Using Machine Learning to Detect Cyberbullying

Kelly Reynolds

Spring 2012

Submitted to the faculty of Ursinus College in fulfillment of the requirements for Distinguished Honors in Computer Science

Distinguished Honors Signature Page

Kelly Reynolds

Using Machine Learning to Detect Cyberbullying

Advisor: _____

April Kontostathis

Committee Members:

_____

April Kontostathis

_____

Akshaye Dhawan

_____

Lynne Edwards

Outside Evaluator:

_____

John P. Dougherty

Approved:

_____

Mohammed Yahdi

Abstract— Cyberbullying is the use of technology as a medium to bully someone. Although it has been an issue for many years, the recognition of its impact on young people has recently increased. Social networking sites, such as MySpace, Facebook, and Formspring.me, provide a fertile medium for bullies. Teens and young adults who use these sites are vulnerable to attacks. Through machine learning, we can detect language patterns used by bullies and their victims, and develop rules to automatically detect cyberbullying content.

The data we used for our project was collected from the website Formspring.me, a question-and-answer formatted website that contains a high percentage of bullying content. The data was labeled using a web service, Amazons Mechanical Turk. In order to test the data we collected, we utilized two main methods of machine learning: rule based learning and a bag-of-words approach. We used the labeled data, in conjunction with machine learning techniques provided by the Weka tool kit, to train a computer to recognize bullying content. Both a C4.5 decision tree learner and an instance-based learner were able to identify the true positives with 78.5% accuracy. The best result from our bag-of-words approach yielded a 40% recall and 30.6% rank-464 statistic.

I. INTRODUCTION

Social networking sites are great tools for connecting with people. Posting links, sharing pictures and videos, creating groups, and creating events are all great ways to extend communication with peers. However, as social networking has become widespread, people are finding illegal and unethical ways to use these communities. We see that people, especially teens and young adults, are finding new ways to bully one another over the Internet. Close to 25% of parents in a study conducted by Symantec in 2011 reported that, to their knowledge, their child has been involved in a cyberbullying incident [1]. The number of children who have been involved with cyberbullying is probably higher than this. The goal of this work is to find ways to detect instances of cyberbullying by creating a language model based on the text in online posts.

There are no well-known datasets for research on cyberbullying. A set of large datasets was made available at the Content Analysis on the Web 2.0 (CAW 2.0) workshop for a misbehavior detection task, however this dataset was unlabeled (i.e. it is not known which posts actually contain cyberbullying). Furthermore, the data was pulled from a variety of sources, and, with the exception of the data from Kongregate (a gaming website), the datasets appear to be discussions among adults. In order to conduct the study reported herein, we developed our own labeled dataset containing data from a webcrawl of Formspring.me. Formspring.me was chosen because the site is populated mostly by teens and college students, and there is a high percentage of bullying content in the data.

A variety of lexical features were extracted from the Formspring.me post data and several data mining algorithms that are available in the Weka toolkit were used to develop a model for the detection of cyberbullying. Our goals when developing the model were two-fold. First, we wanted a model that could be replicated in code. Decision-tree and rule-based learners were preferred because they provide an explicit representation of the model generated that can be easily reproduced in code. Second, we were focused more on recall, the number of positives the model returns, than on precision, the number of correct posts. In other words, were willing to accept some false positives in order to increase the percentage of true positives that could be identified by the tool.

II. BACKGROUND AND RELATED WORK

Patchin and Hinduja define cyberbullying as willful and repeated harm inflicted through the medium of electronic text [2]. A review of the adolescent psychology literature reveals nine different types of cyberbullying that can be distinctly identified [2], [3], [4]. These types are: flooding, masquerade, flaming, trolling, harassment, cyberstalking, denigration, outing, and exclusion. The types of bullying are defined as follows:

- **Flooding** consists of the bully monopolizing the media so that the victim cannot post a message [10].
- **Masquerade** involves the bully logging in to a website, chat room, or program using another user's screenname to either bully a victim directly or damage the victim's reputation [6].
- **Flaming**, or **bashing**, involves two or more users attacking each other on a personal level. The conversation consists of a heated, short lived argument, and there is bullying language in all of the users' posts [6].
- **Trolling**, also known as **baiting**, involves intentionally posting comments that disagree with other posts in an emotionally charged thread for the purpose of provoking a fight, even if the comments don't necessarily reflect the poster's actual opinion [7].
- **Harassment** most closely mirrors traditional bullying with the stereotypical bully-victim relationship. This type of cyberbullying involves repeatedly sending offensive messages to the victim over an extended period of time [6].

- **Cyberstalking** and **cyberthreats** involve sending messages that include threats of harm, are intimidating or very offensive, or involve extortion [6].
- **Denigration** involves gossiping about someone online. Writing vulgar, mean, or untrue rumors about someone to another user, or posting them to a public community or chat room or website, falls under denigration [8].
- **Outing** is similar to denigration, but requires the bully and the victim to have a close personal relationship, either online or in person. It involves posting private, personal or embarrassing information in a public chat room or forum [6].
- **Exclusion**, or ignoring the victim in a chat room or conversation, was the type of cyberbullying reported to have happened most often among youth and teens [8].

Very few other research teams are working on the detection of cyberbullying. As mentioned earlier, a misbehavior detection task was offered by the organizers of CAW 2.0, but only one submission was received. Yin, et. al determined that the baseline text mining system (using a bag-of-words approach) was significantly improved by including sentiment and contextual features. Even with the combined model, a support vector machine learner could only produce a recall level of 61.9% [5].

A recent paper describes similar work is that is being conducted at Massachusetts Institute of Technology. The research is aimed towards detecting cyberbullying through textual context in YouTube video comments. The first level of classification is to determine if the comment is in a range of sensitive topics such as

sexuality, race/culture, intelligence, and physical attributes. The second level is determining what topic. The overall success off this experiment was 66.7% accuracy for detecting instances of cyberbullying in YouTube comments. This project also used a support vector machine learner [9].

III. DATA COLLECTION

In this section we describe the collection and labeling of the data we used in our experiments.

A. Dataset Origin

The website Formspring.me is a question and answer based website where users openly invite others to ask and answer questions. What makes this site especially prone to cyberbullying is the option for anonymity. Formspring.me allows users to post questions anonymously to any other user's page. Some instances of bullying found on Formspring.me include: "Q: Your face is nasty. A: your just jealous" and "Q: youre one of the ugliest bitches Ive ever fucking seen. A: have you seen your face lately because if you had you wouldn't be talkin hun (:".

To obtain this data, we crawled a subset of the Formspring.me site and extracted information from the sites of 18,554 users. The users we selected were chosen randomly. The XML files that were created from the crawl ranged in size from 1 post to over 1000 posts. For each user we collected the following profile information: date the page was created, userid, name, link(s) to other sites, location, and biography.

The name, links and biography data were manually entered by the user who created the page (the Formspring.me account) and we cannot verify the validity of the information in those fields. In addition to the profile information, we collected the following information from each Question/Answer interaction: Asker Userid, Asker Formspring page, Question, and Answer.

B. Labeling the data

We extracted the question and answer text from the Formspring.me data for 50 files, representing 50 formspring pages (50 unique users). These files were chosen randomly from the set of 18,554 users that were crawled. We organized the 50 users into 5 sets of 10 users, and we ensured that there was no overlap between the sets. The first set was used as the original training model for the rule-based learning. The second was used as a test set. We created the additional three sets to train the model for the anonymity section of the rule-based learning. We used the same procedure to identify class labels in all the data sets.

We used Amazon's Mechanical Turk service to determine the labels for our truth sets. Mechanical Turk is an online marketplace that allows requestors to post tasks (called HITs) which are then completed by paid workers. The workers are paid by the requestors per HIT completed. The process is anonymous (the requestor cannot identify the workers who answered a particular task unless the worker chooses to reveal him/herself). The amount offered per HIT is typically small. We paid three workers .05 cents each to label each post. Each HIT we posted displayed a Question and Answer from the Formspring crawl and a web form that requested the following information:

1. Does this post contain cyberbullying (Yes or No)?
2. On a scale of 1 (mild) to 10 (severe) how bad is the cyberbullying in this post (enter 0 for no cyberbullying)?
3. What words or phrases in the post(s) are indicative of the cyberbullying (enter n/a for no cyberbullying)?

