

# Real Time Learning Detection of Flaming in Twitter

A Special Project Presented to the  
Faculty of the Department of Computer Science,  
University of the Philippines Cebu

In Partial Fulfillment  
Of the Requirements for the Degree  
Bachelor of Science in Computer Science

By  
Jessa Mae Y. Cabigas

June 2017

The Special Project entitled

**REAL TIME LEARNING DETECTION OF FLAMING IN TWITTER**

has been examined and recommended for acceptance and approval.

AILEEN JOAN O. VICENTE, MSIM

Adviser

---

Date

-----

Accepted in partial fulfillment of the requirements for the degree of Bachelor of Science in  
Computer Science.

ROBERT R. ROXAS, Ph.D.

Chair, Department of Computer Science

---

Date

## ACKNOWLEDGEMENT

“Anytime you see a turtle up on top of a fence post, you know he had some help.”

-Alex Haley

Likewise with the quote by Alex Haley, this research would have never been completed without the help of several wonderful people. Without them, I wouldn't be able to reach this goal.

To my core inspiration and motivation to finish this research, I express my love and sincerest gratitude to my SP adviser, Miss Aileen Joan O. Vicente. With your persistent support and guidance, you have given me light amidst my darkest research days. I thank you for always pushing me to do better.

To my parents who have never complained about me going home late and not doing some chores at home, thank you for all the love, understanding, and utmost support. I promise, I will clean my room soon.

To my friends who have never whined whenever I can't go out with them, I swear I will make it up to you. Thank you for always being there for me whenever I needed a break.

To my constant person, thank you for quietly staying by my side. For sending me food, handling me at my most frustrating moments, and for always telling me that I can do it, I thank you very much.

To my co-advisees, thank you for pressuring me to finish my research. Seeing you perform well in your respective studies motivated me to achieve the milestones I have set for myself. Thank you for sharing this journey with me.

To my Kultu group, thank you for giving me a smile whenever I needed it the most. Thank you for the solace in times of woe, and for the immediate support in times of need. Without you, college wouldn't be as fun and memorable.

To the endless support you have given me all throughout my entire life, I thank you God for silently expressing your love and guidance. Words won't ever be enough. I offer this to you.

## **RESEARCHER'S PROFILE**

### **Personal Background**

Name: Jessa Mae Y. Cabigas

Birthdate: May 11, 1996

Age: 21

Course and Year Level: BS in Computer Science IV

Address: Sarihville, Poblacion, Talisay City, Cebu

E-mail Address: [jessacabigas@gmail.com](mailto:jessacabigas@gmail.com)

### **Educational Background**

Elementary:

St. Therese's School

Jose Rizal St, Talisay City, Cebu, Philippines

Secondary:

University of San Jose - Recoletos

Basak Pardo Cebu City, Cebu, Philippines

Tertiary:

University of the Philippines Cebu

Gorordo Avenue, Lahug, Cebu City, Cebu, Philippines

## ABSTRACT

Flaming or bashing is a specific manifestation of cyberbullying where users directly fight each other virtually by using messages that are insulting and vulgar. Flaming has become one of the most common manifestation of cyberbullying, especially among minors. Systems have been proposed to detect cyberbullying and to address the matter. The 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. Future works include application of the prototype in a real social media setting as an API and the inclusion of other cyberbullying norms in the classification.

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

## LIST OF TABLES

<i>Table</i>	<i>Page</i>
3.1 Sample Tweets and their labels.....	21
4.1 Final Representation Performance Measure.....	36
4.2 Balanced Dataset Performance.....	38
4.3 Skewed Dataset Performance.....	39

## LIST OF FIGURES

<i>Figure</i>	<i>Page</i>
2.1 Sample of Flaming in Twitter.....	6
2.2 Sample of Flooding in Twitter.....	7
2.3 Sample of Cyberthreats in Twitter.....	10
2.4 Sample of Exclusion in Chat Conversations.....	11
2.5 Cycle in creating an incremental model.....	13
2.6 Stochastic Gradient Descent Contour Plot.....	16
3.1 Study Framework.....	19
3.2 Sample of Stopwords.....	23
3.3 Sample Matrix of TF-IDF Features.....	26
3.4 Sample of Terms Related Context-Wise via LSA.....	27
3.5 Sample Matrix of LSA Features.....	28
3.6 Training Data Representation.....	28
3.7 Confusion Matrix Structure .....	30
3.8 Engine Architecture.....	33
4.1 Class Distribution in the Dataset.....	34
4.2 Confusion Matrix for Classifier Performance .....	35
4.3 Confusion Matrix for Improved Model using Balanced Data .....	37
4.4 Confusion Matrix for Improved Model using Skewed Data .....	39

## TABLE OF CONTENTS

	Page
ACKNOWLEDGEMENT.....	i
RESEARCHER’S PROFILE.....	ii
ABSTRACT.....	iii
LIST OF TABLES.....	iv
LIST OF FIGURES.....	v

### CHAPTER:

#### 1. Introduction

1.1. Background of the Study.....	1
1.2. Research Objectives.....	3
1.2.1. General Objectives.....	3
1.2.2. Specific Objectives.....	4
1.3. Scope and Limitation .....	4
1.4. Significance of the Research.....	5

#### 2. Review of Related Literature

2.1. Flaming and Other Manifestations of Cyberbullying.....	6
2.2. Cyberbullying Detection Systems.....	12
2.3. Instance-based versus Batch-based Learning.....	13
2.4. Stochastic Gradient Descent.....	14



3.	Methodology	
3.1	Study Framework	18
3.2	The Dataset	20
3.2.1	Labeling	20
3.3	Preprocessing	21
3.3.1	Removal of Links and Punctuations and Numbers	21
3.3.2	Cleaning Misspelled Words	22
3.3.3	Removal of Stopwords and Conversion to Lowercase	23
3.4	Feature Extraction	23
3.4.1	Bag of Words	24
3.4.2	Term Frequency times Inverse Document Frequency	25
3.4.3	Latent Semantic Analysis	26
3.4.4	Final Representation	28
3.5	Machine Learning Classifier	29
3.6	Performance Evaluation	30
3.6.1	Confusion Matrix	30
3.6.2	Classification Report	31
3.7	Real Time Flaming Detection Engine	32
3.7.1	Engine Architecture	32
3.7.2	Performance Evaluation	33

4.	Results and Discussion	
4.1	Class Distribution .....	34
4.2	Classifier Performance .....	35
4.3	Real Time Flaming Detection Engine Performance .....	37
4.3.1	Balanced Dataset .....	37
4.3.2	Skewed Dataset .....	38
5.	Conclusions and Recommendations	
5.1	Conclusions .....	41
5.2	Recommendations .....	42
Appendices		
A.	Technical Document.....	45

# CHAPTER 1

## INTRODUCTION

### 1.1 Background of the Study

Twitter is a social networking (SNS) and microblogging service, enabling registered users to read and post short messages 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. The SNS has offered its users with remarkable features such as easy tracking of the most trending topics because of the “Trends” sidebar that keeps its users up-to-date to the most recent hottest topics all over the globe, has less advertisements, keeps content focused and brief because of the 140 character requirement, and provides full control to the user on the content displayed in his timeline. 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” (Patchin J. & Hinduja S., 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. Victims of this predicament were said to be twice as likely to

have attempted suicide compared to those young people who have not experienced cyberbullying (Patchin J. & Hinduja S., 2010).

