Automatic Vandalism Detection in Wikipedia: Towards a Machine Learning Approach

Koen Smets and **Bart Goethals** and **Brigitte Verdonk**
Department of Mathematics and Computer Science
University of Antwerp, Antwerp, Belgium
{koen.smets,bart.goethals,brigitte.verdonk}@ua.ac.be

Abstract

Since the end of 2006 several autonomous bots are, or have been, running on Wikipedia to keep the encyclopedia free from vandalism and other damaging edits. These expert systems, however, are far from optimal and should be improved to relieve the human editors from the burden of manually reverting such edits. We investigate the possibility of using machine learning techniques to build an autonomous system capable to distinguish vandalism from legitimate edits. We highlight the results of a small but important step in this direction by applying commonly known machine learning algorithms using a straightforward feature representation. Despite the promising results, this study reveals that elementary features, which are also used by the current approaches to fight vandalism, are not sufficient to build such a system. They will need to be accompanied by additional information which, among other things, incorporates the semantics of a revision.

Introduction

Since the inception of Wikipedia in 2001, the free encyclopedia, which is editable by everyone, has grown rapidly to become what it is today: one of the largest sources of adequate information on the Internet. This popularity translates itself to an ever growing large amount of articles, readers consulting them, editors improving and extending them . . . and unfortunately also in the number of acts of vandalism committed a day. By vandalism we understand every edit that damages the reputation of articles and/or users of Wikipedia. Priedhorsky et al. (2007) provide a survey of the typical categories of damages together with an empirically determined likeliness of occurrence. We list them here in decreasing order of appearance: introducing nonsense, offenses or misinformation; the partial deletion of content; adding spam (links); mass deletion of an article . . .

To fight vandalism, Wikipedia relies on the good faith of its users that accidentally discover damaged articles and, as in practice turns out, on the time-consuming efforts of its administrators and power users. To ease their job, they use special tools like Vandal Fighter to monitor the recent changes and which allow quick reverts of modifications matching regular expressions that define bad content or are performed by users on a blacklist. Since the end of 2006 some vandal bots, computer programs designed to detect and revert vandalism have seen the light on Wikipedia. Nowadays the most prominent of them are ClueBot and VoABot II. These tools are built around the same primitives that are included in Vandal Fighter. They use lists of regular expressions and consult databases with blocked users or IP addresses to keep legitimate edits apart from vandalism. The major drawback of these approaches is the fact that these bots utilize static lists of obscenities and 'grammar' rules which are hard to maintain and easy to deceive. As we will show, they only detect 30% of the committed vandalism. So there is certainly need for improvement.

We believe this improvement can be achieved by applying machine learning and natural language processing (NLP) techniques. Not in the very least because machine learning algorithms have already proven their usefulness for related tasks such as intrusion detection and spam filtering for email as well as for weblogs.

The remainder of this paper is as follows. First, we give a brief overview of related work, followed by a motivation for using machine learning to solve the problem. Next, we complement the most recent vandalism studies by discussing the performance results of the bots currently active on Wikipedia. Thereafter, we present the preliminary results of using a Naive Bayes classifier and a compression based classifier on the same features that serve as raw input for those bots. Finally, we formulate conclusions and outline the approach we plan to investigate next.

Related Work

Wikipedia has been subject to a statistical analysis in several research studies. Viégas, Wattenberg, and Dave (2004) make use of a visualization tool to analyze the history of Wikipedia articles. With respect to vandalism in particular, the authors are able to (manually) identify mass addition and mass deletion as jumps in the history flow of a page. Buriol et al. (2006) describe the results of a temporal analysis of the Wikigraph and state that 6 percent of all edits are reverts and likely vandalism. This number is confirmed by Kittur et al. (2007) in a study investigating the use of reverting as the key mechanism to fight vandalism. They also point out that only looking for reverts explicitly signaling vandalism is not strict enough to find evidence for most of the vandalism in the history of articles. The most recent study, to the best of our knowledge, by Priedhorsky et al. (2007) categorizes the different types of vandalism and their occurrence rate in a subset of 676 revision chains that were reverted. They confirm that reverts explicitly commented form a good approximation to spot damages, with a precision and recall of respectively 77% and 62%. Our work complements this last one, as we investigate a yet more recent version of the English Wikipedia history, and also analyse the decisions made by two bots. We also try to respond to the authors' request to investigate the automatic detection of damage. The authors believe in intelligent routing tasks, where automation directs humans to potential damage incidents but where humans still make the final decision.