4. Please enter any additional information you would like to share about this post.

The primary advantage to using Mechanical Turk is that it is very, very quick. Our data set was labeled within hours. We asked three workers to label each post because the identification of cyberbullying is a subjective task. Our class labels were ”yes” for a post containing cyberbullying and ”no” for a post without cyberbullying. The data provided by the other questions will be used for future work. At least two of the three workers had to agree in order for a post to receive a final class label of “yes” in our training and testing sets.

From 50 files, we ended up with 13652 posts, 792 of which contained cyberbullying (5.8%). Each of these files represents a Formspring.me page for one user ID. The posts on that page are of other users asking questions to the page owner. These ratios confirmed our suspicion that the percentage of cyberbullying in the Formspring.me data was much higher than in other datasets that we’ve seen.

IV. RULE BASED LEARNING

The first method we used to develop a model for detecting cyberbully was Rule-Based-Learning. We initially developed a model based solely on textual features in the question and answer fields extracted from Formspring.me. We then enhanced the model by adding in an anonymity parameter. We believed that users would be more likely to bully someone if the recipient did not know who was bullying them. Our final experiment was to utilize a bag-of-words approach which will be discussed later in this thesis.

A. Developing Features for Input

This section describes the identification and extraction of features from each Formspring post. We were determined to avoid a bag-of-words approach at first, for several reasons. First, the feature space with a bag-of-words approach is very large. Second, we wanted to be able to reproduce the model in code, and having each term as a feature would make that impractical as well as useless. Third, we wanted to be able to understand why a post was considered as containing cyberbullying as this will inform the development of a communicative model for cyberbullying detection.

One thing was clear from the labeling project; there are "bad" words that make a post more likely to be labeled as cyberbullying. In order to leverage this information, we identified a list of insult and swear words, posted on the website www.noswearing.com. This list, containing 296 terms, was downloaded and each word on the list was given a severity level by our team. The levels were 100 (ex. butt, idiot), 200 (ex. trash, prick), 300 (ex. asshole, douchebag), 400 (ex. fuckass, pussy),

and 500 (ex. buttfucker, cuntass). The classification of these terms into severity levels was subjective.

We were interested in both the number of "bad" words (NUM) and the density of "bad" words NORM) as features for input to the learning tool. We extracted textual feature data in two different ways, one containing the count information, and one containing normalized information. We normalized by simply dividing the number of words at each severity level by the total number of words in the post, and then multiplying by 100 to get an integer value (for example, if there were 6 100-level words in a 10 word post, the 100-level would be reported as 600).

We also generated a feature to assign a value to the severity of the language of a post. We call this feature SUM and computed it as follows:

SUM = 100 * NUM100 + 200 * NUM200 + 300 * NUM300

+ 400 * NUM400 + 500 * NUM500

To reiterate, the following features were extracted:

1. The number of curse and insult words at each level (NUM100, NUM200, etc.)
2. The percentage of curse and insult words in the post at level (NORM100, NORM200, etc.)
3. The total number of words in the post (TOTAL)
4. The weighted sum of the curse and insult words (SUM)

The SUM and TOTAL features were included in both the NUM and the NORM versions of our datasets. The class label (YES, NO) was also extracted from the Mechanical Turk file and included in the input to the machine learning tool.

Once we explored the text-based features of the data, we also extracted the anonymity feature. From the xml files, we were able to determine if the user asking the question provided their username. If the field was empty, the anonymity field in our extracted data was set to true.

B. Learning the Model

Weka is a software suite for machine learning that creates models using a wide variety of well-known algorithms [10]. We identified the following algorithms as most useful for our project.

- J48: The J48 option uses the C4.5 algorithm to create a decision tree model from the attributes provided. An immediate benefit provided by a decision tree learner is that we can easily convert the tree into a software product [10]. When working with decision trees, it is important to consider the size of the tree that is generated, as well as the accuracy of the model. A large, complex tree may be overfitted to the data. A small, simple tree may indicate that the training set is not well balanced and the model cannot be clearly identified.
- JRIP: JRIP is a rule based algorithm that creates a broad rule set then repeatedly reduces the rule set until it has created the smallest rule set that retains the same success rate. Like the decision tree learner, the rules are output and can be used to develop software [9].
- IBK: Instance-based learning is the simplest of all algorithms. The IBK algorithm implemented in Weka is a *k*-nearest neighbor approach. The algorithm works by placing each training instance into an *n*-dimensional vector space, where *n* is the number of features in the training set. The test

instances are then mapped into that same space and cosine similarity is used to identify the training vector(s) that are closest to each test vector. A voting system among the *k* nearest neighbors is used to determine the class label for each test instance. An odd value for *k* is generally preferred because algorithm will use random selection to break ties [10]. We used the IBK method with *k* = 1 and *k* = 3.

- SMO: We wanted to use a support vector machine algorithm for testing also. The papers in the related work section both found reasonable success with support vector machines and SMO is a function-based support vector machine algorithm based on sequential minimal optimization [11]. The disadvantage of using support vector machines is that no representation of the model is provided as output and duplication of the algorithm in order to program the model is not straight-forward. Support vector machine learning is more useful when the feature space is large. SMO was the least successful algorithm in our experiments.

C. Class Weighting

Less than 6% of the training data contained cyberbullying. As a result, the learning algorithms are generated a lot of false negatives (i.e. they can reach accuracy figures of over 94% by almost ignoring the cyberbullying examples). As discussed earlier, we are interested primarily in recall. We would prefer to have innocent posts labeled as cyberbullying (false positives) instead of mislabeling cyberbullying posts as innocent (false negatives).

In order to overcome the problem of sparsicity for the positive instances, we increased the weight of these instances in the dataset. We did this by simply copying the positive training examples multiple times in order to balance the training set and provide an incentive for the learners to identify the true positives.

D. Evaluation

We used two evaluation approaches in our experiments. Our first approach was cross validation, used to evaluate learning algorithms when the amount of labeled data available is small. Cross validation experiments allow for the use of the maximum amount of data for training. The approach is simple. The modeling and evaluation routines iterate *n* times in *n*-fold cross validation. In each iteration 1/*n* percent of the data is held for evaluation and the remaining 1-1/*n* is used to develop the model. The held data is selected using stratification, so that the training and test sets retain approximately the same number of instances in each class as the entire dataset. The selection process also ensures that each instance will be held as part of the test set in one iteration. The statistics reported are the average accuracy figures computed across all *n* iterations. The final model is generated using all instances. Ten-fold cross validation is considered to be the standard approach to evaluation in many machine learning experiments. We report the ten-fold cross validation results for our experiments in the next section.

Our second approach was to develop an independent set to test the success of the model. Using this method, we created a model based on the first data set, called the training set because it creates the model for detecting bullying. We then created an additional set, called the test set, to test the data. This set was created

independently from the first set, with the stipulation that none of the same posts would be in these two sets. We then used the Weka tool kit to run an experiment on the data, using the training set to create the models, and using the test set to evaluate them.

E Results

In this section we describe and discuss our ability to accurately predict instances of cyberbullying in Formspring.me data using a rule-based learner.

i. Nums vs. Norms

This section discusses the results of the NUM and NORM experiments. These experiments were run on the first 10 files, representing user IDs, extracted. This dataset was 2696 instances, 195 of which contain bullying (7.2%).

