Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 132 (2018) 174–182

Procedia Computer Science

www.elsevier.com/locate/procedia

International Conference on Computational Intelligence and Data Science (ICCIDS 2018)

N-Gram Assisted Youtube Spam Comment Detection

Shreyas Aiyar[a], Nisha P Shetty[b]

[a]BTech Student,Department of Information and Communication Technology Manipal Institute of Technology Manipal Academy of Higher Education Manipal -576104 India

[b]Assistant Professor, Department of Information and Communication Technology Manipal Institute of Technology Manipal Academy of Higher Education Manipal -576104 India

Abstract

This paper proposes a novel methodology for the detection of intrusive comments or spam on the video-sharing website - Youtube. We describe spam comments as those which have a promotional intent or those who deem to be contextually irrelevant for a given video. The prospects of monetisation through advertising on popular social media channels over the years has attracted an increasingly larger number of users. This has in turn led to to the growth of malicious users who have begun to develop automated bots, capable of large-scale orchestrated deployment of spam messages across multiple channels simultaneously. The presence of these comments significantly hurts the reputation of a channel and also the experience of normal users. Youtube themselves have tackled this issue with very limited methods which revolve around blocking comments that contain links. Such methods have proven to be extremely ineffective as Spammers have found ways to bypass such heuristics. Standard machine learning classification algorithms have proven to be somewhat effective but there is still room for better accuracy with new approaches. In this work, we attempt to detect such comments by applying conventional machine learning algorithms such as Random Forest, Support Vector Machine, Naive Bayes along with certain custom heuristics such as N-Grams which have proven to be very effective in detecting and subsequently combating spam comments.

Keywords: Spam; Youtube; N-Gram; Naive-Bayes; Random Forest; Support Vector Machine; Word Gram; Character Gram;

1. Introduction

Youtube is a video sharing website that was started back in 2005. It was subsequently bought by Google in 2006 and is now one of Google's subsidiaries. Since then Youtube has emerged as a leading contender in the video sharing space. Users of Youtube are known as channels, and Youtube allows channels to upload, rate, share, add to favorites,

* Corresponding author. Tel.: +91-8217597204 ;
E-mail address: shreyas.aiyar96@gmail.com

10.1016/j.procs.2018.05.181

report, comment on videos, and subscribe to other users. Recently as of late 2017, Youtube generates more than 400 hours of content per minute and more than 1 billion hours of content are consumed by users daily [1]. The website has been ranked as the second most popular website in the world by Alexa Internet - a web traffic analysis company.

One of the most utilised features of Youtube is its commenting system where users can comment on videos uploaded to other channels. This feature allows users to interact with one another and share their opinions, feelings, etc. on the video. However, this has also turned into an opportunity for malicious users to share promotional content also known as spam. Spam comments are often wholly irrelevant to the given video and are usually generated by automated bots disguised as a user. The ability of such bots to perform spam campaigns - large scale orchestrated posting of malicious comments has been explored in [2].

Recently as of November 2017, Youtube has faced increasing criticism about its inability to moderate uploaded content [3]. A large user-base of Youtube consists of children who are often exposed to malicious and harmful material in the form of comments.

1.1. Types Of Spam On Youtube

We can classify the majority of comment spam on Youtube as one of the following types. The table 1 below enumerates the type of spam along with examples for the same.

1.1.1. Link Based Spam

This is a very common form of spam often seen on Youtube. Comments contain Hypertext (HTTP) links to other websites, usually other videos on Youtube itself. Many links redirect the the user to potentially malicious webpages often without the knowledge of the user.

1.1.2. Channel Promotional Spam

This is the most prevalent form of spam on Youtube. These types of comments usually consist of users who attempt to promote their own channel by requesting for subscribers, posting links to their videos etc.

Table 1. Types Of Spam

Nature	Example
Link Based	**make your iPhone 6/6s happy http://www.ebay.com/itm/272739565815?ul_noapp=true**
	I HAVE SOME THING OWSOME FOR YOU , I'M SURE YOU LIKE IT,IT'S OWSOME
Channel Promotion	**Pls some one help me with my channel like can anyone just help by giving me a few subs am trying to come back**
	Go checkout my channel and subscribe I will subscribe back

1.2. Drawbacks Of Current Methods

Youtube has attempted to combat link-based spam by blocking all comments containing Hypertext links (HTTP)[4]. Although effective, this form of filtering has had it's share of problems as it has led to spammers resorting to more creative techniques such as inserting whitespace characters between links to avoid detection. An example of a such a comment is as follows.

