Wiki Vandalysis - Wikipedia Vandalism Analysis
Lab Report for PAN at CLEF 2010

Manoj Harpalani, Thanadit Phumprao, Megha Bassi, Michael Hart, and Rob Johnson

Stony Brook University
{mharpalani, tphumprao, mbassi, mhart, rob}@cs.stonybrook.edu

Abstract 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. Deterring and reverting vandalism has become one of the major challenges of Wikipedia as its size grows. Wikipedia editors fight vandalism both manually and with automated bots that use regular expressions and other simple rules to recognize malicious edits[5]. Researchers have also proposed Machine Learning algorithms for vandalism detection[19,15], but these algorithms are still in their infancy and have much room for improvement. This paper presents an approach to fighting vandalism by extracting various features from the edits for machine learning classification. Our classifier uses information about the editor, the sentiment of the edit, the "quality" of the edit (i.e. spelling errors), and targeted regular expressions to capture patterns common in blatant vandalism, such as insertion of obscene words or multiple exclamations. We have successfully been able to achieve an area under the ROC curve (AUC) of 0.91 on a training set of 15000 human annotated edits and 0.887 on a random sample of 17472 edits from 317443.

1 Introduction

Wikipedia defines vandalism as "any addition, removal, or change of content made in a deliberate attempt to compromise the integrity of Wikipedia"[21]. Vandalism can take many forms, including 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.

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[5]. Researchers have previously suggested more advance vandalism detection algorithms based on content in the edit[16], author information[20], compression ratio of the article with and without the revision[10], and Bayesian-classifiers built on the differences between the new and old revisions[19]. These early approaches leave much room for improvement.

This paper presents a machine-learning-based vandalism detector that uses several features to classify vandalism and that achieves an AUC score of over 0.887. Measuring the performance of the vandalism detection algorithm in terms of AUC instead of other measures 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. Our results significantly outperform a baseline classifier based on a previous approach[19]. Our classifier uses several simple features to catch obvious "silly" forms of vandalism, such as inserting obscenities or long, repeating patterns of text. We also use subtler content-based features, such as misspelled words, grammatical errors, and changes in sentiment, which tend to indicate an edit violates the Wikipedia editing policy. Finally, the classifier uses information about the source of an edit, e.g. whether it is anonymous, to make its decisions.

2 Training corpus & Feature Extraction

The training corpus for the classification was provided by the organizers of the PAN workshop[1,14]. The corpus consisted of 15000 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 edit vandalized the article.

Wikipedia articles in the training set were formatted in WikiMarkup[22]. 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[4], which eliminated the formatting and link information. Therefore, we are unable to detect template or link vandalism.

We focus on content based vandalism as it is the most prominent type of vandalism that exists. After analyzing samples of vandalized and regular edits, we observed that certain attributes of the edits distinguished vandalism from regular edits. We used the following features of the edits for classification:

Edit Distance: Our classifier calculates the Damerau-Levenstein Distance using the LingPipe API[2] to determine the number of changes required to convert the old revision of an article to its new revision.

Edit Type: Our classifier determines whether the edit inserted, deleted and/or modified 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[9]. 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[12] 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[3] and manually added other obscene words that we observed frequently in vandalized edits of our training set.

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.

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[7].

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.

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 count also how many times an article was reverted, regardless if it was explicitly vandalism or not.

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.

3 Classifiers

Modeling Classifiers. We compare in this section two different models for building classifiers. We first investigate the Naive Bayes with Bag Of Words approach described in Smets et al[19]. We introduce a modification to this approach by combining Naive Bayes with Rank Boost[6]. We compare this to a second approach that uses the NBTree

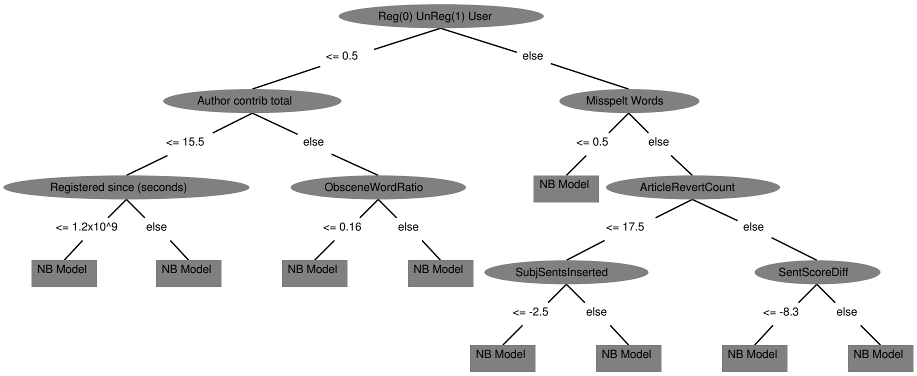


Figure 1. The structure of the NBTree after building the classifier on all training examples. The branches of the tree denote the decision path taken along with the threshold chosen for taking a particular branch. The leaves are Naive Bayesian classifiers used to classify the edit.

classifier (a hybrid between decision trees and Bayesian classifiers) and the features described above.