		NUM Data Set									
		Times Positive Results Repeated									
		1	2	3	4	5	6	7	8	9	10
J48	% correctl	54.40	63.60	72.30	75.60	76.10	76.50	77.70	78.20	78.50	78.50
	GUESS	7.23	13.49	18.96	23.77	28.05	31.87	35.31	38.41	31.24	43.81
	DIFF	47.17	50.11	53.34	51.83	48.05	44.63	42.39	39.79	47.26	34.69
	# of Leave	14	11	7	2	2	2	5	6	26	26
	size of tre	27	21	13	3	3	3	9	11	51	51
JRIP	% correctl	56.90	70.00	75.60	75.10	76.10	76.00	76.80	76.90	77.00	77.30
	GUESS	7.23	13.49	18.96	23.77	28.05	31.87	35.31	38.41	31.24	43.81
	DIFF	49.67	56.51	56.64	51.33	48.05	44.13	41.49	38.49	45.76	33.49
	# of Rules	3	6	6	2	2	5	7	4	3	4
IBK1	% correctl	55.40	75.40	78.10	78.10	78.50	78.50	78.50	78.50	78.50	78.50
	GUESS	7.23	13.49	18.96	23.77	28.05	31.87	35.31	38.41	31.24	43.81
	DIFF	48.17	61.91	59.14	54.33	50.45	46.63	43.19	40.09	47.26	34.69
IBK3	% correctl	54.90	62.30	74.50	77.40	78.50	78.50	78.50	78.50	78.50	78.50
	GUESS	7.23	13.49	18.96	23.77	28.05	31.87	35.31	38.41	31.24	43.81
	DIFF	47.67	48.81	55.54	53.63	50.45	46.63	43.19	40.09	47.26	34.69
SMO	% correctl	36.90	53.80	61.30	61.50	62.20	63.10	65.00	66.70	67.20	67.20
	GUESS	7.23	13.49	18.96	23.77	28.05	31.87	35.31	38.41	31.24	43.81
	DIFF	29.67	40.31	42.34	37.73	34.15	31.23	29.69	28.29	35.96	23.39

Table 1 : The TP percentages for the NUM training set using J48, JRIP, IBK, and SMO algorithms using 10-fold cross validation

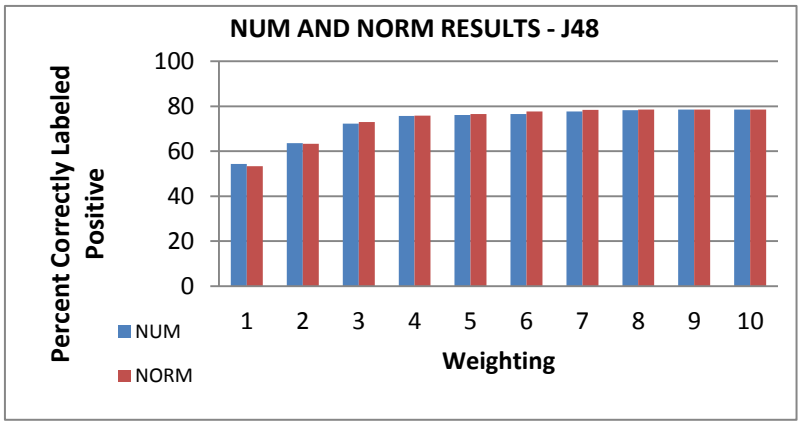

Figure 2 : Comparing the results for each repetition of the J48 algorithm for the NUM and NORM Data Sets

		NORM Data Set									
		Times Positive Results Repeated									
		1	2	3	4	5	6	7	8	9	10
J48	% correctl	53.30	63.30	73.00	75.80	76.50	77.60	78.30	78.50	78.50	78.50
	GUESS	7.23	13.49	18.96	23.77	28.05	31.87	35.31	38.41	31.24	43.81
	DIFF	46.07	49.81	54.04	52.03	48.45	45.73	42.99	40.09	47.26	34.69
	# of Leave	11	12	18	14	11	13	13	13	6	6
	size of tre	21	23	35	27	21	25	25	25	11	11
JRIP	% correctl	57.40	66.40	74.50	75.40	75.60	75.90	76.60	77.20	77.50	77.00
	GUESS	7.23	13.49	18.96	23.77	28.05	31.87	35.31	38.41	31.24	43.81
	DIFF	50.17	52.91	55.54	51.63	47.55	44.03	41.29	38.79	46.26	33.19
	# of Rules	4	4	2	2	6	3	3	3	3	3
IBK1	% correctl	54.40	75.40	78.10	78.10	78.50	78.50	78.50	78.50	78.50	78.50
	GUESS	7.23	13.49	18.96	23.77	28.05	31.87	35.31	38.41	31.24	43.81
	DIFF	47.17	61.91	59.14	54.33	50.45	46.63	43.19	40.09	47.26	34.69
IBK3	% correctl	52.80	65.50	76.10	77.70	78.50	78.50	78.50	78.50	78.50	78.50
	GUESS	7.23	13.49	18.96	23.77	28.05	31.87	35.31	38.41	31.24	43.81
	DIFF	45.57	52.01	57.14	53.93	50.45	46.63	43.19	40.09	47.26	34.69
SMO	% correctl	46.20	54.10	59.00	61.20	63.40	63.90	64.50	65.10	65.20	67.50
	GUESS	7.23	13.49	18.96	23.77	28.05	31.87	35.31	38.41	31.24	43.81
	DIFF	38.97	40.61	40.04	37.43	35.35	32.03	29.19	26.69	33.96	23.69

Table 2 : The TP percentages for the NORM training set using J48, JRIP, IBK, and SMO algorithms using 10-fold cross validation

We compare the results using the NUM training set to the NORM training set in Tables 1 and 2, respectively. These tables report the recall, or percent of true bullying posts labeled bullying, for identifying cyberbullying using 10-fold cross validation. We see that NORM training set outperforms the NUM training set for all repetitions and for all algorithms, with the exception of the SMO algorithm. In Figure 1, we see that as the weighting of positive instances increases, the NORM success rates are slightly higher than the NUM success rates. For SMO, which is our least successful

algorithm, NORM outperforms NUM for all repetitions except the tenth. We conclude from these results that the percentage of "bad" words in a post is more indicative of cyberbullying than a simple count.

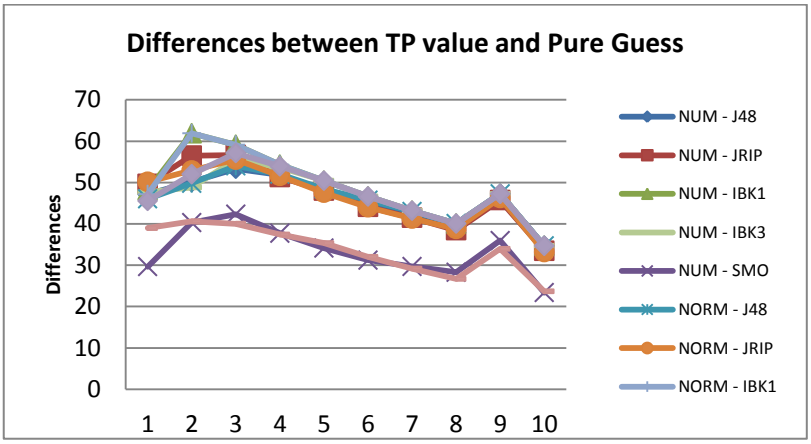

Figure 2: Depicts the DIFF value defined as DIFF= TP-GUESS for NUM and NORM algorithms J48, JRIP, IBK, and SMO

In Tables 1 and 2, we see that the algorithms that reach the maximum success rate are the J48 algorithm, the IBK1, and the IBK3 algorithms. The SMO algorithm is the least successful of the algorithms. The JRIP accuracy figures are also good, and the rules would be an easy model to program; however, the rule sets are extremely small and over simplified. Although IBK1 and IBK3 produce the peak success rate, the IBK model is difficult to reproduce in a program since the model is based on a vector space.

Another metric we decided to measure in order to assess the success of our results was the difference between the true positive percentages and the estimated success we would have if we used a pure guess. The difference value is denoted as "DIFF" and the guess value, "GUESS" in Tables 1 and 2. As shown in Figure 2, the peak difference for most algorithms is between 2 and 4 repetitions. All of the

algorithms have a local maximum at 9 repetitions. This metric gives us an idea of how much more successful the model is at determining if an instance contains cyberbullying. The best increase from GUESS to the TP rate was the IBK1 result for both the NUM and NORM 2 repetition model with a 61.0% increase. Although that is a higher increase than the other models, the TP rates on later repetitions greatly outperform the 2 repetition models and also still have a large increase in success from GUESS.

```
SUM <= 100
|   SUM <= 0: No (2689.0/336.0)
|   SUM > 0
|   |   NORM 100 <= 4
|   |   |   NORM 100 <= 2
|   |   |   |   TOTAL WORDS <= 62
|   |   |   |   |   TOTAL WORDS <= 57
|   |   |   |   |   |   TOTAL WORDS <= 35: Yes (12.0/4.0)
|   |   |   |   |   |   TOTAL WORDS > 35: No (14.0)
|   |   |   |   |   TOTAL WORDS > 57: Yes (8.0)
|   |   |   |   TOTAL WORDS > 62: No (9.0)
|   |   |   NORM 100 > 2: No (7.0)
|   |   NORM 100 > 4
|   |   |   TOTAL WORDS <= 19
|   |   |   |   NORM 100 <= 5: No (4.0)
|   |   |   |   NORM 100 > 5
|   |   |   |   |   NORM 100 <= 10
|   |   |   |   |   |   NORM 100 <= 9
|   |   |   |   |   |   |   TOTAL WORDS <= 15: No (5.0)
|   |   |   |   |   |   |   TOTAL WORDS > 15: Yes (9.0/1.0)
|   |   |   |   |   |   NORM 100 > 9: Yes (8.0)
|   |   |   |   |   NORM 100 > 10: No (2.0)
|   |   |   TOTAL WORDS > 19: Yes (8.0)
SUM > 100: Yes (1286.0/102.0)
```

