Cyberbullying Detection using ML: A comparative study between supervised and semi-supervised methods

Dev Kathuria
Computer Science Engineering Student
Shiv Nadar University
email: dk382@snu.edu.in

***Abstract*—Bullying has been prevalent since the beginning of time, It's just the ways of bullying which have changed over the years, from physical bullying to cyberbullying. According to Williard(2004), there are eight types of cyberbullying such as harassment, denigration, impersonation, etc. It's been around 2 decades since social media sites came into the picture, but there hasn't been a lot of effective measures to curb social bullying and it has become one of the alarming issues in recent times. This paper does a comparative study of various Supervised and semi-supervised learning algorithms. The evaluation of the result shows that Ensemble supervised methods perform better than traditional supervised methods and the semi-supervised methods as well.**

***Index Terms*—Machine Learning, Cyberbullying, Supervised, Semi-supervised, Detection, Ensemble**

I. DATA

This section contains all the aspects of Data from collection to preprocessing and features extraction.

A. Data Collection

I have used a combination of two datasets to arrive at the final results, Initial plan was to use just the first dataset[1], which didn't give good results[refer Appendix A for details], So I decided to merge the two datasets, where the second dataset had a high percentage of cyberbullying instances.

Here is the Detailed Description of both the datasets.

1) Formspring Data for CyberBullying [1]
   a) Data crawled from Formspring.me, crawled in 2010, manually labelled by Amazon's Mechanical Turk
   b) Total Instances: 12852
2) Dataset for Detection of Cyber-Trolls [2]
   a) It is a partially manually labelled dataset.
   b) Total Instances: 20001
   The dataset just has 2 attributes- tweet and label[0 corresponds to No while 1 corresponds to Yes]

TABLE I
3 OCCURRENCES OF LABELLED FORMSPRING DATA

ANSWER	Yes or No whether it contains cyberbullying or not
CYBERBULLYINGWORK	Phrase(s) or word(s) identified by the worker which corresponds to cyberbullying
SEVERITY	0(NO Bullying) to 10

B. Data Cleaning

All the datasets were in CSV format and they had to be converted to ARFF, so that it could be used by Weka to run various ML classifiers, sadly integrated 'CSV to ARFF weka converter' and the online CSV to ARFF converters gave a lot of errors, so it had to be done manually.

I have advance experience in working with Excel, so I've used it for basic data cleaning and then using 'ATOM text editor', I added all the attribute relation and renamed it as a 'ARFF' file.

Various steps performed using Excel:

1) Used inbuilt excel command "clean()" to remove any non-printable characters
2) Used "IsBlank()" to search for any missing records and deleted any row which had any missing value
3) For ARFF format, all the text had to be converted to a string, I did that using format cell options by using a custom processor (\"@\")
4) Any extra newline character had to be removed, which was done using find and replace by using ASCII code for a line character. Moreover any extra characters which were easily noticeable such as 'Q:', 'A:',' <br>','@quot' etc were removed in the same way
5) using hit and trial, multiple other errors producing elements were later removed

TABLE II
INSTANCES

	Formspring	**Twitter**	**Merged**
Total Instances	10928	10708	19752
CB instances	220	2505	2725
Non-CB instances	10928	6312	17027

TABLE III
TEST AND TRAINING INSTANCES

	Test	Training
Total Instance	5926	13826
CB Instances	616	1909
Non-CB instances	5110	11917

C. Data Preprocessing

Since after manually converting data into ARFF format, I applied NumerictoNominal filter on our class to convert it to nominal, then the preprocessing steps were done as follows using the unsupervised StringtoVector filter[check Screenshot-preprocess filter for detailed configuration]:

