Detecting Vandalism in Wikipedia Edits

Mudit Jain

muditjai@stanford.edu

Murugan Ayyappan murugan@stanford.edu

Nikhil Agarwal

nikhilag@stanford.edu

1. INTRODUCTION

1.1 Vandalism in Wikipedia

Wikipedia is an online encyclopaedia that is developed and maintained by the public. Anyone can make changes to its articles, and create new ones. Official Wikipedia sites have been created for 285 languages, with more than 4 million articles in Wikipedia English alone, making it is the largest encyclopaedia ever created.

Everyday more than 800,000 edits are made to articles across all of the Wikimedia websites by people from all over the world [9]. Although this open-source approach has been vital to Wikipedia's success, it has inherent problems, one of the most widespread being rogue edits (vandalisms).

Vandalism is defined as "any addition, removal or change of content made in a deliberate attempt to compromise the integrity of Wikipedia" [10]. The most common types of vandalisms according to Wikipedia are "addition of irrelevant obscenities and crude humour to a page, illegitimately blanking pages and inserting obvious nonsense into a page" [10]. It is estimated that the vandalism rate is around 7% of all edits. These bad-faith edits cause a lot of problems in Wikipedia by reducing article quality, reliability and damages the website's reputation.

Given the inherent problem of vandalisms in the open source model of Wikipedia, mechanisms to automatically detect Vandalisms have been around for almost as long as Wikipedia itself, and many approaches and techniques have been used to solve this problem. In this paper, we discuss some ways to improve existing automatic vandalism detection mechanisms. In particular, we focus on machine learning based approaches given their adeptness in solving such problems.

1.2 Evolution of Wikipedia Vandalism Detection

Early days of automated vandalism detection consisted of rule-based bots that involve blacklisting of IPs and users, grammar rules and static lists of vulgar words and obscenities. Examples of such bots are AntiVandalBot, ClueBot, MartinBot. Later on, Potthast [2] proposed to treat the vandalism detection problem as a classification problem, and suggested amongst the first machine learning and statistical based approaches to tackle this problem. They used Naïve Bayes that was improved by statistical compression models. Since then, many different feature approaches have been tried out with machine learning techniques, with good results.

Other researches such as those in [5] use the WikiTrust system to predict labels based on user reputation features. User reputation increases or decreases depending on the quality of their article revisions (edits). It assumes that quality is directly proportional to the amount of the change that was retained in subsequent revisions. The algorithm also considers reviewer reputation scores, and if the author was anonymous as features (amongst others). In conjunction with these features, they used

Another example is STiki Metadata [3] which was used in Wikipedia's STiki Vandalism detection tool. They used alternating decision trees with metadata features to come up with vandalism label predictions. Metadata features examined fields of edit such as timestamp, editor information, article and revision comment. These were then used to calculate features pertaining to editors registration status, edit time-of-day, day-of-week, geographical origin, revision comment length, etc.

In [6], Si-Chi Chin et al used semantic features and statistical language models to predict vandalisms. Building on statistical language model concepts, they constructed distributions of words from revision history of Wikipedia. Since vandalisms involves use of unexpected words, they used the variances in distributions to predict if an edit was a vandal or not. They also use an active learning model to solve noisy and incomplete labelling of vandalisms.

ClueBotNG [7] improves upon ClueBot to replace its rule-based model to a machine learning based one. It uses Bayesian classifiers (with a multinomial event model along with word whitelisting/blacklisting) and ANN (where inputs are various statistics calculated from the edit + Bayesian Classifier) to classify an edit as Vandal or Regular.

In the recent PAN 2010 and PAN 2011 competitions to detect vandalisms in Wikipedia, feature extraction and machine learning techniques have been the most successful. Velasco et al [1] used a combination of the features discussed above. They built upon the text features proposed by [2], as well as the language and metadata features obtained from [5, 6, 3]. These features were then learnt by LogitBoost, Random Forest and SVM models and were tested against the PAN-WVC-10 corpus.

In the following sections we will talk about the following: We will look at a summary of features we selected for implementation based on manual observation of vandalisms in conjunction with features suggested in [1], its class distribution and some insights into the nature of the problem. We will look at an outline of our overall algorithm followed by the results of our experiments that show how our addition of extra features improved the model's learning. We conclude by exploring some future directions.