Figure 3: Decision Tree for NORM weight 8 using J48

We also developed an independent test set to evaluate the model. Both the training set and test set are based on 10 user IDs each. The test set was 1219 instances, 172 of which contain bullying (14.1%). When testing the model created with the 8-weight NORM training set and J48 algorithm, we obtained a true positive accuracy of 61.6% and an overall accuracy of 81.7%, using the independent test set. Interestingly, the smaller tree produced by the 9 and 10 repetition data seems to hold up better with an independent test set. This indicates that the tree in Figure 3 may be overfitted to the training data. A close analysis of both trees (Figures 3 and 4) tells an interesting story. The smaller tree relies only on the SUM and the TOTAL WORDS features. The larger

```
SUM <= 100
| SUM <= 0: No (2731.0/378.0)
| SUM > 0
| | TOTAL WORDS <= 62
| | | TOTAL WORDS <= 57
| | | | TOTAL WORDS <= 35: Yes (59.0/23.0)
| | | | TOTAL WORDS > 35: No (14.0)
| | | TOTAL WORDS > 57: Yes (9.0)
| | TOTAL WORDS > 62: No (9.0)
SUM > 100: Yes (1434.0/102.0)
```

Figure 4: Decision Tree for NORM weight 9

one also relies on the percentage of 100-level words in the post. It seems counter-intuitive that the 100-level words are most indicative of cyberbullying. Perhaps those words are just used more commonly than some of the esoteric insults that appear at the 500 level.

ii. Anonymity

This section discusses the results of the experiments that explored the prevalence of anonymity of the question-asker. These results are based on the full 50 files in the data set.

In running these experiments, we determined a fair control would be to send the data through the algorithms both with and without the anonymity attribute. Additionally, we wanted to evaluate the number of repetitions and the results size. Because the data set was much larger in these instances, we repeated the positive values 1 through 10 times to focus the algorithm on labeling the positive results correctly. The results of these experiments can be seen in Tables 3, 4, 5, and 6. These are the NUMs with anonymity, NUMs without anonymity, NORMs with anonymity, and NORMs without anonymity, respectively.

		NUM Data Set - WITH ANON									
		Times Positive Results Repeated									
		1	2	3	4	5	6	7	8	9	10
J48	% correctly labeled positive	31.20	45.00	53.20	58.20	62.60	64.90	66.20	66.80	67.20	67.30
	GUESS	5.80	10.97	15.59	19.77	23.54	26.98	30.12	33.01	35.66	38.11
	DIFF	25.40	34.03	37.61	38.43	39.06	37.92	36.08	33.79	31.54	29.19
	# of Leaves	27	71	84	86	112	130	174	177	175	172
	size of tree	51	132	157	162	212	274	329	336	329	375
JRIP	% correctly labeled positive	27.80	41.70	52.20	58.10	61.90	63.50	63.50	64.10	64.40	65.30
	GUESS	5.80	10.97	15.59	19.77	23.54	26.98	30.12	33.01	35.66	38.11
	DIFF	22.00	30.73	36.61	38.33	38.36	36.52	33.38	31.09	28.74	27.19
	# of Rules	8	10	14	26	23	27	22	22	16	12
IBK1	% correctly labeled positive	33.30	55.90	62.50	65.70	66.50	67.60	67.80	68.00	68.00	68.20
	GUESS	5.80	10.97	15.59	19.77	23.54	26.98	30.12	33.01	35.66	38.11
	DIFF	27.50	44.93	46.91	45.93	42.96	40.62	37.68	34.99	32.34	30.09
IBK3	% correctly labeled positive	33.00	47.60	60.30	65.60	66.40	67.60	67.80	68.00	68.00	68.20
	GUESS	5.80	10.97	15.59	19.77	23.54	26.98	30.12	33.01	35.66	38.11
	DIFF	27.20	36.63	44.71	45.83	42.86	40.62	37.68	34.99	32.34	30.09

Table 3: The TP percentages and differences for the NUM training set using J48, JRIP, and IBK algorithms. These results include the anonymous feature.

	NUM Data Set										
		Times Positive Results Repeated									
		1	2	3	4	5	6	7	8	9	10
J48	% correctly labeled positive	30.70	44.10	52.10	56.70	60.40	64.80	65.50	66.20	66.70	67.10
	GUESS	5.80	10.97	15.59	19.77	23.54	26.98	30.12	33.01	35.66	38.11
	DIFF	24.90	33.13	36.51	36.93	36.86	37.82	35.38	33.19	31.04	28.99
	# of Leaves	22	44	48	48	64	68	81	91	90	110
	size of tree	43	87	95	95	127	135	161	181	179	219
JRIP	% correctly labeled positive	24.60	39.60	52.20	57.80	61.60	63.70	64.60	64.50	64.50	65.00
	GUESS	5.80	10.97	15.59	19.77	23.54	26.98	30.12	33.01	35.66	38.11
	DIFF	18.80	28.63	36.61	38.03	38.06	36.72	34.48	31.49	28.84	26.89
	# of Rules	7	10	13	23	22	20	13	23	16	13
IBK1	% correctly labeled positive	33.30	53.00	59.80	63.30	64.60	66.30	67.00	67.30	67.40	67.70
	GUESS	5.80	10.97	15.59	19.77	23.54	26.98	30.12	33.01	35.66	38.11
	DIFF	27.50	42.03	44.21	43.53	41.06	39.32	36.88	34.29	31.74	29.59
IBK3	% correctly labeled positive	32.40	45.90	58.20	63.00	64.60	66.30	67.00	67.30	67.40	67.70
	GUESS	5.80	10.97	15.59	19.77	23.54	26.98	30.12	33.01	35.66	38.11
	DIFF	26.60	34.93	42.61	43.23	41.06	39.32	36.88	34.29	31.74	29.59

Table 4: The TP percentages and differences for the NUM training set using J48, JRIP, and IBK algorithms. These results do not include the anonymous feature.

	NORM Data Set - WITH ANON										
		Times Positive Results Repeated									
		1	2	3	4	5	6	7	8	9	10
J48	% correctly labeled positive	29.20	47.50	53.30	59.10	63.10	65.60	66.60	67.20	67.40	67.40
	GUESS	5.80	10.97	15.59	19.77	23.54	26.98	30.12	33.01	35.66	38.11
	DIFF	23.40	36.53	37.71	39.33	39.56	38.62	36.48	34.19	31.74	29.29
	# of Leaves	41	101	117	135	154	172	190	202	206	208
	size of tree	77	192	222	256	293	327	362	384	389	395
JRIP	% correctly labeled positive	29.90	43.40	53.30	59.10	62.10	62.50	63.70	64.00	65.40	65.30
	GUESS	5.80	10.97	15.59	19.77	23.54	26.98	30.12	33.01	35.66	38.11
	DIFF	24.10	32.43	37.71	39.33	38.56	35.52	33.58	30.99	29.74	27.19
	# of Rules	7	10	18	18	16	15	23	25	23	14
IBK1	% correctly labeled positive	34.10	55.50	62.70	65.70	66.50	67.60	67.80	68.00	68.00	68.20
	GUESS	5.80	10.97	15.59	19.77	23.54	26.98	30.12	33.01	35.66	38.11
	DIFF	28.30	44.53	47.11	45.93	42.96	40.62	37.68	34.99	32.34	30.09
IBK3	% correctly labeled positive	31.60		60.10	65.20	66.40	67.60	67.80	68.00	68.00	68.20
	GUESS	5.80	10.97	15.59	19.77	23.54	26.98	30.12	33.01	35.66	38.11
	DIFF	25.80	-10.97	44.51	45.43	42.86	40.62	37.68	34.99	32.34	30.09

Table 5: The TP percentages and differences for the NORM training set using J48, JRIP, and IBK algorithms. These results include the anonymous feature.