This song is really amazing and i bet JB still busy using authentic views dot c0m and reaching millions of views

Such types of comments are inherently designed to bypass Youtube's link detection filter. Most link filters and blacklists prove ineffective against such obfuscated comments.

1.3. Need For A Better Method

There remains a need for a technique that can accurately classify a comment as spam or ham solely based on its content while simultaneously remaining resistant to transformations. This proves to be a difficult computing problem due to the ambiguous nature of language. The increase in computing power in the recent years have paved the way

for applying Machine Learning techniques to solve such problems. Our contribution in this direction is to attempt to identify algorithms and apply heuristics such as N-Grams that can accurately detect spam with a high F1 Score and hence improve the accuracy of the classifier.

The organisation of the rest of the paper is as follows. Section II considers related work in the field. In section III Proposed Methodology for N-Gram assisted Spam-Detection is discussed. The Results are elaborated in section IV, and finally, Conclusions & Future Work is presented in section V.

2. Related Work

Alex Kantchelian et al. [5] developed a spam detection technique which can quantify fruitless and superfluous features in blogs, making meaningful stories more accessible to the perpetual stakeholders. They suggested extension of their work to broaden the definition of spam such as URLs, short message removal, etc. in addition to inclusion antagonist awareness, online deployment to enable prediction of futuristic comments and so on.

Enhua Tan et al. [6] designed a runtime spam detection scheme called as BARS: Blacklist-Assisted Runtime Spam Detection which constructed a database of Spam URLs against which URL of every new post was analysed to determine if the post was spam or not. Clustering of User IDS based on shared URLs also increased the effectiveness of detection. However, the efficacy of this approach is a consequence of how successfully the blacklisted URL list is fostered.

Seungwoo Choi et al. [7] experimented their algorithm on Ted-Talks videos to find the comment offering coverage and information about the video contents. Nonetheless, the proposed method was found inadequate in analyzing the feelings and opinions expressed in platforms such as YouTube.

Igor Santos et al. [8] applied the concept of anomaly detection wherein the deviation from authentic emails was used as a metric to classify emails as spam or ham. Better accuracy was achieved owing to the limited training sets as seen in labelling based systems.

M. McCord et al. [9] harnessed machine learning algorithms which were trained with content and user-centred facets to identify spammers. They tested their algorithms with twitter data and discovered that the Random Forest Classifier offered the best results.

Web Spammers usually aim to increase their revenue by redirecting users to visit their sites, or to spread malignant content via the installation of malevolent software. Authors Shekoofeh Ghiam et al. [10] studied the various web-spam practices and relevant exposure methods.

Authors Wojciech Indyk et al. [11] administered Map reduce Algorithm to effectively discover spam masses is their research.

The authors, Qingxi Peng et al. [12] employed the concepts of sentiment analysis to detect spam reviews. They concluded by proposing some improvements in the calculation of sentiment score which was the base of their paper.

3. Methodology

In this section, we present an overview of the process used to classify the comments as Spam or Ham as well give a breakdown of the dataset collected and trained upon.

3.1. Overview

Figure 1 shows a concise flowchart of the entire process. We used the public Youtube API to collect our comments and manually labelled them as spam or ham. This forms our dataset which is fed into a Bag Of Words (BoW) vectorizer which transforms words into numerical features (numpy arrays) for training and testing. The transformed dataset is split into training and testing subsets and fed into Naive Bayes, Random Forest and Support Vector Machine pipelines respectively. All implementation was performed on Python 2.7 using the *scikit − learn* library.

