Machine learning to classify bullying messages on Twitter

Bouke Bommerson

University of Twente

October 23rd, 2015

Abstract

In this study we look at cyberbullying on Twitter. We try to automatically classify bullying messages and compare various classification algorithms. We fine-tune the performance of some algorithms and find that oversampling the bullying messages in the minority improves the kappa and accuracy measures of some classifiers.

We explored different external features, and while not giving conclusive results, we suggest directions for future work in that area.

We find that training a classifier incrementally in a feedback loop with manual classification improves the accuracy of the classfier.

1 Introduction

1.1 Bullying on the internet

Bullying can have bad consequences,

> increased levels of depression, anxiety and psychosomatic symptoms in victims (Kaltiala-Heino, Rimpela, Rantanen, & Rimpela, 2000; Kumpulainen et al., 1998; Neary & Joseph, 1994; Roland, 2002). The bullied students also feel more socially ineffective and have greater interpersonal difficulties (Craig, 1998; Forero, McLellan, Rissel, & Baum, 1999), together with higher absenteeism from school and lower academic competence (Rigby, 1997; Zubrick et al., 1997). However, it is still unclear if these symptoms are antecedents or consequences of bullying (Hodges & Perry, 1999; Roland, 2002). Thus the direction of causality may be both ways (Kaltiala-Heino et al., 2000).

Campbell (2005)

Bullying can also happen online. This can take the form of making hurtful comments about someone, humiliating them, or threats and verbal abuse. The effects of cyberbullying can be the same or worse than face-to-face bullying, including increased levels of anxiety and depression.Campbell (2005)

Cyberbullying can happen on social media like Facebook, Twitter, and Instagram. Monitoring this behaviour can be valuable, to increase insight into the behaviour or possibly to help prevent it. Recognizing messages as bullying messages with reasonable accuracy and speed generates opportunities like notifying the sender before the message is sent, or recognizing a spike of bullying in a certain area.

1.2 Problems with manual classification

Manual and automatic approaches for identifying bullying messages are possible. However, manually monitoring this is time-inefficient due to the large amount of data involved (roughly 6000 messages per second on Twitter, according to a tool (Internetlivestats.com) collecting statistics about Twitter, used the 5th of July, 2015), and due to bullying messages only making up a small portion of all messages on Twitter. That means that for manual classification of messages a lot of work needs to be done to get any meaningful results.

1.3 Automatic classification

This task can also be done automatically with machine learning algorithms. In machine learning a classifier is trained on certain features of the data. Based on these features, it makes a model of what a certain class of data 'looks like'. In this case, it could make a model of bullying messages.

This model could then be used to predict what class a message belongs to, and so help to classify bullying messages.

2 Research questions

- What are effective features for identifying bullying messages?

We will explore particular features in tweets as well as tweet metadata to find features that automatic analysis might not.

- Are these features useful in selecting data from Twitter to train on?

Taken as an absolute, not a high proportion of messages on Twitter are bullying. If we can filter data from Twitter before analyzing it, we might retrieve more examples of bullying. This would aid in classifying.

- What are good classifier algorithms and parameters for this task?

There are many different classification methods. We can compare them to find the most suited one.

- Can automatic classification help with classifying bullying messages?

By use of certain classification methods, parameters, and features, we hope to show that automatic classification can be useful in the task of automatically classifying bullying messages.

3 Related work

There have been other studies that are interested in cyberbullying, tweet classification, or both. We have categorized them as such.

3.1 Bullying on Twitter

Sanchez and Kumar (2011)(Sanchez and Humar, 2011) implement lingpipe and a Naive Bayes classifier using a bag-of-words model and get about 67% accuracy. We implemented this too and got 69.8%, comparable to theirs. They crowdsource to classify unlabeled data. Interestingly, a Naive Bayes classifier was marginally more accurate than the crowdsourced classification. This suggests either a limited usability of crowdsourcing this classification, or that improvements in the crowdsourcing set-up can be made.

