Automatic Detection of Cyberbullying on Social Media

Love Engman

June 8, 2016
Master's Thesis in Computing Science, 30 credits
Supervisor at CS-UmU: Lars-Erik Janlert
Examiner: Henrik Björklund

Umeå University
Department of Computing Science
SE-901 87 UMEÅ
SWEDEN

Abstract

Bullying on social media is a dire problem for many youths, leading to severe health problems. In this thesis we describe the construction of a software prototype capable of automatically identifying bullying comments on the social media platform ASKfm using Natural Language Processing (NLP) and Machine Learning (ML) techniques. State of the art NLP and ML algorithms from previous research are studied and evaluated for the task of identifying bullying comments in a data set from ASKfm. The best performing classifier acts as the core component in the detection software prototype. The resulting prototype can monitor selected profiles on ASKfm in real time and display identified bullying comments connected to these profiles on a web page.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The ambition of this thesis is to explore the possibilities of developing a system that can automatically detect cyberbullying on social media. The motivation for this project was originally only the author's desire to learn about natural language processing and machine learning techniques by applying them to the real world problem that is cyberbullying. The idea was later presented to Dohi Agency [15], who encouraged the idea and offered resources and a place to work from as a part of their master thesis program. In short the project involves: give a background on cyberbullying, determine ways to identify bullying in text by analyzing previous research on the subject, experiment with different methods to find the most suitable way of identifying bullying comments and finally implement a demo system that can detect cyberbullying on social media using the resulting method.

1.1 Thesis Outline

The remainder of the thesis is structured as follows:

Chapter 2 - Problem Description
Introduces the problem and the goals of the project.

Chapter 3 - Cyberbullying
Gives an introduction to cyberbullying. The term is defined, the presence of cyberbullying on social media is presented as well as the effects it can have on victims.

Chapter 4 - Bullying Classification
Describes the problem of labeling documents as bullying or non bullying. Explains concepts when measuring the performance of a classifier. Previous research is presented including state of the art methods and algorithms.

Chapter 5 - Experiments
Presents the experiments performed in this study in order to find a suitable method for classifying bullying comments on social media.

Chapter 6 - Prototype Architecture
Discusses some of the challenges and ways to address these challenges when designing an automatic system for cyberbullying detection. Suggests a simple architecture for such a system.

Chapter 7 - Results - Prototype
Describes the implemented prototype.

Chapter 8 - Conclusions
The conclusions of the project are described and tied back to the goals. Short discussion of limitations and future work is included at the end.

Chapter 2

Problem Description

This chapter gives a short introduction to the problem, the desired goals with this thesis and the methods that will be used to reach the goals.

2.1 Problem Statement

Cyberbullying is bullying that takes place in cyberspace through various mediums including online chats, text messages and e-mails. It is a big problem on social media websites like Facebook and Twitter. Many individuals, especially adolescents, suffer negative effects such as depression, sleeplessness, lowered self-esteem and even lack of motivation to live when being targeted by bullies on social media. Much is being done to stop regular bullying in schools. Cyberbullying on the other hand can be difficult to detect and stop due to it happening online, often hidden from the eyes of parents and teachers. The problem we face is to come up with a technological approach that can aid in automatic detection of bullying on social media. The approach we will investigate is a system capable of automatically detecting and reporting instances of bullying on social media platforms.

2.2 Goal

The goal of this project is to generate knowledge of how an automatic system for detecting bullying on social media can be constructed.

2.3 Methods

To achieve the goal we will start by researching cyberbullying. defining the concept and to what extent it takes place on social media. Thereafter we will look at previous research on the subject of bullying detection, determining the state of the art algorithms and methods. With the knowledge from previous research and our own experiments we will come up with a suitable architecture for the system followed by implementing a prototype capable of detecting bullying on a single selected social media platform in real time.

Chapter 3

Cyberbullying

This chapter gives a background on cyberbullying. Starting by defining the term, followed by describing its presence on social media sites. We then look at the negative effects victims can suffer and conclude with a brief discussion of responsibilities.

3.1 Definition

Cyberbullying is a fairly broad term with many definitions. The Cyberbullying Research Center [4] defines cyberbullying as ”willful and repeated harm inflicted through the use of computers, cell phones, and other electronic devices”. They emphasize that the process must be willfull i.e. the action is intended to do harm, and repeated i.e. a single action is not considered bullying. A similar definition is given by NoBullying.com [30] ”the act of using the Internet, cell phones, video games or other technology gadgets to send, text or post images to hurt or embarrass another person”. From the definition we can see that many different actions can be considered as cyberbullying, here are a few examples: taking a private message and forwarding it or posting it publicly, spreading a rumour about someone online, sending an aggressive message to someone, posting an embarrassing picture of someone without permission. Exclusion can also be cyberbullying, for instance excluding someone from a private chat group.

3.2 On Social Media

Cyberbullying is taking place through many different mediums such as e-mails, online chat, text messages, games and social media platforms. Ditch the Label, a UK anti-bullying charity determined by a large survey 2013 [37] that 54 percent of young people on Facebook had experienced cyberbullying, 28 percent on Twitter and 26 percent on ASKfm. These three social media platforms were most common for cyberbullying in terms of percentage of users who had experienced cyberbullying while on the site. The survey asked over 10 000 young people aged 13-22 about cyberbullying.

A similar survey [18] conducted 2015 by the Swedish non-profit organization Friends showed that one third of children aged 10-16 had been exposed to indignities on social media. The most common platforms where children experienced cyberbullying were Kik, Instagram, Facebook and ASKfm. The survey also showed that eight out of ten children knew the offender, typically someone from school.

3.3 Effects of Cyberbullying

Victims of cyberbullying can suffer many negative effects such as not being accepted in their peer groups (leading to loneliness and isolation), low self-esteem, depression, poor academic achievement and decreased emotional well-being [8]. Cyberbullied individuals are also more likely to suffer from headaches, abdominal pain and sleeplessness [34]. In some extreme cases cyberbullying has also been linked to suicide, such as the Megan Meier incident [28]. A 13 year old girl who took her own life after being bullied on the social media platform Myspace.

3.4 Responsibility

When a case of cyberbullying (and regular bullying) is detected one of the recommended actions is to inform a responsible adult, typically a parent or teacher who can investigate and alert the proper authorities [9, 35, 38]. In Sweden, teachers and other adults employed at schools carry a great responsibility when it comes to taking action against bullying. More precisely the organisation Friends describe it as "If a member of staff is made aware that a pupil/student in the school feels that they have been bullied or harassed, the staff member has what is called the obligation to act. The obligation to act applies to all staff members at the school, irrespective of their role, and entails that the school must investigate, deal with and prevent the acts of intimidation at the earliest stage possible" [19].

Social media platforms are not held directly responsible for actions performed by their users. However many social media platforms provide methods for users to report behaviour that violates the terms of service agreement of the platform. Employees working for the platform can then follow up and delete inappropriate content or remove the abusing user. Twitter and Facebook for example, both offer this kind of service [39, 16].

3.5 Summary

Cyberbullying is a significant problem with several dire effects on victims. It takes place on social media networks such as Facebook, Twitter and ASKfm. One recommended action when cyberbullying is detected is to report it to a trusted adult. Cyberbullying can be performed in many different ways, for the remainder of this thesis we will be focusing on the act of sending a hurtful message in text on social media.

Chapter 4

Bullying Classification

This chapter introduces the problem of taking pieces of text, often referred to as documents, and classifying them as either bullying or non-bullying. For our purposes the documents will be messages on social media. We start by explaining a few important concepts to measure performance of a classifier. We then take a look at previous work on the subject to determine the state of the art algorithms and methods. Finally we describe in more detail the algorithms and methods that will be experimented with to find a suitable method to employ in the prototype.

4.1 Precision, Recall and F-Measure

Precision and recall are important concepts when reasoning about the performance of a classifier. They are used as a measure to determine how well a classifier performed. Let us illustrate with an example. Suppose we have a binary classifier capable of inspecting an avocado and classifying it as either ripe or non-ripe. Now imagine that we have 100 avocados and for every one of them we know if it is ripe or not, totaling 30 ripe and 70 non-ripe avocados. We then present the avocados one at a time to the classifier and write down the results. Suppose the classifier marked 35 avocados as ripe, 25 correctly (true positives) and 10 incorrectly (false positives). The classifier missed 5 avocados (false negatives). Precision is then the number of correctly classified ripe avocados (25) divided by the total number of positive classifications (35). Or more generally:

$$P = \frac{tp}{tp + fp}$$

where tp are true positives, and fp false positives.

For the avocado example the precision P is then:

$$P = \frac{25}{25 + 10} \ \approx 0.71$$

Recall can be explained as the number of avocados correctly classified as ripe (25) divided by the total number of ripe avocados (30). More generally:

$$R = \frac{tp}{tp + fn}$$

where tp are true positives, and fn false negatives.

For the avocado example the recall R is then:

$$R = \frac{25}{25+5} \approx 0.83$$

Having a high recall is good because it means that we did not miss any ripe avocados, however to get a recall of 100 percent we could simply label all avocados as ripe and we would not miss any, but doing so will cause our precision value to drop significantly. We realize that we may have to prioritize high precision or high recall depending on our context. For instance if we are classifying mushrooms as edible (positives) or poisonous (negatives) we want to ensure we do not classify poisonous mushrooms as edible i.e. we want to prioritize precision over recall even if it means that we throw away some edible mushrooms. On the other hand let us consider a classifier that inspects a letter or package and classifying it as a bomb letter (positive) or a safe letter (negative). In this case we want to maximize recall to make sure we intercept all bomb letters even if it means we also intercept some safe letters. To more easily measure this trade-off it can be a good idea to have a single measure factoring in both precision and recall, such as the F-measure. F-measure is the weighted harmonic mean of precision and recall and is written as:

$$F = \frac{(\beta^2+1)PR}{\beta^2 P + R}$$

where the weight $\beta \in [0, \infty]$

Using $\beta < 1$ values precision more while $\beta > 1$ values recall, $\beta = 1$ means that we put equal weight on precision and recall and is often written as F_1 which simplifies to:

$$F_1 = \frac{2PR}{P+R}$$

Calculating F_1 for the avocado example gives us:

$$F_1 = \frac{2 \times 0.71 \times 0.83}{0.71 + 0.83} \approx 0.77$$

Another commonly used F-measure is the F_2-measure meaning that $\beta = 2$, and that recall is valued higher than precision.

4.2 Previous Work

Yin et al. 2009 [41] showed that a supervised learning approach could be used to classify harassment. They used test data from three sources: Kongregate, Slashdot and Myspace. A Support Vector Machine (SVM), a supervised learning method learning from examples to classify new data into one of two categories, trained on a Term Frequency - Inverse Document Frequency (TFIDF) model coupled with contextual and sentiment features achieved up to 40 percent precision, 60 percent recall and 45 percent F-measure. TFIDF is a way of calculating how important every word is in a document.