	NORM Data Set										
		Times Positive Results Repeated									
		1	2	3	4	5	6	7	8	9	10
J48	% correctly labeled positive	29.50	45.80	52.60	56.10	60.80	65.10	65.50	66.80	67.00	67.30
	GUESS	5.80	10.97	15.59	19.77	23.54	26.98	30.12	33.01	35.66	38.11
	DIFF	23.70	34.83	37.01	36.33	37.26	38.12	35.38	33.79	31.34	29.19
	# of Leaves	41	64	77	94	114	109	114	118	119	123
	size of tree	81	127	153	187	227	217	227	235	237	245
JRIP	% correctly labeled positive	27.00	43.80	53.30	58.70	61.30	63.10	63.90	64.30	65.40	65.30
	GUESS	5.80	10.97	15.59	19.77	23.54	26.98	30.12	33.01	35.66	38.11
	DIFF	21.20	32.83	37.71	38.93	37.76	36.12	33.78	31.29	29.74	27.19
	# of Rules	8	6	16	19	23	17	21	11	16	20
IBK1	% correctly labeled positive	31.90	52.50	59.80	63.30	64.60	66.30	67.00	67.30	67.40	67.70
	GUESS	5.80	10.97	15.59	19.77	23.54	26.98	30.12	33.01	35.66	38.11
	DIFF	26.10	41.53	44.21	43.53	41.06	39.32	36.88	34.29	31.74	29.59
IBK3	% correctly labeled positive	30.90	45.30	57.80	62.90	64.40	66.30	67.00	67.30	67.40	67.70
	GUESS	5.80	10.97	15.59	19.77	23.54	26.98	30.12	33.01	35.66	38.11
	DIFF	25.10	34.33	42.21	43.13	40.86	39.32	36.88	34.29	31.74	29.59

Table 6: The TP percentages and differences for the NORM training set using J48, JRIP, and IBK algorithms. These results do not include the anonymous feature.

The results from this experiment are lower than those of the NUMs vs. NORMs experiment. This can be attributed to the larger, less dense data set. However, we can see that the anonymous attribute does have a positive influence on the percent of

correctly labeled positive instances. As seen in Figure 4, the results using the anonymous feature are consistently higher than those without the anonymous feature. Although this is true, we were surprised to see that the difference was not as high as we expected it to be.

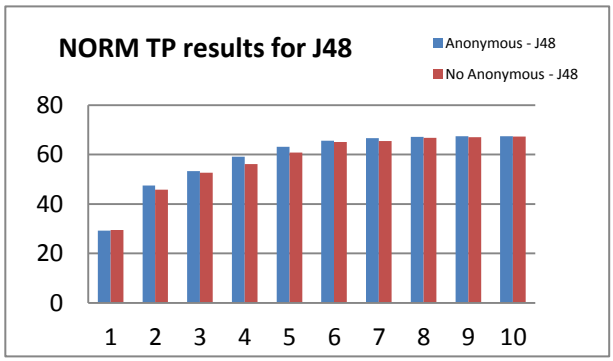

Figure 4: Comparing anonymous posts to non-anonymous posts

In order to further examine the success of the algorithm, we contrasted the TP results with the percent of instances that would be labeled positive through random guess. Again, we examined the DIFF metric described above. We can see in Figures 5 and 6 that there is a peak of the difference between 4 and 6 repetitions. This would indicate that these models are having the greatest increase in success and would be a good choice for the final model we choose.

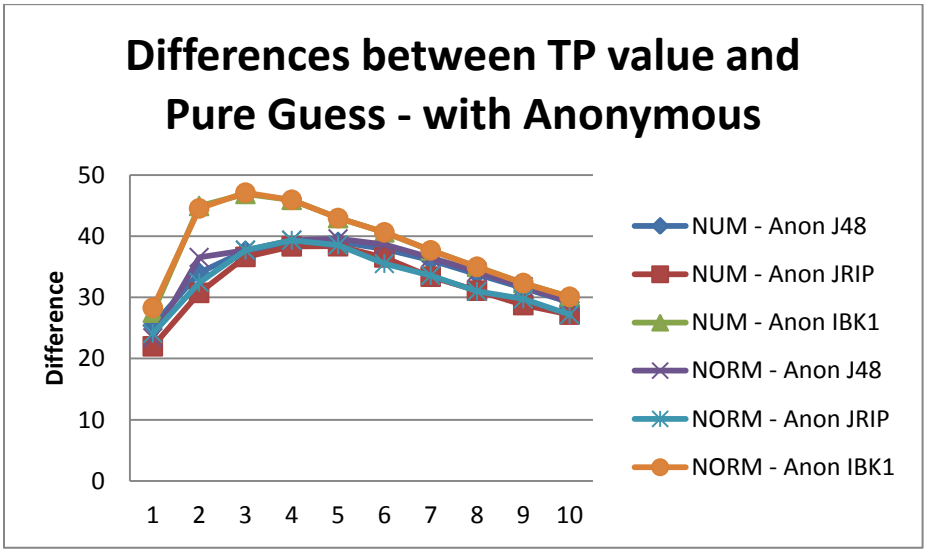

Figure 5: Depicts the DIFF value defined as DIFF= TP-GUESS for NUM and NORM algorithms J48, JRIP, and IBK1 using the anonymity feature

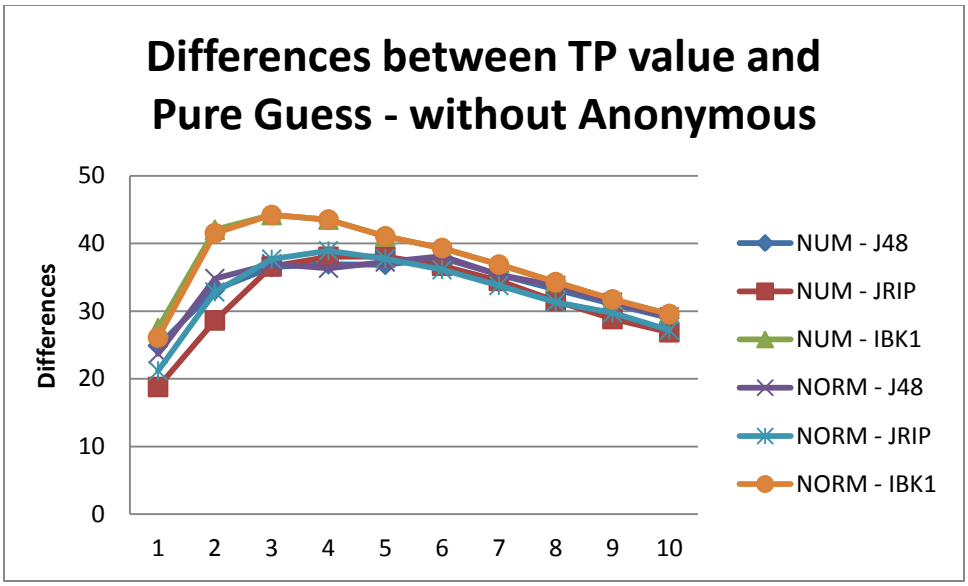

Figure 6: Depicts the DIFF value defined as DIFF= TP-GUESS for NUM and NORM algorithms J48, JRIP, and IBK1 using the anonymity feature

F. Discussion

In the NUMs vs. NORMs experiment, we used a language-based method of detecting cyberbullying. By recording the number of curse and insult words at each severity level, we were able to correctly identify 78.5% of the posts that contain cyberbullying. Our results indicate that text analysis is an effective method for determining posts containing cyberbullying.

The anonymity experiment resulted in a lower true positive rate for both the NUMs vs. NORMs, and also provided evidence in favor of our hypothesis that anonymity is a factor in cyberbullying. The peak result from this set of experiments was a true positive rate of 68.2%, produced by the NORM IBK algorithm with 1and 3 nearest neighbors at 10 repetitions. However, taking into account the DIFF variable, the highest increase of success was by the NORM IBK algorithms at 3 repetitions with a 47.1% increase. We expect that these results are lower than the experiments in

the previous section because these results are based off the entire dataset of 13652 posts. This dataset has a lower percentage of bullying posts, contributing to the lower TP percents.

