Stony Brook University

Wiki Vandalysis - Wikipedia Vandalism Analysis

A Thesis Presented

by

Manoj Harpalani

to

The Graduate School

in Partial Fulfillment of the

Requirements

for the Degree of

Master of Science

in

Computer Science

Stony Brook University

December 2010

Stony Brook University

The Graduate School

Manoj Harpalani

We, the thesis committee for the above candidate for the

Master of Science degree,

hereby recommend acceptance of this thesis.

Rob Johnson – Thesis Advisor
Assistant Professor, Department of Computer Science

Steve Skiena – Chair
Distinguished Teaching Professor, Department of Computer Science

Yejin Choi
Assistant Professor, Department of Computer Science

This thesis is accepted by the Graduate School

Lawrence Martin
Dean of the Graduate School

Abstract of the Thesis

Wiki Vandalysis - Wikipedia Vandalism Analysis

by

Manoj Harpalani

Master of Science

in

Computer Science

Stony Brook University

2010

Wikipedia describes itself as the "free encyclopedia that anyone can edit". Along with the helpful volunteers who contribute by improving the articles, a great number of malicious users abuse the open nature of Wikipedia by vandalizing articles. Wikipedia editors fight vandalism both manually and with automated bots that use regular expressions and other simple rules to recognize malicious edits[Carter, 2010]. Researchers have also proposed Machine Learning algorithms for vandalism detection[Smets *et al.*, 2008; Potthast *et al.*, 2008a], but these algorithms are still in their infancy and have much room for improvement. This paper presents an approach to fighting vandalism using natural language processing and machine learning techniques. Along with basic features of the edit like edit distance, edit type, count of abnormal patterns and slang words, we use features related to information about the editor, past revision history of the article, change in sentiment of the article and PCFG sentence parser score. We have successfully been able to achieve an area under the ROC curve (AUC) of 0.94 and F1 score of 0.53 using LogitBoost in a 10 cross validation setting on a training set [Potthast, 2010] of 32444 human annotated edits. We also analyze the performance of our features by building separate classifier for insert or changes, deletes and template edits in a balanced and unbalanced corpus setting.

Contents

List of Figures

List of Tables

Acknowledgements

I am thankful to Prof. Rob Johnson, who guided and inspired me to take up this challenging research problem and participate in the Plagiarism Detection and Wikipedia Vandalism Detection Workshop (PAN 2010) at the Conference on Multilingual and Multimodal Information Access Evaluation (CLEF 2010) conference.

Special thanks to Prof. Luis Ortiz, Prof. Yejin Choi and Prof. Tamara Berg for their valuable guidance and suggestions in the Machine Learning and Natural Language Processing techniques for vandalism detection.

Thanks to Michael Hart, Thanadit Phumprao, Megha Bassi and Sandesh Singh for all their hard work and valuable contribution in the project. It would not have been possible without all of you.

I also thank my family and friends for all the motivation and support.

The text of this thesis/dissertation in part is a reprint of the materials as it appears in Wiki Vandalysis- Wikipedia Vandalism Analysis Lab Report for Plagiarism Detection and Wikipedia Vandalism Detection (PAN) Workshop at Conference on Multilingual and Multimodal Information Access Evaluation (CLEF) 2010. The co-author(s) listed in the publication directed and supervised the research that forms the basis for this thesis or dissertation.

Chapter 1

Introduction

Web 2.0 introduces new ways of creating, collaborating, editing and sharing user generated content online. With Social Networking sites, blogs and wikis users now have the freedom of expression on the web. Wikipedia [1] - "The free encyclopedia that anyone can edit", takes this advantage of Web 2.0 to enable users to collaborate and share their knowledge freely. It is open to such an extent that even anonymous users can also contribute to a Wikipedia article. With this liberty and the power of anonymity, some malicious users on the web post irrelevant content, criticize other's work, and try all possible ways to harm the content on Wikipedia. This problem is not just applies to Wikipedia, it is a problem with the freedom of content sharing and collaboration in Web 2.0. Some other examples include:

1. Rude comments posted on blogs
2. Bullying on Social Networking websites
3. Content based sharing policies

This makes it very important for service providers to govern and control the content being posted by malicious users. Many websites use publishing policies and rules, but over time these fall short and become unmanageable. Thus there is a need for building adaptive and intelligent techniques to detect such attacks. Wikipedia defines vandalism as "any addition, removal, or change of content made in a deliberate attempt to compromise the integrity of Wikipedia"[Wikipedia, 2010a].

[1] http://en.wikipedia.org/wiki/Wikipedia

Article Integrity refers to:

- Relevance of edit content
- Correctness of markup or formatting syntax
- Stating factual or encyclopedic content

Some examples of vandalism include deleting all the content from a page, modifying a page to be so long that is becomes difficult to load, and inserting profanity, nonsense, unrelated information, inaccurate information, opinionated text, or spam.

1.1 Types of Vandalisms

There are several types of vandalisms performed on an article[Wikipedia, 2010a]. Wikipedia has its own markup language called Wiki Markup [Wikipedia, 2010b] like HTML , users have access to this markup once they start editing the content. Users can change the markup to format the styling and looks of the article or change the content of the article itself. Vandals do not leave any corners and vandalize both the content as well as the markup. This study focus mainly on the content based vandalisms specifically the ones described below :

Silly Vandalism

Adding profanity, graffiti, random characters (gibberish), or other nonsense to pages. A few examples that are adding sentences like "shut up ugly", "this is crazy", "he was a gay" and others including slang and obscene words.

Sneaky vandalism

Vandalism that is harder to spot, or that otherwise circumvents detection. This can include adding plausible misinformation to articles, (e.g. minor alteration of facts or additions of plausible-sounding hoaxes), hiding vandalism (e.g. by making two bad edits and only reverting one), using two or more different accounts and/or IP addresses at a time to vandalize, abuse of maintenance and deletion templates, or reverting legitimate edits with the intent of hindering the improvement of pages.

