Real Time Learning Detection of Flaming in Twitter

Jessa Mae Cabigas
Department of Computer Science
University of the Philippines Cebu
Cebu City, Philippines, 6000
jessacabigas@gmail.com

Aileen Joan O. Vicente
Department of Computer Science
University of the Philippines Cebu
Cebu City, Philippines, 6000
aovicente@up.edu.ph

Abstract

Systems have been proposed to detect cyberbullying in social media sites such as the BullyTracer and ChatCoder. These systems however lack real-time and learning components that are very significant to the changing social media landscape. This paper presents a prototype for a real-time and learning detection of Flaming in Twitter. The prototype learns the model using Support Vector Machine with Stochastic Gradient Descent. An initial model for the prototype was generated with an F-score of 73%. The prototype engine allows for the improvement of the model with increments of training samples over time. A simulation was done to test the model's performance over time using different datasets and it produced a better performance with the addition of data.

Keywords: *cyberbullying, flaming, support vector machine, stochastic gradient descent, twitter, incremental learning*

1 Introduction

Twitter is a social networking (SNS) and microblogging service, enabling registered users to read and post short message known as tweets. Twitter, being one of the most popularly used social networking site (Ebizmba, 2017) (Moreau E, 2017), has allowed people from around the world to communicate and stay connected with each other. While it continues to be a medium for connecting people on the one hand, on the other, Twitter has also become a venue for people to virtually inflict harm onto others. This phenomenon is popularly known as cyberbullying.

Cyberbullying is an act of "willful and repeated harm inflicted through the medium of electronic text" (Hinduja S. & Patchin J., 2006). To consider an exchange of messages/replies containing a manifestation of cyberbullying, the intent to inflict harm in social, physical, or emotional aspects must be present. There are many manifestations of cyberbullying in social media with each has their own intention of harm inflicted upon their victim. The most common form of cyberbullying is Flaming according to Middle School Cyberbullying Curriculum (2008). Flaming, which can also be referred as bashing, is a kind of online fight. The bully sends or posts electronic message/s which are enticingly insulting, vulgar to one or several persons either privately or publicly to an online group (N.E. Willard, 2007).

To help alleviate the matter of cyberbullying, popular systems were implemented and made available: ChatCoder (Kontostathis et al., 2009) and BullyTracer (Bayzick, J., 2011). These two computer softwares perform well at classification performance; however, they lack the ability to be implemented in a real-time setting, and ability to update its knowledge base. Thus, this paper intends to create a prototype that enables dynamic learning and updating its model while detecting a specific type of cyberbullying in Twitter. Specifically, this study intends to:

- Obtain a cyberbullying dataset from Twitter to be used for training and testing
- Represent and extract features from tweet

representation.

- Build an Support Vector Machine with Stochastic Gradient Descent
- Implement learning classification in a Twitter Flaming Detection prototype
- Evaluate classification performance of the TFD prototype and present results

2 Related Studies

2.1 Cyberbullying Systems

A research by Bayzick, J. (2011) implemented a computer software, named BullyTracer,that detects many cyberbullying manifestations. The dataset consisted of thread style conversations from MySpace.com. The software analyzes all files in given directory using a rule-based algorithm. A single post was considered to contain cyberbullying if it contains an insult word and a second person pronoun such as “you” and “your”. The overall accuracy of the BullyTracer is 58.63% suggests that the coding rules for falsely flagging an innocent post free of cyberbullying needs to be significantly refined.

The research that motivated BullyTracer into being studied was ChatCoder. ChatCoder by A. Kontostathis et al. (2009) was a program built to classify predatory language from chat conversations to categories: personal information (the predator asks questions or volunteers information), grooming (any inappropriate or sexual language), or approach (talking about meeting the victim in person or talking on the phone). Each single chat line is classified into one of those classes. In this study, communications theories, text-mining and k-means clustering was utilized.

SafeChat, a third party plugin for Pidgin which is an open source instant messaging client, uses detection algorithms to classify chat participants as potential sexual predators (Thom et al., 2011). This research also makes use of the communication theory presented by A. Kontostathis (2009) and an age function. The rule-based approach of this plugin resulted to a 68% accuracy in categorizing posts.