Nalini and Sheela (2015)(Nalini and Jaba Sheela, 2015) Implement a support vector machine for classying bullying messages on twitter, we get similar results.

3.2 Bullying on other social media

Nahar et al. (2012) (Nahar et al, 2012) try to detect cyberbullying messages, bullies, and victims on MySpace, SlashDot and Kongregate. They extract common features from all messages and specific sentiment features from bullying posts only. To do the latter, they apply Probabilistic Latent Semantic Analysis. Using this method they gain high (98%+) accuracy. Using SVM in combination with LSA gives us an accuracy of 72%, worse than using other attribute selectors, giving 74% in case of gainratio. It might be worth pursuing this further. However, part of their high accuracy stems from a low ratio of bullying messages to all messages. Therefore, even a classier that classifies everything as non-bullying with only false negatives would get an accuracy score of 90+%. See 5.3 on how we dealt with that problem.

3.3 Other classification on Twitter

Wang (2010)(Wang, 2010) proposes a Bayesian classifier to classify spam messages on Twitter. In the study they use the content of the tweet, but also the friends and followers of the user.

Hosseinmardi et al. (2015) (Hosseinmardi et al, 2015) study cyberbullying on Instagram. They find that an SVM classifier gets better results than a Naive Bayes classifier, with an accuracy of 87%. Interestingly, they found that even high amounts of profanity do not necessarily constitute cyberbullying. We used profanity initially to generate a dataset for manual classification, but do see a good proportion of bullying messages. They crowdsource part of their bullying classification, and the crowd seems to agree about bullying messages as opposed to the previously mentioned study.

Nguyen et al. (2014)(Nguyen et al, 2014) use a logistic regression model to predict the gender of the poster of a tweet, which performs similarly to a crowdsourced prediction. This is an interesting approach to crowdsourcing, and comparing the automatic classification of a bullying message to the crowdsourced prediction could be a topic of further study.

4 Approach

4.1 What is machine learning?

Machine learning can be used to classify data according to certain features of that data. For example, it is used on emails to classify spam messages. To filter messages it makes a model of what features the various classes of messages share. To make the model it is provided with a dataset of messages with the messages already annotated according to what class of message they belong to. There are two main ways of training a classifier: one time or iterative. The former method is trained on sample data once, and the latter periodically takes new data with which it improves its model. An advantage of iterative or online learning is that the model improves over time. This is useful, because language changes over time. By periodically re-training, the model will be aware of these changes and will be more accurate. Additionally, the automatic classifier can sort a batch of tweets based on how confident it is that they are bullying messages. This way, manual classification can speed up by presenting actual bullying messages first.

4.2 How is it applied and useful here?

We use a machine learning approach to attempt to classify bullying messages automatically. Then, we run experiments to compare and optimize classifiers as well as to find additional features.

5 Experiments

5.1 Experimental setup

We used the Weka machine learning environment for preprocessing and classification tasks. To do the task of manual classification we wrote programs in Python and Java to integrate the powerful Weka tools in the workflow of the annotator. For testing and comparing different classifiers we used both the KnowledgeFlow mode of the Weka GUI and a self-written test suite.

5.2 Data description

To retrieve data relevant for the problem, we retrieved data from Twitter with two conditions:

- It contains a swearword according to the list at noswearing.com. Bullying can involve verbal violence, so swearwords were chosen to indicate that. However, see related work for a work concluding that swearing is not necessarily an indicator of bullying.
- Secondly, it contains an @-mention. Bullying tends to be directed towards a person. Including an @-mention directs a tweet at someone. In the section 'Feature Exploration' the effect of included @-mentions is examined.

5.2.1 Pre-processing the data

This dataset contains the full tweet metadata. For classification we trimmed each tweet down to contain only the ID number of the tweet and the text of the tweet. We later used the ID of the tweet to retrieve info from the full dataset whenever that was relevant.

We then used the trimmed dataset to manually classify tweets to create a training set for the automatic classification. After we annotated 250 messages manually, we trained an automatic classifier on this data and let it classify the remaining messages. We sorted the output based on the confidence of the classifier of each message being a bullying message. We did this so further rounds of manual classification would find more bullying messages. See also 5.6. Nevertheless, the dataset of 1109 thus manually classified messages still contains only 282 bullying messages (25.4%).

