SVM Constraint Discovery using KNN applied to the Identification of Cyberbullying

D. Ducharme, L. Costa, L. DiPippo, and L. Hamel
Department of Computer Science and Statistics,
University of Rhode Island,
Kingston, Rhode Island,
United States
dducharme@cs.uri.edu, costa.leandrom@gmail.com, dipippo@cs.uri.edu, lutzhamel@uri.edu

Abstract— *In the context of classifying cyberbullying comments, we introduce a k-nearest neighbor/support vector machine hybrid model. A crucial insight into the training of support vector machines is that the training and the resulting model only depend on a few select points of the training data - the support vectors - that represent constraints on the implicit decision surfaces of the models. As part of our hybrid model we propose to identify support vector candidates in the training data using a modified k-nearest neighbor algorithm. The reduced training data is then used to train the support vector machine.*

Keywords: Machine Learning, SVM, KNN, N-Gram, Cyberbullying

1. Introduction

In this paper, we study the application of a hybrid support vector machine (SVM) classifier to the problem of identifying cyberbullying.[1] A crucial insight into the training of SVMs is that the training and the resulting model only depend on a few select points of the training data - the support vectors - that represent constraints on the implicit decision surfaces of the models. As part of our hybrid model, we propose to identify support vector candidates in the training data using a modified k-nearest neighbor algorithm (KNN) [2] that acts like a filter and prunes the size of the training data down to only the points that are possible support vectors. This pruned data set is then used for our cyberbullying study, speeding up our model training and evaluation cycle. Overall, we were able to construct models of cyberbullying that had an accuracy of about 70% to 80% with our hybrid model using less than 50% of the available training data; a significant saving. The data set itself consisted of 350 comments with 909 features split equally between bullying comments and non-bullying comments. From a data perspective the fact that we can successfully train models on less than 50% of the training data is interesting in that real world data, such as the cyberbullying data, can be reduced by these large margins and still produce good models.

These models were generated utilizing several different rules for classification. This helped to represent the different communities that develop on sites. The primary rule set was to classify the comments solely based upon the researchers understanding of the U.S. Federal Code and the Rhode Island General Laws. A secondary rule set was to flag all political and religious comments as cyberbullying to represent a hypothetical websites terms of service. Finally, a third rule set was generated by marking any comment that was flagged as cyberbullying in either rule set to represent a more realistic implementation. These three rule sets were trained and tested on comments taken from both YouTube and Twitter to ensure that the differences in comment length and form did not impact the model.

The remainder of this paper is structured as follows. We survey existing work, both in terms of hybrid SVM models based on KNN and cyberbullying in Section 2. In Section 3, we briefly introduce SVMs and provide an overview of our hybrid model approach. Our data preparation and acquistion is described in Section 4. Results on the performance of our models are given in Section 5. Final remarks and future work are summarized in Section 6.

2. Related Work

A number of works apply KNN to identify support vector constraints, *e.g.* [7], [8], [9]. The approach presented here is very similar to the work found in [8]. Both approaches use KNN to pre-process the training data. However, the main difference is in how KNN is applied: In [8], each point in the training data that belongs to one class uses KNN to select possible support vectors of the other classes while in our approach, KNN investigates if the data point being analyzed is a possible support vector based on all its neighbors.

Utilizing different algorithms to identify cyberbullying online is a very active field of research that is showing promise from several different places. First, there is BullyBlocker from Arizona State University [10]. Its primary purpose is "to exploit social media data and, based on a model built

[1] The use of electronic communication to bully a person, typically by sending messages of an intimidating or threatening nature – Wikipedia

on previous research findings in areas of traditional and cyberbullying in adolescents, to then identify an instance of cyberbullying and notify the parents." To do this the authors have designed a calculation they call the Bullying Rank and, using calculated warning signs and vulnerability, they are able to calculate a risk ranking that will place the child in either a low, moderate, or severe risk category.