2.2 Instance-based versus Batch-based

Learning dynamic and evolving data is important especially when dealing with data where training examples are incoming in any point (Read et al., 2012). Thus, an incremental approach to cyberbullying is a sound method because language and its pattern also evolve in time. Figure 2.1 below shows the cycle in implementing an incremental approach. The model will be created in learning the training examples. Afterwards, the model will be tested and used in predictions. But, it is expected that the model should also be always ready for new data.

Instance-based incremental learning updates the model with new training examples as soon as they are available. Whenever new data is introduced to the system, training the system from scratch will be unnecessary. The algorithm will learn new knowledge from the new data without forgetting the previously acquired information. Instance-based is naturally suited for incremental learning and it is fast compared to the batch-based approach (Read et al., 2012).

Batch-based learning approach observes the batch training technique wherein all the training examples are needed at once because these examples will be operated in bulk. There are many classifiers to choose from when using this approach compared to the instance-based. However, this approach has to approach models over time as memory fill up. This approach is also slow to run in terms of running time (Read et al., 2012). To learn new data in batch-based algorithms, the old classifier will be discarded and training of a new classifier will be performed again using all the data appended with the new one.

Read et al. (2012) performed experimentations on using batch-based and instance-based incremental learning in dynamic and evolving data. Results showed that the Instance-based Incremental Method such as the Naive Bayes, Stochastic Gradient Descent (SGD) are faster than the Batch-based with the SGD performing the fastest. With this, SGD showed promise in implementing a real time online learning system for detecting Flaming.

2.3 Stochastic Gradient Descent

Stochastic gradient descent (SGD), also known as the incremental gradient descent, is a popular algorithm for training a wide range of models in machine learning, including (linear) support vector machines, logistic regression and graphical models (Finkel et al., 2008). It is a simple yet very efficient technique to discriminative learning of linear classifiers under convex loss functions such as the SVM and Logistic Regression. SGD has been successfully applied to large-scale and sparse machine learning problems often encountered in text classification and natural language processing.

In practice, the batch learning setting performance on the computation of the cost and gradient for the entire training set can be very slow and sometimes intractable on a single machine if the dataset is too big to fit in main memory. Moreover, batch optimization methods don't give an easy way to incorporate new data in an 'online' setting. This problem is solved by the Stochastic Gradient Descent.

SGD deals with large scale data by breaking it up into chunks and is dealt with sequentially - which can be referred as minibatch learning. The fact that we only need to load one chunk into memory at a time makes it useful for large-scale data, and the fact that it can work iteratively allows it to be used for online learning as well. This technique tries to find the minima or the maxima at each iteration. As the algorithm performs through the entire training data example, it updates the convergence at each random training example.

3 Methodology

This chapter is divided into sections: data preparation and preprocessing, feature extraction, building of the classifier, and creation of the TFD prototype.

3.1 The Dataset

A total of 2513 English tweets were streamed using Tweepy, a library that enables Python to communicate with Twitter platform and use its API. The posts were tweeted around March 2017 to keep up with the recent trends.

The tweets are manually labeled one by one to represent tweets that exemplify Flaming and those that do not. In determining what constitutes Flaming, two rules are observed in the labelling process of tweets:

1. The presence of a user mention in a tweet should be present.
2. The insult or vulgarity should be directed at the user being mentioned in the tweet.

Tweets that contain a manifestation of Flaming are be labelled 1 and tweets that are free of Flaming presence are labelled 0.

3.2 Preprocessing

In the preprocessing of data, the first step was removing the whitespaces and punctuations. The documents were also transformed into lowercase

In the first phase of preprocessing, each tweet were freed from punctuations and numbers. However, user mentions which are tokens that are tagged with '@' are an exception because Flaming is a type of cyberbullying that is directed to someone and user mentions satisfies that premise.

Next, each tweet was manually checked for misspellings one by one. Manually checking ensures that the content of the tweet will not be altered because using a tool to correct misspellings might change the meaning of the tweet or the term itself. Misspelled words and unnecessary repetition of characters are corrected so that it won't add unnecessary weight to the data.