5.3 Oversampling bullying tweets

The proportion of bullying to non-bullying messages is fairly skewed, with barely over 25% bullying messages. This can create problems because a classifier that, for example, classifies everything as a non-bullying message will still be 75% accurate. One method of dealing with this problem is oversampling the smaller class. This means that in its most basic form oversampling would select a datapoint in the smaller class more than once. Similarly, undersampling the majority class can also be done, where some datapoints of the larger class will not be considered. A more advanced method is called SMOTE Chawla et al. (2002), "Synthetic Minority Oversampling TEchnique". Based on datapoints surrounding a datapoint in the minority class, it generates a new datapoint. In our experiments, we also test the effect of oversampling.

5.4 Classifier comparison

To create a shortlist of suitable classifiers for further refining, we ran the classifiers on the same data to compare them.

We chose to test 6 different classifiers for a spread of different approaches to classifying.

We tested them both with and without SMOTE preprocessing.

The following images show a comparison of the kappa and accuracy scores of the six classifiers. Kappa can be seen as a single value measure of the amount of true and false positives and negatives. A kappa closer to 1 is better, a kappa below 0.4 can be seen as poor. These guidelines are relatively arbitrary.

NaiveBayes is an implementation of a probabilistic classifier.

IBk is an implementation of a lazy learning method.

JRip is an implementation of a rule based method.

J48 is an implementation of a decision tree based method

SMO and LibSVM are implementation of a support vector machine method, with LibSVM implementing SMO.

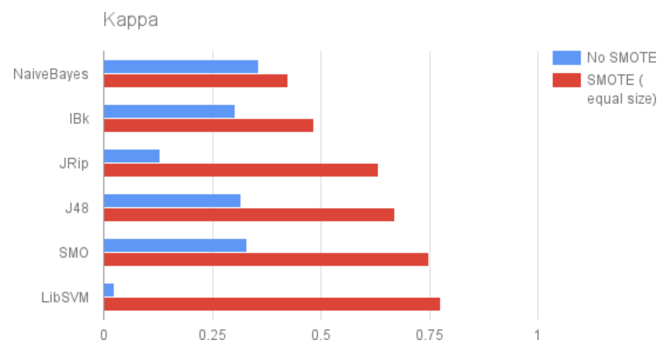

Figure 1: Kappa measure of various classifiers.

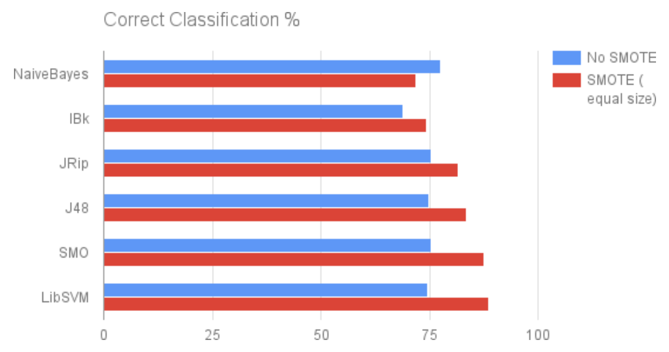

Figure 2: Accuracy of various classifiers.

Comparing the kappa measure and the correct classification percentage of the LibSVM classifier illustrates the problem of unbalanced data. The percentage of correctly classified messages is 74.6% before preprocessing. However, this is only a reflection of the underlying data. Only 17 messages were classified as bullying, the rest as not bullying. Clearly, the percentage of correctly classified messages is not always a good measure of the usefulness of a classifier. The difference preprocessing makes is illustrated better by the kappa measure. Without SMOTE the kappa measure for LibSVM is very low, only 0.0255. This shows that the classifier does not do better than one classifying messages randomly, and that it thus is not very good. After preprocessing all classifiers do better. Also of note is that IBk has a very low proportion of false negatives at the cost of higher proportion of false positives. That is valuable if you want to make sure you miss as few bullying messages as possible. Another method is to use a cost based classifier and set higher costs for falsely classifying a positive. We chose to focus on just one of these methods.