Dinakar et al. 2011 [13] determined that it was possible to get better results by first labeling bullying into categories and then using a binary classifier for every category. They used sexuality, race/culture and intelligence as categories. A decision tree using JRip (an implementation of the propositional rule learning algorithm RIPPER) achieved the best accuracy while an SVM using the Sequential Minimal Optimization (SMO) algorithm for training was the most reliable method. The test data consisted of comments on Youtube videos.

In 2011 Reynolds et al. [32] used a data set from Formspring.me and rated the documents based on occurrences of bad words. They then used a machine learning tool to train a few different classifiers including decision trees and a support vector machine. A decision tree using the C4.5 algorithm performed best and reached a recall level of 78.5 percent.

To further improve classification, gender information can be used as shown by Dadvar et al. 2012 [10]. The study takes advantage of the fact that males and females use different types of vocabularies. More specifically there is a difference in which curse words are typically used. They then trained two separate SVM classifiers for the male and female test cases. The results showed an increase from 31 to 43 percent in precision, from 15 to 16 percent in recall and from 20 to 23 percent in F-measure.

A different approach was suggested in 2012 by Chen et al. [5]. They proposed a Lexical Syntactic Feature (LSF) which determines the offensiveness of a sentence based on the offensiveness of the words and the context. The offensiveness of words were measured from two lexicons. To get the context they parsed the sentence grammatically to identify dependencies between words. When a bad word could be grammatically related to a user name or another bad word the offensiveness of the sentence was adjusted. The LSF method achieved 98.24 percent precision and 94.34 percent recall in detecting offensive sentences in a data set of Youtube comments. Offensive sentences were defined as sentences containing vulgar, pornographic or hateful language. A similar approach was used to predict the offensiveness of a user.

Chen et al. [6] took things a step further in 2012 by implementing a demo system capable of detecting cyberbullying in real time on twitter. The idea behind the system was to let teachers and parents monitor the tweets of children. This was done by adding the childrens' profiles to a filter, it was also possible to add profiles based on location. The system used a Gradient Tree Boosting model for learning to differentiate between regular tweets and bullying tweets. Parents and teachers could also classify tweets manually to further train the system. The research does not reveal any results of how well the system performed and the project seems to have been discontinued.

Dadvar et al. produced another paper in 2013 [11] where they used content-based, cyberbullying-specific and user-based features to improve classification results. Using activity history of users, they trained an SVM to classify bullying in youtube comments. The results were 77 percent precision, 55 percent recall and 64 percent F-measure.

Experimenting on the same data set as Yin et al. in 2009, Nahar et al. 2013 [27] used a weighted TFIDF model and trained an SVM to classify posts as bullying or non-bullying. The experiment showed improved results with values up to 87 percent precision, 99 percent recall and 92 percent in F-measure. They also proposed a social networking graph showing the most active bullies and victims.

Huang et al. [22] determined in 2014 that considering social relationships between users could improve results for classification. The experiment used regular textual analysis combined with social network features to classify bullying in a data set from Twitter.

In 2016 Zhao et al. [42] used a set of features they eventually named EBoW, consisting of a bag of words model combined with Latent Semantic Analysis (an NLP technique to find meaning of words) and word embeddings by calculating word vectors. They then trained an SVM using these features to classify a biased data set from Twitter. The data set only consisted of tweets containing one of the keywords ”bully”, ”bullying” or ”bullied”.

We can see that there has been significant progress over these few years. Two different methods appear to have achieved the best results when looking at previous research. The LSF method and the weighted TFIDF features fed to an SVM both achieved remarkable levels of precision and recall. However the results achieved by different researchers are mostly not comparable due to experimenting on different data sets.

4.3 Classification techniques and algorithms

In this section we describe in more detail Natural Language Processing techniques and algorithms as well as Machine Learning methods that were used successfully in previous research. These methods will then be used to experiment on a data set collected from ASKfm.

4.3.1 Term Frequency - Inverse Document Frequency

Term Frequency - Inverse Document Frequency (TFIDF) is a measure of the importance of a term in a document. It is often used in Information Retrieval systems to determine which documents are relevant when searching in a collection. Let us start by explaining Term Frequency with an example. Imagine that we have a collection of six documents. These documents are the scripts of the first six Star Wars films (Episodes I-VI) containing all the spoken lines in the films. We define the Term Frequency to be the number of times a term occurs in a document. Let us say that the word ”Anakin” occurs 27 times in episode I. Then the Term Frequency (TF) is:

$$TF_{t,d} = 27$$

where the term t = Anakin and the document d = Episode I

Sometimes it can be a good idea to normalize the term frequency by dividing it with the length of the document. This is done to avoid bias towards long documents. It is likely that a longer document has more occurrences of a particular term however it does not necessarily mean that such a document is more relevant than a shorter document.

The Document Frequency (DF) for a term is defined as the number of documents in the collection that contains the term. Let us say that the term Anakin occurs in episodes I,II,III and V. The DF for t = Anakin is then:

$$DF_t = 4$$

Now let us look at the Inverse Document Frequency. It is defined as:

$$IDF_t = \log \frac{N}{DF_t}$$

where N is the total number of documents in the collection and t is the term.

Calculating IDF for the term t = Anakin we get:

$$IDF_t = \log \frac{6}{4} \approx 0.18$$

The benefit of IDF is that we get a small number for terms that occur in many documents and a larger number for very rare terms that occur in only a few documents. We can use this to determine which documents are relevant when searching for a term or in our example which Star Wars episodes to watch if we only want to watch episodes with Anakin. Another feature of IDF is that terms that occur in every document will have a value of 0. The definition for TFIDF is now simply:

$$TFIDF_{t,d} = TF_{t,d} \times IDF_t$$

Suppose we have observed the TF values for the word Anakin displayed in Table 4.1 for the different Star Wars films. If we then calculate the TFIDF for the third and fifth films for t = Anakin, we get:

$$TFIDF_{t,3} = 33 \times \log \frac{6}{4} \approx 5.81$$

$$TFIDF_{t,5} = 2 \times \log \frac{6}{4} \approx 0.35$$

Thus a search for the word ”Anakin” in the collection of these six documents would rank episode III as more relevant than episode V.

Term	I	II	III	IV	V	VI
Anakin	27	19	33	0	2	0
Vader	0	0	15	10	22	19
Padmé	12	15	26	0	0	0
Force	23	12	19	7	16	9

Table 4.1: Fabricated Term Frequency values for the terms Anakin, Vader, Padmé and force in the six Star Wars episodes

Another important feature of TFIDF is the possibility to represent a document as a vector of TFIDF values. For instance if we have a dictionary consisting of all the terms used in the Star Wars films and that the four first elements of the document vector represents Anakin, Vader, Padmé and force. The document vector of episode III would be [5.81, 2.64, 7.83, 0, ...] and for episode V it would be [0.35, 3.87, 0, 0, ...]. Notice that the common term ”force” that occurs in every episode is represented as 0. Using these vector representations of the documents we can more easily make comparisons and rankings of the documents in the collection.

4.3.2 The Stanford Parser

The Stanford Parser by the Stanford Natural Language Processing Group [20] is a parser that works on sentences to identify grammatical dependencies between words. For example, feeding the sentence ”My dog likes cucumbers” to the parser returns the following list of dependencies:

– nmod:poss(dog-2, My-1)

– nsubj(likes-3, dog-2)

– root(ROOT-0, likes-3)

– dobj(likes-3, cucumbers-4)

The nmod:poss is a possession modifier meaning that there is a possessive dependency between the governor "dog" and the dependent "My". A nominal subject dependency is expressed as nsubj and normally involves a noun and a verb. The dobj dependency means direct object and refers to the noun that is being acted on by the verb. The parser uses a tree structure to represent the sentence and the root dependency refers to the root of the parse tree. These are just a few of the dependency types that the parser can identify.

4.3.3 Supervised Learning

The most common way to train a machine learning classifier is to use supervised learning. The idea is to feed training data along with correct answers to the learning algorithm in order for it to learn the desired function. This requires access to an annotated training data set where the correct classification is annotated with every example. When the training is complete the model can classify new data which has not previously been annotated. The training is typically complete when the algorithm determines that it has learned a function that best models the training data. That function can then be applied to new data to classify it. There are many different types of supervised machine learning methods such as Naive Bayes, Artificial Neural Networks and Support Vector Machines.

4.3.4 Support Vector Machine

Manning et al. [25] describes a Support Vector Machine (SVM) as "a vector space based machine learning method where the goal is to find a decision boundary between two classes that is maximally far from any point in the training data (possibly discounting some points as outliers or noise)." We will investigate more precisely what that entails in this section by describing a few different examples based on the SVM material by Burges [3].

Basically an SVM can use training data to learn a classifying function with which it can classify new data, that it has previously not seen, into one of two categories. A training case consists of a vector and the correct classification of that vector. When the SVM is being trained it attempts to find a line or hyperplane that best separates the positive cases from the negative cases. New cases can then be classified by determining which side of the hyperplane they fall on. Wikipedia describes a hyperplane as "a subspace of one dimension less than its ambient space" [40]. For example a hyperplane in 3-dimensional space is a plane (two dimensions), while in a 2-dimensional space a hyperplane is a line (one dimension).

To illustrate how the SVM works we will look at a very simple example. Given the training input in Table 4.2 consisting of ten vectors with a dimension of two and the correct classification of these as either -1 or 1. By plotting these vectors as done in Figure 4.1 we can see that they can easily be separated by a line so that positive examples are on one side and negative on the other.

Vector	x	y	Class
V1	1.33	0.59	1
V2	1.50	0.51	1
V3	0.24	0.95	1
V4	-0.31	1.15	1
V5	2.17	0.11	1
V6	2.32	1.51	-1
V7	2.46	2.56	-1
V8	3.19	3.49	-1
V9	1.58	1.90	-1
V10	2.17	4.66	-1

Table 4.2: Example of training input for a Support Vector Machine.

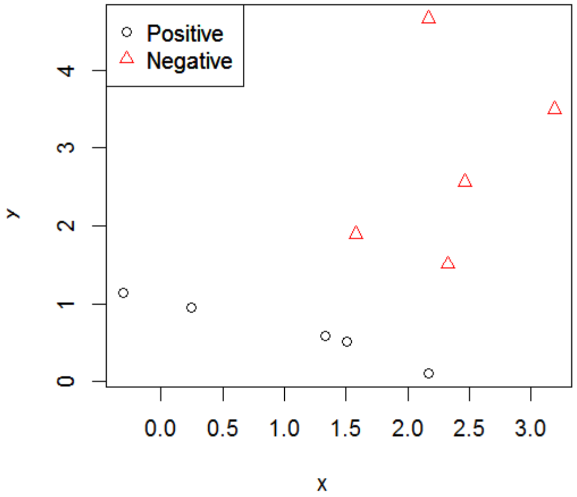


Figure 4.1: The example vectors V1-V10 as input for a Support Vector Machine plotted in 2D-space.