Researchers at Columbia University utilize support vector machine learning in order to classify hate speech [11]. They define hate speech as "any communication that disparages a person or a group on the basis of some characteristic such as race, color, ethnicity, gender, sexual orientation, nationality, religion, or other characteristic." Thus, while not all hate speech is cyberbullying (some hate speech has made it into the common vernacular and at this point is so common as to not be considered bullying), there is an overlap between the two. When classifying anti-semitic speech, the researchers were able to achieve an accuracy of 94%. To process their data, the authors used a template-based strategy which applied various positive and negative templates to the paragraph and kept count of how many occurrences were found, which they called the log-odds.

Finally, starting in December of 2016, Twitch, which is an online video game streaming service owned by Amazon, rolled out a new tool called AutoMod [12], [13]. This is a machine learning based automatic moderation software designed to hold back messages for moderator approval. It looks for and filters based on identity language, sexually explicit language, aggressive language, and profanity. Moderators can also set what level of filtering to use which will ignore some of the filters depending on the tier chosen.

The main difference between this research and the work mentioned above is that it is designed to work as a context insensitive tool to match the classification patterns of a moderator or group of moderators. It does not use any outside information to aid the classification like BullyBlocker does and, in order to generalize, it cannot utilize the template-based strategy of the Columbia University study. Finally, while the AutoMod tool is close, it relies on the Amazon administrators to guide the classification into four distinct levels that moderators can then choose to utilize. This means while the AutoMod tool can aid in some instances, it will not necessarily classify the comments in the same way that the moderation staff of that channel would.

3. Support Vector based Learning

3.1 SVM

The Support Vector Machine (SVM) is a machine learning technique that creates a maximum margin classifier between two classes of labeled data by finding specific data points called support vectors and using them as constraints on the margin of a linear classifier (Figure 1). The support vectors are the result of the dual maximum margin optimization

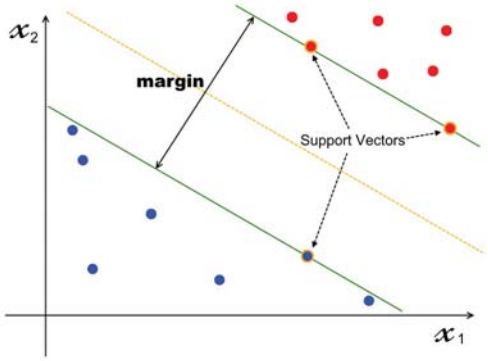

Fig. 1
SVM DECISION SURFACE WITH SUPPORT VECTORS AND MARGIN.

equation (1),

$$\max_{\bar{\alpha}} \phi(\bar{\alpha}) = \max \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{l=1}^{l} \alpha_i \alpha_j y_i y_j \kappa(\bar{x_i}, \bar{x_j}). \quad (1)$$

Here, κ is a kernel function which can be chosen in such a way as to provide optimal fit in the case of data sets that are not linearly separable. Furthermore, the optimization algorithm ensures that the decision surface generated is equidistant from the distributions of both sets of labeled data, that is, in the center of the margin [14].

3.2 The K-Nearest Neighbors Algorithm

K-Nearest Neighbors (KNN) is a machine learning technique that classifies a point based on its k-nearest known data points in the training data. It is a type of instance-based learning, where all computation is deferred until classification. The algorithm is very simple and can be described as follows:

1) **Training Phase:** Store all known points of the training data and their respective labels.
2) **Classification phase:** To classify a new entry *e*:
   a) Calculate the distance between *e* and every point in the training data using the distance function *D*.
   b) Find the *k* closest points in the training data.
   c) Classify the new point *e* as the most frequent class between all the *k* points.

In Figure 2, we show a simple example of KNN. The algorithm will classify the small black circle in the middle of the concentric rings as a blue square if *k*=3. However, this classification would be changed to a red diamond if *k*=5.

The distance function *D* can be specified by the user when creating the classifier. The most common distance functions used are the Euclidean distance, when working with continuous variables, and Hamming distance, when working with discrete values.