1) Word Tokenization, It converts our text to separate words in a list, in which I used certain extra delimiters { , ; : ' " ( ) ? ! / - _ > < & # 1 2 3 4 5 6 7 8 9 0 } after going through the data, I also filtered out any numeric content as it doesn't contribute to cyberbullying.
2) Lowering Text using Lower case only tokens setting on Weka
3) Stop words and encoding cleaning using Weka's MultistopWords Algorithm
4) Stemming: Initially I didn't do stemming which produced bad results, after evaluating "CfsSubset Eval" and "InfoGainAttributeEval using Ranker for searching"[Appendix B], I used "Lovins Stemmer" for the same
5) Now after manually going through the attributes, all the unnecessary attributes which weren't filtered such as [$,=d,=p,*] etc, were removed manually
6) Now the next step was to extract features so that it can be used with ML algorithms, for which I used TF-IDF Transform using Weka. TF-IDF is a statistical measure to evaluate the relevance of a word, which is basically calculated by multiplying the number of times that words appeared in the document by the inverse document frequency of the word

D. Data Resampling

As the data was skewed, Resampling had to be performed on the training data, Firstly the data was split into Training and Test in 70:30 ratio and resampling was performed on the training data. Note: I did a slightly different kind of resampling for Supervised and semi-supervised learning(More details are given later).

1) Supervised Learning:

- As I had ample data to work with(using Merged dataset), I used Undersampling of majority class followed by oversampling of the minority class
- For Oversampling, Supervised Resample filter is used with a bias towards uniform class as 1(100%) and noReplacement set as false
- For Undersampling, the same filter is used with noReplacement set as true

After resampling, training data had 4411 CB & NON-CB instances

2) SemiSupervised Learning: Since Semi-Supervised learning has to have an unlabelled dataset, along with test and training datasets, I decided to use the instances which were labelled as invalid during the cleaning for Formspring dataset (around 2000 instances) as the unlabelled dataset, since Weka's collective semi-supervised framework still has a lot of bugs [Refer screenshot err,err2], I couldn't use the separate unlabelled dataset. To tackle this problem, I used 10 fold cross-validation to estimate the model. It works by automatically dividing the datasets and computing the results for every fold, resulting in an averaged output.

Since we are using cross-validation, we have to carefully resample the data to prevent the model from showing false accuracy. Firstly, I computed the model using unsampled dataset and then I just undersampled the majority class till it was nearly balanced by repeatedly using the beforementioned undersampling filter.

Since Initially, the plan was to use it along with the unlabelled dataset, it had a few extra attributes(Total: 1509 attributes) as compared to the datasets used in supervised learning(Since those attributes have 0 weights in the training dataset, they don't really matter and it is more or less similar to the dataset used in supervised learning)

TABLE IV
SEMI-SUPERVISED LEARNING DATASET

	NoSampling	Sampling
Total Instances	13826	3844
CB instances	1883	1883
Non-CB instances	11943	1961

II. EXPERIMENTAL RESULTS

For supervised learning, I've used NaiveBayes, SVM, J48 and Ensemble methods- AdaBoost-NaiveBayes, AdaBoost-SVM, Bagging-j48, RandomForest. Where Naive Bayes Performed the poorest and RandomForest gave the best result in terms of every metric.[fig1 & fig2] For semisupervised learning, I've used YATSI and Collective IBk for unbalanced and balanced datasets, where the result of the unbalanced dataset came out to be bad as expected while the results of Unbalanced dataset are still considerable, giving an average accuracy of around 63.5%, thus performing poorly in comparison to supervised methods

Metrics used are:

- ROCArea, which denotes the area under the curve formed by plotting TP rate.
- $$F - Measure = \frac{(2 \times Precison \times Recall)}{Precision + Recall}$$
- $$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$
- $$Precision = \frac{TP}{TP + FP}$$
- $$Recall = \frac{TP}{TP + FN}$$

1) Supervised: Traditional Supervised Learning used: NaiveBayes, SVM and J48

Ensemble Learning Methods used: AdaBoost-NaiveBayes, AdaBoost-SVM, Bagging-j48 and RandomForest

Figure 1 and Figure 2 shows a graphical comparison between the aforementioned algorithms

TABLE V
SUPERVISED TRADITIONAL METHODS

	NaiveBayes	SVM	J48
ROCArea	0.693	0.721	0.722
F-Measure	0.648	0.763	0.775
Precision	0.827	0.852	0.840
Recall	0.584	0.723	0.741
Confusion Matrix	2874 2236 231 585	3699 1411 230 586	3881 1229 303 513