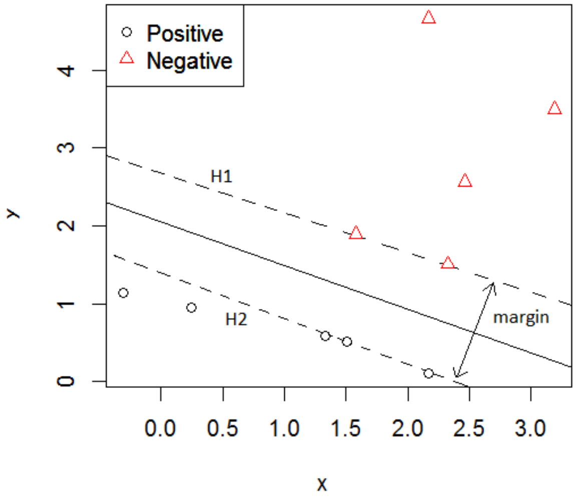


Figure 4.2: Training points for a Support Vector Machine that has been separated with a hyperplane that maximizes the margin.

The SVM algorithm tries to find the line separating the two classes so that the margin between the closest positive example and the closest negative example is as large as possible. In Figure 4.2 we can see how that could look like. Training data points that lie on the hyperplanes H1 or H2, making up the margin, are called support vectors. Removing a support vector would alter the dividing hyperplane, while removing a random other data point would not.

We have now covered the case where the training points could be perfectly separated by a hyperplane. However that is not always possible, especially when dealing with noisy data. Figure 4.3 is an example where a linear separation is not possible. The way an SVM handles this is to disregard certain data points for a cost. There is a trade off between how many errors are allowed and the width of the margin. This trade off can be controlled by the user by changing the parameter **C** for the SVM. A larger value for **C** increases the penalty for each data point inside or on the wrong side of the margin, consequently the SVM will make the margin smaller and keep more data points correctly classified. On the other hand a smaller value for **C** will make the SVM look for a wider margin even if that means more training data points are classified incorrectly. While having a large **C** will decrease the number of training points inside or on the wrong side of the margin, it can also cause overfitting, meaning that the classifier can not generalize well and is too closely tailored to the training data which can include noise. In Figure 4.4 we can see how two data points were marked as errors in order to find a separating hyperplane. It would have been enough to only disregard one data point but that would have significantly decreased the width of the margin.

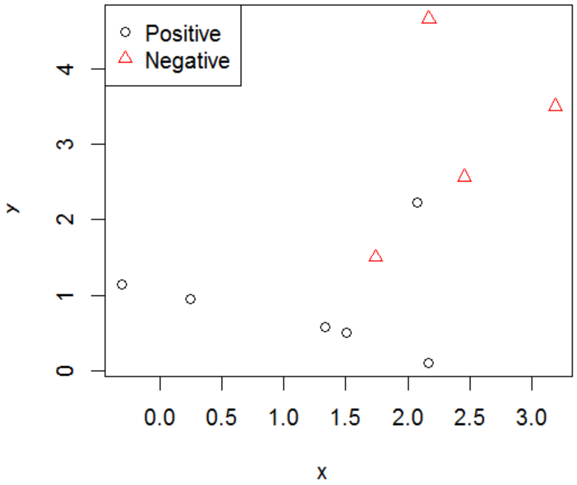


Figure 4.3: Training data points for a Support Vector Machine where a linear separation would cause incorrect classification of training data.

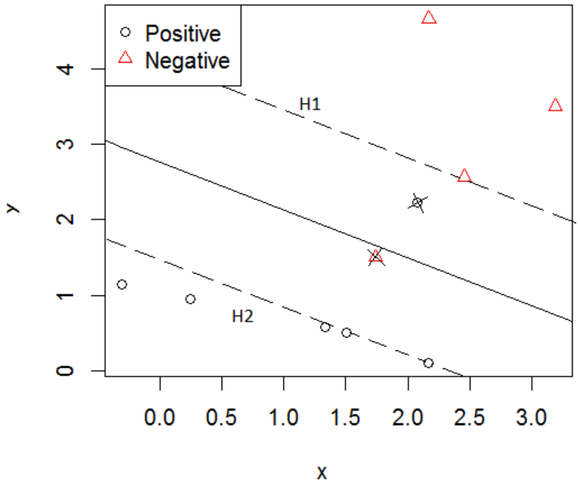


Figure 4.4: Training data points for a Support Vector Machine where the two data points crossed out have been disregarded in order to find a separating hyperplane with a wide margin.

Sometimes there is just no way of making a linear separation that makes sense. For instance in Figure 4.5, even though the data points are clustered nicely, there is no way to divide the test data with a straight line to produce a good classifier. To overcome this problem an SVM can map the training data to a higher dimension where a linear separation is possible.

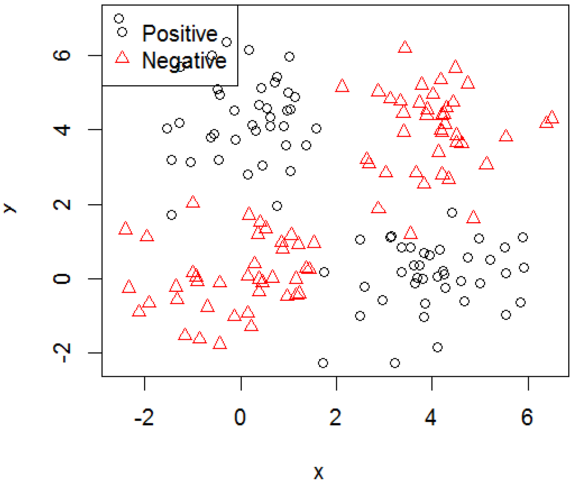


Figure 4.5: Training data points for a Support Vector Machine where a linear separation would not work.

In these examples we have worked exclusively with two dimensional vectors because they are easy to reason about. It is however important to know that SVMs can work with vectors of any dimension. In addition it is good to know that normalizing input vectors before they are fed to an SVM is a good idea because it generally improves classification performance [21].

4.3.5 Word2vec

Another interesting technology used briefly to aid in bullying classification is Word2vec. It has been used to increase the size of dictionaries (required for feature extraction) by finding similar words. Word2vec was proposed in 2013 by Mikolov et al. [26] and is a method for learning the meaning of words by representing them in vector space. Word2vec uses a two layered neural network to learn vector representations of words. It looks at neighbouring words to determine which words are similar. For example if the word ”pizza” occurs next to the word ”eat” very often and the word ”hamburger” also occurs next to the word ”eat”, the algorithm can learn that there may be some similarity between pizza and hamburger. Word2vec takes a text corpus (preferably a very large corpus) as input and the result is a dictionary of words and their vector representations. The dictionary can then be queried, for instance listing the three closest words in vector space to the term ”music” may return ”songs”, ”rock”, and ”classical”. Word2vec works better when trained on a very large data set, up to billions of words, however training on very large data sets can take a long time.

Chapter 5

Experiments

This chapter describes experiments performed in this study to determine a suitable method for identifying bullying comments on a social media platform.

5.1 Evaluation

The results of the classifiers are reported in accuracy, precision, recall, F_1-measure and F_2-measure where we focus on F_2-measure as the main metric for evaluating and comparing methods. The reason is that while precision and recall are both important we want to have an emphasis on recall, with the motivation that it is better to classify innocent comments as bullying than to classify bullying comments as innocent.

5.2 Corpus from ASKfm

The data set to experiment on was acquired from the social media platform ASKfm. It is a platform where users ask questions publicly on other users' pages. Questions can also be asked anonymously. ASKfm has on several occasions been critiqued for being unsafe for children due to cyberbullying [36, 29, 7, 12]. To get comments from various users a simple program was implemented that connects to the start page of ASKfm. The site presents a few sample user profiles on the start page, these user names were read by the program. In order to get English speaking sample profiles the connection was done through various public proxy servers located in the UK and the US. The program then read a fixed number of questions and answers posted on these users' profile pages. To get additional profiles the user names of questioners were saved to be scanned later. This pattern was done recursively until about two thousand comments were gathered. The proxy server was then changed and the program restarted in order to get a more diverse user group. All user names were stored on disk to make sure that the same user was never scanned more than once. Nakatani's Java implementation for language detection [33] was used to ensure that the comments downloaded were written in English. The comments were also cleaned from strange characters, links and user names. Links were replaced with "@link" and user names with "@user". In addition all letters were made lower-case.

In total about 24 000 questions and answers were gathered from the site. These were manually classified by the researcher as either bullying or non-bullying. The comments, both questions and answers, were inspected one by one without context and labeled as positive if they could be interpreted as cyberbullying based on the previously established definition (a message intended to inflict harm). Only 278 comments were labeled as bullying while 23684 were labeled as not bullying.

The entire data set was then split into a training set and a test set. 70 percent of both the negative and positive examples were used as a training set while the remaining 30 percent were used as the test set. This gives a test set of 7190 comments where about one percent are bullying comments. This is important because it resembles what we expect to see on social media and will give us an idea of how the classifier would perform if deployed on a social media platform.

5.3 Experiment I - LSF

The Lexical Syntactic Feature approach proposed by Chen et al. [5] was used to successfully detect offensive sentences in a data set of Youtube comments. This approach uses the Stanford Parser [20] to identify dependencies between words in a sentence. Three dictionaries are used. The first dictionary contains words that are labeled as strongly offensive. The second dictionary contains words labeled as weakly offensive. Examples of strongly offensive words are "whore" and "jerkoff" while examples of weakly offensive words are "stupid" and "idiot". The third dictionary contains personal pronouns like "you", "he", "she". The dictionary of bad words was constructed from words found at various websites listing bad words. See Appendix A for the exact dictionaries. Using these dictionaries the offensiveness of a word in a sentence is then defined as:

$$O_w = \begin{cases} a_1 & \text{if w is a strongly offensive word} \\ a_2 & \text{if w is a weakly offensive word} \end{cases}$$

where $1 \geq a_1 \geq a_2$

The offensiveness is increased when an offensive word is grammatically related to a personal pronoun or user name. Only the dependency relations listed in Table 5.1 are considered. The offensiveness of the word is multiplied by an intensifier I_w. For the word w in sentence s, all dependency sets D including w labeled as $D_{w,s} = \{d_1, ..., d_k\}$ contribute to the intensifier. For each $d_j (1 \leq j \leq k)$:

$$d_j = \begin{cases} b_1 & \text{if } d_j \text{ is a pronoun or user} \\ b_2 & \text{if } d_j \text{ is an offensive word} \end{cases}$$

where $b_1 \geq b_2 \geq 1$ meaning that an offensive word related to a user is more offensive than two related offensive words e.g. "f***ing @user" is more offensive than "f***ing stupid". The idea here is that the sentence "school is stupid" should not be classified as offensive but "you are stupid" should be. The intensifier for a word I_w is then calculated as $I_w = \sum_1^k d$. Finally the offensiveness value of the entire sentence is:

$$O_s = \sum O_w I_w$$

The values used for the constants were $a_1 = 1.0$, $a_2 = 0.5$, $b_1 = 2.0$, $b_2 = 1.5$. A sentence is classified as offensive, or in this experiment as bullying if $O_s \geq 1.0$. These were the same values as the ones used by Chen et al.