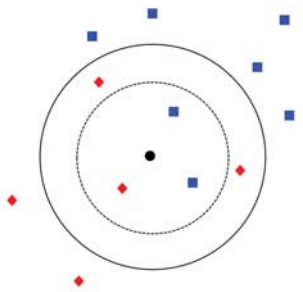

Fig. 2
A K-NEAREST NEIGHBORS EXAMPLE

3.3 KNNFilter

In order to speed-up the training of an SVM, the KNNFilter will use the fact that only points near the border between classes are good support vector candidates and tries to extract them into a new, smaller dataset suitable for efficient training.

The algorithm 1 does so by first creating an empty list *SVC* that will hold the Support Vectors Candidates. Then, for all elements *i* of the training data, it will follow the same procedure.

First, it creates a vector (*DistVector*) to hold the distance between *i* and every other point in the training data calculated using the distance function *D*. With the distance vector filled, the algorithm now appends a new row with the indexes of the other data points creating a new matrix *DistMatrix*.

Next, *DistMatrix* is sorted based on the distance row. Now it holds in one row distances from *i* to all other points of the dataset, in increasing order, and the corresponding index of said point in the original dataset. The algorithm then selects the *k*-nearest neighbors of *i* by using the first *k* elements of the index row.

The method will then compare the class of those *k* neighbors to the class of data point *i*. If any neighbor has a different class than *i*, then *i* is a good candidate to be a support vector and is added to *SVC*.

After this process has been applied to all points of the training data, the set *SVC* will contain data points with a higher chance to be on the margin and they can then be used to train a SVM faster.

To show that KNNFilter can extract the Support Vector candidates correctly, it was tested using artificial datasets with specific formats (an XOR, a circle, and 2 normal distributions). Figure 3 has the plots of both full and the SVC data sets. The full datasets appear on the top half of the figure and the reduced datasets on the bottom half. These examples show that the algorithm was successful in selecting the important points close to the border while maintaining the shape of the data sets. Now, it is important to keep

Algorithm 1 KNNFilter

Input: Training Set T of size n, Distance Function D, k
Output: Support Vector Candidates [*SVC*]
$SVC \leftarrow \emptyset$
for all Data Point p_i in T **do**
  $DistVector \leftarrow \emptyset$
  for all Data Point p_j in T, where $p_i \neq p_j$ **do**
    $DistVector \leftarrow [DistVector, D(p_i, p_j)]$
  end for
  $DistMatrix \leftarrow AddRow(DistVector, [1, ..., n-1])$
  $DistMatrix \leftarrow Sort(DistMatrix[0])$
  for $nn = 0$ **to** $nn = k$ **do**
    if $p_i[class] \neq p_{DistMatrix[nn,1]}[class]$ **then**
      $SVC \leftarrow [SVC, p_i]$
      break
    end if
  end for
end for
return *SVC*

in mind that any SVM classifier is defined only by the points that represent constraints on the margin and therefore training an SVM on the reduced dataset will result in the same decision surface as an SVM trained on the full dataset. However, the former can be trained more efficiently because of the reduced training data size.

An important part of the algorithm is the value picked for *k*. If *k* is too small, the *SVC* set might miss important features of the border and, if it is too large, then the algorithm will lose the desired effect of selecting a small *SVC* data set. The value for k is a dataset dependent parameter. When class borders have a fair amount of overlap, small values of k might be enough to ensure that the border is well represented in *SVC*. However, when classes are well defined and the class distributions are separated by a large margin, k will need to be larger to make sure that the KNNFilter can identify that border correctly and find neighbors from other classes. Empirical tests with real datasets have shown that,

$$k = 0.1 \times n, \tag{2}$$

where n is the size of the training data, is a good starting point when selecting k.

The other component of the KNNFilter that needs to be discussed is the distance function D. All examples in this paper used the Euclidean distance for the distance function D. This means that all border *SVC* sets were the result of the algorithm being applied to the input space. However, other distance functions can be used with KNN to better reflect the dataset being analyzed. But more interesting than that is the possibility of doing the KNN in the same feature space the SVM will use to create its decision surface. This can be accomplished by replacing D with an appropriate kernel function expression, *e.g.* [3].