Afterwards, all tweets are converted to lowercase for uniformity and after which stopwords are removed. Stopwords are words filtered out before or after processing of natural language data or text for they do not contribute important significance to a study. Using a stop word list significantly reduces the number of postings that a system has to store and thus reduces unnecessary weight to the features (Manning, C. D., et al., 2009).

3.3 Tweet Representation and Feature Extraction

After each tweet is preprocessed, it is now ready for feature extraction. A unigram and bigram collection of words are collected for the Bag-of-Words model. Afterwards, the *tf-idf* scheme are applied. The scheme is such follows:

$$w_{i,j} = TF_{i,j} \; x \; log\,(\frac{N}{DF_i})$$

Where TF_{ij} is the term frequency of the i-th word in j-th document, DF_i is the number of documents containing i-th word *N* is the number of documents.

Afterwards, the *tf-idf* features is concatenated with the features extracted by Latent Semantic Analysis or LSA. LSA is a technique that analyzes relationship between a set of documents and terms (K. Landauer et al. 1998). This method will extract contextual-usage meaning of words by statistical computations applied to a large corpus of text. This feature analyzes text and documents and retrieves sets of concepts related to the document and terms.

3.4 Model Building

After the features have been extracted, they were then shuffled and then divided into 70-30 where 70% are used for the training phase of the model and 30% for the testing.

3.5 Twitter Flaming Detector

After the generation of the model, it is then used for the Twitter Flaming Detector prototype. The prototype is for an engine that automatically detects Flaming in Twitter, and then feeds the new data back into the system for the remodeling of the classifier. The prototype serves as the study's proof of concept for incremental learning of Flaming Detection.

The model for classification implemented beforehand is loaded and used for the later part in detecting Flaming. A tweet then undergoes the same predefined preprocessing stages. Afterwards, it is evaluated by the model on whether it contains a manifestation of Flaming or not. The user will then be prompted on the result and will confirm if he/she feels like she is being bullied for proper labeling. The new data is then be used to update the model for online learning.

4 Results and Discussions

4.1 Labelling

Labeling of the gathered tweets prepared the dataset for the training and testing. Out of the total 2513 tweets gathered, 1612 (64.1%) tweets were labeled as 0 - does not contain Flaming, and 901 (35.9%) tweets were labeled as 1 - contains a manifestation of Flaming. The dataset is skewed to begin with and might create problems later on.

4.2 Initial Model Performance

Both Non-Flaming and Flaming class performed fairly. It was noted that majority of the test tweets were classified correctly.

Table 4.1 Initial Model Performance Results

Class	Precision	Recall	F-Score
Non-Flaming	80%	77%	79%
Flaming	61%	66%	63%
Overall	74%	73%	73%

It can be noted that the classifier performed well in the Non-Flaming class. The classifier scored a recall of 77% which signifies a relatively good classification of Non-Flaming tweets. Moreover, with a precision of 80%, this means that the Non-Flaming class was able to retrieve relevant instances of the class. Overall, the non-flaming class performance gave a 79% f-score overall.

On the other hand, the Flaming class performance generated 61% as its precision score, and 66% for its recall score. While the scores are not high, these show that the method was able to extract significant patterns in the data to represent Flaming tweets. The reason for this performance is because of the skewed dataset where the Flaming instances were under-represented compared to the Non-Flaming examples which was vastly more represented. The skewed dataset exhibited a bias towards the majority label which is the Non-Flaming.

In general, the classification produced an overall average precision of 74% and recall of 73%. This means that the model had a good performance overall compared to the BullyTracer, although it considered many norms of cyberbullying, only produced a 58.63% score overall. The initial model generated in this setup is now ready to be tested for actual real-time evaluation.

4.3 TFD Prototype Performance

A real-time experiment was not possible due to time constraints, however, to make up for this, evaluation of the engine's performance were conducted in a simulated environment. For the simulation, new datasets were fabricated for the purpose of testing the capabilities of the Real Time Online Learning System.

4.3.1 Balanced Dataset

A 50 Flaming - 50 Non-Flaming dataset was utilized. In this experiment, the new dataset was used to update the original model and the initial testing dataset was used to evaluate the performance of the updated model.