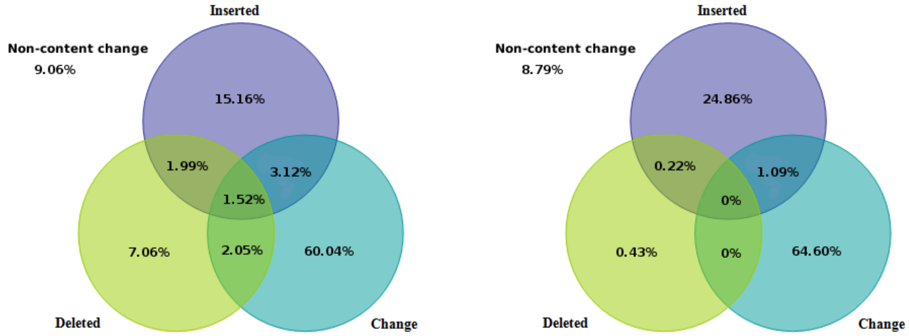
Baseline: Naive Bayes with the Bag Of Words model. In order to set a baseline for comparison, we first started with classification of edits using the Bag of Words approach. We used Mallet[13] to create a vector space model based on unigrams from the edit difference containing inserted and changed text in the new revision of the article to create the Bag of Words and then used word count of the words after removing the stop words as a feature to the Naive Bayes classifier. We used Rank Boost on this weak classifier to maximize its area under the ROC curve. In this experiment, boosting the Naive Bayes classifier 15 times performed the best.

Building classifiers. We built classifiers using the features presented in the previous section with three different algorithms: Naive Bayes, C4.5 Decision Trees[18], and NBTrees[11] using Weka[8], a machine learning and data mining library provided by University of Waikato. The NBTree performed the best among all the other classifiers. An NBTree is a decision tree with Bayesian classifiers as its leaves. The classifier builds a decision tree to optimally partition the training data to build several Bayesian classifiers that will perform better than either a decision tree or Bayesian classifier alone.

We combined all the features described in the previous section and performed a 10-fold cross validation on all three algorithms. NBTree yielded the best result with the default settings, which uses 5-fold cross validation of Naive Bayes at a node to compute its utility and a split threshold of 5% of the relative reduction on error. Figure 1 shows the NBTree we constructed from the training data.

4 Evaluation & Results

Our test corpus contained 15000 edits. Registered users contributed 67% of these edits, while anonymous users contributed 33%. Among these 93.9% of edits were not in-



Regular Edits

Vandalism

Figure 2. The above Venn diagrams show how often the text of an article was inserted into, changed, deleted or any combination of these actions

Feature	Information Gain
Total number of author contributions	0.074
How long the author has been registered	0.067
If the author is a registered user	0.060
How frequently the author contributed in the training set	0.040
How often the article has been vandalized	0.035
How often the article has been reverted	0.034
The number of previous contributions by the author on the article	0.019
Change in sentiment score	0.019
Number of misspelled words	0.019
Sum of metrics	0.018

Table 1. Top ten features ranked by information gain

stances of vandalism. Not surprisingly, unregistered users more frequently vandalized articles than registered users, 16% to 1% respectively. See Figure 2 for Venn diagrams which highlight how often an article 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.

In Table 1, we present our top 10 ranked features according to information gain. Table 2 compares the performance of our classifiers using stratified 10-fold cross validation on the training data. Table 3 presents the results on the PAN 2010 Wikipedia Vandalism test corpus[1].