2. Data

In the Wikipedia research community, two major corpuses have been released for vandalism detection model learning - the PAN-WVC-10 (PAN 2010) and PAN-WVC-11 (PAN 2011) corpuses. For our project, we chose the PAN-WVC-10 corpus for learning our model since its sample size is much bigger than PAN-WVC-11 data (30,046 regular and 2,394 vandals in 2010 as opposed to 8837 regular and 1143 vandal edits). The PAN 2010 data corpus was the first such corpus collected in an attempt to create a standard dataset that researchers could compare their algorithms against. This contains edits got from edit logs in Wikipedia which have been annotated by mechanical turks. For our project, we chose to include old and new revision texts, editor (if no username is present, IP address was stored instead), edit time and edit comment in our base data. We built our features from this data.

3. Preprocessing

Wikipedia edit logs are stored in text form and contain special formatting to denote links, image URLs, tables, section headers, references, alignment information, etc. There are existing parsers available that remove Wikipedia formatting and output plain text. We tried several parsers, however each had its own limitations. They either removed all special characters including those which were not part of Wikipedia formatting or they concatenated original edit text words, which in turn misled our features (see section 4 on Features below). Thus we created our own pre-processing steps that would help us with feature extraction later. It does the following:

- 1) Replace all URLs with <URL:"url">
- 2) Replace all alphanumeric characters with <ALPHANUMERIC:"alphanumeric">
- 3) Replace all numbers with <NUMBER:"number">
- 4) Separate all non alpha-numeric characters from alpha-characters by inserting spaces around them.
- 5) Replace newline character with <NEWLINE>.

Steps 1, 2, 3 allow easier feature extraction for URLs, alphanumerics and numbers respectively. Step 4 helps us in preserving the Wikipedia formatting symbols. Including this in our feature set will help classify cases of template vandalisms, image/url vandalisms, etc.

We ran our pre-processor on the PAN 2010 data, performed feature extraction on this processed data and fed these features to the classifier.

4. Features extracted

There are many classes of vandalism. The most common types of vandalism are addition and deletion of random text, promotion and propagation of spam, selfpromotion, silly vandalism by adding nonsense characters, creating hoaxes by propagating plausible misinformation, vandalisms to Wikipedia page objects such as templates, page name, images, links, etc., personal attacks and defamation attacks on people, countries and communities [11].

From the pre-processed data, we extracted three main classes of features – Metadata features, Text features and Language features that will help in classifying many of the above vandalism types. These are listed in Table 1.

Text features capture characteristics of the inserted text. It gets information on character casings, special characters, word length, diversity of characters, etc. These help in catching vandalisms involving random word inserts, silly vandalisms and also changes to Wikipedia formatting. Since we preserve Wikipedia formatting symbols in our pre-processed data, any changes to these will show up in the features.

Language features help the classifier in understanding the semantics of the edit text. Vulgar term features help in detecting abusive language and personal attacks. Colloquial and slang word related features help detect unrelated casual language in an edit text, which usually indicates vandalism. Using a pronoun list, the algorithm gets features that help in detecting self-promotions.

Metadata features capture extra information about edit text. Currently we have 2 features - is editor anonymous, and edit comment length.

5. Experiments and Results

We tried out different well known machine learning classifiers on the features discussed above for vandalism detection. We picked the best one from this evaluation and varied its parameters to observe the change in performance. We also analysed the different subsets of feature classes to understand their contribution to the entire system.

In each case, we used 10 fold cross validation on our training data and report the precision (P), recall (R), F-score, area under precision-recall curve (AUC-PR) and area under receiver operating characteristic curve (AUC-ROC). We used Weka for our experiments.