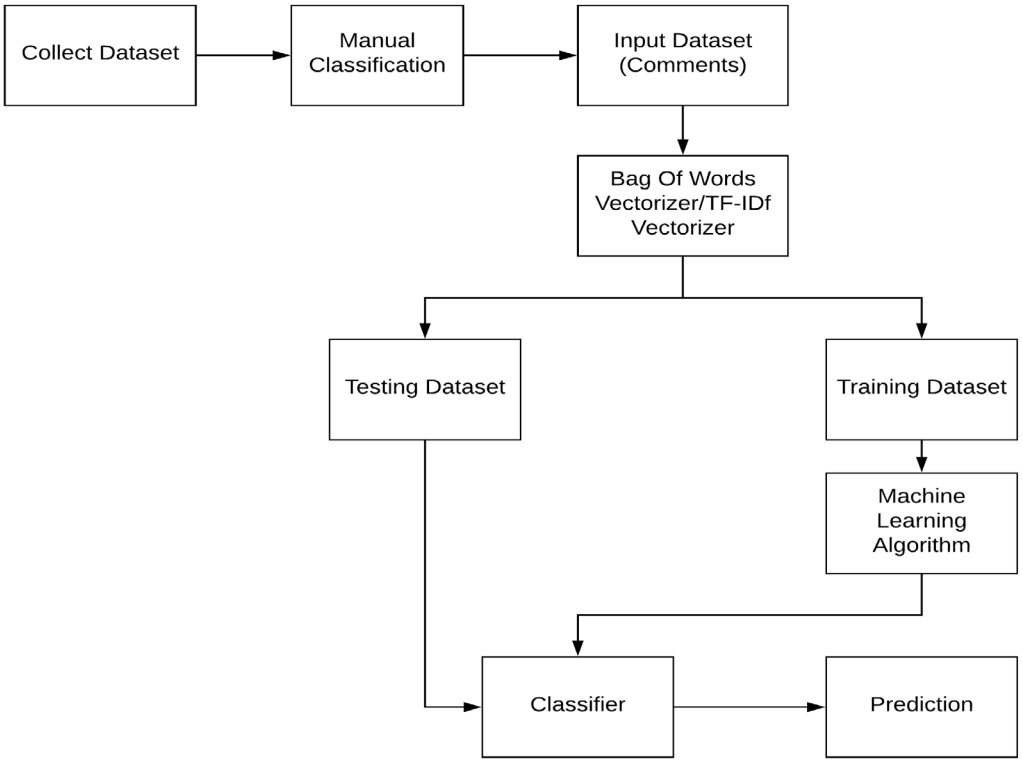

Fig. 1. Flowchart

3.2. Dataset Analysis

We manually extracted around 13000 comments from various channels across Youtube using the public Youtube API [13] and stored them in a database for subsequent analysis. In particular trending music videos with a significantly large view count were targeted. Thus our sample consists of comments from the most popular videos at the time. A simple library filter was used to detect and extract comments with only Latin letters since the aim was to evaluate English language comments. Owing to the relatively low ratio of ham to spam comments a basic hand engineered spam filter was used to extract potentially spammy comments. The hand engineered spam filter consists of a set of simple regular expressions which contains elements of spam based comments. Thus our final dataset contains a relatively equal ratio of both spam and ham. We manually hand labeled comments (i.e assigned a value of 0/1) which were promotional in nature or out of context with the given video and classified them as spam.

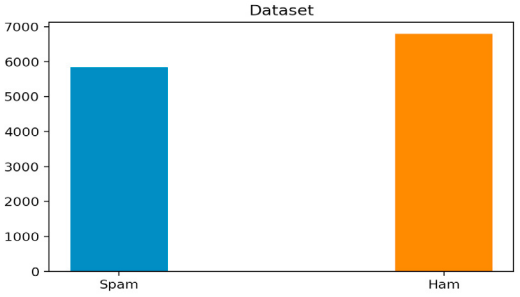

Fig. 2. Dataset

3.3. N-Gram Analysis

A significantly thorough analysis has been performed in [14] where the effectiveness of using n-grams has been demonstrated in the case of email-filtering. We attempt to extend a similar methodology in the case of Youtube comments.
An n-gram is a sequence of n characters or words. In case of a character-gram of size m we extract $m + n - 1$ substrings of consecutive characters. We first define a value for n and extract from each document in the training corpus(here a comment), the frequently occurring n-grams. By performing the same operation over all the comments in the collection, we can obtain a vector L

$$L < x_1, x_2, x_3, x_4 ..... x_L > \tag{1}$$

Let g_i correspond the the n-gram.The values of $x_1, x_2, x_3, x_4 ..... x_L$ correspond to one of the following approaches to convert n-grams into feature vectors for training.