The authors of [17] 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. We report

Metric	NB+BoW	NB+BoW+RankBoost	NB	C4.5 Decision Tree	NBTree
Precision	27.8%	34.1%	15.8%	53.2%	**64.3%**
Recall	32.6%	26.6%	**93.2%**	36.9%	36.4%
Accuracy	87.5%	89.7%	69.2%	94.1%	**94.8%**
F-measure	30.1%	29.9%	27.1%	43.6%	**46.5%**
AUC	69%	62%	88.5%	80.5%	**91%**

Table 2. Overall performance of classifiers with 10 fold stratified cross validation on training set

Metric	Naive Bayes	C4.5 Decision Tree	NBTree
Precision	19.0%	51.0%	**61.5%**
Recall	**92.0%**	26.7%	25.2%
Accuracy	72.0%	91.6%	**92.3%**
F-measure	35.5%	35.1%	**35.8%**
AUC	86.6%	76.9%	**88.7%**

Table 3. Overall performance of classifiers on PAN 2010 testing data

statistics for both registered and unregistered users as well as edits made by those users or IP addresses that frequently contribute to the article in Table 4.

5 Discussion

Our NBTree-based classifier outperforms the baseline classifier for two reasons. First, the features we selected – editor information, sentiment change, etc. – convey more information about each edit than the simple bag of words model. Second, by using an NBTree, we enable the classifier to partition the diverse space of edits into regions that are more amenable to Bayesian classification.

Five of our top ten features involved editor information. Current Wikipedia bots make limited use of editor information, but our results show that they could do more. In fact, our NBTree 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.

Another observation yielded by our highest ranked features is that simple features outperformed our more sophisticated features. The only sophisticated feature that appeared in the top ten ranked attributes was change in sentiment. Figure 3 shows the values of the change in sentiment score. 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 zero change in sentiment score, 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)

Type of user	FP rate	Recall	Precision
Registered users	< 0.1%	22.0%	68.4%
Registered users that edited this article 10 times or more	<0.01%	0.0%	0.0%
Unregistered users	3.9%	40.8%	67.2%
IP addresses that edited this article 10 times or more	1.7%	33.3%	50.0%

Table 4. Performance of our classifier for registered and unregistered users as well as frequent contributors

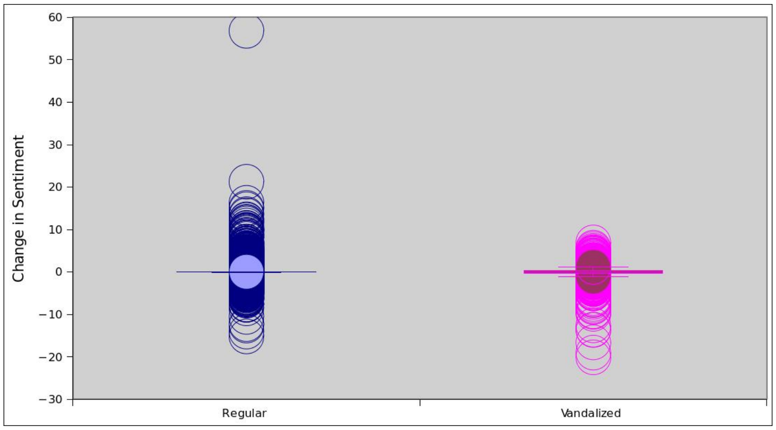


Figure 3. This box plot captures the change in sentiment between regular and vandalized edits. The outliers are shown as circles. The majority of edits had zero change in sentiment, but we do see that vandalized edits skew negatively and regular edits skew in a positive difference

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.

As mentioned before, the initial partition performed by the NBTree evaluated how many contributions a registered user or an IP address has made to Wikipedia. Interestingly the benefit of this feature is not exclusive to registered users. IP addresses that have contributed significantly in the past have a lower vandalism 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 geographic 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.

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.

Our classifier had tremendous difficulty classifying vandalism by registered users. A sizable chunk of these vandalizing edits come from users that registered just before submitting the edit, and therefore are very suspicious, but the rest of these vandalizing 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 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[14], we recommend that the edits labeled vandalism by frequent and active contributors to be examined more closely by an active Wikipedian. Fortunately, this constitutes a small minority of edits, but detecting vandalism among registered users is a challenging obstacle. In the future, we will examine techniques for training with unbalanced datasets to see if we can further improve our detection for this type of vandalism.

Lastly, we hypothesize the success of the NBTree arises from the fact that vandalism is too diverse to be captured by a "one-size-fits-all" approach and requires partitioning the training space to find edits with common features. In our experiments, the Weka toolkit allowed us to test several different types of classifiers including meta-classifiers, neural networks, support vector machines and decision trees. Ultimately, decision trees fared better than other classification schemes and the C4.5 and NBTree the best. Some partitions in the NBTree, as seen in Figure 1, take advantage of the fact that it is highly unlikely that edits which meet a certain criterion will contain vandalism (e.g. a frequent contributor to Wikipedia). We hope to explore how we can further identify particular types of vandalism and build precise and effective classifiers for them automatically.