To help resolve or alleviate the matter, researchers have created cyberbullying detection models by utilizing different machine learning techniques and creating classifiers (Nadali et al., 2013). Nadali et al. (2013) reviewed researches directed at cyberbullying. Researches on cyberbullying detection has been done through the text mining paradigm such as the ChatCoder (Kontostathis et al., 2009) and spam detections; however, their study on technical solutions were inadequate and has thus resulted to insufficient and ambiguous training dataset. Dinakar et al. (2011) performed experiments on the efficiency of binary classifiers against multiclass classifiers using a manually labelled Youtube comments dataset and concluded that binary classifiers outperformed the other one. Reynolds et al. (2011) got a 78.5% accuracy using Decision Trees and replicated positive examples for evaluation. Studies influenced the work of Dadvar et al. (2012) about the effects linguistic features based on gender in the diagnostics of cyberbullying on social networks where men and women have patterns in using pronouns. A graph model for extraction of cyberbullying network was presented by Nahar and Pang (2013) and their study allowed the identification of the most active predators and victims using ranking method. It was noted that most researches were academic in nature and its application to a real-time setting has not been thoroughly if at all, studied.

Two popular implementations for cyberbullying were made available - ChatCoder (Kontostathis et al., 2009) and BullyTracer (Bayzick, J., 2011). BullyTracer is a computer software that classifies chat conversations from MySpace into norms of cyberbullying using a rule-based

algorithm. Meanwhile, ChatCoder is a computer software that analyzes and classifies chat log transcripts into determining the class of the coded dialog - whether it is a predator or a victim class, using a decision tree. These two popular implementations perform well at classification performance. What these systems lack is (a) ability to be implemented in a real-time setting, and (b) ability to update its knowledge base. It is important that cyberbullying detection models should be able to dynamically adapt to new patterns of data or when the data itself is generated as a function of time, most especially in the case of cyberbullying where patterns and the use of languages evolve over time.

## 1.2. Research Objectives

### 1.2.1 General Objective

The present implementations of cyberbullying system work offline, heavily depending on pre-generated models and rules to detect cyberbullying events. This implementation can be further improved by applying a learning aspect to the generation of models. The general objective, therefore, of this study is to develop a real-time and learning detection of Flaming in Twitter.

### 1.2.2 Specific Objectives

To fulfill the objective mentioned above, the study aims the following:

- To obtain a cyberbullying dataset from Twitter to be used for training and testing.
- To extract features from tweet representation.

- To generate classifier in detecting Flaming using Support Vector Machine with Stochastic Gradient Descent.
- To implement a learning classification in a Twitter Flaming Detection prototype.
- To evaluate the classification performance of the Twitter Flaming Detection prototype over time.

### 1.3. Scope and Limitation

There are many types of cyberbullying norms, this study, however, is delimited to Flaming - a specific manifestation of bullying where users directly fight against each other virtually by using messages that are insulting and vulgar. Flaming has become one of the most common manifestation in cyberbullying, especially among minors, that it got the attention of teachers (Middle School Cyberbullying Curriculum , 2008) and thus, addressing this problem can create a great effect in the cyberbullying phenomenon.

The social network platform that is the solid ground of the research is in the context of Twitter. In addition, Flaming is easier to detect in Twitter because the SNS, or Social Networking Site, has a limited way of communication - one of which is posting a tweet directed to a specific user using “@” username mention.

#### 1.4. Significance of the Research

This study is very timely because of the increased awareness on cyberbullying. Although prevalent in the past, the 21<sup>st</sup> century has aggravated bullying by introducing new platforms to inflict harm onto others.

Although other researchers have studied and have laid a strong foundation for the detection of Cyberbullying in some SNS, the classifiers previously implemented did not support features that allows the users to update the knowledge base themselves in real time.

This study shall extend the present state of cyberbullying detection researches by utilizing a continuously learning SVM classifier in real time. As this study aims to implement an incremental SVM, the model will be updateable unlike the previous classifiers implemented using batch-based learning approaches that discards the existing knowledge gained from the initial training and then trains again with the appended data. With this, detecting cyberbullying can be made more intelligent as the model is being continuously updated by each tweet of the user. This approach leads more to a real time continuous remodeling of the classifier compared to the previous implementations.

## CHAPTER 2

### REVIEW OF RELATED LITERATURE

#### 2.1 Flaming and Other Types of Cyberbullying

There are many norms that can be considered in classifying a manifestation of cyberbullying. Each has their own intention of harm inflicted upon their victims.

**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). This is one of the most common manifestation of cyberbullying according to Middle School Cyberbullying Curriculum (2008).

In Twitter, Flaming can be easily detected due to the presence of @username tagging/replying ability in a Twitter feed conversation. For example, Figure 2.1 shows cyberbullying where user2 directly and publicly ridiculed user1.



Figure 2.1 Sample of Flaming in Twitter



**Flooding** is another form of cyberbullying where the bully constantly enters nonsensical comments in order to deny the victim to join in on the conversation or to annoy the victim (D. Maher, 2008). This type of cyberbullying is mostly found in chat conversations. For example, Figure 2.2 presents a manifestation of flooding where user1 creates a conversation thread with user2 just to flood her with annoying contents.



Figure 2.2 Sample of Flooding in Twitter

April enters blank messages in the chat conversations which disables other users in the chatbox to join in the conversation followed by Jessa's protest "Stop that". The series of nonsensical contents or repeated comments by the bully succeeded by a protest from a different user constitutes flooding.

**Masquerade** is another form of cyberbullying wherein the bully takes up a different identity in order to mask his own as he bullies the victim online (N.E. Willard, 2007) . Here are a few examples:

- Jessa can take up the identity of Mae and enters inappropriate posts about Mae - making it look like Mae is embarrassing herself.
- Jessa can take up the identity of Mae and bullies a different user named Faith - making it look like Mae is the bully.
- Jessa can take up an entirely unknown identity and bullies Mae - frustrating Mae in a way that she won't know who and why she is being bullied by an unknown person.

This type of cyberbullying is hard to detect because there is no way to verify the identity of a user account in social media. There has to be proof that a different person is using the account of a different person to detect malicious hacking of an existing account. Moreover, in the case where the bully takes up an unknown identity of a newly created account, it is much harder to detect masquerade. Account information can be easily set to private; thus, making it difficult for the victim to know his bully. Furthermore, the bully can easily lie when it comes to inputting his own information; thus, hiding the bully's actual information.

**Trolling**, can also be known as baiting, is a form of cyberbullying wherein the bully intentionally posts his/her views online that would be in contrast to other people's views – thus, baiting the other users to create an argument among others (Glossary of cyberbullying terms, 2008, January).

For example,

- Jessa joins an LGBT support community thread and posts “There are only two genders.”
- During a time of heavy calamity where there are a lot of casualties, Jessa posts “Well, everyone dies anyway.”
- Jessa posts a racy comment “Black people are slaves to the white men” in an interracial platform.

This type of cyberbullying entails a study of sarcasm which has never been successfully detected since it is difficult to detect a person’s true intentions on the Internet. Sarcasm can be linked to trolling because a person can easily post an opinion in a forum that isn’t really his actual stand on a matter, but just want to provoke other people’s emotions.

**Harassment** is the usual and traditional term applied whenever “cyberbullying” is used – this is the kind of cyberbullying where the bully insults and sends rude messages to the victim over an extended period of time (N.E. Willard, 2007).