Dependency Type	Meaning
abbrev	abbreviation modifier
acomp	adjectival complement
amod	appositional modifier
appos	noun compound modifier
nn	noun compound modifier
partmod	participial modifier
dobj	direct object
iobj	indirect object
nsubj	nominal subject
nsubjpass	passive nominal subject
xsubj	controlling subject
agent	passive verb's subject
conj	conjunct
parataxis	parataxis
poss	holds between the user and its possessive determiner
rcmod	relative clause modifier

Table 5.1: Dependency types found by the Stanford Parser that are relevant for offensive sentence detection.

5.3.1 Results

The algorithm was applied to the test set described earlier. The results measured in precision, recall, F_1-measure and F_2-measure can be seen in Table 5.2. LSF acheived 36 percent precision and 45 percent recall. Table 5.3 lists accuracy, false positives and false negatives. LSF successfully identified 38 out of the 84 bullying comments in the test set while 67 innocent comments were classified as bullying. The total accuracy i.e. the percentage of correctly classified comments were 98 percent.

Method	Accuracy	Precision	Recall	F_1-measure	F_2-measure
LSF	98.43%	36.19%	45.24%	40.21%	43.08%

Table 5.2: Various performance measures for classifying the ASKfm test data set using the LSF method.

Method	Correct	Incorrect	tp	fp	tn	fn
LSF	7071	119	38	67	7039	46

Table 5.3: True positives (tp), false positives (fp), true negatives (tn) and false negatives (fn) for cyberbullying classification using the LSF method.

5.3.2 Discussion and Conclusion

LSF did not perform that well in bullying classification on the data set from ASKfm. The main reason for false negatives is that the sentences written on ASKfm are seldom grammatically correct. They very often contain spelling errors and grammatical errors. This means trouble for the Stanford Parser causing it to not find the grammatical dependencies we are looking for. For example the comment "fuck you i corrected myself", clearly offensive, but not classified as offensive by LSF. However changing the letter "i" to a capital "I" makes the parser recognize the relation dobj(fuck-1, you-2) and labeling it as offensive. Such a minor grammatical error had a devastating effect on the classification. This means that any comment that should be processed by LSF must be corrected with regards to spelling and grammar to make this approach effective. Chen et al. did use a spelling correction algorithm to deal with this issue in their experiments. However in our case adding a spell checker did not improve the results. While the spell checker corrected some misspelled words it also changed commonly used abbreviations such as "lol" and "wtf".

The conclusion of this experiment is that LSF alone is not suitable for detection of bullying comments on the ASKfm data. It may however work well on a different platform where users write more formal and grammatically correct English.

5.4 Experiment II - Feature Selection + SVM

In this experiment we classify bullying comments by extracting features and feeding them to a Support Vector Machine for learning.

5.4.1 Feature Set A

As determined in previous research TFIDF has outperformed other simple features such as N-Grams when used in conjunction with a machine learning tool. Therefore we will use TFIDF as a base feature set. We calculate the inverse document frequency based on the entire data set (training set + test set). The term frequency values are normalized by dividing with the length of the document.

5.4.2 Feature Set B

For our second set of features we extend the base model by scaling the TFIDF values for bad words. All features for bad words are scaled by a factor of two as done by Nahar et al. [27]. The reason for this is that bullying comments very often contain bad words and scaling these features can make it easier to find a good separation in vector space for the SVM.

Bad words are determined from a dictionary. The dictionary consists of two parts. The first is a static dictionary constructed by combining bad words from different web pages listing bad words (See Appendix A). The second part is generated using Google's word2vec. Over one million additional comments were collected from ASKfm using the same program as before. The word2vec implementation from Deeplearning4j [14] was then used to learn the word representations. A bad word was labeled as such if at least three (decided after inspecting several words and their neighbours) of its closest ten neighbours in vector space were in the static dictionary of bad words. The idea of this approach is that some bad words not previously in the static bad words dictionary will be identified by word2vec and essentially increasing the number of bad words in total, which ultimately could lead to higher recall rates.

5.4.3 Feature Set C

As done in previous research counting the number of bad words and pronouns as separate features has showed improved classification. This is due to bullying comments often involving pronouns or user names in addition to bad words. Thus we count the number of bad words as one feature, the number of second person pronouns as another (”you”, ”u”, ”your”, ”@user”) and third person pronouns as a third feature (”he”, ”she”). Words are counted as bad if they are in the static or dynamic dictionary of bad words. Two static dictionaries for pronouns were used (see Appendix A). Additionally while inspecting some comments we realize that on many occasions comments with bad words often also include positive words, for instance ”I love you so fucking much. So thankful I can call you my best friend”. This sentence is a good example of a comment involving a bad word and second person pronoun but is still a positive comment. Therefore we also use the count of positive words in the document as a feature. A dictionary with positive words was constructed based on a list made by Mind Map Inspiration [17]. See Appendix A for the final positive words dictionary.

5.4.4 Feature Set D

Finally for feature set D we experiment with Stanford Dependencies as features. We treat all relations where a bad word is related to another word as the governor or the dependent as separate features. This means that we get three features for every relation: rel(badword, word), rel(word, badword), rel(badword, badword). In addition all dependencies where a bad word is related to a personal pronoun is also added as a feature i.e. rel(badword, pronoun) and rel(pronoun, badword). This gives us a total of 80 features (for this feature set) when using the 16 relations in Table 5.1. The intent for these features is to increase the potential to separate non bullying comments including bad words and user identifiers from actual bullying comments. In other words to combat the weakness of TFIDF having the same representation for ”John is faster than Alice” and ”Alice is faster than John” even though the semantics of the sentences are completely different.

5.4.5 Training the SVM

Joachims’ implementation SVM$^{\text{light}}$ [23] was used to train an SVM and classify the examples. The SVM was trained with a linear kernel on the training data. Since the training data is imbalanced, containing mostly negative examples we have to adjust the cost-factor parameter **J**, a factor representing how much the cost of an error on a positive example should outweigh an error on a negative example. We also need to tune the **C** parameter. Tuning the parameters was done with a parameter sweep where **C** assumed the values [0.002, 0.02, 0.2, 2, 20, 200] and J assumed the values [10, 30, 100]. The parameter setup achieving the highest F_2-measure was recorded and used for evaluation. Before feeding the features to the SVM all features were normalized with the length of the feature vector (L_2 norm). After training the models, they were applied to the ASKfm test set.

5.4.6 Results

The results when classifying the ASKfm test data are displayed in Table 5.4. The table shows various measures for different feature sets and combinations of feature sets. All trained models achieved similar results in terms of accuracy, around 96-98 percent. Precision varies between 24-33 percent while recall varies between 61-75 percent. The highest F_2-measure, 55.89 percent, was achieved when combining features B and C i.e. weighted TFIDF combined with frequencies of bad words, positive words and pronouns. Table 5.5 shows true positive, false positive and false negative counts for the different feature sets.

Feature Sets	Accuracy	Precision	Recall	F_1-measure	**F_2-measure**
A	97.33%	24.53%	61.90%	35.14%	47.44%
B	96.88%	23.68%	75.00%	36.00%	52.32%
B+C	98.09%	33.94%	66.67%	44.98%	55.89%
B+C+D	97.62%	28.78%	70.24%	40.83%	54.53%

Table 5.4: Various performance measures for classifying the ASKfm test data set with a Support Vector Machine, using different feature sets.

Feature Sets	Correct	Incorrect	tp	fp	tn	fn
A	6998	192	52	160	6946	32
B	6966	224	63	203	6903	21
B+C	7053	137	56	109	6997	28
B+C+D	7019	171	59	146	6960	25

Table 5.5: True positives (tp), false positives (fp), true negatives (tn) and false negatives (fn) for classifying the ASKfm test data set with a Support Vector Machine, using different feature sets.

5.4.7 Discussion and Conclusion

Firstly we can note that using only the baseline feature set A, that is plain TFIDF, outperformed the LSF method in F_2-measure. Scaling TFIDF for bad words (feature set B) increased F_2-measure by about 5 percentage points. Adding counts of bad words, positive words and pronouns further increased F_2-measure. The results for features sets B + C are not as good as the ones presented by Nahar et al. [27] on whose research these features were based. The reason for this is likely due to the different data sets used. Upon inspection of the classified comments it appears that the reason for most false negatives (bullying comments classified as innocent) is that they do not involve bad words. For instance the comment ”why do you always post on insta omg im sick of seeing your face” was not classified as bullying when using feature sets B + C. The reason for false positives is that many comments involve bad words but are not bullying. One such comment is ”not really but i recently learned its fat tuesday thanks @user lol party time lol” which contains the bad word ”fat” and a user identifier ”@user”. This clearly demonstrates the weakness of TFIDF, that the order and relations of words are ignored causing loss in precision. The experiment to add Stanford Dependencies (feature set D) was an attempt to include features that would make it possible to combat this problem of TFIDF. However as can be seen in Table 5.4 it barely had any effect on the F_2-measure. The conclusion of the experiments is that out of the methods that we have evaluated, an SVM trained on the extracted feature sets B + C is best suited for cyberbullying detection on ASKfm.

5.5 Future Work

The state of the art in cyberbullying classification involves training a machine learning model using supervised learning. The research conducted is mostly focusing on feature engineering, i.e. finding features that can separate bullying comments from non bullying comments. Finding good features is difficult and problematic. Features that work well for Youtube comments may not work well for comments on ASKfm, due to different social media platforms being likely to have varying vocabulary and expressions in part caused by restrictions on communication (e.g. Twitter 140 character cap), different age groups and users' interests.