5.5 Parameter sweeps

We performed paramater optimalization of the three highest ranked classifiers by kappa. Each classifier was evaluated by 10-fold cross-validation for each parameter value.

For LibSVM we did a grid-search optimalization of the cost and gamma attributes, and for J48 and SMO a simple parameter sweep.

In all cases we did a wide initial sweep before a narrower sweep focusing on the more promising parameter values.

5.5.1 LibSVM

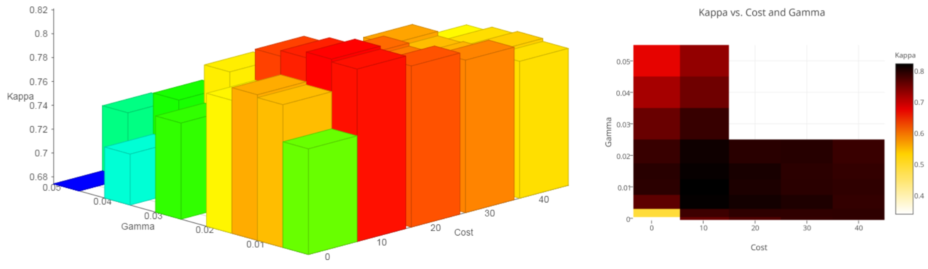

Figure 3: A 3d and 2d graph of kappa at various values of cost and gamma for the LibSVM classifier.

We did an initial wide grid search over cost and gamma, then focused on a gamma from 0 to 0.05 and a cost of 0 to 40. Cost penalizes misclassifying training examples. Gamma is a parameter related to the spread of the datapoints in space and thus is recommended to have a value of 1 / (Amount of features), in this case 0.0002757 after SMOTE preprocessing. However, we get better results with higher values. Cost is recommended to be modified by factors of ten, but we focused on the range in the graph above as it seemed sufficiently promising. It is possible that we have found a local maximum or even that the model now overfits the data. The highest kappa value of 0.82 in this search was found at a Cost of 10 and a Gamma of 0.01. This highest point seems to be in a stable region. Because of the good result and high speed of the LibSVM method, we recommend this method.

5.5.2 J48

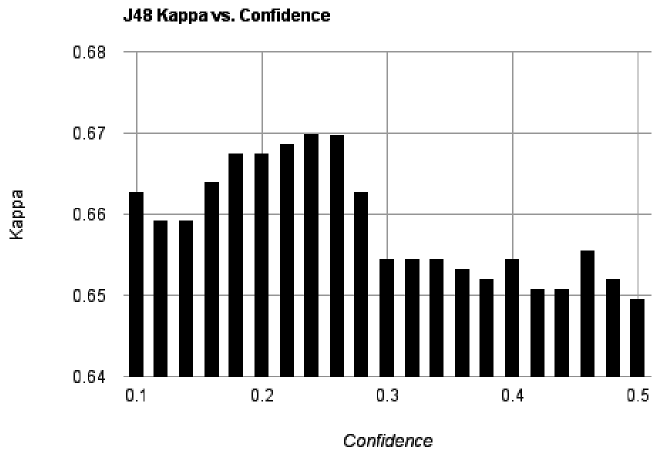

Figure 4: Kappa at various confidence values for the J48 classifier.

We did a sweep over the confidence parameter from 0.1 to 0.5. A smaller confidence factor increases the pruning of the tree. Too much pruning will decrease the accuracy of the classifier, but too much pruning risks overfitting the training data. The highest kappa value we found to be at a confidence factor of 0.24. The results seem to be stable around this point, in the region of 0.2 and 0.3. Interestingly, the small spike at low confidence values might be an indicator of overfitting starting to happen.

