ASTES

Advances in Science, Technology and Engineering Systems Journal Vol. 2, No. 1, 16-22 (2017)

www.astesj.com

Special Issue on Computer Systems, Information Technology, Electrical and Electronics Engineering

ASTESJ

ISSN: 2415-6698

The Class Imbalance Problem in the Machine Learning Based Detection of Vandalism in Wikipedia across Languages

Arsim Susuri[*], Mentor Hamiti, Agni Dika

Faculty of Contemporary Sciences and Technologies, South East European University, Tetovo, Macedonia

ARTICLE INFO

Article history:
Received: 30 November, 2016
Accepted: 14 January, 2017
Online: 28 January, 2017

Keywords:
Machine learning
Wikipedia
Vandalism
Class imbalance

ABSTRACT

This paper analyses the impact of current trend in applying machine learning in detection of vandalism, with the specific aim of analyzing the impact of the class imbalance in Wikipedia articles. The class imbalance problem has the effect that almost all the examples are labelled as one class (legitimate editing); while far fewer examples are labelled as the other class, usually the more important class (vandalism). The obtained results show that resampling strategies: Random Under Sampling (RUS) and Synthetic Minority Oversampling Technique (SMOTE) have a partial effect on the improvement of the classification performance of all tested classifiers, excluding Random Forest, on both tested languages (simple English and Albanian) of the Wikipedia. The results from experimentation extended on two different languages show that they are comparable to the existing work.

1. Introduction

Ever since its inception, in 2001, Wikipedia has continuously grown to become one the largest information source on the Internet. One of its unique features is that it offers the ability to anyone to edit the articles. This popularity, in itself, means that, a number of articles can be read, edited, and enhanced by different editors and, inevitably, be subject to acts of vandalisms through illegitimate editing.

This paper is an extension of work originally presented in [1], by addressing the issue of class imbalance in the detection of vandalism in Wikipedia articles across languages.

Vandalism means any type of editing which damages the reputation of an article or a user in Wikipedia. A list of typical vandalisms along with their chances of appearance was created as a result of empirical studies done by Priedhorsky et al. [2]. Typical examples include massive deletions, spam, partial deletions, offences and misinformation. In order to deal with vandalism, Wikipedia relies on the following users:

- Wikipedia users' ability and willingness to find (accidentally or deliberately) damaged articles
- Wikipedia administrators and
- Wikipedia users with additional privileges

These users use special tools (e.g. Vandal Fighters) to monitor recent changes and modifications that enable retrieval of bad expressions or which are implemented by blacklisted users.

Wikipedia was subject to different statistical analysis from various authors. Viégas et al. [3] uses visualization tools to analyze the history of Wikipedia articles. When it comes to vandalism, authors were able to identify (manually) massive deletions as a jump in the history flow of a particular article page. Since late 2006, some bots (computer programs designed to detect and revert vandalism), have appeared on Wikipedia. These tools are built on the primitive included in the Vandal Fighters. These use lists of common phrases, and consult databases containing blocked users or IP addresses in order to separate legitimate editing from vandalism.

One drawback of these approaches is emphasized that these world use static list of obscenities and grammatical rules which are difficult to maintain and easily "fooled". These detect only 30% of vandalisms committed. Consequently, there is a need to improve the detection of this kind. One of the possible improvements is the application of machine learning.

The prior success implemented in interference detection, spam filtering for email, etc., is a good indicator for the opportunity that the machine learning shows in improvements in detecting vandalism in Wikipedia [4].

[*]Corresponding Author: Arsim Susuri, Faculty of Contemporary Sciences and Technologies, South East European University, Tetovo, Macedonia
Email: arsimsusuri@gmail.com

2. Wikipedia Vandalism Detection

To define the vandalism detection task, we have to define some key concepts of MediaWiki (the wiki engine used by Wikipedia).

An article is composed of a sequence of revisions, commonly referred to as the article history. A revision is the state of an article at a given time in its history and is composed of the textual content and metadata describing the transition from the previous revision [5]. Revision metadata contains, among others, the user who performed the edit, a comment explaining the changes, a timestamp, etc. An edit is a tuple of two consecutive revisions and should be interpreted as the transition from a given revision to the next one.