- Term-Frequency Inverse Document Frequency(TF-IDF)
- Bag Of Words(BoW)
- Word2Vec

The TF-IDF model assigns a score to g_i based on the frequency of its occurrence in each document (here a comment) which is attenuated by a factor of its document frequency df_t to scale down its significance.

$$TF - IDF_{t,d} = tf_{t,d} \times \log \frac{N}{df_t} \tag{2}$$

The Bag Of Words model instead simply uses a term-frequency approach to vectorize g_1. A binary value of 1 is assigned to x_i if the comment contains g_i else a value of 0 is assigned.

$$tf_{t,d} = \begin{cases} 1, & t \\ 0, & otherwise \end{cases} \tag{3}$$

A significant consideration in the N-Gram approach is the selecting an appropriate value of n. Lower values of n significantly increase the dimensionality of the problem thus affecting the performance of certain algorithms. Higher values can significantly increase the training time.

In this study, we have opted to use the Bag Of Words (BoW) approach since in the case of classification we would not like to down weight the frequency of commonly occurring words. Further, we do not apply any form of conventional preprocessing techniques such as stemming, lemmatization, stop word removal. As is pointed out in [15] not all of the steps are mandatory. This is further corroborated by a study performed on a Twitter dataset [16] which concludes that the use of a pre-compiled stop-list negatively affects the performance of the classifier.

3.4. Algorithms

We performed training and classification using Multinomial Naive Bayes (M-NB), Random Forest (RF) & Support Vector Machine (SVM) respectively. Naive Bayes was chosen as the standard for our study due to it's simplicity and high efficiency [17]. Support Vector Machines & Random Forests have been proven by multiple studies as performing well for classification problems [18]. SVM's in particular are especially well suited for high dimensional datasets which are often seen in the case of Natural Language Processing tasks.
The Table 2 depicts the algorithm and respective hyperparameter values used in the study. No significant hyperparameter tuning was performed as we are more interested in the gain leveraged by using n-grams.

Table 2. Hyperparameter Values

Classifier	Hyperparameters	Value
Multinomial Naive Bayes(MNB)	alpha	1
	fit_prior	true
Random Forest(RF)	n_estimators	10
	max_depth	None
Support Vector Machine(SVC)	C	1.0
	kernel	rbf
	degree	3

4. Results

We evaluate the performance of our spam comment detection system using the cross validation and k-fold approach. The dataset is shuffled using a random number. A five-fold cross-validation procedure was used. The entire dataset was divided into five equal parts, in each fold a different part is used as the test set and the remaining parts as training set. The final F1 Score is obtained from averaging the results of each fold. The algorithm is trained over the training subset and the accuracy of the classifier is cross validated with the testing subset.

4.1. Evaluation Metrics

We used Accuracy, Precision, Recall and F1 Score as metrics for evaluation. The performance of the algorithm is however, judged using the F1 score since we would like to obtain both a high precision as well as a high recall.

$$Precision = \frac{TP}{TP + FP} \tag{4}$$

$$Recall = \frac{TP}{TP + FN} \tag{5}$$

$$F1_{Score} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \tag{6}$$

Where TP, TN, FP, FN denote the true positive, true negative, false positive, false negative rates respectively.

4.2. N-Gram Performance Evaluation

6 sets of experiments were performed with n values ranging from 1 to 6 for both character and word grams. In all cases F1 score was used to evaluate the performance of the classifier. The results of the experiment are shown in the Support Vector Machine(SVM) with an n value of 6 and using character-gram produces the highest F1 score of 0.984. Word-grams outperform character-grams at lower values of n. However, we see a significant increase in the F1 Score at higher values of n in the case of character-grams yielding improvements of nearly 1% over word-monograms. Naive Bayes consistently performs the worst across the 3 algorithms while Random Forests interestingly does not seem to be significantly affected by n values or the usage of n-grams.

Figure 3 shows a graphical representation of the F1 Score(Y-Axis) against tested values of n (X-Axis). The score for character-grams increases significantly at n values of 2 and 3 and peaks at an n value of 6. Correspondingly the score for word-grams decreases significantly at an n value of 3 and remains the highest at an n value of 1.