1.2 Prior Work & Research

Vandalism detectors attempt to automatically distinguish integrity-violating edits from integrity-preserving edits. Wikipedia currently uses a combination of manual and automated vandalism detection. The automated "bots" employed by Wikipedia use regular expressions and other simple rules to detect vandalism[Carter, 2010]. Researchers have suggested more advance vandalism detection algorithms based on content in the edit[Potthast *et al.*, 2008b], author information
[West *et al.*, 2010], compression ratio of the article with and without the revision[Itakura and Clarke, 2009], and Bayesian-classifiers built on the differences between the new and old revisions[Smets *et al.*, 2008]. These early approaches leave much room for improvement which researchers further explored in the Plagiarism and Vandalism Detection Workshop [PAN-Workshop, 2010]. The Overview paper [Germany, 2010] of the PAN 2010 workshop summarizes various features like author reputation, article revision history, spelling errors, grammatical errors, comment information along with other meta information about the edit and compares the performance of various approaches. Another recent study [Wang and McKeown, 2010] uses Natural Language Processing techniques of syntactic and semantic modeling to capture syntactically ill-formed and semantically ill-intentional vandalism edits. This study enhances our previous work [Harpalani *et al.*, 2010] by introducing new features like article revision history, revision size ratio and syntax & semantic modeling identified in the PAN 2010 Workshop and recent research by [Wang and McKeown, 2010]. We also introduce a Natural Language Processing technique using Probabilistic Context Free Grammars to distinguish vandalism edits from regular.

1.3 Our contribution

This study presents a machine-learning-based vandalism detector that uses several features to classify vandalism and achieves an AUC score of over 0.94 and F1 score of 0.53 using LogitBoost with 500 iterations in a 10 fold cross validation setting. Our results significantly outperform a baseline classifier based on a previous approach using Naive Bayes - Bag of

Words[Smets *et al.*, 2008]. We also compare our results with winner[Velasco, 2010] of the PAN 2010 workshop[PAN-Workshop, 2010] and [Wang and McKeown, 2010]. Our classifier uses several simple features to catch obvious "silly" forms of vandalism, such as inserting obscenities or long, repeating patterns of text. We use subtler content-based features, such as misspelled words, grammatical errors, and changes in sentiment, which tend to indicate that an edit violates Wikipedia policy. We introduce syntax & semantic features similar to [Wang and McKeown, 2010] so as to capture tricky vandalism edits like off topic or non relevant content addition and evaluate our results on a balanced dataset having equal number of regular and vandalism edits created using random resampling technique. Inspiring from the Authorship Attribution using Probabilistic Context-Free Grammars [Raghavan *et al.*, 2010], we group authors into 2 categories regular and vandals, we use the same technique to capture the syntax and style of regular and vandalism edits using Probabilistic Context Free Grammar (PCFG) and compute the log probability score of a sentence as a feature along with existing features. Other significant features include information about the source of an edit, e.g. whether it is anonymous, edit history of articles and author reputation to make its decisions. We analyze the performance of our features on different type of edits including insert or change, delete and template edit. Majority of the vandalisms are in the insert or change edit type, and the best performance we achieve is a F1 of 0.58 and AUC of 0.93 in a corpus with ratio of vandlism to regular edits being 1:10. Though we do not achieve a very high performance on delete edit types and template change edit types, but we strongly believe that there is scope for improvement in capturing vandalism in these kind of edit types and segregating the edits to handle them differently with features applicable to that particular category will help in improving the performance of the classification task.

Chapter 2

Training corpus & Features

2.1 Training Corpus

The training corpus for the classification was provided by the organizers of the PAN workshop [PAN-Workshop, 2010; Potthast, 2010]. The corpus consisted of 32444 edits coupled with the previous revisions. Along with the training corpus in WikiMarkup format, we were also provided with meta-data including the edit id, the old revision id, the new revision id, the user name or IP of the author who performed the edit, the comment of the author, and whether the editor vandalized the article.

2.2 Problem Definition

Given an edit in Wikipedia, we can use the following information for the vandalism classification task :

1. The edit itself
2. Previous contributions of the editor
3. Comments of the editor
4. Past revisions of the edit
5. Related articles from the web or in Wikipedia itself

2.3 Edit Types

Wikipedia articles in the training set were formatted in WikiMarkup[Wikipedia, 2010b]. WikiMarkup includes not only the content, but link and display instructions, much like HTML. For our purposes, we converted the WikiMarkup directly to plain text using the Bliki engine[axelclk, 2010], which eliminated the formatting and link information. Thus we decided to separately calculate features for delete and template vandalism cases by using their edit diff on the wiki markup text itself. The main idea here is to separate out different type of edits and evaluate the performance of different classifiers for each edit type. We categorize each edit into one of the following 3 types :

Insert or Change (Content Addition)

These include the edits in which the user added content to the article or modified existing content. These changes are visible to a user and are not formatting or markup changes. We calculated these by taking a diff on the edits after converting Wiki Markup to plaintext.

Delete (Content Removal)

These include the edits in which the user deleted the content from the article. Like Inserts or Changes these are not changes in formatting or markup. We calculated these by identifying all the lines deleted from the edits after converting from Wiki Markup to plaintext.

Template Changes (Changes in Wiki Markup)

These include the edits in which the user modified the Wiki Markup instead of the visible content of an article. For these articles there is no difference in plain text thus we calculated the difference on Wiki Markup text itself to identify the changes done in the template for a given edit.

We calculate various features on all the above edit types and build classifiers for each edit type using the relevant features depending on the edit type.