With the available methods it is fairly simple to achieve high recall rates by using large dictionaries of bad words and word vectors to determine offensive language. However achieving high precision by being able to separate comments containing bad language from actual cyberbullying comments is difficult with current methods. That particular challenge is an interesting topic for future research. Another interesting topic would be to investigate the possibilities of using Paragraph Vectors [24] to aid in bullying classification. For instance given a set of known bullying comments it could be possible to find comments with similar semantic meaning. This could be used in classification or simply to construct much larger data sets with positive data, which currently is a very tedious task. Furthermore Google just recently open sourced SyntaxNet, a syntactic language parser similar to the Stanford Parser [31]. They claim it to be the most accurate parser available. It would be interesting to see if it performs better than the Stanford Parser for the bullying classification problem.

Chapter 6

Prototype Architecture

This chapter aims to present some of the challenges for an automatic cyberbullying detection system on social media and then propose an architecture that addresses these challenges.

6.1 Target Audience

There are two main audiences for an automatic cyberbullying detection system. The first is social media sites and their administrators. The second audience is parents, teachers and other responsible adults. Social media sites would like to keep their platforms free from cyberbullying for the sake of safety of their users as well as good publicity. Parents and responsible adults would like to ensure the safety and well being of their minors. With this in mind the approach implemented by Chen et al. [6] seems reasonable. Their approach involved a web page where parents and teachers could specify Twitter user profiles to monitor. When a bullying comment linked to a monitored profile was detected the adult was made aware. This approach is also strengthened by the fact that one of the top recommended actions to take when being cyberbullied is to report the incident to a trusted adult and the responsibility to rectify the situation would then fall upon the adult. A system that monitors selected user profiles can be applied for both major audiences. Parents and teachers can monitor their children and students while social media sites can monitor all users on their platform. For these reasons this is the approach we will consider when reasoning about the architecture of the system.

6.2 Challenges

There are a few challenges that we would like to address in the architecture of an automatic system for cyberbullying detection. The first challenge is that there are hundreds of different social media platforms, and the most popular platforms vary over time. Different platforms have different communication concepts. This is problematic for two reasons. The first is that if communication style varies, identifying cyberbullying can be dissimilar on different platforms. The second problem is that we have to be able to monitor several different platforms and easily adapt to new platforms.

Another challenge is to detect bullying in different languages. To do that any machine learning classifier approach would have to be retrained with data from that particular language. This means that the classifier in the architecture must be relatively easy to retrain or replace. This has the added benefit that if a new perhaps better machine learning technique or a new way to better classify bullying comments is discovered it can easily be implemented in the system. This idea goes for all the parts of the system.

A difficult challenge is to make the system scalable. It is however necessary if the system should be usable by our two main audiences. Monitoring every user on a social media site will take a lot of resources depending on the scope of the platform. Therefore every part of the system must be scalable to some degree.

The common theme of these challenges is that all parts of the system need to be modularized. Meaning that they need to be decoupled from the rest of the system and be able to function independently. Which is exactly what our suggested architecture focuses on.

6.3 Architecture Overview

The architecture for the system is made up of five different modules that all have a single responsibility. A central coordinator utilizes the different modules to form a complete system. In Figure 6.1 the architecture overview is displayed, showing the different modules and their connections.

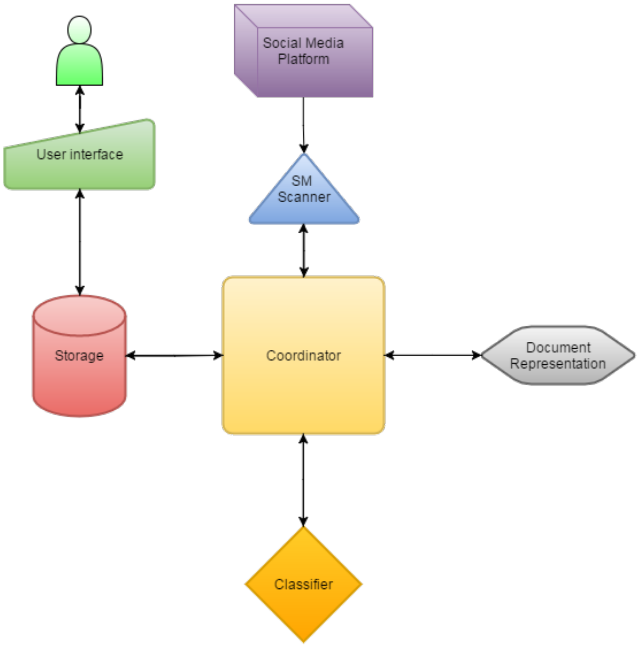


Figure 6.1: An overview of the suggested architecture for an automatic cyberbullying detection system.

6.4 Modules

This section describes the modules that make up the architecture, what the purpose of each module is and their respective input and output. The idea is that every module should work as its own independent unit with the exception of the Coordinator whose purpose is to tie together all the other modules. The reason for this is to make it easier to replace or update independent modules. For instance if it is determined that a new machine learning algorithm should be used as the classifier, that module can easily be replaced without changing any other unit. Same goes for if the social media scanner needs to be changed to target a different platform. The benefit of independent modules is also the scaling factor. It is easy to replicate modules to work in parallel. For example the Document Representation module, whose task is to convert sentences to a representation that the classifier can work with, can be replicated as several instances that work simultaneously on different sentences.

How these modules should be realized depends on the scale of the application. If the system should serve a few hundred users it is enough to place all the modules in a single application with no replication. But if the system should serve a million users it might be a better idea to deploy every module as a replicated web service where additional instances can be launched during heavy load.

6.4.1 Document Representation

The purpose of this module is to transform documents, which in this case are comments on social media into a representation that is suitable for the classifier to work with. For a machine learning classifier the representation is typically a vector of some sort. Hence the module would take a comment and output the vector representation of that comment.

6.4.2 Classifier

The classifier has one clear role. To classify documents as potential cyberbullying or non cyberbullying. It takes as input a document that has been preprocessed by the document representation module and outputs whether that comment was cyberbullying or not.

6.4.3 Storage

The storage module is responsible for keeping information about users, which profiles these users are monitoring and bullying comments connected to any monitored profile.

6.4.4 User Interface

The user interface allows users to add and remove profiles for monitoring, look at monitored profiles and any cyberbullying comments connected to the monitored profiles. The user interface communicates with the storage module for displaying and storing information.

6.4.5 Social Media Scanner

The task of the social media scanner is to scan monitored profiles for any new activity and output it. For instance if three new comments have been posted on the monitored profile since it was last checked, the scanner would return these three new comments.

6.4.6 Coordinator

The coordinator ties it all together by using the other modules. It asks the storage module for profiles that should be scanned. It then requests new activity for these profiles from the social media scanner. New activity in the form of documents are then passed to the document representation module for transformation. The transformed documents are classified by the classifying module and passed back to the coordinator. The coordinator then updates the storage with any new cyberbullying activity.

Chapter 7

Results - Prototype

This chapter describes the resulting prototype which is the final product of the project. The system consists of three different pieces of software and a storage unit. The first part is a web API that does classification of comments. The second part is an application that scans ASKfm profiles. The last piece is a web interface that allows users to employ the system. The storage unit is an SQL Server database.

7.1 SVM Web API

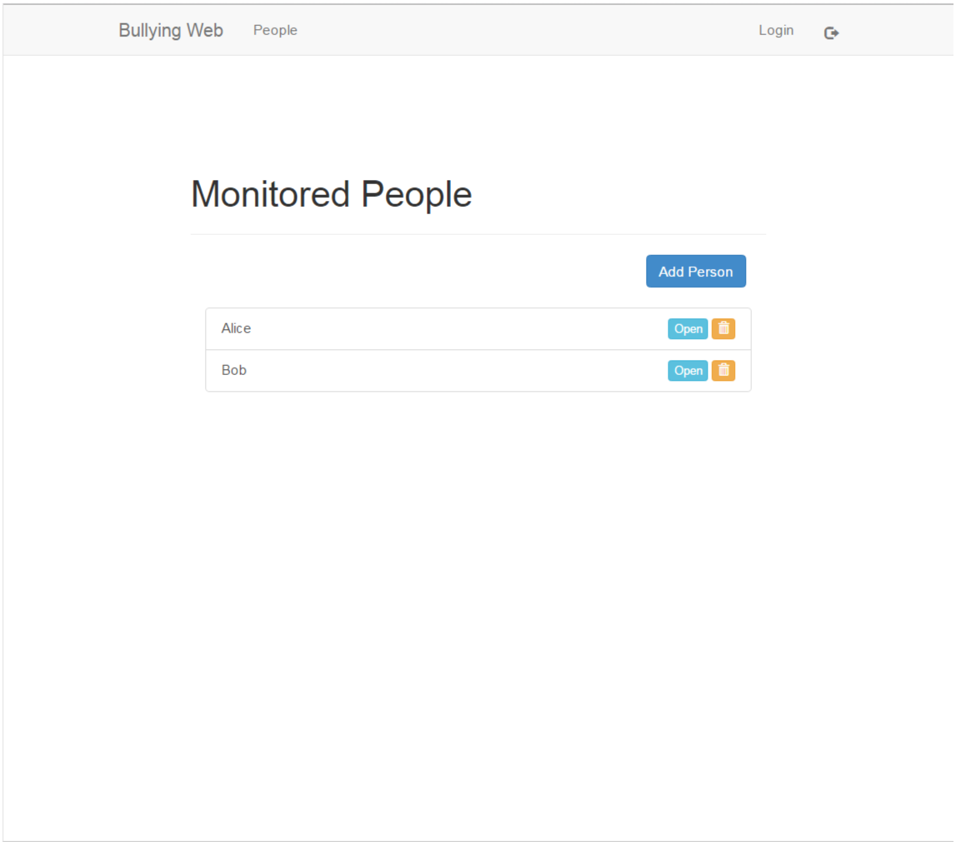
This part of the system has two responsibilities, the first is to convert comments into the document representation format as described in the experiments chapter. The second is to classify comments using an SVM. The service therefore has only one valid request, given a set of comments return the classification for each comment i.e. bullying or not bullying. The web API was implemented as a lightweight Java HTTP Server using NanoHTTPD [2]. The same Java implementation of TFIDF used in the experiments was used in the prototype to calculate the feature vectors. When comments have been converted to document vectors they are written to a file which is then fed to the SVM$^{\text{light}}$ C implementation along with the trained model file developed during the experiments. The results file is read and the results returned to the client.

This implementation comes with the the benefit that the model can be retrained easily by simply replacing the model file that is fed together with the input file to SVM$^{\text{light}}$. It can even be done while the software is running. In addition it is easy to scale this component by using several servers since they are decoupled from the rest of the system. Finally it opens the window for developing other types of applications that can make use of bullying classification, as we will look into later in this chapter.