**Cyberstalking** or **cyberthreats** is another form of cyberbullying wherein the bully sends threatening, extorting, and offensive messages to the victim (N.E. Willard, 2007). For example in Figure 2.3,

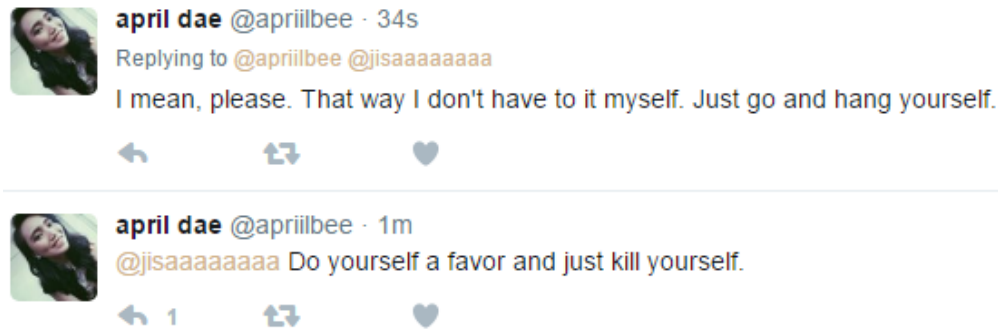


Figure 2.3 Sample of Cyberthreats in Twitter

This type of cyberbullying instills fear in the heart of the victim - the victim may begin to start fearing for his own life, or to start considering taking his own life.

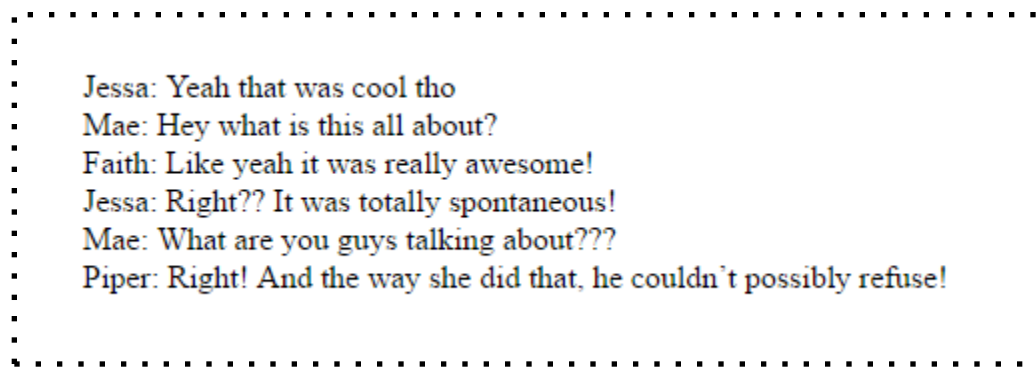
**Denigration** happens when the bully spreads false information about the victim that can cause the latter's reputation to be ruined (N.E. Willard, 2007). Denigration can also be referred to as gossiping about someone online and writing untrue rumors so that other people may turn against the victim.

Another form of denigration is **outing**, however; it has to be taken into account that the bully and the victim are close friends (N.E. Willard, 2007). For example,

- The victim tells the bully his personal information or problems and then the bully spreads it to other people.
- The victim and the bully are having a private conversation, and the bully claims to be the only person reading the conversation when in fact, there are also others reading it.
- The victim and the bully are having a private conversation, and the bully claims to keep the conversations between themselves and then forwards it for others to read.

This type of cyberbullying is more accurate when it comes to the information being shared compared to denigration because of the personal connection between the bully and the victim. The victim trusted the bully enough to confide personal information and thus adding credibility to the information being disseminated.

And lastly, **Exclusion** – it happens generally to society, especially to the youth. This is the type of cyberbullying wherein the bully excludes the victim into joining a particular notion - denying the victim to be involved in a particular social milieu (Patchin, J., & Hinduja, S, 2006). For example, in the Figure 2.4 below presents a



Jessa: Yeah that was cool tho  
Mae: Hey what is this all about?  
Faith: Like yeah it was really awesome!  
Jessa: Right?? It was totally spontaneous!  
Mae: What are you guys talking about???  
Piper: Right! And the way she did that, he couldn't possibly refuse!

Figure 2.4 Sample of Exclusion in Chat Conversations

The conversation above contains the presence of exclusion. It is clear that Mae is being ignored by her peers in the chatbox and they refuse to acknowledge her presence. “What are you guys talking about???” constitutes the presence of exclusion in this conversation. However, it is still difficult to detect this type of cyberbullying since ignoring a topic being brought up might not necessarily mean that they are trying to bully the person.

## 2.2 Cyberbullying Detection Systems

A research by Bayzick, J. (2011) implemented a computer software, named BullyTracer, that detects many of the manifestations mentioned in Section 2.1. 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.

### 2.3 Instance-based versus Batch-based Learning

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 2.5 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.

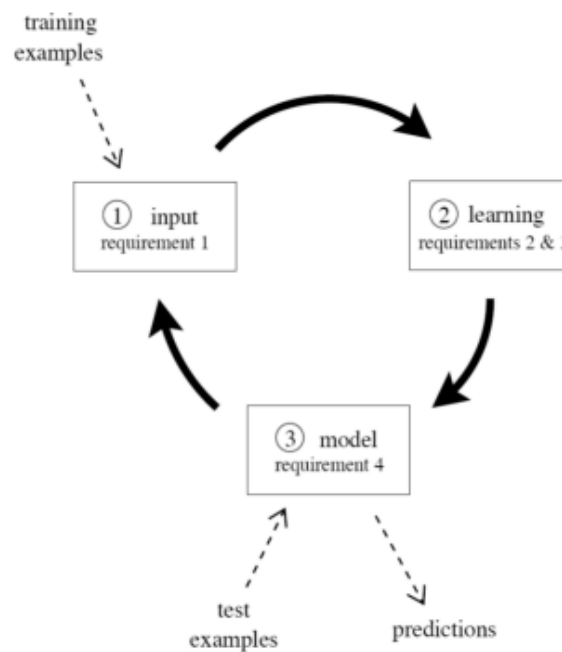


Figure 2.5 Cycle in creating an incremental model (Read et al., 2012).

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.

## 2.4 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 (see, e.g., Vowpal Wabbit) 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 (S., 2016).

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 (Optimization: Stochastic Gradient Descent., n.d.).

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.

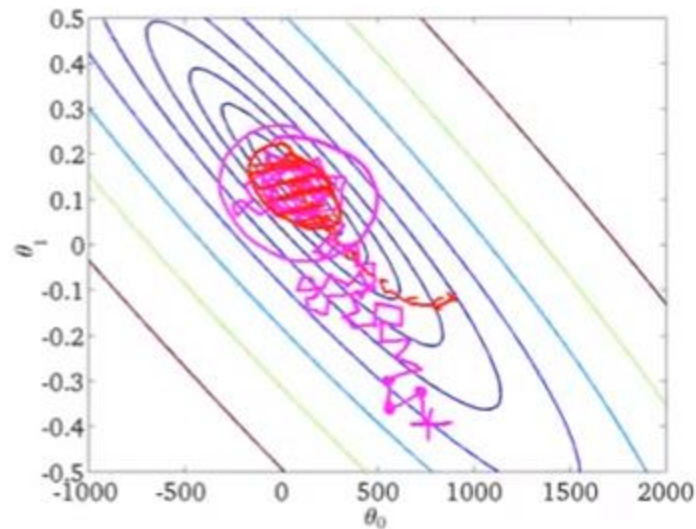


Figure 2.6 Stochastic Gradient Descent Contour Plot (Ng, A., 2013)

As shown in Figure 2.6, the purple lines denote the SGD's performance in finding the convergence and the red lines denote the Batch Gradient Descent. As you can see, the BGD will tend to take a reasonably straight line trajectory to get to the global minimum. In contrast the SGD, every iteration is going to be much faster because we don't need to sum up over all the training examples. Every iteration is just trying to fit the single training example better. SGD doesn't converge as the same sense as BGD does, and it ends up doing a wandering around continuously in the some region close to the global minimum as you can see in the figure. But in practice, this isn't a problem because as long as the parameters end up in some region close to the global minimum, it will produce a pretty good hypothesis (Ng, A., 2013).