Evaluating a vandalism detection system requires a corpus of pre-classified edits. Our focus is on four different corpora:

- PAN-WVC-10 - The PAN Wikipedia Vandalism Corpus 2010 (PAN-WVC-10), compiled in 2010 via Amazon's Mechanical Turk comprises 32439 edits from 28468 English Wikipedia articles of which 2394 have been annotated as vandalism. The dataset was created by 753 human annotators by casting 193022 votes, so that each edit has been annotated at least three times, whereas edits that were difficult to be annotated received more than three votes [6]. The PAN-WVC-10 was first used in the 1st International Competition on Wikipedia Vandalism Detection [7].
- PAN-WVC-11 - The PAN Wikipedia Vandalism Corpus 2011 (PAN-WVC-11) from 2011 is an extension of the PAN-WVC-10. It was used in the 2nd International Competition on Wikipedia Vandalism Detection [8]) and is the first multilingual vandalism detection corpus. The corpus comprises 29949 Wikipedia edits in total (9985 English edits with 1144 vandalisms, 9990 German edits with 589 vandalisms, and 9974 Spanish edits with 1081 vandalism annotations).
- Wikipedia History Dump - Wikipedia records all revisions of all articles and all other Wikipedia pages and releases them as XML or SQL dump files. For this research dumps 29.10.2015 have been used for:
  - simplewiki and
  - sqwiki.

2.1. Approaches based on Machine Learning

Since 2008 Wikipedia vandalism detection based on machine learning approaches has become a field of increasing research interest. In table 1 existing vandalism detection approaches from the literature are shown, depicting Precision, Recall and PR-AUC (Precision Recall – Area Under Curve) values.

Potthast et al. [9] contributed the first machine learning vandalism detection approach using textual features as well as basic metadata features with a logistic regression classifier. Smets et al. [10] used a Naive Bayes classifier on a bag of words edit representation and were the first to use compression models to detect Wikipedia vandalism. Itakura and Clarke [11] used Dynamic Markov Compression to detect vandalism edits on Wikipedia.

Mola Velasco [12] extended the approach of Potthast et al. [9] by adding some additional textual features and multiple wordlist-based features. He was the winner of the 1st International Competition on Wikipedia Vandalism Detection [7]. West et al. [13] were among the first to present a vandalism detection approach solely based on spatial and temporal metadata, without the need to inspect article or revision texts.

Similarly, a vandalism detection system on top of their WikiTrust reputation system was built by Adler et al. [14 and 15]. Adler et al. [16] combined natural language, spatial, temporal and reputation features used in their aforementioned works [12, 13 and 14].

Besides Adler et al. [16], West and Lee [17] were the first to introduce ex post facto data as features, for whose calculation also future revisions have to be considered.

Supporting the current trend of creating cross language vandalism classifiers, Tran and Christen [18] evaluated multiple classifiers based on a set of language independent features that were compiled from the hourly article view counts and Wikipedia's complete edit history.

3. Objectives

The objectives of this research work were to experimentally compare four classifiers on unbalanced data with and without resampling on four different corpora: PAN-WVC-10, PANWVC-11, Simple English Wikipedia (simplewiki) and Albanian Wikipedia (sqwiki) history dumps, respectively.

We compare four different classifiers Logistic Regression, RealAdaBoost, BayesNet, and Random Forest regarding their performances using RUS and SMOTE.

Based on this experiment we try to build a model that would be able to represent the impact of class imbalance on the detection of vandalism across languages, for small scaled datasets.

3.1. The Class Imbalance Problem

The problem of used vandalism corpora (Webis-WVC-07, PAN-WVC-10 and PAN-WVC-11) is that they offer data that are highly skewed. What this means is that the ratio between the number of vandalism and regular edits is highly imbalanced (5-7% of all samples are annotated as vandalism edits).

Learning traditional classifiers with such datasets can cause lower detection performance. Based on surveys of classifying imbalanced data by He and Garcia [20] and Ganganwar [21] we can list three reasons for performance decline:

- If classifier learning is based on minimizing the overall error, then the minority class instances contribute little to the error. This results in an increase of bias of the classifier towards the majority class.
- Many classifiers assume a balanced class distribution of the minority and the majority class, which is not often the case when working with realistic scenarios.
- Often classifiers implicitly assume equal costs for misclassification for both classes, which is often not sensible: for example, the cost for classifying cancer as non-cancer is way higher than the other way round.