7.2 Web Interface

The user interface was implemented as a C# ASP.NET MVC application using AngularJS. The interface consists of two main views. The first allows a user to add and view people that they wish to monitor, for instance a parent would want to monitor their children. See Figure 7.1 for a look at this view. The second view displays all bullying comments connected to social media profiles of that person. Here the user may also add additional profiles for that particular person in order to have the system monitor them. This view can be seen in

Figure 7.2 The web interface is tightly coupled with the storage unit since it is essentially just a more user friendly way of manipulating the database. Requests done through the web interface (such as adding a profile for monitoring or requesting all bullying comments for a person) to the server are simply forwarded as queries to the database.

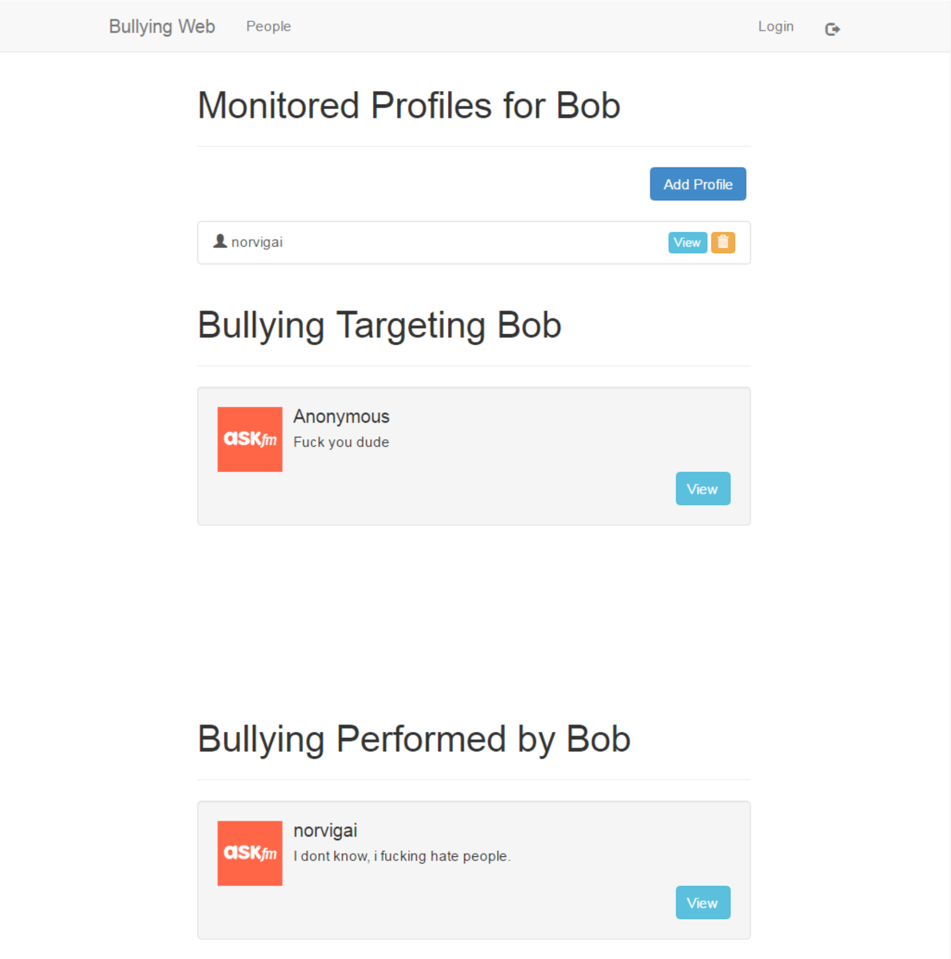


Monitored People

Alice

Bob

Figure 7.1: The People view of the prototype, allowing a user to view and add people for monitoring. Here Alice and Bob have been added for monitoring.



Monitored Profiles for Bob

norvigai

Bullying Targeting Bob

Anonymous
Fuck you dude

Bullying Performed by Bob

norvigai
I dont know, i fucking hate people.

Figure 7.2: The view in the prototype listing social media profiles for a person as well as bullying comments identified by the system. Here we are viewing Bob who has a single ASKfm profile called norvigai (test profile for this project). The system has identified two comments linked to Bob's profile.

7.3 Database

An SQL Server Database was used as storage for the developed prototype. The reason for this particular Database Management System is that interfacing it with an ASP.NET web application is effortless. The database consists of five tables. Figure 7.3 shows the tables, primary keys and foreign keys. When a user adds a new person using the web interface a new row is added to the People table, a foreign key dependency references the ID of the user that added the person. A user can then go ahead and add multiple Profiles that belongs to that newly added person. A profile represents a profile on a social media platform. All profiles in the database are repeatedly scanned by the scanner software. The BullyingComments table is used to store comments that were classified as bullying. The comments are connected to the profile they were read from. BullyingComments can then be retrieved based on a

User, a Person or a Profile. The SMPlatforms table simply contains the different social media platforms and is prefilled with the different supported platforms. For this prototype only ASKfm is supported. The table is there to make it easy to add storage support for additional platforms.

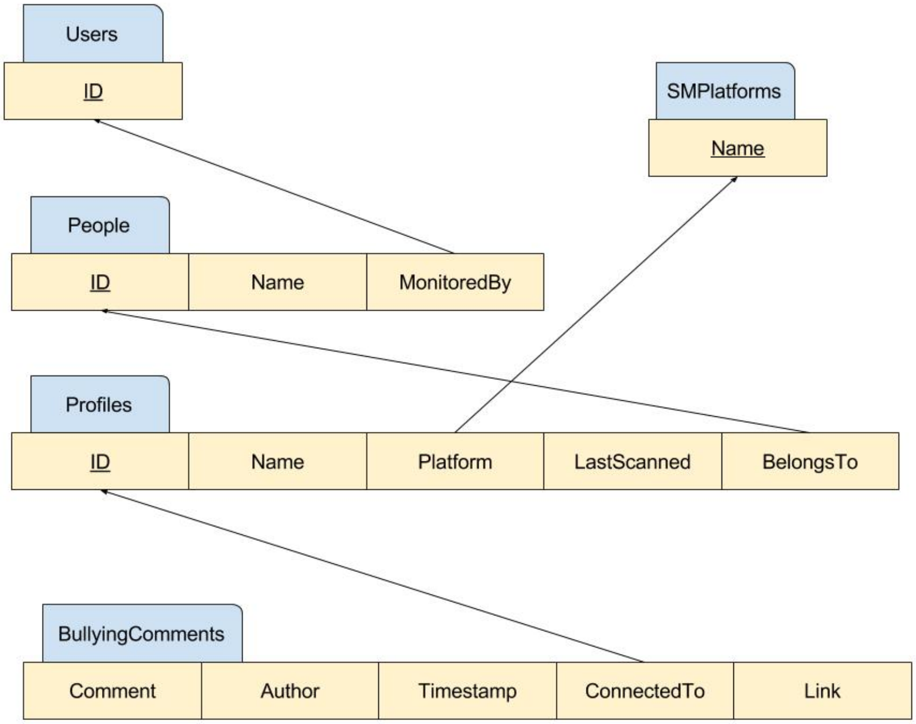


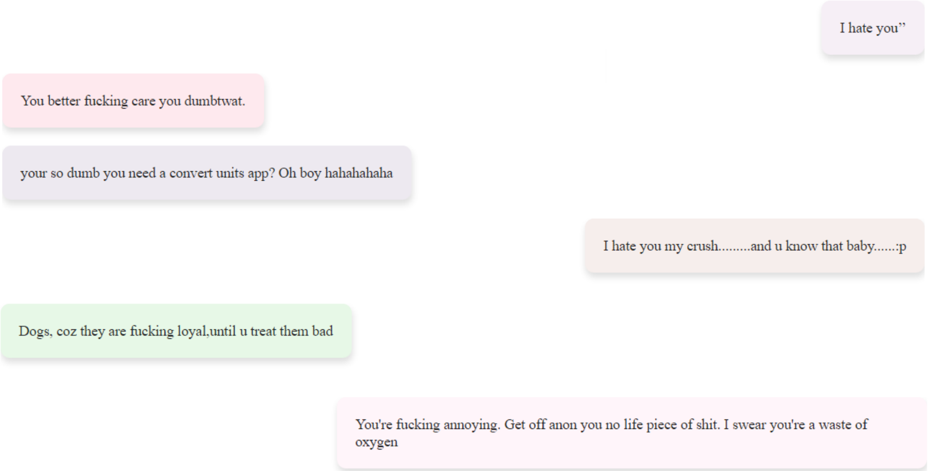
Figure 7.3: Database diagram for the prototype database, showing tables, primary keys and foreign key dependencies. Primary keys are underlined.

7.4 Scanner

The social media scanner is implemented as a separate C# application. Its only task is to continuously scan for new comments posted on any of the monitored profiles. This is done by simply sending a HTTP request to the profile page and parsing the returned HTML for questions and answers. It uses the LastScanned property of the Profiles table to make sure that only comments posted after the profile was last scanned is retrieved. When new comments have been found by the scanner, they are sent to the SVM web API for classification. All comments then classified as bullying are sent to the database for storage.

7.5 Bonus Application - Live Feed

As mentioned earlier, having a service capable of classifying bullying comments opens up possibilities for other applications. One such application was thought of towards the end of the project and quickly implemented. The idea was to have a system that scans random profiles continuously and displays bullying comments on a web page as the system identifies them. This could possibly be used to give an insight into how common bullying is on ASKfm, and therefore be used to raise awareness of the problem that is cyberbullying. Another instance of the scanner software, though slightly modified, was used. This scanner starts with a list of eighty thousand ASKfm profiles. It then scans these profiles and adds any new profiles it encounters to the list. If it does at some point run out of profiles to scan it will restart with the eighty thousand profiles in a different order. The comments scanned are classified using the SVM web API. A simple web frontend was developed using React.js that repeatedly queries the application for the latest identified bullying comments which are then displayed in a chat like fashion on the web page. The React component Chat-Template [1] was used to create the feeling that these messages were being sent at the present moment. Figure 7.4 shows a snapshot of the live feed in action. This application could also be used to acquire more test data for classification, by displaying these messages and allowing visitors to point at comments they interpret as bullying. The displayed comments and visitor's opinions could be used to retrain the classifier.



I hate you”

You better fucking care you dumbtwat.

your so dumb you need a convert units app? Oh boy hahahahaha

I hate you my crush.........and u know that baby......:p

Dogs, coz they are fucking loyal,until u treat them bad

You're fucking annoying. Get off anon you no life piece of shit. I swear you're a waste of oxygen

Figure 7.4: A snapshot of the live feed bonus application displaying positively classified comments as they are identified by the system.