5.5.3 SMO

We did a sweep over the complexity parameter and then finetuned at the more promising values. The complexity parameter tunes how many instances will be used to draw the boundaries between classes. It works as a trade-off between a lack of accuracy and overfitting. Complexity is recommended to be modified by factors of ten which we did for the first sweep, from 0.01 to 100. Then we focused on the complexity from 0.2 to 1.4 as that seemed to maximize kappa. The highest kappa value we found to be at a complexity factor of 0.6, which seems to lie in a relatively stable region.

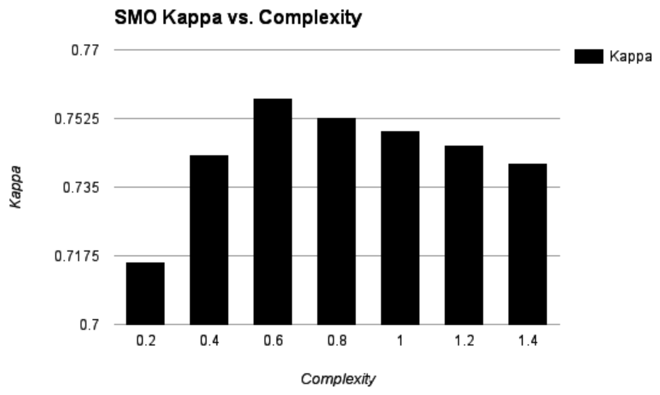

Figure 5: Kappa at various complexity values for the SMO classifier.

5.6 Annotated set size evaluation

To aid a manual classifier with their task we built a tool to integrate automatic classification in their workflow. The user is asked to manually classify a number of messages from a large input dataset. Then a classifier is built, using these messages as training data. The tool uses the classifier to sort the messages in the larger dataset based on the predicted probability of them being bullying messages. The user is presented the messages in order. We expect that in this way the user will encounter more bullying messages over time, and so incrementally improve the classifier. This will create a positive feedback loop. In this way the relatively rare bullying messages will form a larger proportion of the training dataset.

To test this we manually annotated 1109 messages and split this into 559 testing messages and 550 initial training messages. We trained a LibSVM classifier on a learning dataset containing 50 training messages pulled from the training dataset, using 10-fold cross-validation.

Cross-validation results in better classifier training, but might take infeasibly long with larger datasets. At that point, taking a percentage of datapoints randomly and training on those would become more feasible.

We used the parameters Cost = 10 and Gamma = 0.01, as found in 5.5.1. We tested the resulting model on the test dataset and recorded the accuracy and kappa measures. Then we sorted the remaining testing messages based on the probability output of the classifier.

With 11 steps in total we moved the 50 messages with the highest bully probability from the initial dataset to the learning dataset and repeated the procedure until all 550 initial messages were in the learning set.

We repeated this experiment three times and took the average result.

This graph show that the accuracy and kappa measures of the classifier improve as more data is annotated.

Of note is the peak in both measures in the average result. One run shows

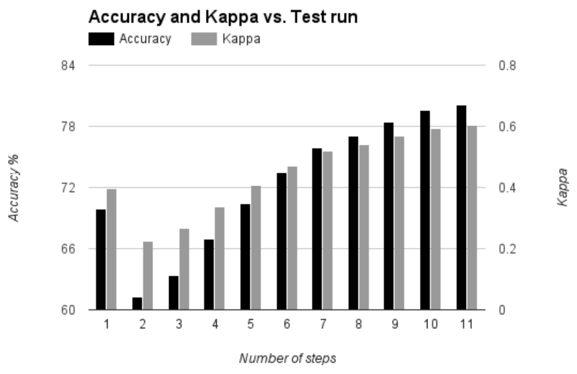

Figure 6: Measuring accuracy and kappa of subsequent runs. Each run includes 50 more messages. Note the peak at the initial run.