There is strong cross-pollination possible between Wikipedia and several research areas. Wikipedia can benefit from techniques from the machine learning, information retrieval and NLP domains in order to improve the quality of the articles. Adler and de Alfaro (2007) build a content-driven system to compute and reflect the reputation of authors and their edits based on the time span modifications remain inside an article. Priedhorsky et al. (2007) use a closely related measure but they do not take into account the lifetime but the expected viewing time to rate the value of words. Rassbach, Pincock, and Mingus (2007) explore the feasibility of automatically rating the quality of articles. They use a maximum entropy classifier to distinguish six quality classes combining length measures, wiki specific measures (number of images, in/out links . . .) and commonly used features to solve NLP problems (part-of-speech usage and readability metrics). The problem to detect damages is related to ours in the sense that we need to rate the quality of a single revision instead of the whole article. The cross-pollination also holds for the other way around as machine learning, information retrieval and NLP can benefit from the use of Wikipedia. Gabrilovich and Markovitch (2007) use an explicit semantic interpreter built using articles from Wikipedia which is capable of measuring the semantic relatedness between text documents.

Recently, Potthast, Stein, and Gerling (2008) also use machine learning to detect vandalism in Wikipedia. Compared to their work, we have a larger labeled data set, use different classifiers, and most importantly, use different features. We aim to summarize an edit by focusing on the difference between the new and old version of an article, while Potthast, Stein, and Gerling use a set of 15 features that quantify the characteristics of vandalism.

Vandalism Detection and Machine Learning

The particular task to detect vandalism is closely related to problems in computer security: intrusion detection or filtering out spam from mailboxes and weblogs. It is a specific kind of web-defacement, but as the accessibility allows everyone to contribute, there is no need for crackers breaking into systems. We can see it as a form of content-based access control, where the integrity constraint on Wikipedia enforces that "All article modifications must be factual and relevant" as stated by Hart, Johnson, and Stent (2007). The problem also shares characteristics intrinsic to computer security problems. We need to deal with a skew and ever changing class distribution as the normal edits outnumber vandalism and both vandalism and legitimate edits are likely to change, due to respectively the adversarial environment and the rise of new articles or formatting languages.

Machine learning provides state of the art solutions to closely related problems. We put two techniques from the world of spam detection to the test. On one hand we use a well-known Naive Bayes classifier and on the other hand, as results from Naive Bayes models are significantly improved by state-of-the-art statistical compression models, a classifier based on probabilistic sequence modeling provided by Bratko et al. (2006).

Although we are aware that we will not be capable of identifying all types of vandalism (e.g. detecting misinformation in the pure sense is regarded as impossible without consulting external sources of information), we believe that machine learning might cope with this interesting, but far from trivial, problem.

Performance Analysis of Bots on Wikipedia

In this section we complement the work done by Priedhorsky et al. (2007) by analysing the results of the bots on one hour of data from the English version of Wikipedia. We show that there is still significant room for improvement in the automatic detection of vandalism. Furthermore, we provide additional evidence that the labeling procedure based on edit reverts, is quite sound. Next, we introduce the Simple English Wikipedia and present the results of a modified version of ClueBot on this data set, which we also use in our machine learning experiments later on. We start however with a short introduction to ClueBot's inner working.