Chapter 8

Conclusions

In this thesis we have investigated the possibilities of building a system capable of automatically identifying cyberbullying on social media. This chapter summarizes what we have learned, discusses limitations of the achievements and concludes with a brief look at future work.

8.1 Goals

The goal with the project was to generate knowledge of how an automatic system for detecting bullying on social media could be constructed. During the project we have learned how to employ state of the art methods in NLP and ML for bullying classification. Which is the core in any bullying detection system. We have learned the limitations of these methods in the form of differentiating between common bad language and cyberbullying. We have learned about some of the challenges that we are facing when trying to automatically scan a social media platform. Most importantly we have learned that an automatic system for bullying detection is possible to some extent as shown by the implemented prototype. The greatest limiting factor is how well classification can be performed. With the knowledge gained during this project it is the author's opinion that the step to a real world application is not very far.

8.2 Prototype Limitations

The prototype is limited in some of its abilities. As described earlier the classification of comments is limited in precision. The prototype has a hard time differentiating between bullying comments and regular comments including bad language. Another limitation lies in the ability to scan comments from ASKfm. A question does not show up on a profile page unless it has been answered, which means that if the victim does not answer a bullying question there is no way for this particular scanner to find it. Furthermore when monitoring a profile the scanner does not look for questions posted by that user on other people's profile pages. Finally with the way ASKfm displays questions on the profile page, only the last 25 questions and answers are displayed, unless the page is scrolled down causing the page to load more questions. This results in the scanner only finding the last 25 questions meaning that if a user is very active and answers more than 25 questions since the profile was last scanned, some questions may be missed.

8.3 Future work

To improve upon the weaknesses of the prototype the priority should be to improve classification. Firstly it would be beneficial to acquire more training data. While selecting good features is important it is likely that more training data will boost the performance of the classifier the most. Acquiring and manually labeling training data is however a very tedious and time consuming task. For this project at least 40 hours were spent collecting and labeling data.

When the main issues with classification have been improved upon, future work on the prototype could involve for instance: adding support for additional social media platforms and incorporating the live feed to involve users in labeling training data. It could also be a good idea to work on spelling and grammar correction which could help the classifier.

Chapter 9

Acknowledgements

I would like to thank Dohi Agency and their employees for their help and resources during this project. I would also like to thank my university supervisor Lars-Erik Janlert for giving valuable feedback during the work on this thesis.

References

[1] Chat-template. https://github.com/sevenleaps/chat-template. [Online; accessed 28-April-2016].

[2] Nanohttpd. https://github.com/NanoHttpd/nanohttpd. [Online; accessed 28-April-2016].

[3] Christopher JC Burges. A tutorial on support vector machines for pattern recognition. *Data mining and knowledge discovery*, 2(2):121–167, 1998.

[4] Cyberbullying Research Center. What is cyberbullying? http://cyberbullying.org/what-is-cyberbullying/. [Online; accessed 25-January-2016].

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

[6] Yunfei Chen, Lanbo Zhang, Aaron Michelony, and Yi Zhang. 4is of social bully filtering: identity, inference, influence, and intervention. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, pages 2677–2679. ACM, 2012.

[7] Cnet.com. Ask.fm, the troubling secret playground of tweens and teens. http://www.cnet.com/news/ask-fm-the-troubling-secret-playground-of-tweens-and-teens/. [Online; accessed 24-February-2016].

[8] Helen Cowie. Cyberbullying and its impact on young people's emotional health and well-being. *The Psychiatrist*, 37(5):167–170, 2013.

[9] cybersafetysolutions.com. What can i do if i am cyberbullied. http://www.cybersafetysolutions.com.au/fact-what-to-do-if-i-am-bullied.shtml. [Online; accessed 26-January-2016].

[10] Maral Dadvar, FMG de Jong, RJF Ordelman, and RB Trieschnigg. Improved cyberbullying detection using gender information. 2012.

[11] Maral Dadvar, Dolf Trieschnigg, Roeland Ordelman, and Franciska de Jong. Improving cyberbullying detection with user context. In *Advances in Information Retrieval*, pages 693–696. Springer, 2013.

[12] Dailymail.co.uk. Pupils and parents warned over social networking website linked to teen abuse. http://www.dailymail.co.uk/news/article-2261588/Ask-fm-Pupils-parents-warned-social-networking-website-linked-teen-abuse.html. [Online; accessed 24-February-2016].

[13] Karthik Dinakar, Roi Reichart, and Henry Lieberman. Modeling the detection of textual cyberbullying. In *The Social Mobile Web*, 2011.

[14] DL4J. Word2vec. http://deeplearning4j.org/word2vec. [Online; accessed 26-January-2016].

[15] Dohi. Dohi agency. http://dohi.se/agency. [Online; accessed 27-April-2016].

[16] Facebook. Report something. https://www.facebook.com/help/263149623790594/. [Online; accessed 26-January-2016].

[17] Paul Foreman. A to z positive words. http://www.mindmapinspiration.co.uk/. [Online; accessed 25-January-2016].

[18] Friends. Friends online report. http://friends.se/wp-content/uploads/2015/03/Natrapporten-final-webb-eng.pdf. [Online; accessed 25-January-2016].

[19] Friends. The responsibility of the school. http://friends.se/en/what-we-do/school/the-responsibility-of-the-school/. [Online; accessed 26-January-2016].

[20] Stanford Natural Language Processing Group. The stanford parser. http://nlp.stanford.edu/software/lex-parser.shtml. [Online; accessed 24-February-2016].

[21] Ralf Herbrich and Thore Graepel. A pac-bayesian margin bound for linear classifiers. *Information Theory, IEEE Transactions on*, 48(12):3140–3150, 2002.

[22] Qianjia Huang, Vivek Kumar Singh, and Pradeep Kumar Atrey. Cyber bullying detection using social and textual analysis. In *Proceedings of the 3rd International Workshop on Socially-Aware Multimedia*, pages 3–6. ACM, 2014.

[23] Thorsten Joachims. Making large-scale svm learning practical. LS8-Report 24, Universität Dortmund, LS VIII-Report, 1998.

[24] Quoc V Le and Tomas Mikolov. Distributed representations of sentences and documents. *arXiv preprint arXiv:1405.4053*, 2014.

[25] Christopher D Manning, Prabhakar Raghavan, Hinrich Schütze, et al. *Introduction to information retrieval*, volume 1. Cambridge university press Cambridge, 2008.

[26] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[27] Vinita Nahar, Xue Li, and Chaoyi Pang. An effective approach for cyberbullying detection. *Communications in Information Science and Management Engineering*, 3(5):238, 2013.

[28] NoBullying.com. The tragic megan meier story. http://nobullying.com/the-megan-meier-story/. [Online; accessed 26-January-2016].

[29] Nobullying.com. Understanding the reasons behind ask.fm bullying. http://nobullying.com/ask-fm-cyber-bullying/. [Online; accessed 24-February-2016].

[30] NoBullying.com. What is cyberbullying? http://nobullying.com/what-is-cyberbullying/. [Online; accessed 25-January-2016].

[31] Slav Petrov. Announcing syntaxnet: The world's most accurate parser goes open source. http://googleresearch.blogspot.se/2016/05/announcing-syntaxnet-worlds-most.html. [Online; accessed 16-May-2016].

[32] Kelly Reynolds, April Kontostathis, and Lynne Edwards. Using machine learning to detect cyberbullying. In *Machine Learning and Applications and Workshops (ICMLA), 2011 10th International Conference on*, volume 2, pages 241–244. IEEE, 2011.

[33] Nakatani Shuyo. Language detection library for java, 2010.

[34] Andre Sourander, Anat Brunstein Klomek, Maria Ikonen, Jarna Lindroos, Terhi Luntamo, Merja Koskelainen, Terja Ristkari, and Hans Helenius. Psychosocial risk factors associated with cyberbullying among adolescents: A population-based study. *Archives of general psychiatry*, 67(7):720–728, 2010.

[35] StopBullying.gov. What you can do. http://www.stopbullying.gov/kids/what-you-can-do/. [Online; accessed 26-January-2016].

[36] The Telegraph. Cyberbullying suicides: What will it take to have ask.fm shut down? http://www.telegraph.co.uk/news/health/children/10225846/Cyberbullying-suicides-What-will-it-take-to-have-Ask.fm-shut-down.html. [Online; accessed 24-February-2016].

[37] Ditch the Label. The annual cyberbullying survey. http://ditchthelabel.org/downloads/cyberbullying2013.pdf. [Online; accessed 25-January-2016].

[38] Ditch the Label. Top 10 tips for overcoming bullying. http://www.ditchthelabel.org/top-10-tips-for-overcoming-bullying/. [Online; accessed 26-January-2016].

[39] Twitter. Reporting abusive behavior. https://support.twitter.com/articles/20169998. [Online; accessed 26-January-2016].

[40] Wikipedia. Hyperplane. https://en.wikipedia.org/wiki/Hyperplane. [Online; accessed 1-March-2016].

[41] Dawei Yin, Zhenzhen Xue, Liangjie Hong, Brian D Davison, April Kontostathis, and Lynne Edwards. Detection of harassment on web 2.0. *Proceedings of the Content Analysis in the WEB*, 2:1–7, 2009.

[42] Rui Zhao, Anna Zhou, and Kezhi Mao. Automatic detection of cyberbullying on social networks based on bullying features. In *Proceedings of the 17th International Conference on Distributed Computing and Networking*, page 43. ACM, 2016.