2.4 Block Diagram

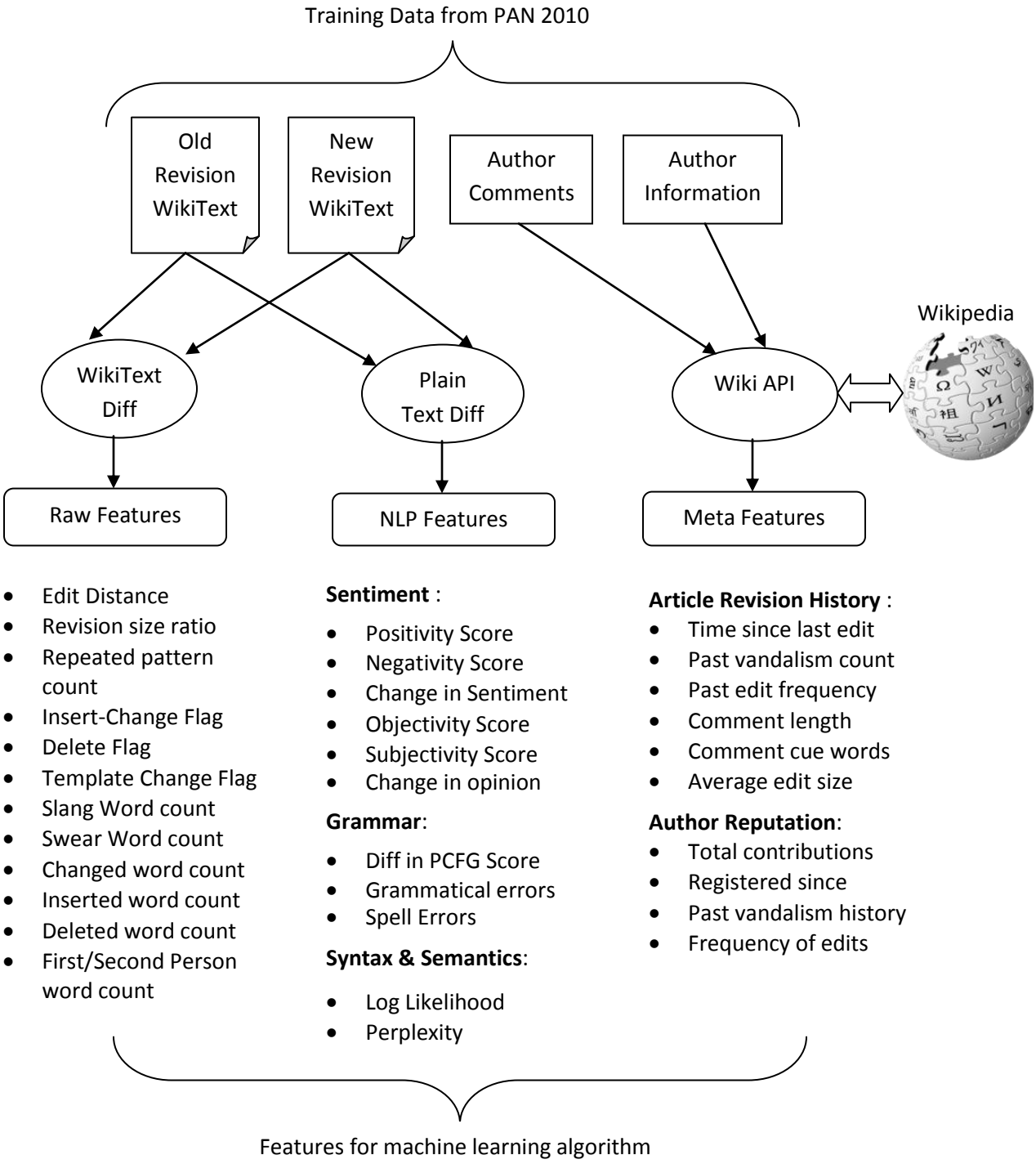


Figure 2.1: Block Diagram showing the feature extraction process.

2.5 Features

The main focus of our research has been on content based vandalism in which the insert or change type of edits, as this is the most prominent type of vandalism. We also analyze the delete and template vandalism cases separately. We used the following features of the edits for classification:

Edit Distance:

Our classifier calculates the Damerau-Levenstein Distance using the LingPipe API[Alias-i., 2008] to determine the number of changes required to convert the old revision of an article to its new revision.

Revision Size Ratio:

This feature is a simple ratio of number of words in the new revision to the number of words in the old revision of an edit, thus representing the amount of content added or deleted.

Edit Type:

Our classifier determines whether the edit consisted of insertion, deletion and/or modification of text or a combination of these actions.

Text Changes:

Our classifier determines the edit length, word count, words inserted, words deleted, and words changed using java-diff[Incava.org,]. It also uses LingPipe's English Sentence chunker to tokenize an article into sentences and calculate exact changes sentence-by-sentence using java-diff.

Spelling Errors:

Our classifier counts the number of apparent spelling mistakes in the edit and the ratio of spelling errors to correctly spelled words. Our spell-checking[LLC., 2010] software contained 200K English words, including named entities such as proper names and geographic places.

Obscene Words:

Our classifier enumerates the total number of obscene words in the edit and the ratio of obscene words to benign words. We started with a dictionary of obscene words [ExpressionEngine, 2010] and manually added other obscene words that we observed frequently in vandalized edits of our training set.

Patterns Detected	Regular Expressions
Repetition of exclamation mark	"!(3,)"
Appearance of multiple capital letters in a row	"\b[A-Z][â-z](3,)\b"
Repetition of same letter multiple times in a row	"(\w) \1(3,)+"
Repetition of same word multiple times in a row	".*(([A-Za-z])(3,))\1+.*"

Figure 2.2: Java regular expressions used to capture repeated patterns in vandalized edits

Repeated Patterns:

"Silly" vandalism often employs repeated patterns like upper case words, exclamation marks, and repetition of words/letters (e.g. "heyyyyyy", "oh!!!!!", "wow wow wow", "hahahaha", "WIKIPEDIAAA"). Our classifier counts these patterns using the regular expressions in Figure 2.2.

Sum of metrics:

This simple but effective meta-feature is the sum of the number of Repeated Letters, Repeated Words, Capitalized Words and Multiple Exclamation Marks.

Article History:

Vandals do not necessarily perform vandalism across all articles at the same rate. Rather, some articles receive a disproportionate amount of vandalism than others (e.g. Michael Jackson). The feature is the number of times an article was vandalized in the previous 5000 edits on the article. We denote vandalism as a comment left by an editor that contains "reverted", "user" and "vandalism" in this order. We also count how many times an article was reverted, regardless if it was explicitly vandalism or not. Along with these features we have temporal features like time since last edit, average frequency of edits and average edit size per month for previous 2 years and beyond.

Slang and Swear Word Counts:

Along with obscene words we also add these features so as to capture the common slang vocabulary used in many vandalism edits. We refer to [NoSlang.com, 2010] and [NoSwearing.com, 2010] for the slang and swear word dictionary. We compute these features for the plain text diff for the insert or change edit types as well as on the wiki markup diff for template and delete changes.

Sentiment Analysis:

Statements expressing opinions are common features of vandalism. Our classifier performs a sentiment analysis of the edit to uncover subjective or opinionated (positive or negative) text. We use LingPipe's Sentiment Analysis Tool trained on movie review data. The classifier counts objective or opinionated sentences and measures the change in total number of positive or subjective sentences.

Syntax & Semantics

Large number of vandalisms are off topic, these are tricky to be captured by any of the rule based or a black list dictionary based raw features. We reimplement the technique of shalow syntactic and semantic modeling by [Wang and McKeown, 2010] to capture such kind of vandalisms.

Grammatical errors:

Wikipedia editors strive for good grammar, but vandals do not generally follow these rules. They may insert words or phrases into the middle of existing sentences or write ungrammatical sentences deliberately or unintentionally. Our classifier parses the edits into sentences that had been inserted or changed by using java-diff and LingPipe's sentence tokenizer and counts the grammatical errors in them using CMU's Link Grammar Parser[Grinberg *et al.*, 1995].