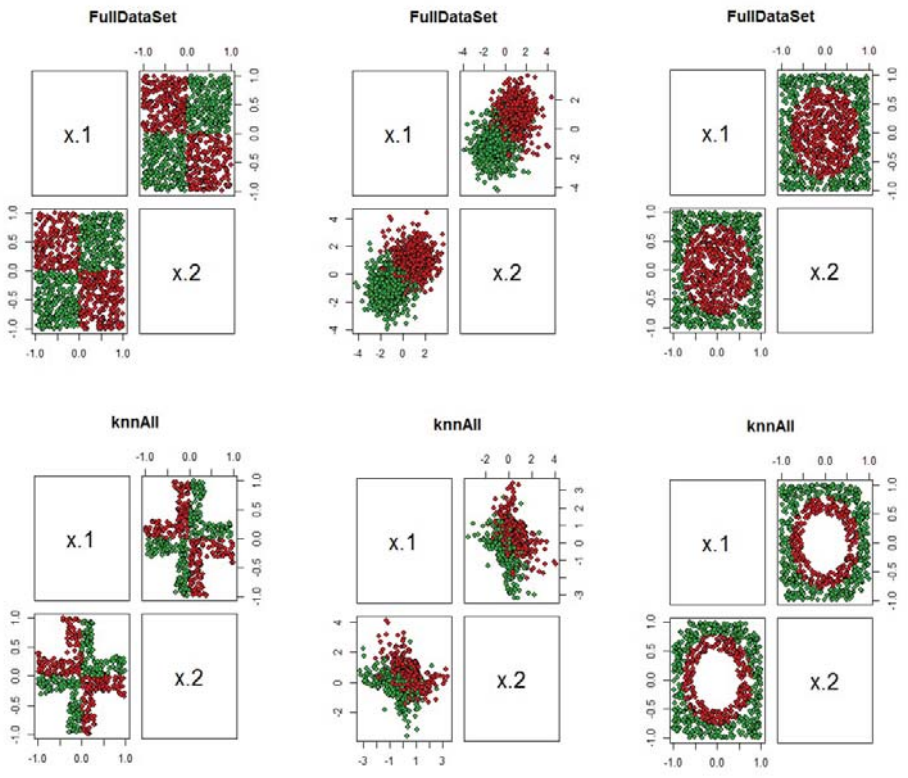

Fig. 3
KNNFILTER IMPACT ON THE TRAINING SET

4. Data Preparation

Cyberbullying is a growing phenomenon that is plaguing today's youth and is increasing at an alarming rate. As technology advances and becomes prevalent in more facets of our lives, the potential for bullies to reach into a teen's life increases, causing additional hardship leading to depression and, in some cases, even suicide. The Cyberbullying Research Center's research [15] showed that, in 2013, about one in four teens had been the victim of cyberbullying and one in six teens was involved in the bullying. In every study, their research also shows that cyberbullying is on the rise. Extrapolating from the studies, they estimate that 2.2 million teens were cyberbullied nationwide in 2011 up from an estimated 1.9 million in 2009. This number is expected to increase as both teens and adults continue to have an increased online presence.

The current method of dealing with the problem is to use human moderators and administrators to remove offending comments and ban repeat offenders. However, on most sites, the number of comments far outweighs the ability of moderators to read and approve every comment. Thus, in order to combat this, moderators typically rely on users to flag or report offending comments. This means that users have already seen and been affected by the cyberbullying at which point it is too late to remove it. The algorithm, created by this research, will improve this situation by automatically flagging offending comments. At that point, a human moderator could approve or deny them without the intended victims having ever had to read it [16].

4.1 Data Set Details

To create a good representative data set, a web crawler was utilized to harvest comments from YouTube. This web crawler was designed to grab all of the comments from the videos found in the YouTube playlist Popular Right Now by #PopularOnYouTube [17] where the top 200 videos at that time are displayed. The web crawler was designed in such a way that no user information was recorded and any username in the replies were stripped out. These comments were grabbed while the video was still popular which helped to ensure that the comments present at the time the site was crawled did not have time to be fully moderated.