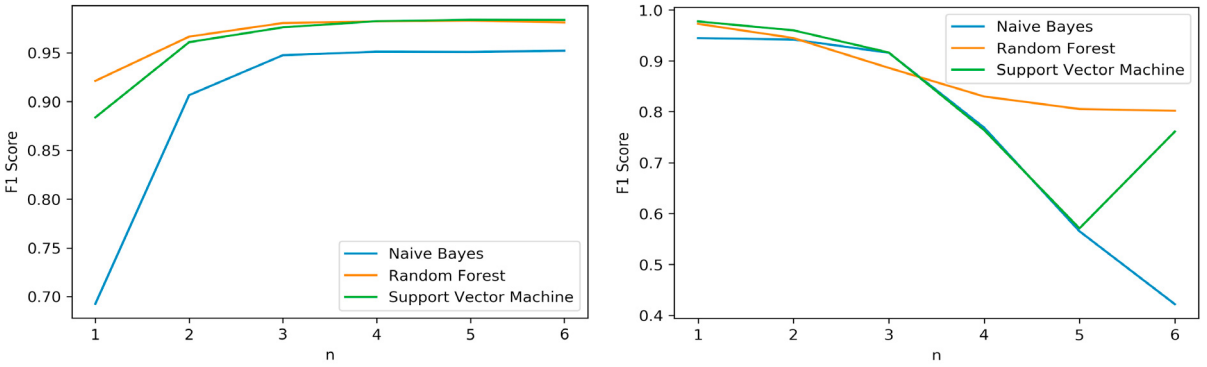

Fig. 3. (a) Char-Gram Plot (b) Word-Gram Plot

The F1 Scores for the 6 n values for character grams and word grams can be seen in Table 3 and Table 4 respectively below.

Table 3. F1 Scores for Character Grams

Classifier	$n = 1$	$n = 2$	$n = 3$	$n = 4$	$n = 5$	$n = 6$
Multinomial Naive Bayes	0.6927	0.9058	0.9470	0.9510	0.9506	**0.9521**
Random Forests	0.9203	0.9665	0.9795	0.9830	0.9821	**0.9813**
Support Vector Machine	0.8690	0.9607	0.9775	0.9818	0.9832	**0.9841**

5. Conclusion & Future Work

In this paper, we have presented a method for automated machine assisted detection of spam comments on the Youtube platform and have highlighted the effectiveness of using character-grams(substrings of n characters) over

Table 4. F1 Scores for Word Grams

Classifier	$n = 1$	$n = 2$	$n = 3$	$n = 4$	$n = 5$	$n = 6$
Multinomial Naive Bayes	**0.9444**	0.9419	0.9158	0.7687	0.5652	0.4220
Random Forests	**0.9726**	0.9447	0.8861	0.8299	0.8053	0.8019
Support Vector Machine	**0.9774**	0.9599	0.9163	0.7639	0.5708	0.7609

word-grams in improving the accuracy of the classification. We see that character-grams are better able to identify the measure of spamminess in the comment as opposed to word-grams.

As corroborated by [19], Support Vector Machines & Random Forests indeed outperform other traditional Machine Learning algorithms for classification and are very well suited for high dimensional datasets.

For future work, with the advent of neural network based implementations of embedded word vectors (such as Word2Vec) [20] we may obtain better word representations which can produce better models.

Acknowledgments

I am very grateful to Ms. Nisha P Shetty, Assistant Professor, Manipal Institute Of Technology, Manipal for her invaluable guidance, inspiration and constructive suggestions that helped me in the preparation of this paper.