V. BAG-OF-WORDS

The next logical step for us to take was to move to a bag-of-words model. We decided it would give us a good look at what words were being used in the posts, especially those involving cyberbullying. Bag-of-words is a keyword based vector-space model. This model "recognizes that the use of binary weights is too limiting and proposes a framework in which partial matching is possible. [12]" Bag-of-words will allow us to create a matrix of the entire lexicon used in all 13,652 posts. We then use this matrix to run queries and determine the relative closeness of the query to each post in the matrix [12].

A. Developing the Model

To create the bag of words model, we used a compressed-row term by document sparse matrix. The matrix is comprised of all words used in the posts on the x-axis and all posts in the data set on the y-axis. The value in each element of the matrix is the number of times each word was used in that particular post. Suppose we have a set of 3 posts that use a total of 5 words: dogs, cats, and, run, lay. And suppose the posts are: "dogs run", "cats lay", and "dogs run and run." In Figure 7, we see that the matrix is mostly 0-values. Although this example is small, it illustrates the wasted space that representing the 0-values would create. Imagine this matrix with millions of words in the *x*-axis and 13,652 posts in the *y*-axis. It is easy to see that this matrix would have even a higher percentage of 0-values. Because most words will not be

	P1	P2	P3
dogs	1	0	1
cats	0	1	0
and	0	0	1
run	1	0	2
lay	0	1	0

Figure 7: Bag-of-Words model

used in most posts, this is a sparse matrix – a matrix with mostly null values. Storing the 0-values would be a waste of query time and space, and we used a compressed-row model to reduce memory and run time.

The compressed-row model uses a map that indicates the value in the y-axis as the key and the number of occurrences as the value. The map will only store a key and value when the value is greater than 0. The compressed-row model also has a row pointer list that indicates the starting location for each row in the map. This method compresses the matrix significantly. Using the example from above, the compressed row representation of this matrix is depicted in Figure 8.

value	1	1	1	1	1	2	1
col index	1	3	2	3	1	3	2

row pointer	1	3	4	5	7

Figure 8: Sparse Matrix version of the Bag-of-Words model example

B. Trimming the Model

Some words in the matrix will be unimportant data, for example, words that appear too often or infrequently in the total results. Words that appear too often are poor discriminators because when running a query on these terms, too many posts would be returned as relevant – clouding the results. In the running example, a word like "lay" or "and" may appear infrequently in the overall scheme, while "run" may be appearing too often. Although these aren't necessarily unimportant for the example, in a much larger scheme, having a word appear only once in a set of millions makes a difference. Additionally, if a certain word is appearing too many times in that same set, it may be clouding the data and is usually a commonly used word, such as "to," "you," "and," etc. By removing this noise from the data, we are able to focus more on the words that may be real indicators of bullying.

C. Querying the Data

In order to extract the useful information from the matrix we created, we need to query the matrix. Running a query will give us the relative closeness between the query vector we submit and each post that is represented as a column in the matrix. We will use vector of values for our query and each non-zero value in the query vector represents the weight we are giving to that query term. When the query vector is multiplied with the term by post matrix, a result vector will be produced. The result vector will have an entry for each post and the value will indicate how close the query vector is to that post vector. The matrix multiplication is similar to the result that would be obtained by normalizing each vector and taking the cosine of the angle between the matrix and the query vector. If these vectors coincide, the cosine will be 1, if they are far apart, the result will be near 0. Similarly, in our result matrix, larger values indicate posts that are more relevant to the query, and smaller or zero values indicate posts that are less relevant or non-relevant.

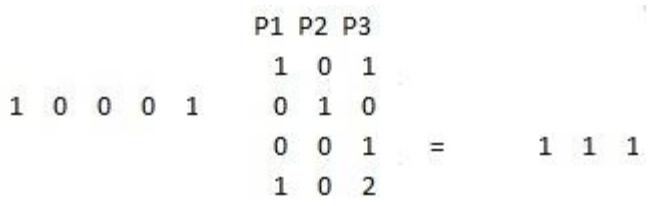

Figure 9: Matrix Multiplication of "dogs lay" vector.

i. How to Query

A query is a matrix multiplication function. Using our example matrix, if we want to find out how close the sentence "dogs lay" is to any of the sentences in the matrix, we would create a vector to multiply with the matrix, as pictured in Figure 9. The result gives us a vector of scores and each score corresponds to a post. We can then rank the posts and create a list of most relevant posts.

ii. What to Query

There are many avenues to explore when querying our matrix. We have identified three methods: querying based on posts we know to contain bullying individually, querying based on posts we know to contain bullying through vector combination, and querying based on our dictionary of curse and insult words. The first, querying on individual posts gives us the relative closeness of other posts to one that we know contains bullying. Positive results from this will indicate a pattern in word usage that determines bullying. This method is the example above in Figure 9.

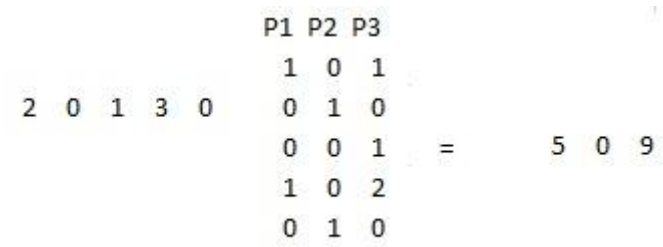

Figure 10: Matrix Multiplication of combined post 1 and 3.

The second method, querying on a combination of posts in a vector may bring more severe examples of bullying to the top of our results list. The query vector is created using vector addition of the posts we want to combine. Following our example from before, suppose we want to combine post 1 and post 3 to query, we would result in the vector query depicted in Figure 10.

The final method we used is to query on the dictionary of curse and insult words. This will give us a good indicator of what types of words are being used more often in posts, especially in those that contain bullying. We also can query on just certain levels of curse and insult words as well – allowing us a chance to compare the severity of words being used.

D. Results

This section discusses the results of the query experiments in the bag-of-words model. A description of each query can be found in Table 7. Each of the results tables (Tables 8-12) evaluates the success of the queries with 8 statistics that can be broken down into three categories:

1. Result>*x*: These results measure the number of posts that have a result value greater than *x* and the percentage of those that contain bullying. This statistic provides evidence for trimming the query results, as well as a score for how relevant the query results are to the query. We noted this metric for *x*=0, 5, 10 (i.e. the number of posts with any score greater than 0, the number with a score greater than 5, and the number with a score greater than 10).
2. Rank Statistic *n*: A rank statistic gives us *precision* at rank *n.* *Precision* is defined as the number of true positives divided by the number of results returned. Precision at rank *n* is the number of true positives with rank less than or equal to *n* (number of true positives returned within the top *n* highest scores). For example, if 100 posts had scores greater than 0, and 50 of these were true positives, the precision at rank 100 would be .5 (50%). If moving to rank 150 finds another 5 true positives, the precision at rank 150 would be 55/150 = .367 (36.7%). Because we used word counts, we often have many posts with the same score returned for a given query. If *n* falls in a group of scores that are all equal, we measure precision at rank *m*, where *m* is the rank for the last post in the "tie" situation. For example, if we have 150 posts with positive scores, and we want to measure precision at rank 100, but the documents ranked from 92 to 113 all have a score of 7, we will measure precision at rank 113. So if we have 34 true

positives within the top 113 scoring posts, the precision will be 34/113 = .300 (30.0%). We measured precision at rank *n* where *n* =100, 250, 500, and 792. 792 was chosen because it is the total number of true positives in our collection.

3. Percent Recall: This statistic indicates the percentage of bullying posts returned out of the total 792 posts that contain bullying. In this case, we looked at posts which had a score greater than 0. If we had 562 posts with a score greater than 0, and 345 were true positives, the recall is 345/792 = 43.5%.