6 Conclusion

This paper presents new features and algorithms to improve classification of vandalism. We found that features regarding information about the editor, misspellings, obscene words, detection of repetitive patterns and change in sentiment effectively classified vandalism. These features combined with the NBTree classifier gave an AUC score of 91% for a 10-fold stratified cross validation of our training corpus and 88.7% on the PAN 2010 testing corpus. We hope to expand upon these features and look for new ways to compare the content inserted, modified or deleted with previous revisions to determine if such actions constitute vandalism.

7 Acknowledgements

We would like to acknowledge Tamara Berg and Luis Ortiz for their guidance and suggestions.

References

1. 2010, P.W.: 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)
2. Alias-i.: Lingpipe 4.0.0. `http://alias-i.com/lingpipe/` (October 2008)
3. anonymous: English obscene wordlist. `http://expressionengine.com/?ACT=51&fid=8&aid=1830_mvLZ2WkoRucNvRegThbL&board_id=`
4. axelclk: gwtwiki - Java Wikipedia API (Bliki engine). `http://code.google.com/p/gwtwiki/`
5. Carter, C.: ClueBot. `http://en.wikipedia.org/wiki/Wikipedia:CLUEBOT`
6. Freund, Y., Iyer, R., Schapire, R.E., Singer, Y.: An efficient boosting algorithm for combining preferences (1998)
7. Grinberg, D., Lafferty, J., Sleator, D.: A robust parsing algorithm for link grammars. In: In Proceedings of the Fourth International Workshop on Parsing Technologies (1995)
8. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The weka data mining software: an update. SIGKDD Explor. Newsl. 11(1), 10–18 (2009)
9. Incava.org: Difference algorithm for Java. `http://www.incava.org/projects/java/java-diff/`
10. Itakura, K.Y., Clarke, C.L.A.: 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. pp. 822–823. ACM, New York, NY, USA (2009)
11. Kohavi, R.: Scaling up the accuracy of naive-bayes classifiers: A decision-tree hybrid. In: Second International Conference on Knoledge Discovery and Data Mining. pp. 202–207 (1996)
12. LLC., S.: English dictionary. `http://softcorporation.com/products/spellcheck/dictionaries/english/`
13. McCallum, A.: Mallet: A machine learning for language toolkit (2002)
14. Potthast, M.: Crowdsourcing a Wikipedia Vandalism Corpus. In: 33rd Annual International ACM SIGIR Conference (to appear). ACM (Jul 2010)

15. Potthast, M., Stein, B., Gerling, R.: Automatic vandalism detection in wikipedia. In: Macdonald, C., Ounis, I., Plachouras, V., Ruthven, I., White, R.W. (eds.) Advances in Information Retrieval, Lecture Notes in Computer Science, vol. 4956, chap. 75, pp. 663–668. Springer Berlin Heidelberg, Berlin, Heidelberg (2008)
16. Potthast, M., Stein, B., Gerling, R.: Automatic vandalism detection in wikipedia. In: ECIR'08: Proceedings of the IR research, 30th European conference on Advances in information retrieval. pp. 663–668. Springer-Verlag, Berlin, Heidelberg (2008)
17. Priedhorsky, R., Chen, J., Lam, S.T.K., Panciera, K., Terveen, L., Riedl, J.: Creating, destroying, and restoring value in wikipedia. In: GROUP '07: Proceedings of the 2007 international ACM conference on Supporting group work. pp. 259–268. ACM, New York, NY, USA (2007)
18. Quinlan, R.: C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, San Mateo, CA (1993)
19. Smets, K., Goethals, B., Verdonk, B.: 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). pp. 43–48. AAAI Press (2008)
20. West, A.G., Kannan, S., Lee, I.: Detecting wikipedia vandalism via spatio-temporal analysis of revision metadata. In: EUROSEC '10: Proceedings of the Third European Workshop on System Security. pp. 22–28. ACM, New York, NY, USA (2010)
21. Wikipedia: Definition of vandalism. `http://en.wikipedia.org/wiki/Wikipedia:Vandalism`
22. Wikipedia: WikiMarkup. `http://en.wikipedia.org/wiki/Help:Wiki_markup`