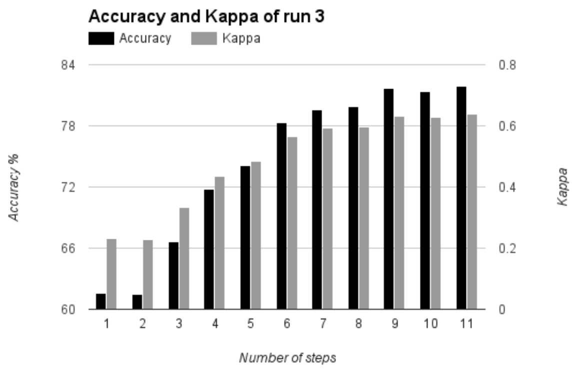

Figure 7: One run that no peak at the initial run.

that the accuracy and kappa still increase when there is no initial peak.

We surmised that the cause of that might be due to SMOTE having been applied to the testset too, and that the additional synthetic samples may automatically be classified as the bully messages that they are.

We tested this by re-running the experiment, comparing the classifier with a testset to which SMOTE had not been applied. However, the spike remained.

As yet, the cause of this spike remains unknown.

6 Feature exploration

Aside from bag-of-words textual analysis we considered some other possible features of bullying messages. Bag-of-words analysis is limited to just the text of the tweet. It is possible that there are other features that won't show up in text of a tweet. For example, the time a tweet is posted might help classify messages.

6.1 @-mention

Because bullying is can be surmised to be directed at a person, the initial dataset is of tweets that contain an '@-mention'. An @-mention serves as a tool to notify a specific user.

In the bag of words model these mentions are treated like any other word. To test if the presence of an @-mention improves the accuracy of the classifier, the classification was done on the same dataset but with all @-mentions removed. This is not entirely representative, and see the Recommendation section for thoughts on how to handle this. Comparisons of classifying with versus without @-mentions included in the bag of words show no appreciable difference.

6.2 Time of tweet

The time of day a tweet is posted was also examined as a possible indicator of a bullying message. For example, bullying messages might be posted after school more often than non-bullying messages. For the 1200 manually classified messages the time of posting and the timezone of the user were extracted and added to result in the local time of posting. See figure 8. According to statistical methods, the difference is significant. A problem is that the messages were collected from a 48 hour consecutive period. Any spikes might be from an event starting at that specific time. Thus, we did not implement this as a feature. To make this worthwhile as feature to implement we suggest sampling messages from a longer period of time.

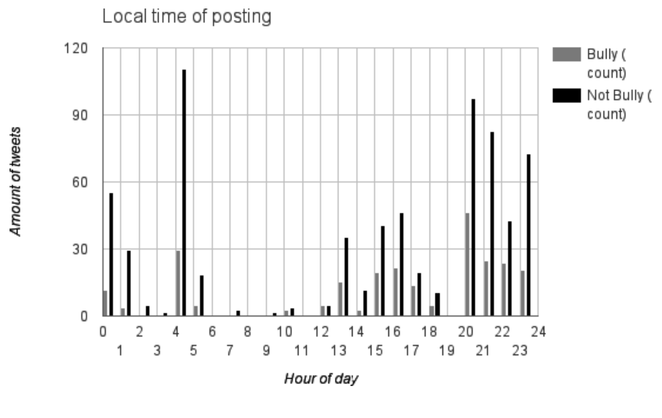

Figure 8: A histogram showing at what local time a tweet is posted.

Additionally, some 15 percent of messages were deleted or otherwise inaccessible at the moment the time data was collected. We thought that they might turn out to have a high proportion of bullying messages that were either deleted by Twitter for breaking any rules or by the user for regretting their comment. However, this turned out not be the case. The proportion of bullying to non-bullying messages is lower than that of the original dataset. If the proportion were higher it could be added as a an additional feature: After some weeks the classifier could 'double check' earlier messages classified as bullying, and if they were deleted, it could increase the confidence of that particular message being a bullying message. There is slight statistical evidence that the deletion of a message can be an indicator of a non-bullying message, but we didn't implement this because of the low expected gain for the cost of implementation.