Sentence Parser Features:

Vandals have a different writting style and syntax than regular editors. Inspired from [Raghavan *et al.*, 2010] we introduce a new technique to segregate vandalism edits from regular ones using Probabilistic Context Free Grammar sentence parser. Here we train two Stanford PCFG Sentence parsers [Klein and Manning, 2003], one on regular and another on vandalism edits and then use these custom trained parser to get the log probability score of each sentence in an edit. For each insert or change edit type we compute various statistics like mean, median, standard deviation, sum, min and max on the log probability/ PCFG score of all sentences in the edit. This feature models the syntax of regular and vandalism sentences and thus helps the classifier to separate them from each other.

Comment Features:

Vandals hardly take any time to insert a helpful comment while editing an article, whereas regular edit comments include use of common keywords like 'edited', 'revised', 'cleaned' etc.

We built a list of common terms used in the regular edit comments along with comment length for including them as a feature in our technique.

First and Second Pronouns:

A lot of vandals tend to write opinionated sentences and use first and second pronouns in the text inserted. Hence we adopted this simple feature to count the number of first and second pronouns in the edit difference.

Editor information:

Knowing who contributed an edit can give us some expectation of the quality of the change. For example, we expect an active and registered Wikipedia editor with several thousand edits to be more trustworthy compared to an unregistered editor that is only identifiable from an IP address and who has not contributed before. Our classifier uses several editor-based features: whether the editor is registered or unregistered, how long the editor has been registered, the total number of contributions made to Wikipedia during the period that training data was collected, the total number of edits made to Wikipedia by the editor up to the date the training data was finalized, the number of reverts on previous revisions by the author deemed to be vandalism (using the same heuristic as for article history) and the total number of previous modifications made by the editor on the article they are revising. We were careful to make sure for edit totals to not include future modifications. If the user is not registered, the classifier uses their IP address as a proxy for their identity.

2.6 Probabilistic Context-Free Grammar Details

We train two separate Stanford PCFG sentence parsers [Klein and Manning, 2003] one on regular edits and one on vandalism edits. We use 15000 edits from the PAN 2010 corpus of 32444 edits for training the PCFG parsers. Out of the remaining 17444 instances we consider only insert or changes edit type, since template vandalisms and deletes would not make sense to be parsed through the sentence parsers. Thus we are left with 10085 insert or changes which we use for experimenting with our new features. This makes sure that we do not include the edits which were used in the training of PCFG parser in the test set. Thus all classifiers built with PCFG feature would exclude these 15000 edits.

PCFG sentence parser training steps:

1. Calculate all the insert or change diff between the plain text of old and new revision separately for regular and vandalism edits.

2. Treebank all the sentences extracted above using Stanford parser.

3. Train the Stanford Parser using the treebanked regular edits to generate a sentence parser for regular edits.

4. Similarly train another sentence parser using the treebanked vandalism edits to generate a sentence parser for vandalism edits.

PCFG Sentence Parse Score Calculation:

1. The Stanford parser generates a PCFG score for each sentence parsed. This score represents the log likelihood probability of the best parse.

2. Compute the PCFG Score of each sentence using the trained sentence parser.

3. For every edit in the left over 10085 insert or changes we calculate the PCFG score for each sentence in the edit and then calculate statistics like Min, Max, Mean, Sum and Standard Deviation for each edit for both the parsers.

4. Finally we include the difference in these scores returned from both the regular and vandalism sentence parsers along with all the other features.

2.7 Syntactic & Semantic modeling

For the syntactic and semantic modeling, we use the same approach as [Wang and McKeown, 2010].

Calculating the syntax & semantic features:

1. For every edit use the edit title to get top 100 search results from Bing [1].

2. Using the Stanford Tagger [Toutanova *et al.*, 2003] POS tag the search results once they are cleaned using the HTMLParser library [Oswald and Walters,].

3. Once the POS Tagging is complete, extract POS tags and output them to separate files. Now we have the unigrams with POS tags for syntax & semantics modeling and only POS tags for modelling just the syntax.

[1] http://www.bing.com

4. Build a trigram language model on these 2 data sets for each edit using the SRILM toolkit [Stolcke, 2002].

5. Test the plain text edit diff for the edit on this language model and get the log likelihood and perplexity output using the SRILM toolkit.

2.8 Balanced & Unbalanced Corpus

We evaluate the performance of different classifiers on various datasets depending on the edit type, features and ratio of vandalism to regular edits. The details of the various training and test corpus used are as follows :

Balanced Corpus:

We create a balanced corpus to compare our results with [Wang and McKeown, 2010]. We replicate their experiment, so as to set a baseline and analyze the impact of adding our new features to the balanced corpus. We create a uniformly distributed data set by using random resampling technique for a set of 2018 edits from each class. We evaluate this corpus in a 10 fold cross validation setting with and without the PCFG feature to evaluate its performance.

Unbalanced Corpus:

For rest of the experiments we use the unbalanced corpus provided by PAN Workshop, However we use the corpus in different settings. Experiments that do not include the PCFG features utlized the whole corpus. We used 15000 edits from the PAN corpus to train the PCFG sentence parser and excluded these edits from the final test dataset. We also analyzed the performance of our classifier separately on inserts or changes, deletes and template edits. The insert or change dataset contains 17832 edits, the delete dataset contains 3808 edits and the template change dataset contains 13308 edits. The unbalanced corpus we use for PCFG consists of 10085 insert or change edits. We noted that all the datasets have approximately 8-10% vandalism edits.

Chapter 3

Classifiers

3.1 Modeling Classifiers

We build classifiers on both balanced and unbalanced corpora to compare our approach with a recent study usiing Shallow Syntactic & Semantic Modeling [Wang and McKeown, 2010] and the participants of PAN 2010 workshop. After experimenting various algorithms like C4.5 decision trees, AdaBoost, SVM, Naive Bayes Tree and LogitBoost on the features that we computed and found LogitBoost to perform the best. We discretize the numeric values to appropriate bins so as to have nominal features instead of numeric since we observed the results with discretized features were far better than with numeric features. For all our experiments we use the LogitBoost algorithm in a setting of 500 iterations and 10 fold cross validation. We set the following baselines and evaluate the performance of our new features in comparison with the ones previously used by [Wang and McKeown, 2010] and [PAN-Workshop, 2010].

3.2 Baseline

Baseline 1: Balanced Corpus Scenario

For the experiments on balanced corpus, we set our baseline as "Got You - Vandalism Detection" study [Wang and McKeown, 2010]. We replicate the experiment they performed and compute the results of Syntax & Semantics features for both balanced and unbalanced

corpus. Our features along with Syntax & Semantics improve the results and outperform the baseline classifier in the balanced corpus setting.