SGD is made available in the sklearn Python library. Their implementation of SGD is based on Léon Bottou's work on Stochastic Gradient SVM. Similar to his work, the weight vector is represented as the product of a scalar and a vector which allows an efficient weight update in the case of L2 regularization. In the case of sparse feature vectors, the intercept is updated with a

smaller learning rate (multiplied by 0.01) to account for the fact that it is updated more frequently. Training examples are picked up sequentially and the learning rate is lowered after each observed example (S., 2016.).

## CHAPTER 3

### METHODOLOGY

#### 3.1 Study Framework

Figure 3.1, shows the pipeline of the entire study. First thing to be done is to connect to the Twitter API in order to generate a collection of tweets that shall comprise the Flaming dataset. Section 3.2 discusses the methodologies done upon the creation of the dataset. Afterwards, each tweet is preprocessed to prepare for feature extraction. Each preprocessing procedure is discussed in detail in Section 3.3. These procedure shall prepare the data for feature extraction which will be discussed in Section 3.4. Features will be gathered and concatenated in preparation for the training and testing datasets. In Section 3.5, the concatenated data will be fed into the classifier. This will be the creation of the knowledge base of the research. Afterwards, the knowledge base will be tested and evaluated using f-score metrics in Section 3.6. A Real Time Online Learning Flaming Detection Engine will then be created and discussed further down in Section 3.7.

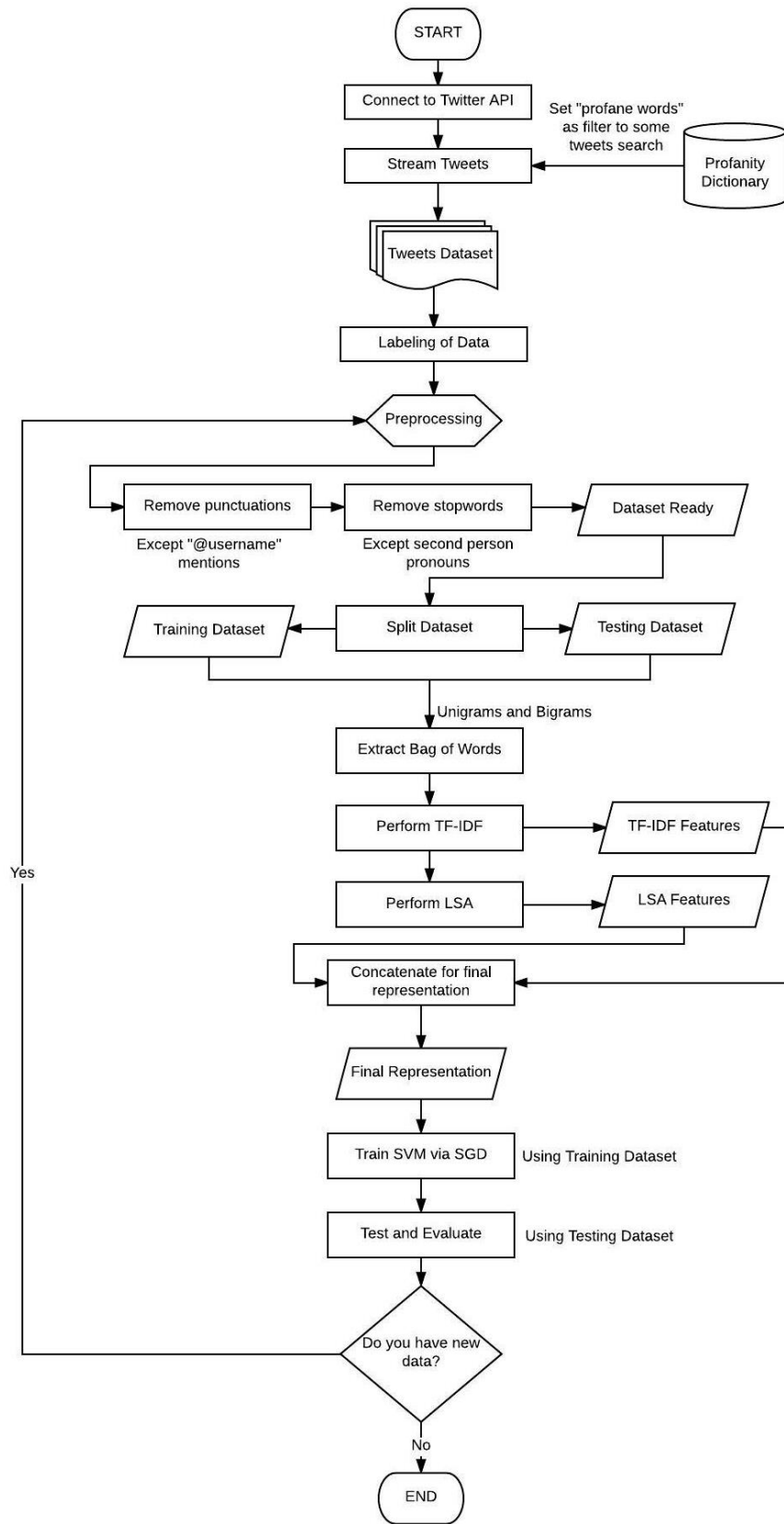


Figure 3.1 Study Framework

## 3.2 The Dataset

A total of 2513 English tweets were streamed using Tweepy (Tweepy, 2016), 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.

### 3.2.1 Labeling

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.

For example:

Table 3.1 Sample Tweets and their labels

Tweet	Label
@thetomska fuck you and fuck your stupid ass fans	1
Ahh this dish tastes like shit fuck this	0
@VisageAwko you nasty bastard	1
Shaina you bitch	0

### 3.3 Preprocessing

Each tweet in the dataset will undergo several preprocessing stages in preparation for feature extraction.

#### 3.3.1 Removal of Links and Punctuations and Numbers

Tweet tokens that are tagged with 'RT' (retweets) and 'U' (links) or tokens that start with 'http' or 'www' are removed from the tweet. Numbers and punctuations are also removed from the tweets, except the user mentions which are tokens that are tagged with '@' because Flaming is a type of cyberbullying that is directed to someone and user mentions satisfies that premise.

An example preprocessing is shown below.

**~ Original Tweet:**

*"I am such an arsehole when i am sober; as soon as i am intoxicated ... i am the nicest person you could meet"*

**~ Cleaned Tweet:**

*"I am such an arsehole when i am sober as soon as i am intoxicated i am the nicest person you could meet"*

### 3.3.2 Cleaning Misspelled Words

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.