References

[1] Youtube, Youtube Press Statistics, https://www.youtube.com/yt/about/press/.
[2] D. O'Callaghan, M. Harrigan, J. Carthy, P. Cunningham, Network analysis of recurring youtube spam campaigns, CoRR abs/1201.3783.
[3] BBC, Glitch in YouTube's tool for tracking obscene comments), http://www.bbc.com/news/blogs-trending-42060357.
[4] Youtube, Youtube Moderate & review comments, https://support.google.com/youtube/answer/111870?hl=en.
[5] A. Kantchelian, J. Ma, L. Huang, S. Afroz, A. Joseph, J. D. Tygar, Robust detection of comment spam using entropy rate, in: Proceedings of the 5th ACM Workshop on Security and Artificial Intelligence, AISec '12, ACM, New York, NY, USA, 2012, pp. 59–70. doi:10.1145/2381896.2381907.
URL http://doi.acm.org/10.1145/2381896.2381907
[6] E. Tan, L. Guo, S. Chen, X. Zhang, Y. Zhao, Spammer behavior analysis and detection in user generated content on social networks, in: 2012 IEEE 32nd International Conference on Distributed Computing Systems, 2012, pp. 305–314. doi:10.1109/ICDCS.2012.40.
[7] S. Choi, A. Segev, Finding informative comments for video viewing, in: 2016 IEEE International Conference on Big Data (Big Data), 2016, pp. 2457–2465. doi:10.1109/BigData.2016.7840882.
[8] I. Santos, C. Laorden, X. Ugarte-Pedrero, B. Sanz, P. G. Bringas, Spam Filtering through Anomaly Detection, Springer Berlin Heidelberg, Berlin, Heidelberg, 2012, pp. 203–216. doi:10.1007/978-3-642-35755-8_15.
URL https://doi.org/10.1007/978-3-642-35755-8_15
[9] M. McCord, M. Chuah, Spam Detection on Twitter Using Traditional Classifiers, Springer Berlin Heidelberg, Berlin, Heidelberg, 2011, pp. 175–186. doi:10.1007/978-3-642-23496-5_13.
URL https://doi.org/10.1007/978-3-642-23496-5_13
[10] S. Ghiam, A. N. Pour, A survey on web spam detection methods: Taxonomy, CoRR abs/1210.3131. arXiv:1210.3131.
URL http://arxiv.org/abs/1210.3131
[11] W. Indyk, T. Kajdanowicz, P. Kazienko, S. Plamowski, Web Spam Detection Using MapReduce Approach to Collective Classification, Springer Berlin Heidelberg, Berlin, Heidelberg, 2013, pp. 197–206. doi:10.1007/978-3-642-33018-6_20.
URL https://doi.org/10.1007/978-3-642-33018-6_20
[12] Q. Peng, M. Zhong, Detecting spam review through sentiment analysis 9.
[13] Youtube, Youtube Data API (v3), https://developers.google.com/youtube/v3/docs/commentThreads#Retrieve_comments.
[14] I. KANARIS, K. KANARIS, I. HOUVARDAS, E. STAMATATOS, Words versus character n-grams for anti-spam filtering, International Journal on Artificial Intelligence Tools 16 (06) (2007) 1047–1067. arXiv:http://www.worldscientific.com/doi/pdf/10.1142/S0218213007003692, doi:10.1142/S0218213007003692.
URL http://www.worldscientific.com/doi/abs/10.1142/S0218213007003692
[15] T. S. Guzella, W. M. Caminhas, A review of machine learning approaches to spam filtering, Expert Systems with Applications 36 (7) (2009) 10206 – 10222. doi:https://doi.org/10.1016/j.eswa.2009.02.037.
URL http://www.sciencedirect.com/science/article/pii/S095741740900181X
[16] H. Saif, M. Fernández, Y. He, H. Alani, On stopwords, filtering and data sparsity for sentiment analysis of twitter, in: LREC 2014, Ninth International Conference on Language Resources and Evaluation. Proceedings., 2014, pp. 810–817.
URL http://oro.open.ac.uk/40666/

[17] H. Zhang, The optimality of naive bayes (01 2004).
[18] M. Fernández-Delgado, E. Cernadas, S. Barro, D. Amorim, Do we need hundreds of classifiers to solve real world classification problems?, Journal of Machine Learning Research 15 (2014) 3133–3181.
URL http://jmlr.org/papers/v15/delgado14a.html
[19] T. C. Alberto, J. Lochter, T. Almeida, Tubespam: Comment spam filtering on youtube (12 2015).
[20] A. A. Septiandri, O. Wibisono, Detecting spam comments on indonesias instagram posts, Journal of Physics: Conference Series 801 (1) (2017) 012069.
URL http://stacks.iop.org/1742-6596/801/i=1/a=012069