ClueBot

ClueBot, written by Carter (2007), uses a number of simple heuristics to detect a subset of the types of vandalism mentioned above. First, it detects page replaces and page blanks relying on an auto-summary feature of MedaWiki software. Next, it categorizes mass delete, mass addition and small changes based on absolute difference in length. For the last three types, vandalism is determined by using a manually crafted static score list with regular expressions specifying the obscenities and defining some grammar rules which are hard to maintain and easy to by-pass. Negative scores are given to words or syntactical constructions that seem impossible in good articles, while wiki links and wiki transcludes are considered as positive. The difference between the current and the last revision is calculated using a standard *diff* algorithm. Thereafter, the inserted and deleted sentences are analysed using the score list and if this value exceeds a certain threshold vandalism is signaled. ClueBot further relies on the user whitelist for trusted users and increases its precision by only reverting edits done by anonymous or new users.

English Wikipedia (enwiki)

We analyse one hour of data from the first of March 2008 (00:00:00 - 00:59:59), restricting ourselves to the recent

	legitimate	reverted	mislabeled
1 hour	6944	323	26 (8.00%)
5 hours	28 312	1 926	

	ClueBot	VoABot II
1 hour	68 (22.89%)	33 (11.11%)
5 hours	349 (18.12%)	154 (8.00%)

Table 1: Edit statistics on English Wikipedia (2008.03.01).

changes of pages from the main namespace (0), the true encyclopedic articles, and ignore revisions from user or talk and discussion pages.

The data is automatically labeled by matching revision comments to regular expressions that signal a revert action, i.e. an action which restores a page to a previous version. This approach closely resembles the identification of the set of revisions denoted by Priedhorsky et al. (2007) as Damaged-Loose, a superset of the revisions explicitly marked as vandalism (Damaged-Strict).

While labeling based on commented revert actions is a good first order approximation, mislabeling cannot be excluded. If we regard vandalism as the positive class throughout this paper, then there will be both false positives and false negatives. The former arises when reverts are misused for other purposes than fighting vandalism like undoing changes without proper references or prior discussion. The latter occurs when vandalism is corrected but not marked as reverted in the comment, or when vandalism remains undetected for a long time. Estimating the number of mislabelings is very hard and manual labeling is out of question, considering the vast amount of data.

From the total of 6944 revisions, 4.65% are considered vandalism. Manual inspection demonstrates that of these 323, 11 are mislabeled as vandalism and for 15 others we are in doubt. So in the worst case we have to cope with a false positive rate of 8%.

Of the correctly labeled acts of vandalism 68 are identified by ClueBot and 33 by VoABot II, the two active vandal fighting bots on Wikipedia nowadays. Together this corresponds to a recall of 33%. Hence the bulk of the work is still done by power users and administrators. All vandalism identified by the two bots is true vandalism so the precision during this one hour is 100%.

Priedhorsky et al. (2007) identify that around 20% of their labeled data is misinformation, a number confirmed by our manual inspection. Even disregarding those, the above analysis reveals there is much room for improvement with respect to the recall. Numerical analysis on a data set including the next four hours, see Table 1, shows that these numbers remain invariant, as they are only multiplied by a factor of 5.

Simple English Wikipedia (simplewiki)

As a proof of concept and because of storage and time constraints, we run the preliminary machine learning experiments on Simple English Wikipedia, a user-contributed online encyclopedia intended for people whose first language is not English. This encyclopedia is much smaller in size compared to the standard English Wikipedia as shown in Table 2. There are no bots in operation that try to remove spam or vandalism. Nevertheless the articles are also subject to vandalism, which often last longer as fewer readers and users are watching the pages.

	pages	revs	xml.bz2
simplewiki	53 449	499 395	88.7 MB
enwiki	11 405 052	167 464 014	133.0 GB

Table 2: Size (Simple) English Wikipedia expressed in terms of the total number of pages, revisions and the compressed file size of the pages-meta-history files available at `http://download.wikimedia.org`.

period	nr (%) of vandalism	revs	pages
2003 - 2004	21 (1.12)	1 870	784
2004 - 2005	276 (2.03)	13 624	2541
2005 - 2006	2 194 (5.60)	39 170	6626
2006 - 2007	12 061 (8.33)	144 865	17 157
2007 - ...	12 322 (6.96)	177 165	22 488
2003 - ...	26 874 (7.13)	376 694	28 272

Table 3: Estimated vandalism statistics of the revisions together with the number of revisions and pages from the main namespace in Simple English Wikipedia (2007.09.27).