Table 1: Vandalism detection classification obtained from various authors

Authors	Data	Classifier	Precision	Recall	PR-AUC	Corpora
Smets et al. [10]	Unbalanced	Probabilistic Sequence Modeling	0.3209	0.9171	-	Simplewiki
Smets et al. [10]	Unbalanced	Naive Bayes	0.4181	0.5667	-	Simplewiki
Tran and Christen [18]	Balanced	Gradient Tree Boosting	0.870	0.870	-	Historical Dump
Potthast et al. [9]	Unbalanced	Logistic Regression	0.830	0.870	-	Webis-WVC-07
Velasco [5]	Unbalanced	Random Forest	0.860	0.570	0.660	PAN-WVC-10
Adler et al. [14]	Unbalanced	ADTree	0.370	0.770	0.490	PAN-WVC-10
Adler et al. [16]	Unbalanced	Random Forest	-	-	0.820	PAN-WVC-10
West and Lee [17]	Unbalanced	ADTree	0.370	0.770	0.490	PAN-WVC-10
Harpalani et al. [19]	Unbalanced	LogitBoost	0.606	0.608	0.671	PAN-WVC-10
West and Lee [17]	Unbalanced	ADTRee	-	-	0.820	PAN-WVC-11

In general, there are two approaches to overcome the class imbalance problem:

- The data level that involves several training data resampling techniques.
- The algorithmic level that involves adjusting the misclassification costs or probabilistic estimates, e.g., at the tree leaves of decision tree classifiers, as well as learning classifier models solely based on minority class samples (so-called one-class classification).

During the examination of the impact of training dataset resampling on vandalism detection performance we find that, in most cases, resampling reduces the performance of the tested classifiers. Logistic Regression, Real Ada Boost and Bayesian Network classifiers benefit from certain resampling strategies, whereas a Random Forest classifier turns out to be relatively unaffected by resampling approaches.

3.2. Evaluating Resampling Techniques

One approach to overcome performance issues of classifiers is resampling the training dataset in order to balance the classes. There are several common approaches to do so, namely random under sampling, random oversampling, directed over and under sampling and hybrid methods which combine the aforementioned [20].

3.2.1. Resampling Strategies

Random under sampling (RUS) removes a certain amount of randomly picked majority class instances from the training dataset. RUS leads to class balancing or, in an extreme case, even to majority class removal. However, a disadvantage of RUS is the loss of possibly decisive instances. Since important information for the class separation is likely to be removed, this technique might induce a lower classification performance.

Random over sampling (ROS) reproduces a certain amount of randomly chosen minority class samples. Thus, the class distribution can be adjusted towards a uniform distribution. Since classifiers, after oversampling, are trained by using some minority class values multiple times, the learned model is likely to over fit.

The Synthetic Minority Oversampling TEchnique (SMOTE) by Chawla et al. [22] over samples the minority class by computing artificial instances. The feature values of these samples are calculated by random interpolation of the K-nearest neighbors' feature values (typically K = 5). The method aims at avoiding over fitting while oversampling minority class instances. Han et al. [23] extend SMOTE to use only the minority class samples at the class borderline (borderlineSMOTE) in order to generate artificial data which is more important for classification.

3.2.2. The Classifiers

Our focus is on Logistic Regression and Random Forest, since they are used by Potthast et al. [9], Mola Velasco [12], and Adler et al. [16]. Additionally, we consider RealAdaBoost as a state-of-the-art Boosting algorithm and a Bayesian Network classifier as a Bayes approach that is reported to outperform the Naive Bayes classifier used by Smets et al. [10].

Logistic Regression analysis estimates the relationship between a dependent variable and one or more independent variables.

RealAdaBoost (Friedman et al. [24]) is a boosting algorithm based on Adaptive Boosting (AdaBoost) by Freund and Schapire [25]. Boosting is a method to enhance classification performance by combining many weak base classifiers (weak hypotheses) in order to create a more powerful classifier.