Baseline 2: Unbalanced Corpus Scenario

We use the same setting of LogitBoost with 500 iterations and 10 fold cross validation. We experiment the syntax & sematic features in an unbalanced corpus scenario and notice that the results improve as compared to classifier without syntax & semantic features. The results of our classifier on the unbalanced corpus were lower than those on the balanced as expected due to bias towards the regular edits. The original PAN 2010 corpus contains 7% vandalism edits. We first evaluate the results in the same setting as the original competition which was 15000 edits in the training corpus and 17444 edits in the test corpus. We improve the results from the ones we previously submitted [Harpalani *et al.*, 2010] to the competition. The AUC increased from 0.88 to 0.91, which is comparable to the results of the winner of the PAN competition [Velasco, 2010]. We also analyze our features with different type of edits including insert or change, delete and template edits. We present the results of different combination of corpora in the Evaluation and Results section.

3.3 Building Classifiers

We built classifiers using the features presented in the previous section with different algorithms using Weka[Hall *et al.*, 2009] - A machine learning and data mining library provided by University of Waikato. In our previous work [Harpalani *et al.*, 2010]
NB Tree which is a hybrid version of Naive Bayes and Decision Tree performed well with the numeric features we computed, but the LogitBoost algorithm outperformed NB Tree when we discretized the features. LogitBoost is a boosting approach which uses logistic regression as the cost function.

Chapter 4

Evaluation & Results

4.1 Evaluation Metrics

Vandalism accounts only for 8-10% of the edits, thus choosing the right metric for determining the best classifier is tricky. Accuracy of the classifier is not helpful as even a simple classifier that classifies all edits as regular can achieve a high accuracy of 94%. Precision, Recall and F1 score is important in the task of vandalism detection as they represent the correctness and completeness of the classifier. The detector should capture as many vandalisms as possible and have a very low false positive rate so as not to deter genuine editors from contributing to Wikipedia. Recent studies have started using area under the ROC curve (AUC) [PAN-Workshop, 2010] to measure the performance of vandalism detectors. Measuring the performance of the vandalism detection algorithm in terms of AUC instead of other measures also makes sense as AUC score denotes the probability by which a classifier is able to distinguish a randomly sampled vandalism edit from a randomly sampled regular edit. We compute our results in both unbalanced and balanced dataset scenario to compare our results with existing studies.

4.2 Balanced Corpus

We use random sampling to generate a corpus of 4036 edits having equal proportion of regular and vandalism edits. We present the results of classification using LogitBoost algorithm in

Experiment	Precision	Recall	F–Measure	AUC
"Got you!" [Wang and McKeown, 2010]	0.85	0.85	0.86	–
Our features with Syntax & Semantics without PCFG	0.83	0.89	0.86	0.93
Our features with Syntax & Semantics with PCFG	0.84	0.89	0.87	0.94

Table 4.1: Results with Syntax & Semantic Features in a balanced corpus setting

Experiment	Precision	Recall	F–Measure	AUC
Our features including PCFG without Syntax & Semantics	0.74	0.43	0.54	0.91
Our features including PCFG with Syntax & Semantics	0.74	0.48	0.58	0.92

Table 4.2: Results with Syntax & Semantic Features in an unbalanced corpus setting

the following table and compare with the results obtained by [Wang and McKeown, 2010]. We observe that we get comparable results when we use the syntax & semantic features along with our features. Including the PCFG features improves the F1 score and AUC by 1%. Adding our features along with syntax & semantics improve the Recall and F1 as compared to the baseline results.

4.3 Unbalanced Corpus

Syntax & Semantics Feature We generate an unbalanced corpus of 4036 edits out of which 493 edits were vandalisms having approximately the same class distribution as the original PAN corpus. Here we evaluate the performance of syntax & semantics feature in an unbalanced corpus setting. The results for this experiment are shown in the table 4.2. We see that the results in the unbalanced corpus setting are low because of the bias or skew towards the regular edits which form the majority of the training corpus. But the results we achieve with Syntax & Semantic feature show significant improvement as compared to those without syntax & semantic feature. This shows that syntax & semantic feature improve the result even in an unbalanced corpus. We further believe that the results can be improved if we utilize Wikipedia itself to search for related articles to develop the syntax & semantic

Experiment	Precision	Recall	F–Measure	AUC
PAN 2010 Winner	0.86	0.56	0.67	0.92
Our PAN 2010 setting results	0.64	0.35	0.45	0.91
10 fold cross validation on complete PAN 2010 corpus	0.74	0.41	0.53	0.94

Table 4.3: Results with complete PAN 2010 corpus with our new features except for syntax & semantics and PCFG in an unbalanced corpus setting

language model since the amount of noise in Wikipedia would be much lesser than web search results.

Improved Results with new features on the PAN 2010 With the features identified in the PAN Competition and new feature of PCFG score we see that we improve our results from an AUC of 88.5 to 9192A 10 fold cross validation on the complete corpus of 32444 edits gives us a result of 93The table 4.3 shows the comparison of results for the complete PAN corpus of 32444 edits.

4.4 Classifier on edit types

We evaluate the performance of our features individually on different type of edits. The Table 4.4 shows the results of classifiers on inserts or changes, deletes and template edits. About 80% of the vandalism edits are performed in the insert or change edit category, 17% are performed in the template change category and the rest 3% fall in the delete category. Hence we strongly believe that there is a need to segregate the classification task according to the edit type. The insert or change corpus without PCFG contains 17820 edits out of which 1904 are vandalisms. The insert or change with PCFG contains 10080 edits since we ensure that we do not have the 15000 instances which were used for training the PCFG parser in the test results. The delete corpus contains 3808 edits out of which 56 are vandalisms. The template change corpus contains of 13223 edits out of which 405 are vandalisms. We do not achieve high results for the delete and template vandalism. This is because of the sparsity of vandalism edits, or skew towards regular and the difficulty of the template changes to be parsed through any of our sophisticated features like grammar checker, sentence parser or sentiment analyzer.