We work with the dump from 2007.09.27 and again we only consider the main articles disregarding pages from other namespaces. Labeling using the same procedure shows that the amount of vandalism, as we see in Table 3, is fairly stable and comparable with the percentage on enwiki.

As a reference, we provide the performance of a modified version of ClueBot on the simplewiki data set in Table 4. We use our own implementation based on the source code of the one running at enwiki, with that difference that we only consider the heuristics to detect vandalism and do not take into account the dynamic user whitelist.

We notice in Table 4 a drop in both precision and recall. The former can possibly be explained by not using the dynamic user white list, while the fact that the static score list of the ClueBot is manually tailored towards the English Wikipedia could explain the drop in recall. A more thorough study, including manually analysing the decisions of the ClueBot, is required before we can further explain the decreased performance.

	ACC	PRE	REC	F_1
2003 - 2004	0.9752	0.1250	0.0476	0.0689
2004 - 2005	0.9722	0.2577	0.0905	0.1340
2005 - 2006	0.9346	0.4185	0.0761	0.1288
2006 - 2007	0.9144	0.6207	0.1306	0.2158
2007 - ...	0.9320	0.6381	0.1774	0.2777
2003 - ...	0.9270	0.6114	0.1472	0.2372

Table 4: Performance of ClueBot (without user whitelist) on Simple English Wikipedia.

delete	Vandalism is almost always a crime; different types of vandalism include: graffiti, smashing the windows of cars and houses, and rioting. {{stub}}
insert	Being *** is almost always a crime; different types of **** *** include: doggy style.
change delete	Vandalism property vandal graffiti website vandals funny attention vandal Vandals
change insert	******* of as *** **** *** **** site **** *** *** *** ****** ***_*******
comment	
user group	anonymous

Table 5: Censored feature list of revision 29853 from the Vandalism page in Simple Wiki English.

Experiments

In this section, we will discuss the setting for our machine learning experiment conducted on simplewiki, the Simple English version of Wikipedia. We first consider the data representation. Thereafter we give a brief description of two learning algorithms put to test: a Naive Bayes classifier on bags of words (BOW) and a combined classifier built using probabilistic sequence modeling (Bratko et al. 2006), also referred to in the literature as statistical compression models.

Revision Representation

In this case study we use the simplest possible data representation. As for ClueBot and VoABot II, we extract raw data from the current revision and from the history of previous edits. This first step could be seen as making the static scoring list of ClueBot dynamic. This should provide a baseline for future work. In particular, for each revision we use its text, the text of the previous revision, the user groups (anonymous, bureaucrat, administrator . . .) and the revision comment. We also experimented with including the lengths of the revisions as extra features. The effect on overall performance is however minimal and thus we discard them in this analysis. Hence the focus lies here more on the content of an edit.

As the modified revision and the one preceding it differ slightly, it makes sense to summarize an edit. Like ClueBot, we calculate the difference using the standard *diff* tool. Processing the output gives us three types of text: lines that were inserted, deleted or changed. As the changed lines only differ in some words or characters from each other, we again compare these using *wdiff*. Basically, this is the same as what users see when they compare revisions visually using the MediaWiki software. Table 5 gives an example of the feature representation used throughout this paper, applied to a vandalized revision.

To evaluate our machine learning experiments we use 60% of the labeled data for training and the remaining 40% for evaluation purposes. We do not aim to statistically analyse the different approaches but use it more as a guide to conduct our search towards a machine learning based vandalism detection tool.

BOW + Naive Bayes

As a first attempt we use the Naive Bayes implementation from the 'Bow' toolkit (McCallum 1996) as learning mechanism to tackle the problem. This tool treats each feature as a bag of words and uses Porter's stemming algorithm and stop word removal to decrease the size of the feature space. Next, we train a Naive Bayes classifier on each of the features separately. Our final classifier combines the results of the individual classifiers by multiplying the obtained probability scores.