TABLE VI
SUPERVISED ENSEMBLE METHODS

	AdaBoost NaiveBayes	AdaBoost SVM	Bagging J48	Random Forest
ROCArea	0.745	0.731	0.802	0.803
F-Measure	0.749	0.764	0.781	0.809
Precision	0.842	0.837	0.845	0.841
Recall	0.707	0.727	0.748	0.789
Confusion Matrix	3635 1475 263 553	3793 1317 302 514	3906 1204 288 528	4221 889 360 456

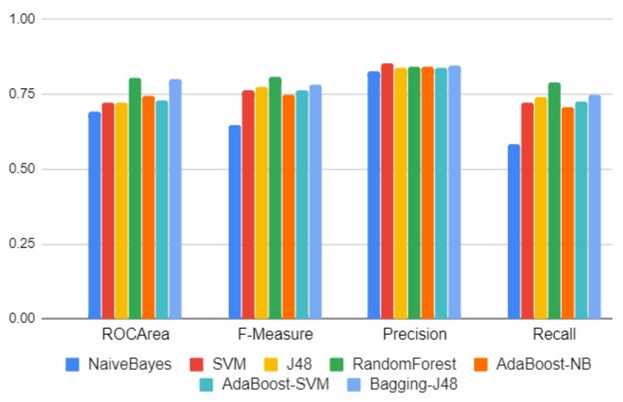

Fig. 1. ROCArea, Fscore, Precision, Recall

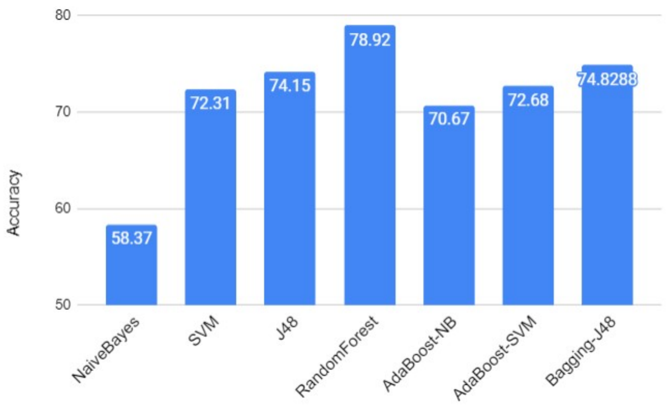

Fig. 2. Accuracy

2) Semi-Supervised: Algorithms Used:

1) YATSI
"Yet another Two-Stage Idea" is a collective classifier that uses the given classifier to train on the training set and labelling the unlabelled dataset, As a classifier, I chose J48 decision trees to generate a pruned or unpruned C4.5 decision tree. Predictions are then done by k nearest neighbours, for which I chose the KDTree search algorithm that uses Euclidean distance as a distance function. [3]
2) Collective IBk
It uses IBk to determine the optimal K for the neighbourhood on the training set. This K is used to build for each instance of the test set a neighbourhood, consisting of instances from the test AND training set.
Majority vote is used to determine the class label for an instance, based on its neighbourhood (in case of ties the first encountered class is taken). Sooner or later all labels of the test set are determined.
An instance that is presented for classification is

TABLE VII
SEMI-SUPERVISED METHODS

	UNBALANCED		**BALANCED**	
	YATSI	IBk	YATSI	IBk
	(Weighted, average, Value for CB class)		*(Weighted Average)*	
ROCArea	0.73, 0.73	0.59,0.59	0.676	0.653
F-Measure	0.82, 0.19	0.81,0.30	0.62	0.650
Precision	0.80, 0.36	0.81,0.31	0.62	0.657
Recall	0.85, 0.12	0.81,0.30	0.625	0.652
Accuracy	84.5	81.5	62.5	65.1
Confusion Matrix	11513 430 1645 238	10685 1258 1312 571	1132 829 611 1272	1135 826 512 1371