**~ Original Tweet:**

*"@jessa sucha bitch butttt nah nevermindddddd"*

**~ Cleaned Tweet:**

*"@jessa such a bitch but nah nevermind"*



### 3.3.3 Removal of Stopwords and Conversion to Lowercase

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).

a	an	and	are	as	at	be	by	for	from
has	he	in	is	it	its	of	on	that	the
to	was	were	will	with					

Figure 3.2 Sample of Stopwords

#### **~ Original Tweet:**

*"I am such an arsehole when i am sober as soon as i am intoxicated i am the nicest person you could meet"*

#### **~ Cleaned Tweet:**

*"i arsehole i sober soon i intoxicated i nicest person you could meet"*

### 3.4 Feature Extraction

This section presents the methods used to extract feature for analysis. In this section, two set of features will be extracted - Term Frequency times Inverse Document Frequency (TF-IDF) and Latent Semantic Analysis (LSA).

TFI-IDF has notably been used among many researches especially in the field of text analysis and information retrieval. This feature determines the importance of the word in a given text; thus, making it an efficient feature in text analysis, sentiment analysis (Shinde, G., & Deshmukh, S. N. ,2016), and unstructured text document classification (Yoo, J., & Yang, D., 2015).

LSA has also been used widely in terms of text analysis. This feature analyzes text and documents and retrieves sets of concepts related to the document and terms thus it overcomes the problem of synonymy among terms. It has been used in document categorization (Landauer, T., et al., 2008), text summarization (Gong, Y., and Liu, X., 2001), automatic keyword annotation (Monay, F., and Gatica-Perez, D., 2003) and more.

### 3.4.1 Bag of Words

The Bag-Of-Words (BoW) feature is used in this research. It will be a collection of all the words both with the unigram and bigram. The words will be stored to the collection for reference later on. The Bag-Of-Words will be the basis of the vector space for each processed tweet content. Each tweet will be mapped to a vector by counting the occurrence of each word. Words that only appear in one document will be dropped to save vector space. This setup is utilized by most Natural Language Processing research such as (Zhao et al, 2006) (Xiang et al., 2012).

### 3.4.2 TF-IDF

Term Frequency times Inverse Document Frequency or TF-IDF is a formal measure used by many NLP techniques in determining how concentrated into a relatively few documents are the occurrences of the given word (Rajaraman, A and Ullman, J. D., 2011). This feature reflects how important a word is to a document in a corpus.

TF-IDF has been widely used especially among Natural Language Processing researches. For example, according to a literature survey of recommender systems by Breitinger et al. (2015), 85% of the systems and researches created utilized the TF-IDF term weighting scheme.

Term weight is used to give gravity or significance to words that occur in the tweets for it may contain terms that belongs in the predefined profanity dictionary. This study shall utilize the *tf-idf* weighting scheme with the *i*-th word in *j*-th document is calculated as:

$$w_{i,j} = TF_{i,j} \times \log\left(\frac{N}{DF_i}\right)$$

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

Each term in the BoW will be crosschecked with the terms in the processed tweet. If a word in the BoW is found in the term, then the scheme will then be applied. Else, it will be set to zero. To maximize efficiency, terms with document frequencies less than 2 will be ignored as also proposed by Zhao et al. (2006) . This paper utilizes the TFIDF Vectorizer by Scikit Learn (2016).

```
[[ 0.  0.  0. ..., 0.  0.  0.]
 [ 0.  0.  0. ..., 0.  0.  0.]
 [ 0.  0.  0. ..., 0.  0.  0.]
 ...,
 [ 0.  0.  0. ..., 0.  0.  0.]
 [ 0.  0.  0. ..., 0.  0.  0.]
 [ 0.  0.  0. ..., 0.  0.  0.]
```

Figure 3.2 Sample Matrix of TF-IDF Features

### 3.4.3 Latent Semantic Analysis

Latent Semantic Analysis (LSA), also known as Single Value Decomposition (SVD), 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.

$$X \approx USV^T$$

$X_{mn}$  is a matrix where m is the number of tweets and n is the number of concepts. The matrix X will be decomposed into three matrices called the U, S, and T. In doing the decomposition, a value of  $k$  will have to be picked since it will represent the number of concepts kept.

$U_{mk}$  is a matrix where the rows are the tweets and the columns will be the mathematical concepts.  $S_{kk}$  is a diagonal matrix where elements will be the amount of variation captured from each concept.  $V_{mk}$  transpose where the rows will be terms and the columns will be concepts.

According to Zhao et al. (2006) and Perez et al. (2005), LSA is effective when it comes to training data for detection of cyberbullying (Xu et al., 2012) (Zhao et al, 2006). This paper utilizes Scikit Learn's SVD Truncated or Single Value Decomposition (Scikit-learn, 2016).

SVD is known to suffer to "sign indeterminacy", which means the sign of the components and the output from transform depend on the algorithm and random state. To work around this, fit instances of this class to data once, then keep the instance around to do transformations (Halko, et al., 2009).

Below is a sample of related concepts extracted by LSA.

<b>Concept 1:</b>	<b>Concept 269:</b>
asap	ass cake
asses you	blessed
asked	ariroseee
ass sucker	bleady
approved	bet
ass bitch	boob hurts
blessing	bunk
banks	bleached anus
bad you	ass fuck
around multiple	biatch you

Figure 3.4 Sample of Terms Related Context-Wise via LSA

In Figure 3.5 below shows the matrix representation result of the LSA technique where the rows are the tweets and the columns are the related concepts nominal value. Values close to 1 represent very similar words while the values that are close to 0 represent very dissimilar words.

```
[ [ 2.01e-01 -4.01e-02 -5.45e-02 ..., 1.97e-02 1.76e-02 -2.60e-02]
[ 1.33e-01 -2.39e-02 -4.34e-02 ..., -2.03e-02 5.44e-02 3.86e-02]
[ 1.52e-01 -5.27e-02 1.83e-01 ..., 3.35e-02 -2.60e-02 1.00e-02]
...,
[ 6.79e-02 -4.40e-03 -1.75e-02 ..., -1.04e-02 -1.80e-03 2.26e-03]
[ 1.09e-01 -1.52e-02 7.86e-03 ..., 1.52e-02 6.88e-03 6.52e-03]
[ 2.08e-01 9.21e-01 1.62e-01 ..., -4.34e-03 3.75e-04 -2.49e-03]]
(1759, 500)
```

Figure 3.5 Sample Matrix of LSA Features

### 3.4.3 Final Representation

The features gathered are concatenated to form the final representation of tweets to be fed into the classifier. The features gathered in the LSA setup will be appended after the TF-IDF features.



Figure 3.6 Training Data Representation

### 3.5 Machine Learning Classifier

Support Vector Machines have been widely used in text classification and natural language processing studies and it has produced exemplary results as stated in the review of Nadali et al. (2013).

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 Support Vector Machine 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 (Scikit-learn., 2016). In contrast to (batch) gradient descent, SGD approximates the true gradient of  $E(w, b)$  by considering a single training example at a time.

This study shall use a Support Vector Machine trained using Stochastic Gradient Descent to equip the classifier with incremental learning. Scikit Learn's SGD Classifier (Scikit-Learn, 2016) which is based on the work of Bottou (n.d). To train a linear Support Vector Machine, we can tweak the parameters of the SGDClassifier by Scikit-Learn with L2 as the regularization penalty and a hinge as it loss parameter.

## 3.6 Performance Evaluation

The performance measure is one of the way to evaluate a solution to the chosen problem. This study shall extract performance measures from a confusion matrix of test results.