Probabilistic Sequence Modeling

Probabilisitic sequence modeling (PSM) forms the foundation of statistical compression algorithms. The key strength of compression-based methods is that they allow constructing robust probabilistic text classifiers based on character-level or binary sequences, and thus omit tokenization and other error-prone pre-processing steps. Nevertheless, as clearly stated by Sculley and Brodley (2006), they are not a "parameter free" silver bullet for feature selection and data representation. In fact they are concrete similarity measures within defined feature spaces. Commonly used statistical compression algorithms are dynamic Markov compression (DMC) and prediction by partial matching (PPM), both described in detail by Bratko et al. (2006). Basically these are n-gram models where weights are implicitly assigned to the coordinates during compression. Empirical tests, in above references, show that compression by DMC and PPM outperforms the explicit n-gram vector space model due to this inherent feature weighting procedure. For the implementation we use PSMSlib (Bratko 2006), which uses the PPM algorithm.

During the training phase a compression model M_c^f is built (Bratko et al. 2006) for each feature f in Table 5 and for each class c (vandalism or legitimate). The main idea is that sequences of characters generated by a particular class will be compressed better using the corresponding model. In theory, an optimal compression can be achieved if one knows the entropy given that model. In order to classify a revision r, we estimate for each of its feature values x the entropy H by calculating,

$$H_c^f(r) = \frac{1}{|x|} \log \prod_{i=1}^{|x|} p(x_i | x_{i-k}^{i-1}, M_c^f),$$

where $p(x_i | x_{i-k}^{i-1}, M_c^f)$ is the probability assigned by model M_c^f to symbol x_i given its k predecessors. In order to score the revision, we combine all features by summing over the entropies,

$$S_c(r) = \sum_f H_c^f(r)$$

and then calculating the log ratio

$$S(r) = \log \frac{S_{van}(r)}{S_{leg}(r)}.$$

If the value S exceeds a prespecified threshold, default 0, we assign the revision to the vandalism class otherwise we consider it as legitimate. The threshold parameter trades off the precision and the recall.

Analysis and Discussion

In this section we discuss the results of the two attempts to put machine learning to work on the Simple English data set.

BOW + Naive Bayes

Table 6 shows the results on the test set of the final Naive Bayes classifier only taking into account the revision diff features as bags of words. Table 7 shows the same, this time including the user group information together with revision comments. While the precision in these tables is almost the same as in Table 4, a significant increase can be noticed in terms of recall and F_1, especially when including user group information and comment.

Table 8 shows the results on the whole data set of the classifiers based on a single feature ignoring the probability of the class priors. This provides more insight in the influence of the different features.

As expected, we see that the '(change) delete'-feature contributes little more than noise, while the 'change insert' is the most decisive factor. Next, we observe a seemingly important contribution of the 'change delete'-feature with respect to the recall. This may be due to the fact that some pages are vandalised more than others. It is, however, not a decisive feature as it contributes little to the overall result in terms of precision.

The domination of the 'user group'-feature on the recall can be easily explained by combining the facts that anonymous users commit most of the vandalism, but that their overall legitimate contribution to Wikipedia is rather small.

Note that when ignoring the probability of the class prior in the Naive Bayes classifier on all features, as shown by the last line in Table 8, the recall is higher but at the same time there is a drop in the precision.

	ACC	PRE	REC	F_1
2003 - 2004	0.9748	0.4000	0.4444	0.4210
2004 - 2005	0.9648	0.3007	0.3603	0.3278
2005 - 2006	0.9235	0.3701	0.2941	0.3278
2006 - 2007	0.9266	0.6975	0.3266	0.4449
2007 - ...	0.9310	0.5949	0.1960	0.2948
2003 - ...	0.9303	0.6166	0.2503	0.3561

Table 6: Results Naive Bayes using the revision diff features in a BOW.