Experiment	Precision	Recall	F–Measure	AUC
Insert or Changes without PCFG	0.73	0.41	0.52	0.92
Insert or Change with PCFG	0.73	0.48	0.58	0.93
Delete without PCFG	0.58	0.25	0.35	0.95
Template Change edits	0.71	0.14	0.23	0.93

Table 4.4: Results with subset of corpora segregated based on edit type in an unbalanced corpus setting

Feature	**Information Gain**
Total number of author contributions	0.105
How long the author has been registered	0.098
How frequently the author contributed in the training set	0.097
If the author is a registered user	0.088
Maximum PCFG Score Difference	0.043
How often the article has been reverted	0.037
Total contributions of author to Wikipedia	0.034
Previous vandalism count of the article	0.032
Length of edit comment	0.029
Revision Size Ratio	0.024

Table 4.5: Top ten features for Insert or Change edit type ranked by information gain

4.5 Best Performing Features

In Table 4.5, we present our top 10 ranked features for the insert or change edit type according to information gain. We see that author information, PCFG sentence score, article revision history, length of comment and revision size ratio play a major contribution in conveying information about the edits.

Interstingly 5 of the features are related to the authorof the edit indicating the importance of author information. Edit specific features like previous revert and vandalism counts also make their way here.

In Table 4.6, we see that the features including revision size ratio, edit distance, delete word count, author info, deletion of objective sentences are helpful in classifying delete vandalisms. We do not include PCFG features here since deletions in regular edit may delete regular sentences or vandalism sentences to revert past vandalism. Comment length matters significantly since vandals hardly comment when performing the edit. Revision size and edit distance rank the highest, indicating large change in deletes identify vandalism edits

Feature	Information Gain
Revision Size Ratio	0.027
Edit Distance	0.026
Deleted Word Count	0.023
Total Author contributions in Wikipedia	0.022
Total sentences deleted in	0.018
No. of objective sentences deleted	0.016
Is the author registered on Wikipedia?	0.015
How long the author has been registered	0.014
No. of subjective sentences deleted	0.013
Comment Length	0.009

Table 4.6: Top ten features for Delete edit type ranked by information gain

Feature	Information Gain
Total contributions of author to Wikipedia	0.044
How long the author has been registered	0.035
If the author is a registered user	0.032
How frequently the author contributed in the training set	0.025
How many times the article has been reverted previously	0.016
How many revisions have been made previously for the article	0.015
How many times the articles has been vandalized in the past	0.015
Average number of edits per month	0.014
Comment Length	0.014
Average time between edits	0.012

Table 4.7: Top ten features for Template Changes edit type ranked by information gain

well.

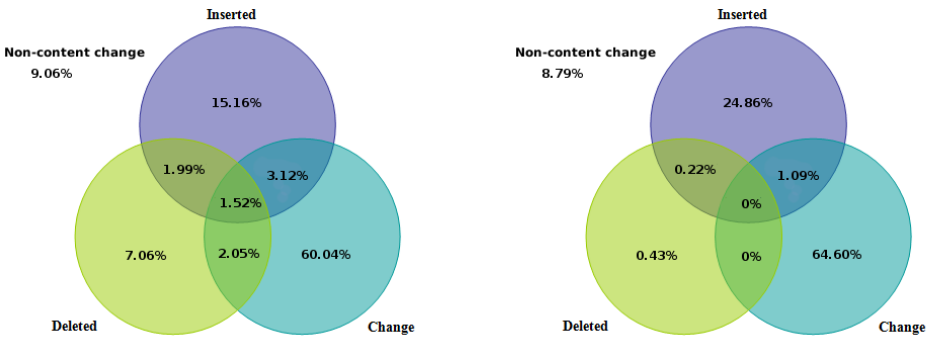
In Table 4.7, we see that similar to Insert or Change edits, the author information, article revision history and size of edits help classify template change vandalisms. We do not use PCFG feature for template vandalisms since they contain special characters from the WikiMarkup and do not always have complete sentences. Interestingly the Average Number of Edits per Month and Average Time Between Edits feature indicates that template changes on frequently or rarely revised articles help in classifying them as regular or vandalism.

Chapter 5

Discussion

5.1 Proportion of vandalism in different edit type

Our training corpus in the PAN competition contained 15000 edits in the training set and 17444 edits in the test set. In this set of 15000 edits, we noted that registered users contributed 67% of these edits, while anonymous users contributed 33%. Among these 93.9% of edits were not instances of vandalism. Not surprisingly, unregistered users vandalized articles more frequently than registered users, 16% to 1% respectively. See Figure 5.1 for Venn diagrams which highlight how often an article has had text modified, deleted, inserted, or any combination of these actions. Note that vandals are significantly more likely to insert text and are much less likely to make multiple changes in one revision.



Regular Edits Vandalism

Figure 5.1: Venn diagrams showing how often the text of an article was inserted into, changed, deleted or any combination of these actions

5.2 Registered v/s Anonymous Vandals

The authors of [Priedhorsky *et al.*, 2007] observed that there are a small number of registered users (0.1%) that generate over half the content on Wikipedia. Therefore, we report not only the overall performance, but also statistics relating to important subsets of editors. This will let us gauge, for example, if the false positive rate is much too high to be acceptable for frequent contributors. We encourage future papers relating to the efficacy of vandalism detectors to analyze the performance for important subsets of editors. Classifying vandalism by registered users is tricky. A sizable chunk of these vandalizing edits come from users that registered just before submitting the edit, and therefore are very suspect, but the rest of these edits are much more difficult to detect. Registered users infrequently vandalize articles, so we have little training data to learn from. We noted in two of the ten cases that we manually inspected that there was a strong argument to not classify the edit as vandalism. For example, one edit introduced a link to "Sex Magic", which may appear on face value to be nonsense. On further investigation, it turns out that this is indeed an article in Wikipedia. As we are indebted to Potthast et al. for their dataset[Potthast, 2010], we recommend that the edits labeled vandalism by frequent and active contributors to be examined more closely, perhaps by an active Wikipedian. Fortunately, this constitutes a small minority of edits, but detecting vandalism among registered users is a challenging obstacle.