### 3.6.1 Confusion Matrix

A confusion matrix indicates the quality of the output of the classifier or the model. This metric presents the performance of the model in detail: true positives - ones with the positive label, and then predicted as positive; the false positives - ones with negative labels but predicted as positive; the false negatives - ones with positive as their actual labels, but predicted as negative; and lastly, the true negatives - ones with negative actual value and then also predicted as negative.

		PREDICTED LABEL	
		Non-Flaming	Flaming
TRUE LABEL	Non-Flaming	<i>True Negative</i>	<i>False Positive</i>
	Flaming	<i>False Negative</i>	<i>True Positive</i>

Figure 3.7 Confusion Matrix Structure



The diagonal elements represent the number of points for which the predicted label is equal to the true positive label, while the non-diagonal elements are those that are mislabeled by the classifier. This means that the higher the diagonal values of the confusion matrix is, the better the performance because it indicates that many true positives were predicted. These values are also used in measuring the Precision, Recall, and the combination of the two called F-Score.

### 3.6.2 Classification Report

Recall, Precision, and F-Score measurements will be calculated to check the efficiency of the classification performance.

Recall is referred as the true positive rate or the rate of sensitivity. In simpler terms, the recall addresses the question, “Given a positive example, will the classifier detect it?”. It is

$$\text{Recall} = \frac{tp}{tp + fn}$$

calculated as:

Where:

$tp$  is the number of true positives

$fn$  is the number of false negatives

Precision is referred as the positive predictive value. In simpler terms, the precision addresses the question, “Given a positive prediction from the classifier, how likely is it to be correct?”. It is calculated as:

$$\text{Precision} = \frac{tp}{tp + fp}$$

Where:

$tp$  is the number of true positives

$fp$  is the number of false positives

### 3.7 Real Time Flaming Detection Engine

This study created a prototype 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.

#### 3.7.1 Engine Architecture

Figure 3.8 shows the architecture of the prototype for the engine. It presents the flow of the engine upon utilization. The model for classification implemented beforehand will be loaded and be used for the later part in detecting Flaming, A tweet will then undergo the same predefined preprocessing stages as discussed in Section 3.3. Afterwards, it will be evaluated by the model on whether it contains a manifestation of Flaming or not. The user will be prompted on the result and will confirm if he/she feels like she is being bullied for proper labeling. The new data will then be used to update the model for online learning.

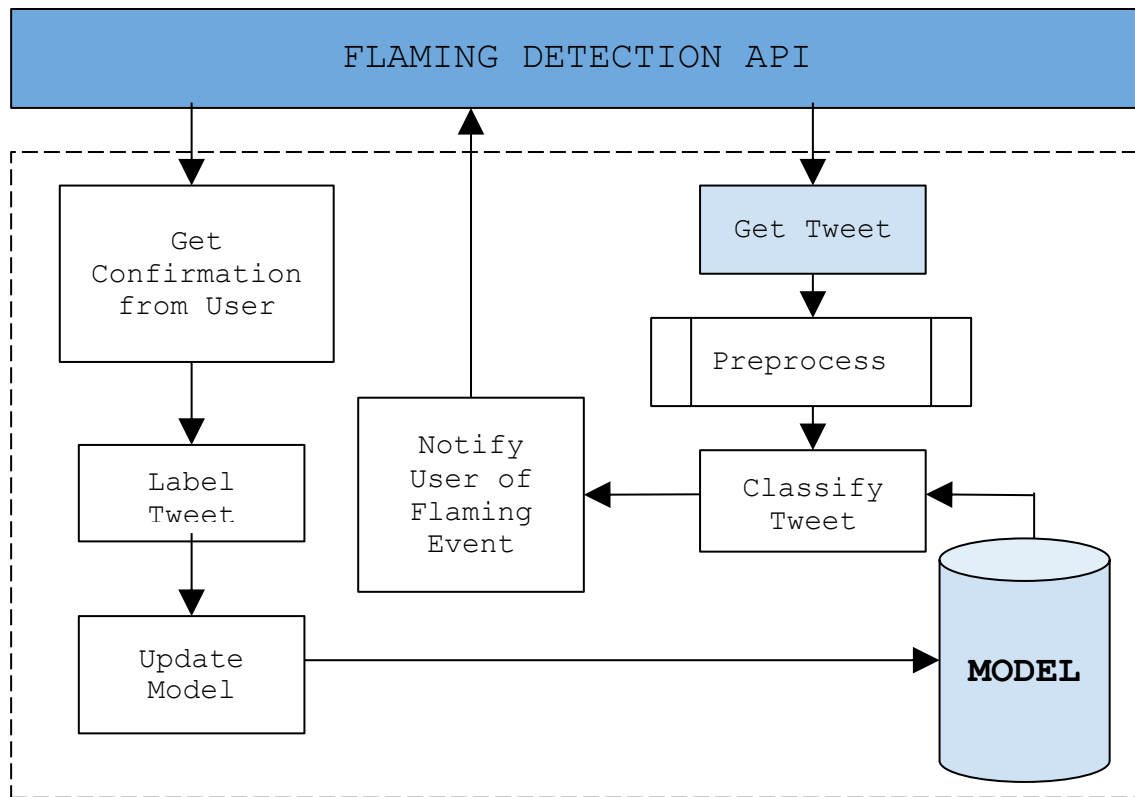


Figure 3.8 Engine Architecture

### 3.7.2 Performance Evaluation

Due to time constraints, the researcher was unable to test the prototype in a real-time setting. To make up for this, a dataset of fabricated tweets that contain Flaming and Non-Flaming characteristics were created and fed into the prototype. Afterwards, the same classifier performance evaluation was conducted.

## CHAPTER 4

### RESULTS AND DISCUSSIONS

In this section, the results of the procedures are presented and discussed. This entire chapter will cover the performance measurement of the model created and the engine that will be the medium for the real time online learning Twitter Flaming detector.

#### 4.1 Class Distribution

Labeling of the gathered tweets prepared the dataset for the training and testing. Out of the total 2513 tweets gathered, 1612 tweets were labeled as 0 - does not contain Flaming, and 901 tweets were labeled as 1 - contains a manifestation of Flaming. The dataset is skewed to begin with and might create problems later on. Figure 4.1 below shows the class distributions in the dataset.

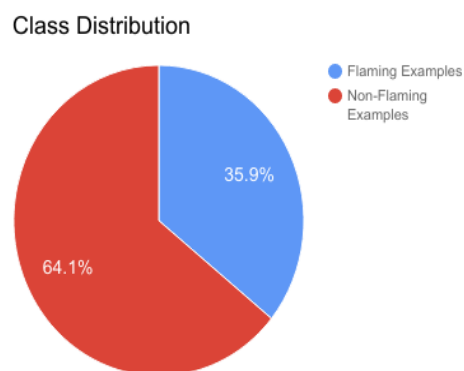


Figure 4.1 Class Distribution in the Dataset

## 4.2 Classifier Performance

The dataset was randomly divided into 70-30 partitions for training and testing. 70% of the dataset was used for training, and 30% of the dataset was used for testing the classifier's performance. The confusion matrix in Figure 4.2 presents the raw results of tests against the remaining 30% test data set.