	ACC	PRE	REC	F_1
2003 - 2004	0.9794	0.5000	0.4444	0.4705
2004 - 2005	0.9635	0.2937	0.3783	0.3307
2005 - 2006	0.9165	0.3427	0.3439	0.3433
2006 - 2007	0.9261	0.6161	0.4800	0.5396
2007 - ...	0.9342	0.5911	0.3453	0.4359
2003 - ...	0.9314	0.5882	0.3694	0.4538

Table 7: Results Naive Bayes including user group information and revision comments.

2003 - ...	ACC	PRE	REC	F_1
delete	0.8618	0.1476	0.2813	0.1936
insert	0.9585	0.2636	0.2670	0.2653
change delete	0.5002	0.1079	0.5307	0.1794
change insert	0.9068	0.6486	0.2068	0.3136
comment	0.8729	0.2360	0.2894	0.2600
user groups	0.8444	0.3102	0.8319	0.4520
	0.9057	0.4181	0.5667	0.4812

Table 8: Results individual Naive Bayes classifiers (ignoring class priors).

Probabilistic Sequence Modeling

Table 9 shows the overall performance together with the results of the individual models on the same test set. Interesting to note is that the recall is much higher, but that the precision drops unexpectedly. We lack a plausible explanation for this strange behaviour, but the effect can be diminished by setting the threshold parameter to a score higher than zero. This is shown in Figure 1, where we plot the precision/recall curves for varying thresholds for the probabilistic sequence models and for the Naive Bayes models, both with and without user groups and comments. The marks show the results when the log ratio threshold is equal to 0. The tendency is that, despite the worse behavior shown in Table 9, the overall accuracy measured in term of precision and recall is better for the compression based models than for the bag of words model using Naive Bayes.

2003 - ...	ACC	PRE	REC	F_1
delete	0.1568	0.0809	0.9567	0.1493
insert	0.5031	0.1274	0.9281	0.2241
change delete	0.2891	0.0805	0.7867	0.1461
change insert	0.5028	0.1177	0.8362	0.2064
	0.8554	0.3117	0.7201	0.4351
comment	0.7978	0.2667	0.9233	0.4138
user groups	0.8460	0.3171	0.8598	0.4633
	0.8436	0.3209	0.9171	0.4755

Table 9: Results Probabilistic Sequence Modeling classifiers.

To boost the overall performance we will need additional information. We believe that incorporating weighted semantics derived from explicit semantic analysis, as described by Gabrilovich and Markovitch (2007), is necessary. The intuition is that the semantics of offenses, nonsense and spam are likely to differ from the semantics of the revised article and hence are an important feature for classification. Moreover, we believe that the 'text deleted'-feature contains more

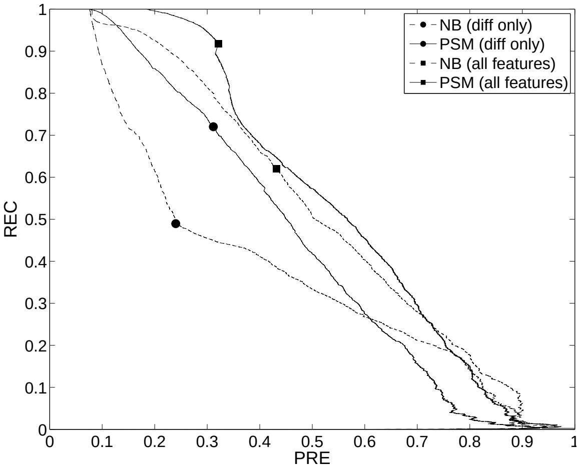

Figure 1: Precision/Recall curves: Naive Bayes versus Probabilistic Sequence Modeling for revision diff features with(out) user groups and comment.

information than is apparent from the current results, where it appears to be merely a noise factor. To exploit the usefulness of this feature, we will take into account its effect on the semantic level by measuring the text life, i.e. the value of the deleted words, as suggested by Adler and de Alfaro (2007).