Table 1 – List of Features

Metadata Features	Feature Description			
IsEditAnonymous	Boolean to indicate whether edit was done by a registered wikipedia user or an anonymou user.			
EditCommentLength	Length of editor's comment.			
	otherwise, each of the following feature is computed on inserted text only)			
UpperToAllRatio	Ratio of uppercase characters to all characters			
UpperToLowerRatio	Ratio of uppercase characters to lower case characters.			
DigitToAllRatio	Ratio of digits to all characters.			
NonAlnumToAllRatio	Ratio of non alpha-numeric characters to all characters.			
CharacterDiversity	A function to measure number of unique character.			
AvgTermFrequency	Average frequency of each inserted word in the original article.			
MaxInverseCharacterDiversity	Maximum of (total characters/unique characters) over each inserted word			
VandalizedOneWord	Boolean to indicate whether a single word was converted from correctly spelt to misspelt.			
LongestNonURLWord	Length of the longest inserted word			
LongestRepeatedCharSequence	Length of longest contiguous character sequence			
LongestRepeatedWordSequence	Maximum number of repeats of a word.			
RevisionLengthRatio	Ratio of new revision length to old revision length.			
RevisionLengthIncrement	Difference between new revision length and old revision length.			
Language Features (Unless menti	oned otherwise, each of the following feature is computed on inserted text only)			
ColloquialFrequency	Frequency of words belonging to colloquial lexicon			
ColloquialImpact	Ratio of (added words - deleted words)/(total words in original) belonging to colloquial lexicon			
DictionaryFrequency	Frequency of words belonging to dictionary lexicon			
DictionaryImpact	Ratio of (added words-deleted words)/(total words in original) belonging to dictionary lexicon			
NonVulgarAdultTermsFrequency	Frequency of words belonging to non-vulgar adult lexicon			
NonVulgarAdultTermsImpact	Ratio of (added words-deleted words)/(total words in original) belonging to non-vulgar adult lexicon			
PronounsFrequency	Frequency of words belonging to pronoun lexicon			
PronounsImpact	Ratio of (added words-deleted words)/(total words in original) belonging to pronoun lexicon			
SlangWordsFrequency	Frequency of words belonging to insults and slangs lexicon			
SlangWordsImpact	Ratio of (added words-deleted words)/(total words in original) belonging to insults and slangs lexicon			
VulgarWordsFrequency	Frequency of words belonging to vulgar lexicon			
VulgarWordsImpact	Ratio of (added words-deleted words)/(total words in original) belonging to vulgar lexicon			
ColloquialDeletedFrequency	Frequency of deleted words belonging to colloquial lexicon			
DictionaryDeletedFrequency	Frequency of deleted words belonging to dictionary lexicon			
PronounsDeletedFrequency	Frequency of deleted words belonging to pronoun lexicon			
SlangWordsDeletedFrequency	Frequency of deleted words belonging to insults and slangs lexicon			
VulgarWordsDeletedFrequency	Frequency of deleted words belonging to vulgar words lexicon			
NonVulgarSexTermsDeleted Frequency	Ratio of (added words-deleted words)/(total words in original) belonging to non vulgar adult terms lexicon			

In Table 3, we analyse contributions of the different feature categories. Individually, text features are the most informative followed by language and metadata features. Effectiveness of text features indicates that random text (gibberish) insertion is the most common form of vandalism.

Language features, which target lexically well-formed but abusive/malicious edits, are less important than text features which indicate relatively lower frequency of such vandalisms. Metadata features are least informative since we have only two such features currently - namely comment length and whether editor was anonymous. Introducing additional metadata features such as reputation of author, local day and time of edit can improve them further, however these were not part of the PAN-2010 dataset and hence not considered for this project.

Table 2: Evaluating different classifiers						
Classifier	Р	R	F-Score	AUC PR	AUC ROC	
Naïve Bayes	0.524	0.375	0.437	0.394	0.839	
C4.5	0.709	0.496	0.584	0.634	0.907	
Logit Boost	0.811	0.494	0.614	0.719	0.942	
Random Forest	0.821	0.500	0.621	0.720	0.939	

Table 3 - Evaluation of feature classes using						
Random Forest (100 trees)						
				AUC	AUC	
Features	P	R	F-Score	PR	ROC	
Metadata	0.000	0.000	0.000	0.203	0.812	
Language	0.524	0.337	0.410	0.419	0.770	
Text	0.710	0.317	0.438	0.525	0.883	
M+L	0.655	0.396	0.494	0.540	874	
M+T	0.727	0.413	0.527	0.617	0.922	
T+L	0.819	0.439	0.572	0.673	916	
M+T+L	0.822	0.500	0.622	0.721	0.939	

Table 4: Evaluating different no. of trees for Random Forest					
Trees	Р	R	F-Score	AUC PR	AUC ROC
10	0.805	0.453	0.580	0.638	0.901
50	0.811	0.493	0.614	0.710	0.933
75	0.815	0.507	0.625	0.717	0.936
100	0.821	0.500	0.621	0.720	0.939
500	0.825	0.504	0.625	0.730	0.943
1000	0.828	0.507	0.629	0.732	0.944