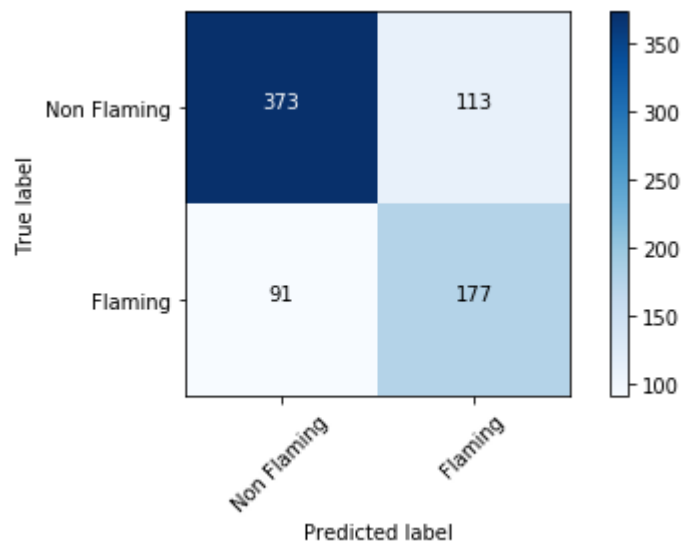


Figure 4.2 Confusion Matrix for Classifier Performance

As seen the figure above, both of the Non-Flaming and Flaming class performed fairly. It was noted that majority of the test tweets were classified correctly. Performance measures were calculated from the matrix and presented in Table 4.1. 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.

Table 4.1 Final Representation Performance Measure

<b>Class</b>	<b>Precision</b>	<b>Recall</b>	<b>F-Score</b>	<b>Support</b>
Non-Flaming	80%	77%	79%	486
Flaming	61%	66%	63%	268
<b>Overall</b>	74%	73%	73%	754

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 Real Time Flaming Detection Engine 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

In the first part, 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. Figure 4.3 shows the detailed performance of the updated model using a confusion matrix.

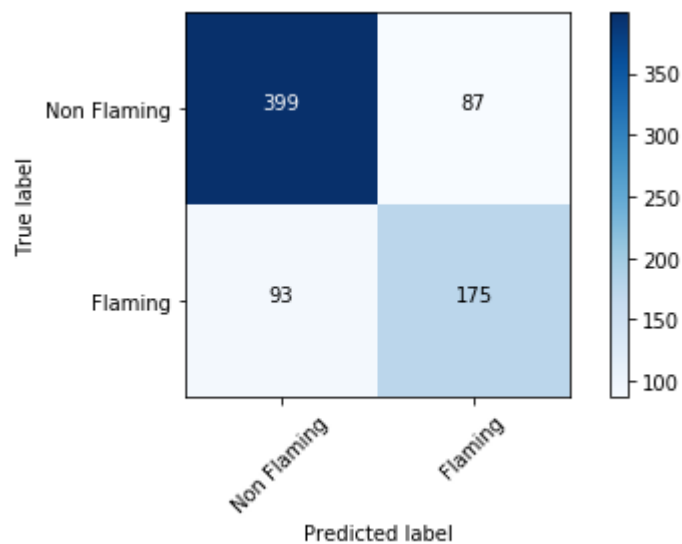


Figure 4.3 Confusion Matrix for Improved Model using Balanced Data

Compared to the performance of the initial model as shown in Figure 4.2, it can be seen that this experiment improved the performance of the model as seen in Figure 4.3. Using the confusion matrix in Figure 4.3, the following measurements were calculated in Table 4.2 below.

Table 4.2      Balanced Dataset Performance

<b>Class</b>	<b>Precision</b>	<b>Recall</b>	<b>F-Score</b>	<b>Support</b>
Non-Flaming	81%	82%	82%	486
Flaming	67%	65%	66%	268
<b>Overall</b>	76%	76%	76%	754

As seen in the table above, 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 Flaming 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.

Figure 4.4 below shows the detailed performance evaluation of this setup.

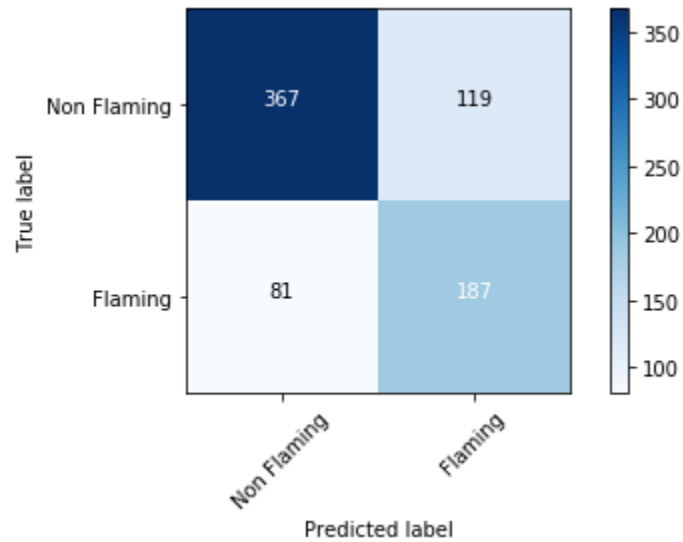


Figure 4.4 Confusion Matrix for Improved Model using Skewed Data

Table 4.3 Skewed Dataset Performance

Class	Precision	Recall	F-Score	Support
Non-Flaming	82%	76%	79%	486
Flaming	61%	70%	65%	268
<b>Overall</b>	75%	73%	74%	754

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.

## CHAPTER 5

### CONCLUSIONS AND RECOMMENDATIONS

#### 5.1 Conclusions

This study was able to obtain a cyberbullying dataset from Twitter for training and testing data using a streaming API called Tweepy. After each tweet in the dataset were cleaned and preprocessed, TF-IDF and LSA features were extracted from each tweet for representation. These features were concatenated to form the final representation per tweet. A Support Vector Machine classifier was then generated and trained using Stochastic Gradient Descent which allowed the SVM incremental learning. 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 a promise for implementing a real-time, learning environment for detecting flaming.

#### 5.2 Recommendations

This study recommends a further research on other cyberbullying norms mentioned in Section 2.1 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

- A. Kontostathis, L. Edwards, A. Leatherman (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: <http://webpages.ursinus.edu/akontostathis/BayzickHonors.pdf>
- Bottou, L. (n.d.). Stochastic Gradient Descent (v.2). Retrieved May 24, 2017, from <http://leon.bottou.org/projects/sgd>
- Breitinger, Corinna; Gipp, Bela; Langer, Stefan (2015-07-26). "Research-paper recommender systems: a literature survey". *International Journal on Digital Libraries*. **17** (4): 305–338.
- C. Morris, and G. Hirst (2012), "Identifying sexual predators by svm classification with lexical and behavioral features" - notebook for pan at clef 2012.
- D. Maher (2008), "Cyberbullying: an ethnographic case study of one australian upper primary school class". *Youth Studies Australia*, 27(4), 5057, 2008.
- E. (2017). The Most Popular Social Networking Sites.
- Glossary of cyberbullying terms. (2008, January). Retrieved from [http://www.adl.org/education/curriculum\\_connections/cyberbullying/glossary.pdf](http://www.adl.org/education/curriculum_connections/cyberbullying/glossary.pdf)
- Gong, Y., and Liu, X. (2001), Creating Generic Text Summaries, Proceedings, Sixth International Conference on Document Analysis and Recognition, 2001, pp. 903–907.
- Halko, et al., (2009). Finding structure with randomness: Stochastic algorithms for constructing approximate matrix decompositions (arXiv:909)
- Hinduja, S. & Patchin, J. W. (2010). Bullying, Cyberbullying, and Suicide. *Archives of Suicide Research*, 14(3), 206-221.
- J.F. Chisholm (2006), "Cyberspace violence against girls and adolescent females." *Annals of the New York Academy of Sciences* 1087, 2006. pp. 74–89.
- Jenny Rose Finkel, Alex Kleeman, Christopher D. Manning (2008). Efficient, Feature-based, Conditional Random Field Parsing. Proc. Annual Meeting of the ACL.
- K. Dinakar, R. Reichart, H. Lieberman (2011), "Modelling the Detection of Textual Cyberbullying." In: ICWSM 2011, Barcelona, Spain, July 17-21 2011.