A Bayesian Network (Pearl and Russell [26]) is a directed acyclic graph. The nodes in the graph represent random variables; the arcs signify direct correlations between these variables. Tree Augmented Naive Bayes (TAN) described by Friedman et al. [27] has been used in the experiments. In our experiment, each attribute in the graph has only the class value and at most one other attribute as parents.

Random Forest Random Forest (Breiman [28]) is an ensemble learning technique, constructing a defined number of decision tree predictors and combining them to a predictor set (forest). The individual trees are learned from randomly chosen feature subsets and represent independent and identically distributed random vectors. Each tree is grown to full depth.

To classify a new data sample, the final class is determined by the mode of classes that are predicted by the individual trees.

3.2.3. Datasets

We use two datasets: the complete edit history of Wikipedia in Simple English and Albanian[1], and the hourly article view count[2].

The edit history data dump is that of 1 December 2015 for both, the Simple English Wikipedia and for Albanian Wikipedia [1]. In figure 1 the number of articles and edits revisions (per month and per year) are shown.

3.2.3.1. Labeling Vandalized Revisions

From the raw revision data, every revision is reduced to a vector of features described in table 2. The reasons for selecting these features are independence from language and simplicity.

Every revision's comment is scrutinized for keywords of "vandal" and "rvv" (revert due to vandalism), which would signal a vandalism in the previous revision. Afterwards, these revisions are marked as vandalism.

In order to correctly arrange the timestamp of revisions with the corresponding article view dataset, we round up the revision time to the next hour. This ensures that the hourly article views reference the correct revision when combining the two datasets.

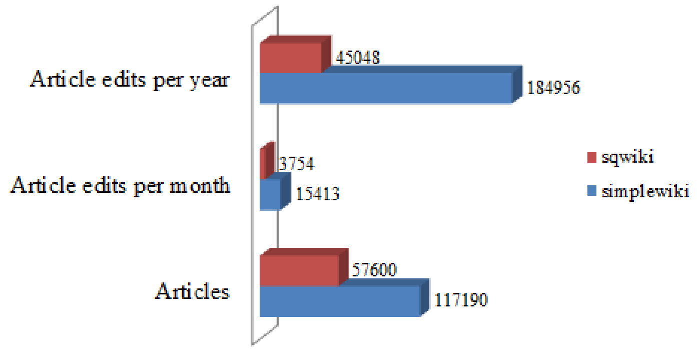

Figure 1: Statistical data of editing history for simplewiki and sqwiki

The arrangement is implemented on all revisions and should not affect classification.

1.1.1.1. Article Views

The raw article view dataset is structured by views of article aggregated by hour. We use the process of applying the transformation and filtering of articles viewed in the revision dataset above (table 3), as used in [1].

Extraction of the redirect articles from the revision dataset is applied and then all access to redirect articles is changed to the canonical article. These extra view counts are aggregated accordingly. These article views are important to seeing the impact of vandalism on Wikipedia [2]. Having in mind that the average time vandalism is active is 2.1 days [5], a lot hours are for unsuspecting readers to face vandalized content.

Table 2: Description of editing history dataset

Attribute	Description
Title of Article	Unique identifier of a Wikipedia article.
Hour Timestamp	The timestamp of the revision.
Anonymous Edit	The editor of this revision is considered to be anonymous if an IP address is given. 0 for an edit by a registered user, and 1 for an edit by an anonymous user.
Vandalism	This revision is marked as vandalism by analyzing the comment of the following revision. 0 for regular edit, and 1 for vandalism.

However, the behavior of vandals may also be seen in a change in access patterns, which may be from vandals checking on their work, or that article drawing attention from readers and their peers [29].

Table 3: Description of article view dataset

Attribute	Description
Name of Project	MediaWiki project, (Simple English - "simplewiki" and Albanian -"sqwiki").
Hour Timestamp	In the format of DDMMYYYY-HH0000.
Title of Article	The title of the Wikipedia article.
Number of Requests	The number of requests during that hour.

The edit history dataset is scanned in order to be sure that these article views occurred when articles are in a vandalized state. Then we apply labelling of all article views of observed vandalized or non-vandalized revisions.

The unknown views from revisions made before 2015, or articles without revisions in this 4-month period under study, are discarded. Thus, we have an article view dataset labelled with whether the views are of vandalized revisions.