Appendix A

Dictionaries

A.1 Bad Words Dictionary

anal
anus
arrse
arse
ass
asses
assfucker
assfukka
asshole
assholes
asswhole
bad
ballbag
balls
ballsack
bastard
bbs
bch
biatch
bich
bitch
bitcher
bitchers
bitches
bitchin
bitching
blowjob
blowjobs
boiolas
bollock
bollok
boner
boob
boobs
booobs
boooobs
booooobs
booooooobs
breasts
btch
buceta
bugger
bunnyfucker
butt
butthole
buttmuch
buttplug
butts
carpetmuncher
cck
ccksucker
clitoris
cock
cockface
cockhead
cockmunch
cockmuncher
cocks
cocksuck
cocksucked
cocksucker
cocksucking
cocksucks
cocksuka
cocksukka
cok
cokmuncher
coksucka
coon
cox
cum
cummer
cumming
cums
cumshot
cunilingus
cunillingus
cunnilingus
cunt
cuntlick
cuntlicker
cuntlicking
cunts
cyalis
cyberfuc
cyberfuck
cyberfucked
cyberfucker
cyberfuckers
cyberfucking
dck
dick
dickhead
dildo
dildos
dink

dinks
dirsa
dirty
discusting
disgusting
dlck
dogfucker
doggin
dogging
donkeyribber
doosh
duche
dumb
dumbass
dyke
ejaculate
ejaculated
ejaculates
ejaculating
ejaculatings
ejaculation
ejakulate
fack
fag
fagging
faggitt
faggot
faggs
fagot
fagots
fags
fanny
fannyflaps
fannyfucker
fanyy
fat
fatass
fcuk
fcuker
fcuking
feck
fecker
felching
fellate
fellatio
fingerfuck
fingerfucked
fingerfucker
fingerfuckers
fingerfucking
fingerfucks
fistfuck
fistfucked
fistfucker
fistfuckers
fistfucking
fistfuckings
fistfucks
fk
flange
fook
fooker
fuck
fucka
fucked
fucker
fuckers
fuckhead
fuckheads
fuckin
fucking
fuckings
fuckingshitmotherfucker
fuckme
fuckoff
fucks
fuckwhit
fuckwit
fucxking
fudgepacker
fuk
fuker
fukker
fukkin
fuks
fukwhit
fukwit
fux
fuxr
gangbang
gangbanged
gangbangs
gaylord
gaysex
goatse
gtfo
hardcoresex
hate
hell
heshe
hoar
hoare
hoe
hoer
hoes
homo
hore
horniest
horny
hotsex
idiot
idiotic
jackoff
jap
jerk
jerkoff
jism
jiz
jizm
jizz
kawk
kick
kill
knob
knobead
knobed
knobend
knobhead
knobjocky
knobjokey
kock
kondum
kondums
kum
kummer
kumming
kums
kunilingus
labia
lich
lick
litch
lmfao
lust
lusting
masochist
masterb

masterbat
masterbate
masterbation
masterbations
masturbate
materb
materbate
mean
meanie
mf
mfo
mof
mofo
mothafuck
mothafucka
mothafuckas
mothafuckaz
mothafucked
mothafucker
mothafuckers
mothafuckin
mothafucking
mothafuckings
mothafucks
motherfuck
motherfucked
motherfucker
motherfuckers
motherfuckin
motherfucking
motherfuckings
motherfuckka
motherfucks
mterbate
muff
mutha
muthafecker
muthafuckker
muther
mutherfucker
nazi
ngga
ngger
nigga
niggah
niggas
niggaz
nigger
niggers
niggh
niggr
nobhead
nobjocky
nobjokey
numbnuts
nutsack
orgasim
orgasims
orgasm
orgasms
penis
penisfucker
phonesex
phuck
phuk
phuked
phuking
phukked
phukking
phuks
phuq
pigfucker
pimpis
piss
pissed
pisser
pissers
pisses
pissflaps
pissin
pissing
pissoff
poop
prick
pricks
prn
pron
pube
punch
pusse
pussi
pussies
pussy
pussys
queer
rectum
retard
rimjaw
rimming
sadist
schlong
screwing
scroat
scrote
scrotum
semen
sex
shag
shagger
shaggin
shagging
shemale
shi
shit
shitdick
shite
shited
shitey
shitfuck
shitfull
shithead
shiting
shitings
shits
shitted
shitter
shitters
shitting
shittings
shitty
sht
shut
skank
slap
slut
sluts
smegma
smell
smelly
smut
snatch
sob
sonofabitch
spac
spunk
stfu
stupid

suck
teets
teez
testical
testicle
tit
titfuck
tits
titt
tittie
tittiefucker
titties
tittyfuck
tittywank
titwank
tosser
ttte
ttties
turd
twat
twathead
twatty
twt
twunt
twunter
ugly
vagina
vgra
wang
wank
wanker
wanky
whoar
whore
wse

A.2 Second Person Pronoun Dictionary

you
yourself
u
ur
you're
youre
yours
yourselves
your
@user

A.3 Third Person Pronoun Dictionary

he
she
he's
hes
she's
shes
they
theyre
they're
him
her
them
his
hers
theirs
himself
herself
themselves

A.4 Positive Words Dictionary

able
abundance
accelerate
accept
acclaim
accolade
accomplish
accord
accredit
accrue
ace
achieve
activate
adept
admirable
adorable
advance
advantage
adventure
affable
affirm
ageless
agree
agreeable
aid
alacrity
alight
alive
amaze
amazing
ambition
amiable
amity
amuse
anew
appealing
applaud
appreciate
approve
arouse
ascend
aspire
assent
assert
assist

associate
assure
astir
astonish
attain
attentive
attest
attraction
attribute
attune
augment
auspicious
authentic
available
avid
award
awash
awesome
aye
beatific
beatify
beatitude
beauteous
beautiful
beautify
beauty
befriend
beloved
benefaction
beneficial
benefit
benevolent
best
bestow
better
betterment
bijou
bless
blessed
blessing
bliss
bloom
blossom
bonafide
bonanza
boost
bountiful
bounty
bright
brighten
brill
brilliant
bubbly
budding
buddy
build
calm
capable
celebrate
certain
charitable
charity
charm
charmer
charming
cheerful
cheers
chirp
chirpy
choice
chortle
chuckle
cinch
civility
classy
clean
clear
comely
comfort
comfortable
comic
comical
compliment
confidence
confirm
congenial
congratulate
conscious
consciousness
considerate
constant
constructive
content
contribute
cool
cooperate
cope
cordial
cosy
could
courage
courteous
creative
credit
cuddly
cushy
cute
decency
decent
delectable
delicate
delicious
delight
desirable
do
dreamy
dynamic
eager
ease
easily
easy
economic
ecstasy
edify
educate
effective
efficiency
efficient
elate
elegant
elevate
eligible
emphasis
emphasize
emphatic
enable
enchant
encourage
endear
endearment
endeavour
endorse
endow
energetic
energize
energy
engage

engaging
engross
enhance
enjoy
enlighten
enlist
enliven
enormous
enough
enrapture
enrich
ensure
enterprise
enterprising
entertain
entertainment
enthral
enthuse
enthusiasm
enthusiastic
entire
entrust
equal
equality
equally
equilibrium
equitable
equity
equivalent
erudite
especial
essence
essential
establish
ethic
ethical
euphony
euphoria
eureka
evolution
exalt
exceed
exceedingly
excel
excellence
excellent
excite
exotic
expert
expertise
exquisite
extensive
extraordinary
exult
fabulous
fair
faithful
fame
family
fancy
fantastic
fare
fascinate
favour
favourite
feasible
felicity
fellowship
festive
fetching
fine
finesse
fit
fitting
flamboyant
flash
flexible
flower
focus
fond
fondly
foresee
foresight
forgive
forgiveness
forward
freedom
fresh
friend
friendly
friendship
fruitful
fulfil
fully
fun
funny
gallant
galore
generate
generous
genial
genius
gentle
genuine
gift
gifted
giggle
gist
give
glad
glorious
glory
glossy
glow
going
good
goodness
goodwill
gorgeous
gosh
grace
graceful
gracious
grand
grandeur
grateful
gratify
gratitude
great
greet
greeting
grow
guarantee
guidance
guide
happily
happy
harmonious
harmonize
harmony
healthy
heart
heaven
heavenly
helpful
helping
highly

hilarious
hilarity
hip
holy
homely
honest
honestly
honesty
honeyed
honorary
honour
honourable
hooray
hope
hopeful
hopefully
hospitable
hot
humane
humanitarian
humorous
humour
ideal
ideally
immense
immerse
immune
impartial
impeccable
impress
impressive
improve
improvement
increase
incredible
indeed
ingenious
ingenuity
initiate
initiative
innocent
innovate
input
inspiration
inspire
inspired
interest
interested
interesting
invitation
invite
inviting
jest
joke
jolly
jovial
joy
joyful
joyous
jubilant
jubilation
juicy
just
keen
keep
kind
kind-hearted
kindly
kiss
kudos
large
lark
laugh
lavish
learn
learned
learning
leisure
leisured
leisurely
liberate
liberation
life
light
light-hearted
lighten
likable
like
liking
lively
lovable
love
lovely
loving
loyal
lucid
luck
lucky
lucrative
luminous
luscious
lush
lustre
lustrous
luxuriant
luxuriate
luxurious
luxury
magnificent
magnify
magnitude
maintain
majesty
majority
manifest
marvellous
master
mediate
meditate
mellow
mercy
merit
miracle
miraculous
morale
motivate
neat
new
newly
nice
nicety
nifty
nippy
nirvana
noble
nod
notable
notice
noticeable
nourish
nurse
nurture
obliging
offer
onward
oodles
oomph

open-minded
openly
opportune
opportunity
original
outgoing
outstanding
pacify
palatable
palpable
paradise
paragon
pardon
passion
passionate
passive
patience
patient
peace
peaceable
peaceful
peak
pep
perfect
perfection
persevere
perspective
placid
pleasant
pleasurable
pleasure
plenitude
plenteous
plenty
plush
poise
polite
positive
possible
potential
practical
praise
precious
prize
pro
productive
proficient
progress
promote
promotion
prosper
pukka
pure
purify
purity
quality
quiet
radiant
rapture
ready
reason
reassure
receive
reception
receptive
reciprocate
recommend
refreshing
relax
release
reliable
relief
remarkable
remedy
reputable
respect
responsible
rest
restful
restore
result
reward
rewarding
rich
richly
sacred
sacrosanct
safety
salubrious
satisfaction
satisfactory
satisfy
saving
saviour
self-assertive
self-confidence
self-discipline
self-esteem
self-help
sense
sensible
share
simplicity
simplify
sincere
smart
smashing
smile
sociable
social
special
spectacular
splendid
splendiferous
splendour
steady
straightforward
succeed
success
successful
succinct
suffice
sufficiency
sufficient
sumptuous
superabundant
superior
supple
supply
support
supporter
supporting
supportive
supreme
sweet
swell
sympathetic
sympathise
sympathy
tact
teacher
teaching
testament
testimonial
testimony
thankful
thanksgiving

therapeutic
therapy
thorough
thoughtful
thrill
thrive
tidy
timeless
timely
training
tranquil
tranquillity
transcend
transient
transparent
triumph
triumphant
trust
trustworthy
trusty
truth
tuition
ultimate
unconditional
uncritical
understanding
unequalled
unequivocal
unerring
unfetter
unflagging
ungrudging
upbeat
upgrade
uplift
upstanding
urbane
useful
user-friendly
utmost
valid
validate
valuable
value
venerable
veracious
verify
versatile
viable
vibrant
virtue
virtuosity
virtuoso
virtuous
vitality
vivacious
vivid
warmth
wellbeing
wholehearted
wholesome
wholly
winner
winning
winsome
wisdom
wise
witt
wonderful
wonderment
wondrous
workable
worth
worthwhile
worthy
yippee
young
youth
youthful
zeal
zealous
zest