Query name	Description
Q1	single post: top post with highest SUM value
Q2	single post: post with 2nd highest SUM value
Q3	single post: post with 3rd highest SUM value
Q4	single post: post with 4th highest SUM value
Q5	single post: post with 5th highest SUM value
Q6	multiple posts: posts from Q1 and Q2
Q7	multiple posts: posts from Q1, Q2, and Q3
Q8	multiple posts: posts from Q1, Q2, Q3, and Q4
Q9	multiple posts: posts from Q1, Q2, Q3, Q4, and Q5
Q10	multiple posts: 10 posts with highest SUM values
Q11	Dictionary: 100 level
Q12	Dictionary: 200 level
Q13	Dictionary: 300 level
Q14	Dictionary: 400 level
Q15	Dictionary: 500 level
Q16	Dictionary: 200 and 500 level
Q17	Dictionary: 100 and 200 level
Q18	Dictionary: 100 and 500 level
Q19	Dictionary: 100, 200, and 500 level
Q20	Dictioanry: whole dictionary
Q21	multiple posts: top 5 results from 200 level
Q22	multiple posts: top 10 results from 200 level
Q23	multiple posts: top 5 results from 500 level
Q24	multiple posts: top 10 results from 500 level
Q25	multiple posts: top 5 results from each level
Q26	multiple posts: top result from each level
Q27	multiple posts: all posts containing bullying
Q28	any suggested bullying words from Amazon Mechanical Turk

Table 7: Describes the queries for Tables 8-12

We began by using single posts as our query text. The SUM metric, which is available for all of our posts (defined in section IV.A), provides a good indicator for the severity of bullying in a post. The five posts with the top SUM value were:

1. “Q: NIGGERS NIGGERS NIGGERS BALLS BALLS TITTIES DICK SUCKING FAGGOT SHIT NIGGER ASSFUCK TITTIES FUCK NIGGERS
A: That's horrible.”
2. “Q: Aw see even nico does nott want too go near you. I like that you try and typeall ghetto. YOU dont fucking scare me you bitch. Ill fucking kill you if we faught. You'd probably bring your "ghetto friends" Bahahaha. Who would that be? Bella and kendall? Pff
 A: Actually i would love to "go near her" but im not the kind of asshole who puts that sort of unfair pressure on a girl :) and shut the fuck up you're not hardcore you cunt....you probably know nothing about fighting and im sure you'd get fucked up in a real fight so stop acting badass and go aay unless you're gonna stop being a bitch and do something about it.”
3. “Q: how old r u?? ur soooo cute and cool.....NAWTTTTTTT your a ugly little slut i bett ur gonna have a baby by the time u turn 13 you think its all cute to be on every guy its not everyone just thinks ur annoying!!!!!!! act ur age
A: really bitch...r grow the fuck up and why dont "you act your age!"r why the fuck do you actually take your time writing this shit if i obviously DONT GIVE A SHIT WHAT YOU THINKKK....bt dubbs i am 13 you fucking dumbass!r and "SLUT?"really?...r wow ... how more pathetic and imature can you get???r AGAIN r GROW UP!”
4. “Q: oh frances hunny dont try and talk all ghetto your white ass doesnt fucking scare me i will break you in fucking half.
A: Dude back the fuck off....i probably don't even know you so get the FUCK off of my fucken page and stay the FUCK away from Frances.”
5. “Q: ok really you put stupid skank smd or haha for your comebacks get some new ones..your just embrassing yourselfr -elizabeth u062a rodriguez
A: i love how your so fucking two faced man you say hi to me every day. what the fuck is your problem man. you know what heres a new comeback for you hop off my dick bitch and go comb the hairs on your mole. you fucking two faced bitch like all your lil friends. thanks for your time bitch <3”

The results for querying using each of these posts as the query text are shown in Table 8. The results indicate that individual queries will not be sufficient. For Q1 only 28 posts had a positive score. This makes sense because there are only a handful of unique terms in this query. On the other hand, Q4 returned a lot of posts, but most of them were not bullying terms, and the precision was only 5.2% (as opposed to 28.6% with Q1, because most of the terms in that post were indicative of bullying).

		num>0	num>=5	num>=10	at 100	at 250	at 500	at 792	% recall
	total/logical stop	28	0	0	28	28	28	28	
	yes	8	0	0	8	8	8	8	
Q1	%	28.57			28.57	28.57	28.57	28.57	1.01
	total/logical stop	1193	8	0	364	364	1193	1193	
	yes	111	2	0		54	111	111	
Q2	%	9.30	25.00		14.84	14.84	9.30	9.30	14.02
	total/logical stop	1592	7	0	248	1592	1592	1592	
	yes	81	0	0		81	81	81	
Q3	%	5.09	0.00		7.26	5.09	5.09	5.09	10.23
	total/logical stop	3717	68	1	161	352	1078	1078	
	yes	192	3	0	5	18	65	65	
Q4	%	5.17	4.41	0.00	3.11	5.11	6.03	6.03	24.24
	total/logical stop	392	0	0	329	329	329	329	
	yes	31	0	0	31	31	31	31	
Q5	%	7.91			9.42	9.42	9.42	9.42	3.91

Table 8: Statistics for Queries 1-5, Single Post Queries.

The results from the single post queries led us to the next logical step, to query using multiple posts. We used a combination of these posts to experiment with querying multiple posts. We first combined Q1 and Q2, then added each sequential query through Q5. These results are represented in Q6 through Q9 in Table 9. The tenth query is the 10 posts with the highest SUM values. We see from Table 9 that our recall is improving dramatically as we add more posts into our query text (from 14% to 49%), furthermore, we are improving recall without loss of precision (which stabilizes around 6%). Unfortunately the precision is very low. This is probably because we still have a lot of non-bullying terms in our query.

		num>0	num>=5	num>=10	at 100	at 250	at 500	at 792	% recall
	total/logical stop	1211	9	0	378	378	1121	1121	
	yes	113	3	0	58	58	113	113	
Q6	%	9.33	33.33		15.34	15.34	15.34	15.34	14.27
	total/logical stop	2455	39	1	109	717	717	2455	
	yes	170	5	0	16	71	71	170	
Q7	%	6.92	12.82	0.00	14.68	9.90	9.90	6.92	21.46
	total/logical stop	4644	258	26	129	258	909	909	
	yes	281	25	1	14	25	66	66	
Q8	%	6.05	9.69	3.85	10.85	9.69	7.26	7.26	35.48
	total/logical stop	4778	476	37	137	354	1061	1061	
	yes	288	38	5	16	29	71	71	
Q9	%	6.03	7.98	13.51	11.68	8.19	6.69	6.69	36.36
	total/logical stop	5647	760	134	134	350	546	1402	
	yes	393	75	21	21	41	56	121	
Q10	%	6.96	9.87	15.67	15.67	11.71	10.26	8.63	49.62

Table 9: Statistics for queries 6- 10, multiple post queries.

Our next step was to query based on the bad word dictionary. We thought that these words were good indicators of bullying when we trained the rule-based classifiers. We first queried each dictionary level separately (see Table 10). It is obvious that some levels, namely level 200 and level 500 have greater success in returning posts that contain bullying, returning 174 and 113 posts, respectively. The corresponding precision for 200 and 500-level words was 29.5% and 33.7%. Thus we are getting much better precision with the dictionary words as query text instead of the post text as queries text (Tables 8 and 9). Q16 in Table 11 shows the results of combining the 200-level and the 500-level words into a single query. This yielded a recall of 31.6%, and maintained the precision at 28.9%. The third highest recall from the dictionary queries was from 100-level words, so we added the 100-level words first to the 200 and 500-level words separately, then all combined all three (Q17, Q18, Q19, resp.). Our recall consistently improved, but we suffered a loss of precision when the 100-level words were added. Q20 provides the resulting statistics when we queried using all five levels of dictionary terms.

		num>0	num>=5	num>=10	at 100	at 250	at 500	at 792	% recall
	total/logical stop	1756	4	0	239	1756	1756	1756	
	yes	84	0	0	7	84	84	84	
Q11	%	4.78			2.93	4.78	4.78	4.78	10.61
	total/logical stop	589	3	0	589	589	589	589	
	yes	174	2	0	174	174	174	174	
Q12	%	29.54	66.67		29.54	29.54	29.54	29.54	21.97
	total/logical stop	242	0	0	242	242	242	242	
	yes	17	0	0	17	17	17	17	
Q13	%	7.02			7.02	7.02	7.02	7.02	2.15
	total/logical stop	82	0	0	82	82	82	82	
	yes	47	0	0	47	47	47	47	
Q14	%	57.32			57.32	57.32	57.32	57.32	5.93
	total/logical stop	335	0	0	335	335	335	335	
	yes	113	0	0	113	113	113	113	
Q15	%	33.73			33.73	33.73	33.73	33.73	14.27