The resulting size of the data is identical to the resulting dataset in the following subsection. This labeled article view dataset allows us to determine whether view patterns can be used to predict vandalism.

From this resulting dataset, we split the "Hour Timestamp" attribute into an "hour" attribute. This allows the machine learning algorithm to learn daily access patterns.

1.1.1.2. Resulting Dataset

The resulting dataset is created by merging two-time series datasets for each language. The dataset is constructed by adding

[1] http://dumps.wikimedia.org/backup-index.html

[2] http://dumps.wikimedia.org/other/pagecounts-raw/

features from the labeled revision dataset to the labeled article view dataset by repeating features of the revisions. Thus, for every article view, we have information on what the properties are and whether this revision was viewed.

We use the "hour" attribute split from the timestamp in the article views dataset. Thus, we have the following 6 features in our resulting dataset: hour, size of the comment, size of article, anonymous edit, number of requests, and vandalism.

These features are language independent and capture the metadata of revisions commonly used, and access patterns. Article name is not included in the resulting dataset because access patterns of vandalized articles may be similar to other vandalized articles, regardless of the name of the article.

To apply the classification algorithms, we split the resulting dataset by date into a training set (September to November) and a testing set (December), as shown in figure 2.

2. Experiments and Results

For SMOTE oversampling we use an amount of 50% and 100% of the original vandalism class instances ($SMOTE_{50}$ and $SMOTE_{100}$).

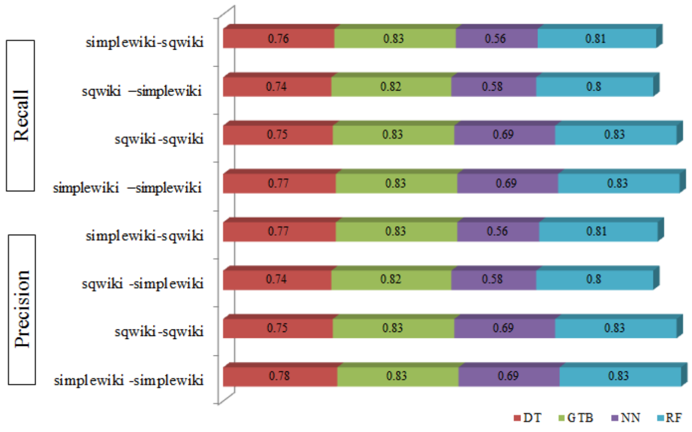

Figure 2: Revisions Dataset – Classifications results

Additionally, we chose an oversampling amount of 1300% which leads to an almost balanced dataset without loss of any majority class instances - 13613 (48.81%) vandalism, 14281 (51.19%) regular.

On PAN-WVC-11 we use a SMOTE oversampling of 1100% instead of 1300%, due to the different class distribution in that corpus (1100% oversampling leads to 28728 (48.88%) vandalism and 30045 (51.12%) regular samples). Table 5 provides the corresponding PR-AUC values.

Using RUS on the training data, all classifiers but RealAdaBoost on PAN-WVC-11 show a performance drop on all four corpora.

For Logistic Regression (on both corpora) and Random Forest (on PAN-WVC-10) RUS leads to the lowest overall performance.

If a classifier already handles class imbalance internally, RUS only removes majority class data that is needed to train the model without benefiting from a balanced data set. For Real Ada-Boost the loss of regular samples seems not to be as influential as the training on a balanced dataset.

SMOTE Logistic Regression benefits from $SMOTE_{50}$ (on PAN-WVC-10) and from $SMOTE_{100}$ (on PAN-WVC-11). Both oversampling strategies result in the best overall performances on the respective corpora. Similar results have also been obtained on simplewiki and sqwiki.

SMOTE oversampling leads to a high performance drop for the RealAdaBoost classifier. Over-sampling the target class with SMOTE causes a slight decrease of performance if the Random Forest classifier is used on both corpora. The performance for BayesNet increases for lower oversampling proportions (50% and 100%) on PAN-WVC-11, $SMOTE_{50}$ even leads to the highest overall performance. On PAN-WVC-10 all SMOTE proportions result in a performance drop for the BayesNet classifier.