Table 4.2 Balanced Dataset Performance

Class	**Precision**	**Recall**	**F-Score**
Non-Flaming	81%	82%	82%
Flaming	67%	65%	66%
Overall	76%	76%	76%

As seen in Table 4.2, this experiment improved the performance of the initial model. The performance of the old model improved when trained using the new balanced data. Compared to the initial model's results, the model improved given the 100 new examples that were introduced. The Non-Flaming class generated an improved f-score of 82% compared to the initial model's 80%. Moreover, the Flaming class improved as its 66% f-score is higher than the initial model's 63%. Overall, the new data improved the initial model's performance. This proves that over time, the model can be improved.

4.3.2 Skewed Dataset

To check the model's performance when the data is skewed, a new dataset was also fabricated for evaluation. 74 Non-Flaming - 26 instances made up the dataset for this experimentation. Like the preceding experiments, the new dataset was used to update the original model and the initial testing dataset was used to evaluate the performance of the updated model.

Table 4.3 Skewed Dataset Performance

Class	**Precision**	**Recall**	**F-Score**
Non-Flaming	82%	76%	79%
Flaming	61%	70%	65%
Overall	75%	73%	74%

In Table 4.3 above, it can be observed that the model was also able to generate an improved performance using the skewed dataset. Overall, this setup induced an overall performance f-score of 74% which means that the model was able to update itself and improved its performance using the fabricated tweets even though it is skewed. Moreover, the Flaming class generated better results compared to the initial model's.

In general, the simulation processes' results proved that over time, the Flaming Detection Model can be improved and can adapt to new patterns of Flaming.

5 Conclusions and Recommendations

5.1 Conclusions

The classifier's initial model generated a 73% f-score, but had problems in correctly classifying tweets with Flaming attributes because of the vastly skewed dataset. The learning classification was implemented in a Flaming Detection prototype. Controlled tests confirmed that with the addition of new data, the model for classifying flaming can be improved. This research's output shows promise for implementing a real-time, learning environment for detecting flaming.

5.2 Recommendations

This study recommends a further research on other cyberbullying norms and to consider these norms in the engine. This study have already established an environment for learning models with introduction of new data. It is highly recommended that classification of the other cyberbullying norms be included in the prototype towards the development of full-blown real-time and learning detection of cyberbullying system.

Moreover, this study was unable to test out the prototype in a real-time setting due to time constraint; thus, supplementary testing of the API should be done to further evaluate its efficiency. Finally, the engine prototype is recommended to be applied as a third-party application of social networking sites for this research to serve its purpose which is to help alleviate the big issue of cyberbullying.

References

Ebizmba. (2017). The Most Popular Social Networking Sites.

Moreau, E. (2017, April 28). The Top Social Networks People Are Using Today. Retrieved May 04, 2017

Hinduja, S. & Patchin, J. W. (2010). Bullying, Cyberbullying, and Suicide. Archives of Suicide Research, 14(3), 206-221.

Middle School Cyberbullying Curriculum . (2008). Retrieved December 31, 2016.

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

Kontostathis A.,Edwards L., Leatherman A.,(2009), "ChatCoder: Toward the tracking and categorization of internet predators." In: Proceedings of SDM 2009, Sparks, NV, May 2 2009

Bayzick, J. (2011). Detecting the Presence of Cyberbullying Using Computer Software

Thom, B (2011) SafeChat: Using Open Source Software to Protect Minors from Internet Predation

Read J., Bifet A., Pfahringer B., Holmes G. (2012) Batch-Incremental versus Instance-Incremental Learning in Dynamic and Evolving Data. In: Hollmén J., Klawonn F., Tucker A. (eds) Advances in Intelligent Data Analysis XI. IDA 2012. Lecture Notes in Computer Science, vol 7619. Springer, Berlin, Heidelberg

Finkel J.R, Kleeman A., Manning C. (2008). Efficient, Feature-based, Conditional Random Field Parsing. Proc. Annual Meeting of the ACL.

Manning, C. D., Raghavan, P., & Schütze, H. (2009). *Introduction to information retrieval*. New York: Cambridge University Press.

Landauer K, Foltz P.W, and Laham D. (1998). An introduction to latent semantic analysis. Discourse processes, 25(2-3):259–284, 1998.