Figure 1: Evaluation of different classifiers

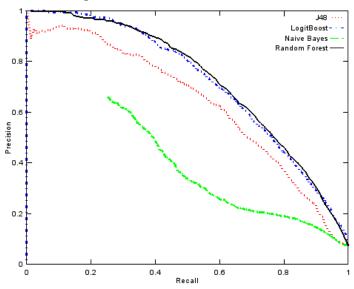
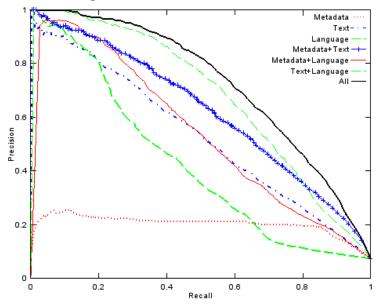


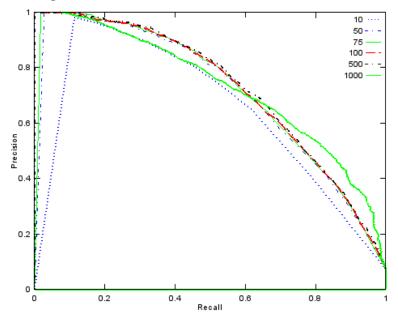
Figure 2: Evaluation of different feature classes



Combining feature categories always shows an improvement over either of the individual categories and the best feature group is obtained by adding all the three feature categories.

In Table 2, we analyse results of different types of classifiers. Random Forest performed better than other classifiers like LogitBoost, C4.5 decision trees and Naïve Bayes in terms of precision, recall and area under precision-recall curve. Hence we selected Random Forest as the algorithm for further analysis.

Figure 3: Evaluation of Random Forest with different no. of trees



In Table 4, we analyse the effect of increasing the complexity of our learning algorithm. Increasing the number of trees in Random Forest from 10 to 100 quickly increases the PR numbers until they gradually asymptote after 100. We observe that forest size of 100 trees achieves a good trade-off between complexity and performance.

The figures 1-3 show the P-R curve for Tables 2-4. We can choose the operating point of our algorithm to be either high precision or high recall depending on the specific requirements of the vandalism detection application.

6. Conclusions and Future Work

In this project we evaluated features with different classifiers for the task of Wikipedia Vandalism detection. We found out that Random Forest worked best with a combination of all the three classes of features (Metadata, Language and Text).

There is further scope for improvement by adding additional features such as author and edit reputations, better semantic features using NLP frameworks and using better lexicons for language features.

7. Acknowledgements

We would like to acknowledge Professor Andrew Ng and the TAs for their assistance throughout the course. We would also like to thank Santiago Moises Mola Velasco (winner of PAN-2010 competition) for his guidance.

8. References

- [1] Velasco et al, "Wikipedia Vandalism Detection Through Machine Learning: Feature Review and New Proposals," 2010.
- [2] G. Potthast, "Automatic Vandalism detection in Wikipedia," in Advances in Information Retreival -Lecture notes in Computer Science, vol. 4956, Craig Macdonald, Ed., Berlin, Springer, 2008, pp. 663-668.
- [3] "STiki," 2012. [Online]. Available: http://en.wikipedia.org/wiki/Wikipedia:STiki#Metada ta_scoring_and_origins. [Accessed 1 12 2012].
- [4] "WikiTrust," 2012. [Online]. Available: http://en.wikipedia.org/wiki/WikiTrust. [Accessed 2012].
- [5] B Adler et al, *Detecting Wikipedia Vandalism using Wikitrust*, Padua, 2010.
- [6] Si-Chi Chin et al, "Detecting Wikipedia Vandalism with Active Learning and Statistical Language Models," in WICOW'10, Raleigh, NC, USA, 2010.
- [7] Cluebot. NG, "Wikipedia user ClueBot NG," 2012.
 [Online]. Available: http://en.wikipedia.org/wiki/User:ClueBot_NG.
- [8] W. Vandalism, "Wikipedia," 2012. [Online]. Available: http://en.wikipedia.org/wiki/Wikipedia:Vandalism#T ypes_of_vandalism.
- [9] Wikimedia, "Wikimedia," 2012. [Online]. Available: http://toolserver.org/~emijrp/wikimediacounter/.
- [10] Wikipedia, "Wikipedia Vandalism," 2008. [Online]. Available: http://en.wikipedia.org/wiki/Wikipedia:Vandalism. [Accessed 2012].
- [11] W. VandalismTypes, "Wikipedia," 2012. [Online]. Available: http://en.wikipedia.org/wiki/Wikipedia:Vandalism_ty pes.
- [12] University of Waikato, "Weka".