- K. Landauer, P. W. Foltz, and D. Laham (1998). An introduction to latent semantic analysis. *Discourse processes*, 25(2-3):259–284, 1998.
- K. Reynolds, A. Kontostathis, and L. Edwards (2011), "Using Machine Learning to Detect Cyberbullying," In *Proceedings of the 2011 10th International Conference on Machine Learning and Applications Workshops (ICMLA 2011)*, vol. 2, December 2011. pp. 241-244,.
- Landauer, T., et al. (2008), *Learning Human-like Knowledge by Singular Value Decomposition: A Progress Report*, M. I. Jordan, M. J. Kearns & S. A. Solla (Eds.), *Advances in Neural Information Processing Systems 10*, Cambridge: MIT Press, 1998, pp. 45–51
- M. Dadvar, F. d. Jong, R. Ordelman, and D. Trieschnigg (2012), "Improved cyberbullying detection using gender information," In *Proceedings of the Twelfth Dutch-Belgian Information Retrieval Workshop (DIR 2012)*, February 2012. pp. 23-25,.
- Manning, C. D., Raghavan, P., & Schütze, H. (2009). *Introduction to information retrieval*. New York: Cambridge University Press.
- Middle School Cyberbullying Curriculum . (2008). Retrieved December 31, 2016.
- Monay, F., and Gatica-Perez, D. (2003), On Image Auto-annotation with Latent Space Models, *Proceedings of the 11th ACM international conference on Multimedia*, Berkeley, CA, 2003, pp. 275–278.
- Moreau, E. (2017, April 28). The Top Social Networks People Are Using Today. Retrieved May 04, 2017
- N.E. Willard (2007), "Cyberbullying and Cyberthreats: Responding to the Challenge of Online Social Aggression, Threats, and Distress." Champaign, IL: Research, 2007.
- Nadali, S., Murad, M. A., Sharef, N. M., Mustapha, A., & Shojaee, S. (2013). A review of cyberbullying detection: An overview. *2013 13th International Conference on Intelligent Systems Design and Applications*. doi:10.1109/isddiscovery over a large scale twitter corpus. *Proceedings of the 21st ACM International Conference on Information and Knowledge Management - CIKM '12*, 1980-1984. doi:10.1145/2396761.2398556
- Xu, J.-M., X. Zhu and A. Bellmore, "Fast learning for sentiment analysis on bullying", in "Workshop on Issues of Sentiment Discovery and Opinion Mining", p. 10 (ACM, 2012b).
- Yoo, J., & Y56, D. (2015). Classification Scheme of Unstructured Text Document using TF-IDF and Naive Bayes Classifier. doi:10.14257/astl.2015.111.50
- Zhao, R., Zhou, A., & Mao, K. (2016). Automatic detection of cyberbullying on social networks based on bullying features. *ICDCN '16 Proceedings of the 17th International Conference on Distributed Computing and Networking* (p. 6). New York: ACM New York, NY, USA.

## APPENDIX

### Streaming Tweets:

```
#You need to have these in order to connect to the Twitter API
consumer_key = ''
consumer_secret = ''
access_token = ''
access_token_secret = ''
auth = OAuthHandler(consumer_key, consumer_secret)
auth.set_access_token(access_token, access_token_secret)
api = tweepy.API(auth)

def getTweet():
    for tweet in tweepy.Cursor(api.search, lang="en").items():
        if not((tweet.text).startswith("RT")):
            # this means retweeted tweets are not included
            if "http" not in tweet.text and "www" not in tweet.text and "https" not in
tweet.text:
                #this means that tweets with links are not included
                print tweet.text

f.write(json.dumps(tweet.text).decode('unicode_escape').encode('ascii','ignore')+'\n')
```

### Remove Stopwords:

```
def remove_stopwords(tweet):
    new_tweet = ""
    for i in tweet.lower().split():
        if i not in stop:
            #stop is an array containing all the stopwords
            new_tweet += i + " "
    return new_tweet
```

### Remove Punctuations and Numbers:

```
def removePunctuations(tweet):
    out = tweet.translate(None, r"!\"#$%&'()*+,-./:;<=>?-[\\]^_`{|}~")
    #removes all the punctuations except the "@"
    out = out.translate(None, '0123456789')
    #removes all the numbers
    tweet = out.split(" ")
    return ' '.join(tweet)
```

### Preparing the dataset:

```
df = pd.read_csv('Training_Dataset/2500_training_data_2.csv')
X,y = df['tweet'],df['label']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30)
#Means that training data is 70% of the dataset and the 30% is for testing

#The initial testing set is saved for a controlled testing environment
of the improved models later on
with open("Evaluating_Dataset/Test_Dataset.csv", 'w') as w:
    fieldnames = ['tweet','label']
```

```

writer = csv.DictWriter(w, fieldnames=fieldnames)
writer.writeheader()
for i,j in zip(X_test,y_test):
    writer.writerow({'tweet': i, 'label':j})

```

### Extracting TF-IDF:

```

# - Filters out terms that occur in only one document (min_df=2). Ngram_range(1,2) also
means that unigrams and bigrams are collected for the Bag of Words
vectorizer = TfidfVectorizer(use_idf=True, ngram_range=(1,2),min_df=2)
X = vectorizer.fit_transform(X_train)
train_tfidf_feature = X.toarray()

```

### Extracting LSA:

```

lsa = TruncatedSVD(n_components = 300, n_iter=500)
#the n_components parameter are number of concepts you want to collect
train_lsa_feature = lsa.fit_transform(X)

```

### Concatenating the Features:

```

train_final_representation = np.concatenate((train_tfidf_feature,train_lsa_feature),axis=1)

```

### Preparing the SVM Classifier:

```

class_weights = compute_class_weight('balanced', [0, 1], y)
class_weight_dictionary = {1:class_weights[0], 1:class_weights[1]}
clf3 = SGDClassifier(class_weight=class_weight_dictionary,loss="hinge",
                    penalty="l2", shuffle=True,)
clf3.partial_fit(train_final_representation,y_train,classes=[0,1])

```

### Testing the Classifier:

```

test_final_representation = np.concatenate((test_tfidf_feature,test_lsa_feature),axis=1)
accuracy3 = clf3.score(test_final_representation,y_test) *100
prediction3 = clf3.predict(test_final_representation)
cf3 = confusion_matrix(y_test,prediction3)
#for confusion matrix generation
print (classification_report(y_test, prediction3))
#for classification report computation (precision, recall, f-score)

```

### Save models:

```

with open('Models/classifier.pkl', 'wb') as f:
    pickle.dump(clf3, f)
with open('Models/vectorizer.pkl', 'wb') as f:
    pickle.dump(vectorizer, f)
with open('Models/lsa.pkl', 'wb') as f:
    pickle.dump(lsa, f)

```