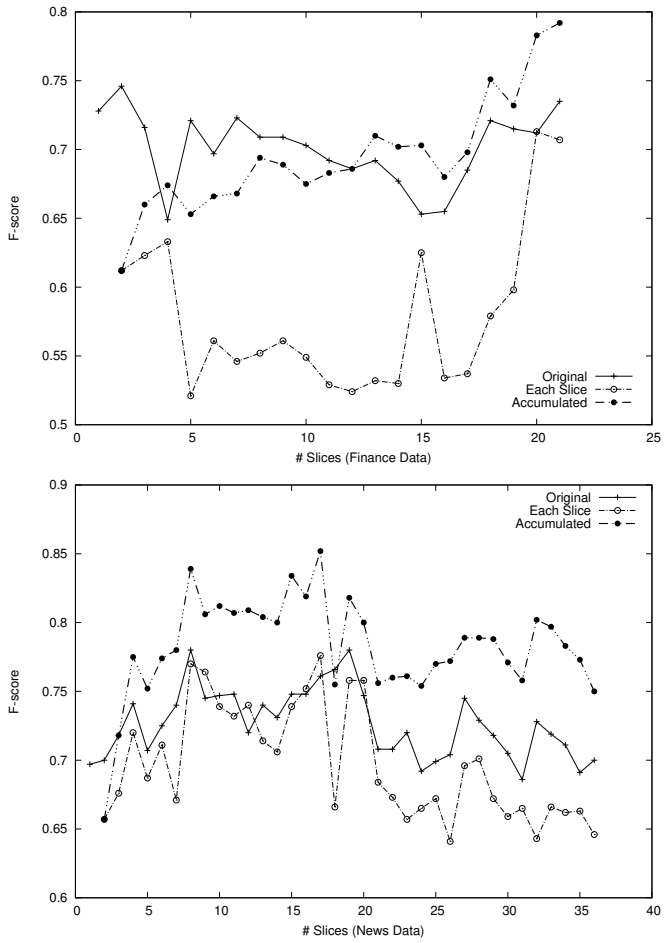

Figure 1: Temporal Evaluations

every iteration. Next, we used our model to perform an analysis of hate speech over the course of one year, providing practical insight into how much data and what kind of data is necessary for this task.

Most work has so far focused on abuse found in *English*, but it remains to be seen how our approach or any of the other prior approaches would fare in other languages. Given how powerful the two n-gram features were in English, these would probably fare well in other languages given enough training data.

Another area of future work includes using the context of the comment as additional features. The context could include the article it references, any comments preceding or replied to, as well as information about the commenter's past behavior or comments.

Acknowledgements

The authors would like to thank the three anonymous reviewers for their helpful comments. We also want to thank our Yahoo colleagues in the Standard Moderation Platform and editorial groups for their insight, guidelines and labeling of data.

7. REFERENCES

[1] S. Brody and N. Diakopoulos. Cooooooooooooooollllllllllllll!!!!!!!!!!!!!! using word lengthening to detect sentiment in microblogs. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 562–570, Edinburgh, Scotland, UK., July 2011. Association for Computational Linguistics.

[2] M. D. Buhrmester, T. Kwang, and S. D. Gosling. Amazon's mechanical turk: A new source of inexpensive, yet high-quality, data? *Perspectives on Psychological Science*, 6(1):3–5, Jan 2011.

[3] Y. Chen, Y. Zhou, S. Zhu, and H. Xu. Detecting offensive language in social media to protect adolescent online safety. In *Privacy, Security, Risk and Trust (PASSAT), 2012 International Conference on and 2012 International Confernece on Social Computing (SocialCom)*, pages 71–80. IEEE, 2012.

[4] N. Djuric, H. Wu, V. Radosavljevic, M. Grbovic, and N. Bhamidipati. Hierarchical neural language models for joint representation of streaming documents and their content. In *International World Wide Web Conference (WWW)*, 2015.

[5] N. Djuric, J. Zhou, R. Morris, M. Grbovic, V. Radosavljevic, and N. Bhamidipati. Hate speech detection with comment embeddings. In *Proceedings of International World Wide Web Conference (WWW)*, 2015.

[6] M. Faruqui and C. Dyer. Non-distributional word vector representations. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 464–469, Beijing, China, July 2015. Association for Computational Linguistics.

[7] J. Horton, D. G. Rand, and R. J. Zeckhauser. *The online laboratory: Conducting experiments in a real labor market*. National Bureau of Economic Research Cambridge, Mass., USA, 2010.

[8] Q. Le and T. Mikolov. Distributed representations of sentences and documents. In T. Jebara and E. P. Xing, editors, *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1188–1196. JMLR Workshop and Conference Proceedings, 2014.

[9] B. Liu. *Sentiment Analysis and Opinion Mining*. Morgan Claypool Publishers, 2012.

[10] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013.

[11] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc., 2013.

[12] G. Paolacci, J. Chandler, and P. G. Ipeirotis. Running experiments on amazon mechanical turk. *Judgment and Decision Making*, 5(5):411–419, 2010.

[13] E. Pitler and A. Nenkova. Using syntax to disambiguate explicit discourse connectives in text. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 13–16, Suntec, Singapore, August 2009. Association for Computational Linguistics.

[14] S. Sood, J. Antin, and E. Churchill. Profanity use in online communities. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1481–1490. ACM, 2012.

[15] S. O. Sood, J. Antin, and E. F. Churchill. Using crowdsourcing to improve profanity detection. In *AAAI Spring Symposium: Wisdom of the Crowd*, 2012.

[16] M. Surdeanu, M. Ciaramita, and H. Zaragoza. Learning to rank answers to non-factoid questions from web collections. *Computational Linguistics*, 37:351–383, 2011.

[17] S. Suri and D. J. Watts. Cooperation and contagion in web-based, networked public goods experiments. *PloS One*, 6(3), 2011.

[18] W. Warner and J. Hirschberg. Detecting hate speech on the world wide web. In *Proceedings of the Second Workshop on Language in Social Media*, pages 19–26, Montréal, Canada, June 2012. Association for Computational Linguistics.

[19] B. Yang, W. Yih, X. He, J. Gao, and L. Deng. Embedding entities and relations for learning and inference in knowledge bases. *CoRR*, abs/1412.6575, 2014.

[20] D. Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, pages 189–196. Association for Computational Linguistics, 1995.

[21] D. Yin, Z. Xue, L. Hong, B. D. Davison, A. Kontostathis, and L. Edwards. Detection of harassment on web 2.0. *Proceedings of the Content Analysis in the WEB*, 2:1–7, 2009.