5.3 Editor Reputation

Five of our top ten features involve editor information. Current Wikipedia bots make limited use of editor information, but our results show that they could do more. In fact, our approach benefited most from how many contributions a particular Wikipedia User or IP address has made. Other important editor attributes were edit frequency, i.e. whether the editor is currently active, and whether the editor had contributed to the article being revised. The number of contributions a registered user or an IP address has made to Wikipedia interestingly benefits not only registered but unregistered users too based on IP address. Several IP addresses that have contributed significantly in the past have a lower vandalism

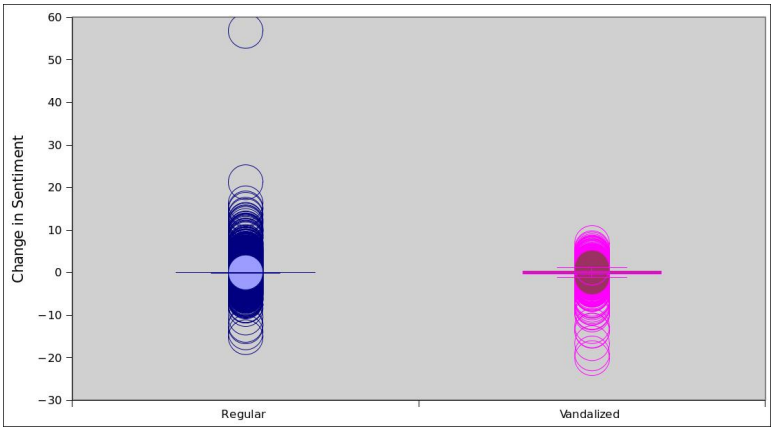


Figure 5.2: Box plot showing the change in sentiment between regular and vandalized edits.

to regular edit ratio in this dataset than those IP addresses that have only contributed a handful of edits. In our training data, the ratio of regular edits to vandalized edits for IP addresses with more than 10000 previous edits and IP addresses with less than 100 previous edits is 76.0 and 3.6 respectively. We gathered information about 5 IP addresses in the training set with the most contributions and found that they were located near universities and one was owned by Microsoft (according to the reverse DNS). Therefore, what does this suggest for Wikipedia? Certain IP addresses and areas are conducive to recruiting future Wikipedians - that these individuals have demonstrated a commitment to the integrity of the site. This would grant more accountability and perhaps better expose the malicious edits from these "good" IP addresses.

5.4 Change in Sentiment

Figure 5.2 shows the values of the change in sentiment score. Outliers are shown as circles. Majority of edits had a change in sentiment of zero, but we do see that vandalized edits skew negatively and regular edits skew in a positive difference. We note that for vandalizing edits, the mean change in sentiment was -0.14 with a standard deviation of 2.0 and for regular edits the mean change was 0.03 with a standard deviation of 1.1. Although most edits, vandalism and otherwise, had a change in sentiment score of zero, we note that vandalism skews towards a negative change in sentiment and regular edits positively, which appears to make sense intuitively. Because of time and resource constraints, we were not able to

confirm if either A) vandalism introduces more polarizing than subjective statements, B) the polarity detector is more effective than the subjectivity detector software or C) subjective or polarizing statements do not constitute a significant percentage of vandalism. We approached the more sophisticated features like the sentiment analysis with an off-the-shelf approach. Therefore, we used pre-existing datasets and software APIs to gather feature values. We hypothesize that employing Wikipedia specific examples may perform more precisely in detecting subjectivity and polarity in article revisions.

5.5 Edit Distance, Spelling & Grammar

Two of our features unexpectedly behaved similarly to the edit distance feature. First, we note that the larger the edit distance, the more likely the edit is not vandalizing the article. We found that our grammar and misspelled words feature ultimately behaved in a similar fashion. For misspelled words, the larger the edit, the more likely our dictionary found misspelled words. Analyzing the top 10 edits with the most misspelled words (all non-vandalizing edits), we observed that these articles contained many non-English or unique pronouns. For example, one article on the movie "Blue Hills Avenue" had many misspellings because the dictionary could not recognize several of the character names. Therefore, we suggest for future dictionaries to build an article specific dictionary that incorporates words that survive several revisions or appear to be integral to the article (a summarization technique could be a good first approximation for this approach). Our grammar checker also had similar issues, arising from difficulties with part-of-speech tagging certain words. Another problem unique to the grammar checker is that several different types of tabulations and enumerations are used throughout Wikipedia. An obvious improvement for implementing a grammar checker is to ignore any modification that occurs in either a table, list or infobox. In future testing, we plan to implement these changes.

5.6 PCFG Analysis

We compute the PCFG scores and evaluate this feature only on Insert or Change type of edits, since deletions in regular edit could be removal of regular sentences or vandalism sentences which could confuse the machine learning algorithm. We do not consider template changes since they include special characters used in WikiMarkup and would not have complete sentences in english as such.

We see that the results on Insert or Changes features without PCFG improve by 6% in F1 score, 1% in AUC, 7% in Recall. The MaxPCFGScore feature ranks 5th in the top 10 features by information gain. This indicates that PCFG sentences parsers can be helpful in separating out the syntax and style of vandalism edits pretty well.

5.7 Syntax & Semantics

The Syntax & Semantics feature without PCFG when analyzed in a balanced dataset setting, perform extremely well and yield very high results - 86% F1 score, 93% AUC and 89% Recall. Adding PCFG score increases the result of all metrics by 1%.

The results of Syntax & Sematic features in an unbalanced dataset is significantly lower than the balanced setting results. The maximum F1 score we achieved using the Syntax & Semantic features along with PCFG was 58% with an AUC of 93%. Thus we observe that the bias towards regular edits in the actual scenario impacts the results of the classifier significantly and future research should take this into consideration.

5.8 Future Vandalism Detection Techniques

An Ideal Vandalism Detector would be the one which has Zero Delay [Adler *et al.*, 2010] and is able to capture tricky vandalism types like Sneaky Vandalism. For Wikipedia to implement this Ideal Vandalism detector it would need a Precision of 1 so that all vandalisms classified by the detector are actually vandalisms and not regular edits. This is very important because regular editors must not be deterred from contributing to Wikipedia. High Recall indicates that the detector captures as much vandalisms as it can indicating the completeness of the

detector. Vandals may also come up with other tricky edits or explore image vandalism further. Thus future vandalism detectors should consider these scenarios and focus on the above metrics so that the detectors are practical enough to be implemented.

Chapter 6

Conclusion

This study presents new features and algorithms to improve the task of vandalism detection in Wikipedia. We found that features such as information about the editor, revision history of the article, misspellings, obscene words, number of repetitive patterns, change in sentiment, and PCFG sentence parse score effectively classify vandalism. These features combined with the LogitBoost classifier gave an AUC score of 94% and F1 score of 53% for a 10-fold stratified cross validation on the PAN 2010 corpus of 32444 edits. We analyzed the performance of our classifier using both balanced and unbalanced dataset. We also evaluate the performance of our classifier on data sets segregated based on their edit type. Our classifier performs well on Insert or change vandalisms but we need to improve our feature set to deal with Template and Delete vandalisms as they are tricky to detect.