On PAN-WVC-10 for Logistic Regression and Random Forest, a higher oversampling proportion leads to lower performance. This is also the case for all classifiers but Logistic Regression on PAN-WVC-11. For all classifiers on both corpora $SMOTE_{1100/1300}$ lead to the lowest classification performance using SMOTE approaches. An exception is the RealAdaBoost classifier (on PAN-WVC-11), for which $SMOTE_{1100}$ outperforms the other proportions.

A reason for the observed lower performance using SMOTE might be the absence of significant data in the training and test corpora. If the vandalism samples given in the test dataset represent other vandalism types than those given in the training set, some kinds of vandalism will never be found.

Wikipedia vandalism has been found to be a heterogeneous problem [30]. Hence, an underrepresentation of vandalism edits from certain categories in the training corpora would not be surprising, since the samples have been chosen randomly ([6], and [8]). In the case of missing decisive vandalism samples, oversampling would not produce a more accurately defined vandalism class region, but would only insert further weak samples.

3. Conclusions and Future Work

Summarizing our experiments, we can conclude that RealAdaBoost is most affected by the imbalance of the training data. Random Forest shows only little sensitivity to resampling approaches. However, it turns out to be the best performing classifier of all evaluated approaches, without applying resampling strategies, as shown in table 4.

We compared different resampling strategies applied on four classifiers, Logistic Regression, RealAdaBoost, BayesNet and Random Forest. We observed that examined resampling strategies (RUS and SMOTE) had a partial increase of the classification performance for all tested classifiers, except for Random Forest. However, regarding the total classification performance, Random Forest, trained with the original data set, outperforms all other approaches.

The reasons for the poor improvement by resampling techniques can be found in the class overlapping or due to class imbalance of the four corpora training datasets, given our chosen feature set.

With these experiments we have shown that the class imbalance has a similar impact on various datasets across languages in terms of the detection of vandalism rates.

Table 4: Values of PR-AUC for the resampling strategies

Classifier	Resampling strategies				
	None	RUS	SMOTE$_{50}$	SMOTE$_{100}$	SMOTE$_{1300/1100}$
PAN-WVC-10					
Logistic Regression	.584	.478	.586	.578	.545
RealAdaBoost	.612	.561	.476	.485	.443
BayesNet	.627	.596	.615	.616	.537
RandomForest	.675	.621	.667	.664	.634
PAN-WVC-11					
Logistic Regression	.633	.615	.636	.646	.616
RealAdaBoost	.534	.536	.495	.453	.499
BayesNet	.651	.643	.663	.656	.562
RandomForest	.744	.726	.738	.732	.706
Simple English					
Logistic Regression	.636	.615	.632	.623	.608
RealAdaBoost	.498	.536	.497	.443	.487
BayesNet	.623	.643	.652	.642	.560
RandomForest	.736	.734	.726	.722	.716
Albanian					
Logistic Regression	.621	.615	.636	.626	.610
RealAdaBoost	.487	.536	.495	.439	.479
BayesNet	.611	.632	.628	.640	.552
RandomForest	.726	.728	.718	.717	.709

For future work, more investigation is needed to point out the within-class imbalance properties in the PAN-WVC corpora and in the Wikipedia history dumps regarding certain feature sets.

References

[1] Susuri, M. Hamiti and A. Dika, Machine Learning Based Detection of Vandalism in Wikipedia across Languages. In proceedings of the 5th Mediterranean Conference on Embedded Computing (MECO), Bar, Montenegro, (2016).

[2] R. Priedhorsky, J. Chen, S. T. K. Lam, K. Panciera, L. Terveen, and J. Riedl. Creating, destroying, and restoring value in Wikipedia. In proceedings of the international ACM conference on supporting GroupWork (GROUP), Sanibel Island, FL, 259-268 (2007).

[3] F. B. Viégas, M. Wattenberg, and K. Dave. Studying cooperation and conflict between authors with history flow visualizations. In proceedings of the ACM Conference on human factors in computing systems (CHI), Vienna, Austria, 575-582 (2004).

[4] M. Hamiti, A. Susuri and A. Dika, Machine Learning and the Detection of Anomalies in Wikipedia, Recent Advances in Communications, Proceedings of the 19th International Conference on Communications, Zakynthos Island, Greece, (2015).