6.3 Negation of swearwords

We thought that a significant portion of swearing might be canceled by nearby negations. Example: "I hate you" compared to "I don't hate you". Both contain the word 'hate' that would be an indicator of a bullying message. However, the meaning of the latter phrase is not offensive despite containing an offensive word. Of the 1200 classified messages all instances containing "not" and its contraction "n't" were extracted. Of the 163 messages with a negation there was only one instance of a negation being used in the specified way. In fact, a negation is more often used in a phrase of the construction "Don't be an -expletive-" or "Don't give an -expletive-". Thus, a negation near an offensive word wasn't used as a feature for a non-bullying message.

7 Conclusion and discussion, recommendations

We have found that a dataset with a relatively large proportion of bullying messages will be retrieved with a filter based on @-mentions and swearwords. However, we have not examined the @-mention feature as a potential bullying feature in unfiltered datasets. Future work might consider this.

Time of day could be a useful feature in indicating bullying messages, but our limited dataset that consists of messages out of 48 consecutive hours is not sufficiently spread in time to provide conclusive evidence. There did appear to be differences, so a recommendation for future work is to sample messages randomly from a larger period of time.

Negation of swearwords did not appear to have an impact on a tweet with regards to the difference between bullying and not bullying. Future work may also consider such phrases as "just kidding".

We have found that the LibSVM method with a cost factor of 10 and a gamma factor of 0.01 was a classifier giving good results. Furthermore, using SMOTE to pre-process the data and oversample the bullying messages in the minority increases the accuracy and kappa measures of the tested classifiers.

We have found that incrementally training a classifier on manually annotated messages ordered by the output of an automatic classifier increases the accuracy of that classifier. This aids the manual annotator by providing them with a dataset with messages that are more likely to be bullying messages. So, the training set encounters more of the relatively rare bullying messages and the automatic classifier is better able to classify new ones. Thus, automatic classification can help with classifying bullying messages.

Future work may also look at the possibilities that word2vec Mikolov et al. (2013) gives with regard to finding words similar to words featuring in bullying messages. It could assist in finding unexpected features to consider.

References

Marilyn A Campbell. Cyber Bullying: An Old Problem in a New Guise? *Australian Journal of Guidance and Counselling*, 15(1):68–76, jul 2005. doi: 10.1375/ajgc.15.1.68. URL `http://dx.doi.org/10.1375/ajgc.15.1.68`.

Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. SMOTE: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, pages 321–357, 2002.

H. Hosseinmardi, S. Arredondo Mattson, R. Ibn Rafiq, R. Han, Q. Lv, and S. Mishra. Detection of Cyberbullying Incidents on the Instagram Social Network. *ArXiv e-prints*, March 2015.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

Vinita Nahar, Sayan Unankard, Xue Li, and Chaoyi Pang. Sentiment Analysis for Effective Detection of Cyber Bullying. In *Web Technologies and Applications*, pages 767–774. Springer Science + Business Media, 2012. doi: 10.1007/978-3-642-29253-8_75. URL `http://dx.doi.org/10.1007/978-3-642-29253-8_75`.

K. Nalini and L. Jaba Sheela. Classification of Tweets Using Text Classifier to Detect Cyber Bullying. In *Emerging ICT for Bridging the Future - Proceedings of the 49th Annual Convention of the Computer Society of India CSI Volume 2*, pages 637–645. Springer Science + Business Media, 2015. doi: 10.1007/978-3-319-13731-5_69. URL `http://dx.doi.org/10.1007/978-3-319-13731-5_69`.

Dong-Phuong Nguyen, RB Trieschnigg, AS Doğruöz, Rilana Gravel, Mariët Theune, Theo Meder, and FMG de Jong. Why gender and age prediction from tweets is hard: Lessons from a crowdsourcing experiment. 2014.

Huascar Sanchez and Shreyas Kumar. Twitter bullying detection. *ser. NSDI*, 12:15–15, 2011.

Alex Hai Wang. Detecting Spam Bots in Online Social Networking Sites: A Machine Learning Approach. In *Lecture Notes in Computer Science*, pages 335–342. Springer Science + Business Media, 2010. doi: 10.1007/978-3-642-13739-6_25. URL `http://dx.doi.org/10.1007/978-3-642-13739-6_25`.