For the web crawler comment process, YouTube was chosen for several reasons. First, the site has a variety of comments, users, and viewpoints resulting in arguments that can get heated. Next, it allows users to post with user-

Comment	Class	Reasoning
Jorge Ramos is such a fucking race baiter. “It’s because of the color of his skin!"	1	This was not bullying because while it fails step two, it is legitimate and because the subject of the harassment is not present in the conversation
We’ll see this lil shit on Ellen	-1	This one is bullying because the subject of the comment is the person who posted the video and is likely to read the comment and be bothered by it
Please show this to the narrow-minded right wing fucks!	-1	While this comes close to serving a purpose, the likely-hood of a right wing reader being upset by this outweighs the need for this comment to exist
Butterface bitch. I bet if she didn’t have those big ass tits the comments here would be she‘s ugly and everything lol.	-1	This one fails every step of the test

Table 1
EXAMPLE COMMENT CLASSIFICATIONS

names that are completely removed from their non-internet alias. This insures users that there are no consequences for anything they might say, outside of having their comments moderated or their account shutdown. Another key factor was the fact that the site has no differentiation between public and private comments. This means that any comment made by a user, that has not been removed by a moderator, is visible and public to every other user. Facebook and Twitter, on the other hand, allow users to specify who can view comments and as such there is a greater expectation of privacy on those sites.

After crawling for only two hours across two different days, over 118,000 comments were recorded into a database. From this database, comments were then randomly selected and classified as cyberbullying (-1) or not (1) based on the following three criteria:

1) Is the comment sexual in nature?
2) Was the comment intended to seriously alarm, annoy or bother the reader?
3) And does the comment serve a legitimate purpose?

These criteria were created after a thorough review of the applicable federal and Rhode Island state laws. By using these three tests, all of the state and federal laws are more than satisfied. See table 1 for four examples of comments that were classified and the reasoning as to why.

In total 4017 comments out of the 118,000 were manually classified by the researcher, which provided 265 cyberbullying comments.

4.2 N-grams

In order to convert the comments into a numerical form capable of being utilized in a SVM, we converted the comment text into N-grams. An N-gram is a word, called a token, or a group of tokens, that can be used to predict a statistical language model [18]. These N-grams, along with the percent of capitalization, make up all of the features that are used by the model.

$$Probability = \frac{CountwithNGram}{TotalCount} \quad (3)$$

Fig. 4
N-GRAM PROBABILITY

n=350	Predicted Cyberbullying	Predicted Clean
Actual Cyberbullying	161	14
Actual Clean	23	152

Table 2
CONFUSION MATRIX OF THE SVM TRAINED WITHOUT KNNFILTER.

As an example for N-gram processing, take the following phrase:

The quick brown fox jumps over the lazy dog.

This phrase contains 9 words, 1 capital letter and one punctuation character. For the purposes of n-grams capitalization is ignored and punctuation and spaces are removed as well. So for the length 1 N-grams there is the, $quick$, $brown$, fox, $jumps$, $over$, the, $lazy$, dog. Now if the length 2 N-grams are calculated they would contain $the|quick$, $quick|brown$, $brown|fox$, $fox|jumps$, $jumps|over$, $over|the$, $the|lazy$, $lazy|dog$. This would continue for the length 3 N-grams all the way up through the length 9 n-grams at which point there would be no further difference with this phrase.

Once the N-grams are constructed, they are used to calculate the frequency of occurrence since machine learning algorithms work on numbers. In our case, this is done by the formula shown in Figure 4. In this equation you take the count of how many times the N-gram token is found within the phrase, (Count with NGram) and divide it by the total number of tokens found with that length.

So again, looking at the phrase, we see that in length 1 N-grams the token “the" has a 22.2% occurrence while all the other words have a 11.1%.