Table 10: Statistics for queries 11-15, each level of the bad word dictionary

		num>0	num>=5	num>=10	at 100	at 250	at 500	at 792	% recall
	total/logical stop	864	5	0	165	864	864	864	
	yes	250	4	0	80	250	250	250	
Q16	%	28.94	80.00		48.48	28.94	28.94	28.94	31.57
	total/logical stop	2229	2	0	412	412	2229	2229	
	yes	227	2	0	72	72	227	227	
Q17	%	10.18	100.00		17.48	17.48	10.18	10.18	28.66
	total/logical stop	2033	0	0	327	327	2033	2033	
	yes	180	0	0	38	38	180	180	
Q18	%	8.85			11.62	11.62	8.85	8.85	22.73
	total/logical stop	2456	12	0	109	522	522	2456	
	yes	291	6	0	33	115	115	291	
Q19	%	11.85	50.00		30.28	22.03	22.03	11.85	36.74
	total/logical stop	2664	12	0	464	628	628	2664	
	yes	321	7	0	142	132	132	321	
Q20	%	12.05	58.33		30.60	21.02	21.02	12.05	40.53

Table 11: Statistics for queries 16-20, multiple levels of the bad word dictionary

Because the results from the 200-level and 500-level dictionary word queries appear to be promising, we decided to query using the post text from posts that had the highest scores for the NUM200 and NUM500 attributes. The idea behind this is that although the dictionary may represent the bulk of the curse on insult words that we think indicate bullying, it cannot represent the volume in which they are used. Using the actual post will also provide content, in the form of other words that are commonly used with these bullying terms. We decided to query using the text from the posts with the top 5 and 10 scores for NUM200 and NUM500. The results appear in Table 12 (Q21-Q24). These results were promising, especially those from the top

10 results from each, yielding recall levels of 52.1% and 50.0%. Additionally, we queried the top 1 and 5 results from each level: Q26 and Q25. As indicated in Table 12, we see that these found 39.5% and 69.3% of the posts. The precision remains near 6%, which is disappointing.

		num>0	num>=5	num>=10	at 100	at 250	at 500	at 792	% recall
	total/logical stop	5156	2468	586	102	275	486	972	
	yes	368	132	32	4	17	32	61	
Q21	%	7.14	5.35	5.46	3.92	6.18	6.58	6.28	46.46
	total/logical stop	6356	2591	898	113	297	570	898	
	yes	413	155	62	7	17	32	62	
Q22	%	6.50	5.98	6.90	6.19	5.72	5.61	6.90	52.15
	total/logical stop	4974	895	133	133	289	705	894	
	yes	305	65	18	18	30	51	65	
Q23	%	6.13	7.26	13.53	13.53	10.38	7.23	7.27	38.51
	total/logical stop	6799	2029	471	115	302	643	937	
	yes	396	138	39	17	29	47	70	
Q24	%	5.82	6.80	8.28	14.78	9.60	7.31	7.47	50.00
	total/logical stop	8704	4069	2782	105	258	556	856	
	yes	549	262	183	9	23	41	67	
Q25	%	6.31	6.44	6.58	8.57	8.91	7.37	7.83	69.32
	total/logical stop	4564	490	72	141	331	938	938	
	yes	313	42	7	13	29	77	77	
Q26	%	6.86	8.57	9.72	9.22	8.76	8.21	8.21	39.52
	total/logical stop	13146	11636	9986	102	251	503	794	
	yes	792	724	656	11	26	63	81	
Q27	%	6.02	6.22	6.57	10.78	10.36	12.52	10.20	100.00
	total/logical stop	12735	10402	7991	101	253	512	797	
	yes	765	694	571	10	25	49	72	
Q28	%	6.01	6.67	7.15	9.90	9.88	9.57	9.03	96.59

Table 12: Statistics for queries 21-28, top results from specific levels, all posts containing bullying, and Amazon Mechanical Turk suggested words

Our final queries were all posts that contain bullying (Q27) and any words suggested by the Amazon Mechanical Turk workers (Q28). Obviously, the query of all posts that contain bullying would result in a recall of 100.0%, but we were surprised by the number of posts not containing bullying that were returned at the highest ranks (see the precision at Rank *n* columns, particularly the "at 792" which shows that for the top 794 scoring posts, only 72 contained bullying). It is likely that many posts had a lot of words in common with the bullying posts, and the noise dominated the search results. Unfortunately, the query that used the words that the

Mechanical Turk workers suggested (Q28) was also disappointing. Some of the workers returned whole posts as their suggested words or phrases, and this seems to have again added in a lot of noise that dominated the search results.

E. Conclusions

From these experiments, we can see that longer, more complex queries provide better precision and recall overall for this task. In this future we will test with no less than 5 or 10 posts when using a combination of posts as our query text.

Additionally, we can conclude that the dictionary that we created for rule-based learning is a good indicator of what words are in bullying posts. Although it is not complete, it provided the best overall results in our bag-of-words experiments.

VI. COMPARING THE MODELS

When comparing the rule-based model to the bag-of-words model we have to understand the nuances of the metrics used in each set of experiments to determine which approach gave us better results. In the rule-based model, our best result was 78.5% correctly labeled positive instances (recall). This means that the model was able to correctly identify 78.5% of the results labeled "containing bullying" (using 10-fold cross validation). In these results, we were most concerned with recall because we would rather err on the side of caution. That is, we were willing the risk labeling posts that do not contain bullying as containing bullying in order to maximize this metric. Additionally, the results from the anonymous experiment indicate a 68.2% TP percentage.

In the bag-of-words approach, we used different metrics, in addition to recall. If we ignore the query containing all of the positive instances, the query with the highest recall returned 96.6% of the true positives. However, this query also returned many innocent posts (the precision levels were very low). The best trade-off between recall and precision appeared to be with Q16 (the 200 and 500-level words from the dictionary). Q20 was also interesting. It produced a recall of 40.50% and the rank-464 statistic is 30.6%.

VII. FINAL CONCLUSIONS

By comparing the results of both the rule-based and bag-of-words methods, we believe that the rule-based method performed better. The recall for the rule-based method is much higher, in most cases higher than the recall achieved using the bag-of-words method. To get a high recall with bag-of-words we had to severely affect the precision. With this, it is clear to see that the rule-based model outperforms the queries we conducted. However, we believe that the query set that we presented in this paper is not complete and that it is possible that with more testing, we would be able to find queries that are more effective.

VIII. ACKNOWLEDGEMENTS

This material is based upon work supported by the National Science Foundation under Grant No. 0916152. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

VIX. REFERENCES

[1] "Study: A Quarter of Parents Say Their Child Involved in Cyberbullying."(2011, July). *PC Magazine Online*. *Academic OneFile*. Web.

[2] Witten, I.; Frank, E. (2005). "*Data Mining: Practical machine learning tools and techniques, 2nd Edition*". Morgan Kaufmann, San Francisco. http://www.cs.waikato.ac.nz/~ml/weka/book.html.

[3] Dinakar, K; Reichart, R.; Lieberman, H. (2011). *Modeling the Detection of Textual Cyberbullying*. Thesis. Massachusetts Institute of Technology.

[4] Quinlan, R. (1993). *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers, San Mateo, CA.

[5] Maher, D. (2008). Cyberbullying: an ethnographic case study of one australian upper primary school class. *Youth Studies Australia*, *27*(4), 50-57.

[6] Willard, N. E. (2007). *Cyberbullying and Cyberthreats: Responding to the Challenge of Online Social Aggression, Threats, and Distress*. Champaign, IL: Research. Print.

[7] *Glossary of cyberbullying terms*. (2008, January). Retrieved from http://www.adl.org/education/curriculum_connections/cyberbullying/glossary.pdf

[8] Patchin, J., & Hinduja, S. (2006). Bullies move beyond the schoolyard; a preliminary look at cyberbullying. *Youth violence and juvenile justice*. 4:2. 148-169.

[9] Aha, D. and D. Kibler (1991). Instance-based learning algorithms. *Machine Learning*. 6:37-66.

[10] Cohen, W. (1995). Fast Effective Rule Induction. In: *Twelfth International Conference on Machine Learning*, 115-123, 1995.

[11] Platt, J. (1998). Fast Training of Support Vector Machines using Sequential Minimal Optimization. In B. Schoelkopf and C. Burges and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*.

[12] Baeza-Yates, R., and Ribeiro-Neto, B. (1999). *Modern Information Retrieval*. New York: ACM. Print.