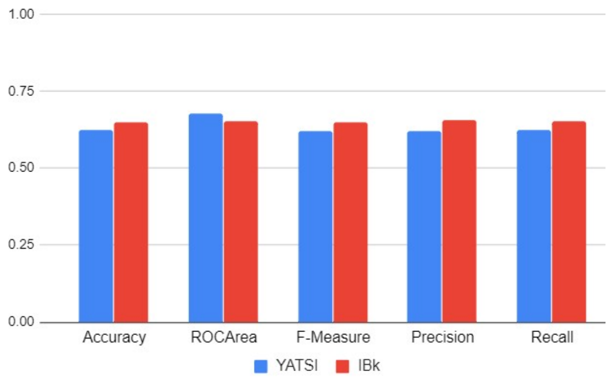

Fig. 3. Yatsi and IBK on Balanced Dataset

then only looked up and the determined label is returned [4]

Running the classifiers on the Unbalanced dataset gave dissatisfying results, it looks like they have high accuracy, which is just a result of base rate fallacy, I've specifically mentioned individual scores for the Cyberbullying class for the unbalanced dataset in table VII so that it becomes clear that running the classifiers on unbalanced dataset gives a poor result, with high false-positive rate.

Running both the classifiers on the undersampled data, provided much better output, and since, the weighted average didn't have a lot of deviation from the scores of either of the classes, I've just mentioned the weighted average scores for both of them The average accuracy came out to be about 63%, which in comparison to the Supervised algorithms is poor

III. RELATED WORK

For Detecting Cyberbullying, numerous approaches have been developed, majorly using Natural Language Processing and Information Retrieval which are then used to classify textual data by extracting it's features by using TF-IDF, Sentiment Analysis, Dimensionality Reduction etc. and they have recieved commendable accuracies

Sambhagadi et al. [5] Detects nastiness using NLP techniques, they use a list of profanities as an extra feature along with analyzing the context as in which they are used, whether they are insulting or neutral in nature. They used modified linear SVM to classify the ASK.FM data, they even took the emojis into the picture. They got an F1 score of 0.59.

Yao et al. [6] acknowledges the repetitive nature of cyberbullying on social media i.e a sequence of aggressive messages sent from bully to a victim with the intent of harm, previous paper, and multiple other papers used profanity to classify toxic comments as cyberbullying, disregarding its repetitive process. This paper uses sequential hypothesis testing formulation to drastically reduce the number of features used in classification, while still maintaining high accuracy.

Nandhini et al. [7] Proposes a naive Bayes based learning model and used the dataset of MySpace.com, they achieved a high accuracy of 91%. [8] uses Formspring data, available at Kaggle.com by Kelly Reynolds which initially had about 12000 instances, but after preprocessing, they got a total of 1608 instances where half of them corresponds to cyberbullying, basically, they used this small dataset to train a Neural network and an SVM Classifier.

TABLE VIII
RESULTS OF [8] (IN %)

Classifier	Average Accuracy	Average Recall	Average Precision	Average F-score
Neural Network	91.76	91.7	92.4	91.9
SVM	89.87	90.1	89.6	89.8

[9] uses Logistic Regression and SVM to train on cyberbullying data

TABLE IX
RESULTS OF [9] (IN %)

Classifier	Average Accuracy	Average Recall	Average Precision	Average F-score
Logistic Regression	73.76	61.47	64.4	62.9
SVM	77.65	58.29	70.29	63.7

Study by Huang et al. [10] focuses on analyzing the social network structure between users and deriving features such as a number of friends, network embeddedness, and relationship centrality, by integrating textual features with social network features, detection of cyberbullying can be achieved. The study claims that past researches haven't fully utilized social media features, this paper proposes cyberbullying detection beyond textual analysis to also consider the social relationships in which these bullying messages are exchanged. It uses twitter corpus and uses Weka 3.0 to implement it with different classification algorithms such as Naive Bayes, J48, SMO, Bagging and dagging.