5. Results

Our training data for this study consisted of 350 comments sampled from our original data with 909 features which consisted of n-grams up to the length 3. The training data included 175 bullying comments and 175 non-bullying comments ensuring that the two classes were balanced. The optimal SVM model trained on the training data set was obtained using the radial basis function kernel with a

k	XV-Acc (%)	Resub Acc (%)	95% Conf Int	Data Red. (%)
2	68.2	68.3	(63.0, 92.6)	61.4
3	73.1	84.0	(65.3, 91.8)	31.1
4	73.5	87.1	(69.5, 89.9)	16.9
5	75.0	88.3	(71.9, 90.7)	9.7
6	77.0	89.7	(71.2, 92.4)	5.7
7	76.4	89.4	(72.5, 91.3)	1.4
8	77.6	89.4	(74.3, 94.3)	0.6
9	77.3	89.4	(72.8, 92.9)	0.6
10	77.7	89.4	(71.4, 92.9)	0.3
11	77.1	89.4	(72.8, 92.9)	0.3
12	77.1	89.4	(74.3, 92.9)	0
13	77.1	89.4	(72.9, 92.9)	0

Table 3
ACCURACY OF THE SVM MODEL AS A FUNCTION OF K IN THE KNNFILTER.

penalty constant C of 1000 and a gamma of 0.01. The cross-validated accuracy of this was 77.7% with a bootstrapped 95% confidence interval of (72.9%, 92.9%). Table 2 shows the confusion matrix of the resubstitution error of 89.0%. The resubstitution error is computed by applying the optimal cross-validated model to the whole training data. The confusion matrix reveals that the model commits slightly more false-positive than false-negative errors which in this context is an advantage.

Next, we applied our hybrid model to the training data. Table 3 shows the cross-validated accuracy of our hybrid SVM model as a function of k, the neighbor parameter for our KNNFilter algorithm. The table shows the cross-validated accuracy, the resubstitution error, the 95% bootstrapped confidence interval, and the amount by which KNNFilter reduced the training data. Here the resubstitution error is computed by applying a model trained on the reduced training data to the full training data set.

We can see that reducing the training data by 60% still produces a classifier with a 68.2% accuracy. As we take out less and less of the training data, the performance of the models increases all the way up to 77.1% when no data is removed from the training set. What is perhaps most striking about these results is that even if we knock out 60% of the training data using our KNNFilter we are still able to train a model whose performance difference to the best performing model is not statistically significant. Note that the model at $k = 2$ has a 95% confidence interval of $(63.0, 92.6)$ whereas the model at $k = 12$ has a confidence interval of $(74.3, 92.9)$. These intervals overlap and therefore the performance difference between the models at $k = 2$ and $k = 12$ is not statistically significant. Precisely the result we were hoping to achieve.

The table also shows that if KNNFilter does not remove any training data the performance of the hybrid model is the same as the original SVM model above.

Table 4 shows the confusion matrix of our hybrid model

n=350	Predicted Cyberbullying	Predicted Clean
Actual Cyberbullying	155	20
Actual Clean	36	139

Table 4
$k = 3$ CONFUSION MATRIX

with $k = 3$. This model has a resubstitution error of 84% and, as the confusion matrix shows, it is a reasonable model in terms of false-negative and false-positive errors and compares well to the original SVM model whose confusion matrix was given in Table 2.

6. Conclusions

In the context of classifying cyberbullying comments, we introduced a KNN/SVM hybrid model. We have shown that we are able to reduce training data sizes with our KNNFilter algorithm by more than 50% and still obtain a high performing model. Most of the hybrid models built on a reduced data set had a cross-validated accuracy between 70% and 80% comparing favorably with the SVM model constructed on the full training data. That seems to indicate that our hybrid model approach works well even in the case of real world data.

The models obtained were reasonable in terms of false-positives and false-negatives and certainly could be deployed as a first-line automatic online moderation tool.

For future work, we would like to take a closer look at the ratio of support vectors actually captured in the reduced set produced by KNNFilter. Furthermore, we would like to extend the algorithm with ideas of handling outliers, especially around the decision boundary, making the generated reduced training data even more representative of support vectors.

References

[1] J. Platt *Sequential minimal optimization: A fast algorithm for training support vector machines*, " in *Advances in kernel methods - Support Vector Learning*, 1998.