Bibliography

[Adler *et al.*, 2010] B. Thomas Adler, Luca de Alfaro, and Ian Pye. Detecting wikipedia vandalism using wikitrust - lab report for pan at clef 2010. In Martin Braschler, Donna Harman, and Emanuele Pianta, editors, *CLEF (Notebook Papers/LABs/Workshops)*, 2010.

[Alias-i., 2008] Alias-i. Lingpipe 4.0.0. http://alias-i.com/lingpipe/, October 2008.

[axelclk, 2010] axelclk. gwtwiki - Java Wikipedia API (Bliki engine). http://code.google.com/p/gwtwiki/, 2010.

[Carter, 2010] Cobi Carter. ClueBot. http://en.wikipedia.org/wiki/Wikipedia:CLUEBOT, 2010.

[ExpressionEngine, 2010] ExpressionEngine. English obscene wordlist. http://expressionengine.com/?ACT=51&fid=8&aid=1830_mvLZ2WkoRucNvRegThbL&board_id=, 2010.

[Germany, 2010] Bauhaus-University Weimar Germany. Overviewof the 1stinternational competitionon wikipedia vandalism detection martinpotthast,bennostein, andteresaholfeld, 2010.

[Grinberg *et al.*, 1995] Dennis Grinberg, John Lafferty, and Daniel Sleator. A robust parsing algorithm for link grammars. In *In Proceedings of the Fourth International Workshop on Parsing Technologies*, 1995.

[Hall *et al.*, 2009] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The weka data mining software: an update. *SIGKDD Explor. Newsl.*, 11(1):10–18, 2009.

[Harpalani *et al.*, 2010] Manoj Harpalani, Thanadit Phumprao, Megha Bassi, Michael Hart, and Rob Johnson. Wiki vandalysis- wikipedia vandalism analysis lab report for pan at clef 2010, 2010.

[Incava.org,] Incava.org. Difference algorithm for Java. http://www.incava.org/projects/java/java-diff/.

[Itakura and Clarke, 2009] Kelly Y. Itakura and Charles L. A. Clarke. Using dynamic markov compression to detect vandalism in the wikipedia. In *SIGIR '09: Proceedings of the*

32nd international ACM SIGIR conference on Research and development in information retrieval, pages 822–823, New York, NY, USA, 2009. ACM.

[Klein and Manning, 2003] Dan Klein and Christopher D. Manning. Fast exact inference with a factored model for natural language parsing. In *In Advances in Neural Information Processing Systems 15 (NIPS*, pages 3–10. MIT Press, 2003.

[LLC., 2010] SoftCorporation LLC. English dictionary. http://softcorporation.com/products/spellcheck/dictionaries/english/, 2010.

[NoSlang.com, 2010] NoSlang.com. Slang Words Dictionary. http://www.noslang.com/dictionary/full, 2010.

[NoSwearing.com, 2010] NoSwearing.com. Swear Words Dictionary. http://www.noswearing.com/dictionary, 2010.

[Oswald and Walters,] Macfarlane Oswald, Raha and Walters. HTML Parser. http://htmlparser.sourceforge.net.

[PAN-Workshop, 2010] PAN-Workshop. Evaluation Campaign on Plagiarism Detection and Wikipedia Vandalism Detection. http://www.uni-weimar.de/medien/webis/research/workshopseries/pan-10/task2-vandalism-detection.htm, 2010.

[Potthast *et al.*, 2008a] Martin Potthast, Benno Stein, and Robert Gerling. Automatic vandalism detection in wikipedia. In Craig Macdonald, Iadh Ounis, Vassilis Plachouras, Ian Ruthven, and Ryen W. White, editors, *Advances in Information Retrieval*, volume 4956 of *Lecture Notes in Computer Science*, chapter 75, pages 663–668. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[Potthast *et al.*, 2008b] Martin Potthast, Benno Stein, and Robert Gerling. Automatic vandalism detection in wikipedia. In *ECIR'08: Proceedings of the IR research, 30th European conference on Advances in information retrieval*, pages 663–668, Berlin, Heidelberg, 2008. Springer-Verlag.

[Potthast, 2010] Martin Potthast. Crowdsourcing a Wikipedia Vandalism Corpus. In *33rd Annual International ACM SIGIR Conference (to appear)*. ACM, July 2010.

[Priedhorsky *et al.*, 2007] Reid Priedhorsky, Jilin Chen, Shyong (Tony) K. Lam, Katherine Panciera, Loren Terveen, and John Riedl. Creating, destroying, and restoring value in wikipedia. In *GROUP '07: Proceedings of the 2007 international ACM conference on Supporting group work*, pages 259–268, New York, NY, USA, 2007. ACM.

[Raghavan *et al.*, 2010] Sindhu Raghavan, Adriana Kovashka, and Raymond Mooney. Authorship attribution using probabilistic context-free grammars. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 38–42, Uppsala, Sweden, July 2010. Association for Computational Linguistics.

[Smets *et al.*, 2008] K. Smets, B. Goethals, and B. Verdonk. Automatic vandalism detection in wikipedia: Towards a machine learning approach. In *Proceedings of the Association for the Advancement of Artificial Intelligence (AAAI) Workshop on Wikipedia and Artificial Intelligence: An Evolving Synergy (WikiAI08)*, pages 43–48. AAAI Press, 2008.

[Stolcke, 2002] Andreas Stolcke. Srilman extensible language modeling toolkit. In *In Proceedings of the 7th International Conference on Spoken Language Processing (ICSLP 2002*, pages 901–904, 2002.

[Toutanova *et al.*, 2003] Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In *In Proceedings of HLT-NAACL 2003*, pages 252–259, 2003.

[Velasco, 2010] Santiago M. Mola Velasco. Wikipedia vandalism detection through machine learning: Feature review and new proposals * lab report for pan at clef 2010, 2010.

[Wang and McKeown, 2010] William Yang Wang and Kathleen R. McKeown. got you!: Automatic vandalism detection in wikipedia with web-based shallow syntactic-semantic modeling. In *23rd International Conference on Computational Linguistics (Coling 2010)*, page 11461154, 2010.

[West *et al.*, 2010] Andrew G. West, Sampath Kannan, and Insup Lee. Detecting wikipedia vandalism via spatio-temporal analysis of revision metadata. In *EUROSEC '10: Proceedings of the Third European Workshop on System Security*, pages 22–28, New York, NY, USA, 2010. ACM.

[Wikipedia, 2010a] Wikipedia. Definition of vandalism. http://en.wikipedia.org/wiki/Wikipedia:Vandalism, 2010.

[Wikipedia, 2010b] Wikipedia. WikiMarkup. http://en.wikipedia.org/wiki/Help:Wiki_markup, 2010.