TABLE X
RESULTS OF [10]

	ROC	TP
Bagging	0.700	0.211
J48	0.628	0.259
SMO	0.703	0.733
Dagging	0.755	0.763
Naive Bayes	0.695	0.723

The results of this study is very much comparable to my work as well, as I got Similar values for those metric as well, Furthermore, Bagging using J48 as base and RandomForest Classifier got even better results than this.

The best result in [10] is recieved using Dagging, giving a ROC value of 0.755, which when compared to Bagging(ROC: 0.802) and RandomForest(ROC: 0.803) is less.

[11], This paper uses Formspring dataset available on Kaggle, it trains classifiers and Neural networks, firstly on the Unbalanced Dataset and then on the Balanced Dataset, Similar to what I did for semi-supervised learning, This paper achieves the best result using Twitter Embeddings along with C-LSTM (Kim et al 2015) and achieves a Precision value of 0.485, Recall of 0.448 and F-measure of 0.444. It again obtains the best result using the same architecture on the balanced dataset:(Refer table X)

TABLE XI
RESULTS OF [11] IN COMPARISON TO SEMI SUPERVISED METHODS TESTED ON BALANCED DATA IN MY PAPER

	Precision	Recall	F-measure
Rosa et al [11]	0.845	0.842	0.842
Semi-supervised (Balanced data)	0.62	0.625	0.62

IV. CONCLUSION

In this paper, I did a comparative study between various Supervised and semi-supervised algorithms, Additionally, I compared various Supervised Ensemble methods as well. The overall best performance was shown by RandomForest classifierAll other Ensemble methods performed equally or better than the Supervised methods but still, I observed a high True positive rate for the cyberbullying class in all the ensemble methods, which is much more desirable. NaiveBayes performed the worst, giving just 58% accuracy.

I used 2 semi-supervised Learning algorithms with cross-validation on Balanced and Unbalanced datasets, Both of them performed much better on the Balanced dataset, giving approximately the same performance. Average accuracy for Semi-Supervised methods came out to be approximately 63.5%, which is better than NaiveBayes but worse than all other tested algorithms

In this paper, I evaluated my approach and compared it with other papers in the section “Related Work” I also observed that none of the studied past researches used any semi-supervised methods, probably because they are not that popular or effective as in our case they didn’t give any commendable result in comparison to the supervised methods.

REFERENCES

[1] ”Formspring Data for Cyberbullying Detection.” Kaggle: Your Machine Learning and Data Science Community, www.kaggle.com/swetaagrawal/formspring-data-for-cyberbullying-detection. Accessed 19 Mar.

[2] Dataturks – Online Tool to Build Image Bounding Box, NER, NLP and Other ML Datasets, dataturks.com/projects/abhishek.narayanan/Dataset%20for%20 Detection%20of%20Cyber-Trolls.

[3] ”A Collective Learning Approach for Semi-Supervised Data Classification.” ResearchGate, 1 Jan. 2018, www.researchgate.net/publication/328358286_A_Collective_Learning_Approach_for_Semi-Supervised_Data_Classification. Accessed 22 Mar.

[4] ”Fracpete/collective-classification-weka-package.” GitHub, 7 Aug. 2019, github.com/fracpete/collective-classification-weka-package.

[5] Samghabadi, Niloofar Safi, et al. ”Detecting nastiness in social media.” Proceedings of the First Workshop on Abusive Language Online. 2017.

[6] Yao, Mengfan, Charalampos Chelmis, and Daphney? Stavroula Zois. ”Cyberbullying ends here: Towards robust detection of cyberbullying in social media.” The World Wide Web Conference. 2019.

[7] Nandhini, B. Sri, and J. I. Sheeba. ”Cyberbullying detection and classification using information retrieval algorithm.” Proceedings of the 2015 International Conference on Advanced Research in Computer Science Engineering Technology (ICARCSET 2015). 2015.

[8] Hani, John, et al. ”Social Media Cyberbullying Detection using Machine Learning.”