[2] T. Cover, P. Hart. *Nearest neighbor pattern classification*, " in *IEEE Trans. Inf. Theor.*, 1967, vol. 13, no.1, pp21–27.

[3] K. Yu, L. Ji, X. Zhang *Kernel nearest-neighbor algorithm*, " in *Neural Processing Letters*, 2002, vol. 15, no. 2, pp. 147–156.

[4] H. Zhang, A C. Berg, M. Maire, J. Malik *SVM-KNN: Discriminative Nearest Neighbor Classification for Visual Category Recognition*," in *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2006

[5] S. S. Keerthi, S. K. Shevade, C. Bhattacharyya, K. R. K. Murthy *A fast iterative nearest point algorithm for support vector machine classifier design*," in *IEEE Transactions on Neural Networks*, 2000, vol. 11, no. 1, pp. 124-136.

[6] H. S. Ridha *Constructing Support Vector Classifier Depending On The Golden Support Vector*," in *Basrah Journal of Agricultural Sciences*, 2014, vol. 40, no. 2, pp. 68-94.

[7] X Jiantao, H. Mingyi, W. Yuying, F. Yan *A fast training algorithm for support vector machine via boundary sample selection*," in *International Conference on Neural Networks and Signal Processing*, 2003, vol. 1, pp. 20-22.

[8] H. Xiao, F. Sun *A Fast Learning Algorithm for SVM Based on K Nearest Neighbors*," in *Electronics, Optics and Control*, 2008, vol. 6.
[9] H. Xiao, F. Sun, Y. Liang *A Fast Incremental Learning Algorithm for SVM Based on K Nearest Neighbors*," in *International Conference on Artificial Intelligence and Computational Intelligence*, 2010, pp. 413-416.
[10] Y. Silva, "Bullyblocker: Towards the identication of cyberbullying in facebook." Arizona State University. [Online; accessed 2-January-2014]. Available: http://www.public.asu.edu/ynsilva/BullyBlocker/index.html
[11] J. Hirschberg, W. Warner, "Detecting Hate Speech on the World Wide Web." Columbia University. [Online; accessed 25-May-2014]. Available: http://aclweb.org/anthology//W/W12/W12-2103.pdf
[12] Twitch Interactive. "How to use automod." [Online; accessed 26-December-2016], December 2016. Available: https://help.twitch.tv/customer/portal/articles/2662186-how-to-use-automod
[13] M. Kamen. Wired. "Twitch now blocks trolls and hatespeech in real-time." [Online; accessed 3-February-2017], December 2016. Available: http://www.wired.co.uk/article/twitch-introduces-anti-troll-automod-for-game-streams
[14] L. Hamel *Knowledge Discovery with Support Vector Machines*, John Wiley & Sons Inc., Hoboken, New Jersey, 2009
[15] J. Patchin "Cyberbullying research: 2013 update," Cyberbullying Research Center. [Online; accessed 2-January-2014]. Available: http://cyberbullying.us/cyberbullying-research-2013-update/
[16] D. Ducharme "Machine Learning for the Automated Identification of Cyberbullying and Cyberharrasement," (2017) University of Rhode Island, Kingston, RI, [PhD Dissertation]. Available: http://pqdtopen.proquest.com/pubnum/10259474.html
[17] YouTube. "Popular right now," [Online; accessed 20-October-2015]. 2015. Available: https://www.youtube.com/playlist?list=PLrEnWoR732-BHrPp_Pm8_VleD68f9s14-
[18] D. Jurafsky, J. H. Martin, *Speech and Language Processing*, Pearson Education, Inc., Upper Saddle River, New Jersey, 2009
[19] C.C. Chang, C.J. Lin "LIBSVM – A Library for Support Vector Machines," (2012). [Online; 12-September-2012]. Available: http://www.csie.ntu.edu.tw/ cjlin/libsvm/
[20] M. Reed III, "Basic Concepts of Statistics - Class 23," (2010) University of North Carolina, Chapel Hill. [Online; accessed 1-October-2016]. Available: http://www.unc.edu/ rls/s151-2010/class23.pdf