Conclusions and Future Work

As far as we know, we are among the first to try machine learning techniques to answer the need of improving the recall of current expert systems, which are only capable of identifying 30% of all vandalism. We demonstrate that, by applying two machine learning algorithms, a straight forward feature representation and using a set of noisy labeled examples, the accuracy of the actual running bots can be improved. We feel confident that this study is merely a starting point and that there is much room for improvement. In the end almost all vandalism that is not related to misinformation should be detectable automatically, without consulting third-party information.

For future work, we will combine the ideas from Gabrilovich and Markovitch (2007) and Adler and de Alfaro (2007) to enhance the feature representation. We aim to rebuild their explicit semantic interpreter and use it for semantic comparison between the current modified revision and the previous versions of an article. We will compare the concepts related to text inserted and deleted, and weight these features using respectively the authority of authors and the value of words expressed in text life or expected viewing rate. In this context, we plan to compare our effort to the work of Potthast, Stein, and Gerling (2008).

Acknowledgements

We thank Walter Daelemans for suggesting statistical compression models, Jacobi Carter and Aaron Schulz for providing the source code of ClueBot and VoAbot II. Koen Smets is supported by a Ph. D. fellowship of the Research Foundation - Flanders (FWO).

References

Adler, B. T., and de Alfaro, L. 2007. A Content-Driven Reputation System for the Wikipedia. In *Proceedings of the 16th International World Wide Web Conference (WWW)*. ACM Press.

Bratko, A.; Cormack, G. V.; Filipič, B.; Lynam, T. R.; and Zupan, B. 2006. Spam Filtering using Statistical Data Compression Models. *Journal of Machine Learning Research* 7(Dec):2673–2698.

Bratko, A. 2006. PSMSLib: Probabilistic Sequence Modeling Shared Library. Available at `http://ai.ijs.si/andrej/psmslib.html`.

Buriol, L. S.; Castillo, C.; Donato, D.; Leonardi, S.; and Stefano, M. 2006. Temporal Analysis of the Wikigraph. In *Proceedings of the IEEE/WCIC/ACM International Conference on Web Intelligence (WI)*, 45–51.

Carter, J. 2007. ClueBot and Vandalism on Wikipedia. Unpublished. Available at `http://24.40.131.153/ClueBot.pdf`.

Gabrilovich, E., and Markovitch, S. 2007. Computing Semantic Relatedness using Wikipedia-Based Explicit Semantic Analysis. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI)*, 1606–1611.

Hart, M.; Johnson, R.; and Stent, A. 2007. Content-Based Access Control. Submitted to the IEEE Symposium on Privacy and Security.

Kittur, A.; Suh, B.; Pendleton, B. A.; and Chi, E. H. 2007. He Says, She Says: Conflict and Coordination in Wikipedia. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*.

McCallum, A. K. 1996. Bow: a Toolkit for Statistical Language Modeling, Text Retrieval, Classification and Clustering. Available at `http://www.cs.cmu.edu/~mccallum/bow`.

Potthast, M.; Stein, B.; and Gerling, R. 2008. Automatic Vandalism Detection in Wikipedia. In *Proceedings of the 30th European Conference on IR Research (ECIR)*, 663–668.

Priedhorsky, R.; Chen, J.; Lam, S. T. K.; Panciera, K.; Terveen, L.; and Riedl, J. 2007. Creating, Destroying, and Restoring Value in Wikipedia. In *Proceedings of the International ACM Conference on Supporting Group Work*.

Rassbach, L.; Pincock, T.; and Mingus, B. 2007. Exploring the Feasibility of Automatically Rating Online Article Quality. In *Proceedings of the International Wikimedia Conference (Wikimania)*. Wikimedia.

Sculley, D., and Brodley, C. E. 2006. Compression and Machine Learning: a New Perspective on Feature Space Vectors. In *Proceedings of the Data Compression Conference (DCC)*, 332–341.

Viégas, F. B.; Wattenberg, M.; and Dave, K. 2004. Studying Cooperation and Conflict between Authors with *history flow* Visualizations. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*.