[9] Vikas S Chavan and SS Shylaja. Machine learning approach for detection of cyber-aggressive comments by peers on social media network. In Advances in computing, communications and informatics (ICACCI), 2015 International Conference on, pages 2354–2358. IEEE, 2015.

[10] Huang, Qianjia, Vivek Kumar Singh, and Pradeep Kumar Atrey. ”Cyberbullying detection using social and textual analysis.” Proceedings of the 3rd International Workshop on Socially-Aware Multimedia. 2014.

[11] Rosa, Hugo, et al. ”A “deeper” look at detecting cyberbullying in social networks.” 2018 International Joint Conference on Neural Networks (IJCNN). IEEE, 2018.

Appendix A
Experiments on the Formspring Dataset

1) Cleaning and Processing was done in the same way as mentioned in the paper
2) Original Dataset had 10708 NON-CB instances and 220 CB instances
3) After Initial preprocessing, it had 1524 features
4) Data was then split in the ratio of 70:30 for training and testing
5) Training data had 7495 NON-CB instances and 155 CB instances
6) Testing data had 3213 NON-CB instances and 65 CB instances
7) Resampling Training data using Weka's supervised Resampling filter resulted in 3825 CB and NON-CB instances
8) Result
    a) I ran SVM on the unbalanced dataset first

TABLE XII
METRICS FOR CB CLASS

FP rate	0.033
Precision	0.125
F-Measure	0.162

These results are obviously very poor and give a false idea of accuracy which came out to be 95%

    b) Then I resampled the data and ran Naive Bayes, Adaboost with Cost-sensitive Classifier and Adaboost with Decision Stump as the base classifier and the results came out to be better but they were not satisfactory. I'll just mention the metric for CB class since that's what matters
    i) Naive Bayes [Cost sensitive ADaboost had similar results as well, So I won't mention them]

TABLE XIII
CONFUSION MATRIX FOR NAIVEBAYES

a	b
2054	1159
29	36

Accuracy : 63%, FP rate 0.361,Precision:0.030, ROC Area: 0.673, F-measure: 0.057
ii) Adaboost with Decision Stump
Accuracy : 84%, FP rate 0.152,Precision:0.065, ROC Area: 0.728, F-measure: 0.116

9) Because of these poor results, I decided to merge my dataset with the labelled Twitter dataset

TABLE XIV
CONFUSION MATRIX FOR ADABOOST WITH DECISION STUMP

a	b
2724	489
31	34

Appendix B
Attribute Evaluation

1) Not stemming initially and not using proper delimiters didn't give good results which were evident just by just looking at the attributes (for eg: /, //www, @#039 etc) and there were numbers too, still, I did attribute evaluation using CfsSubset Eval and also by InfoGainAttributeEval using Ranker, and it became clear that preprocessing was not done well, but all of it was solved by doing the necessary changes [check the preprocess filter2 for detailed configuration]
*Note: All detailed Result buffers are attached
2) Top raked words for Merged Dataset: [Fuck, what, do, bitch, as, ever, gay, would, or, hat, suck], it was a significant improvement since without stemming, few of the top-ranked words were just random characters or numbers.
3) Top ranked words for Semi-supervised Datasets: [fuck, what, ever, do, gay, bitch, favor, suck]
4) Top few ranked words for Formspring dataset: [bitch, fuck, ugly, shit, in what, tin, fak]
*Note: Check Appendix A for results with this dataset

Appendix C
Semi-Supervised Learning on Less Skewed Distribution

I ran semi-supervised YATSI algorithm on less skewed data distribution as well. Since the results were comparable to the balanced dataset, I chose to mention it in the appendix.

a) Data with 2511 Non-Cyberbullying instances & 1883 Cyberbullying instances

TABLE XV
RESULT FOR DATA I

Accuracy	ROC Area	F-measure	Precision	Recall
63%	0.681	0.632	0.639	0.631

b) Data with 4398 Non-Cyberbullying instances & 1883 Cyberbullying instances

TABLE XVI
RESULT FOR DATA II

Accuracy	ROC Area	F-measure	Precision	Recall
67.8%	0.708	0.686	0.701	0.678