[5] Santiago M. Mola-Velasco. Wikipedia Vandalism Detection. - WWW 2011 - Ph.D. Symposium, Hyderabad, India. (2011).

[6] Martin Potthast. Crowdsourcing a wikipedia vandalism corpus. Proceeding of the 33rd international ACM SIGIR conference on research and development in information retrieval - SIGIR '10, 789, (2010).

[7] Martin Potthast, Benno Stein, and Teresa Holfeld. Overview of the 1st International Competition on Wikipedia Vandalism Detection. In Martin Braschler, Donna Harman, and Emanuele Pianta, editors, Working Notes Papers of the CLEF 2010 Evaluation Labs, September (2010).

[8] Martin Potthast and Teresa Holfeld. Overview of the 2nd International Competition on Wikipedia Vandalism Detection. In Vivien Petras, Pamela Forner, and Paul D. Clough, editors, Notebook Papers of CLEF 11 Labs and Workshops, September (2011).

[9] Martin Potthast, Benno Stein, and Robert Gerling. Automatic vandalism detection in wikipedia. In advances in information retrieval, 663-668. Springer Berlin Heidelberg, (2008).

[10] Koen Smets, Bart Goethals, and Brigitte Verdonk. Automatic vandalism detection in wikipedia: Towards a machine learning approach. In WikiAI '08: Proceedings of the AAAI Workshop on Wikipedia and Artificial Intelligence, (2008).

[11] Kelly Y. Itakura and Charles L. a. Clarke. Using dynamic markov compression to detect vandalism in the Wikipedia. Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval - SIGIR '09, 822, (2009).

[12] Mola Velasco Santiago Moisés Mola Velasco. Wikipedia vandalism detection through machine learning: Feature review and new proposals - lab report for pan at clef 2010. In CLEF (Notebook Papers/LABs/Workshops), (2010).

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

[14] B. Thomas Adler, Luca De Alfaro, and Ian Pye. Detecting wikipedia vandalism using wikitrust. Notebook papers of CLEF, (2010).

[15] B. Thomas Adler and Luca De Alfaro. A content-driven reputation system for the wikipedia. Proceedings of the 16th international conference on World Wide Web WWW 07, 7(Generic):261, (2007).

[16] B. Thomas Adler, Luca De Alfaro, Santiago M. Mola-Velasco, Paolo Rosso, and Andrew G. West. Wikipedia vandalism detection: Combining natural language, metadata, and reputation features. In proceedings of the 12th International Conference on Computational Linguistics and Intelligent Text Processing - Volume Part II, 277-288, Berlin, Heidelberg, (2011).

[17] Andrew G. West and Insup Lee. Multilingual vandalism detection using language-independent & ex post facto evidence - notebook for pan at clef 2011. In CLEF (Notebook Papers/Labs/Workshop), (2011).

[18] Khoi-Nguyen Tran and Peter Christen. Cross Language Prediction of Vandalism on Wikipedia Using Article Views and Revisions. Advances in Knowledge Discovery and Data Mining, 268-279 (2013).

[19] Manoj Harpalani, Michael Hart, S Signh, Rob Johnson, and Yejin Choi. Language of Vandalism: Improving Wikipedia Vandalism Detection via Stylometric Analysis. ACL (Short Papers), 83-88 (2011).

[20] Haibo He and Edwardo A. Garcia. Learning from imbalanced data. IEEE Transactions on Knowledge and Data Engineering, **21**(9): 1263-1284 (2009).

[21] Vaishali Ganganwar. An overview of classification algorithms for imbalanced datasets. International Journal of Emerging Technology and Advanced Engineering, **2**(4): 42-47 (2012).

[22] Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. Smote: Synthetic minority over-sampling technique. Journal of Artificial Intelligence Research," **16**(1): 321-357 (2002).

[23] Hui Han, Wen-Yuan Wang, and Bing-Huan Mao. Borderline-smote: A new over-sampling method in imbalanced data sets learning. In Proceedings of the 2005 International Conference on Advances in Intelligent Computing - Volume Part I, ICIC'05, 878-887, Berlin, Heidelberg, (2005).

[24] J. Friedman, T. Hastie, and R. Tibshirani. Additive Logistic Regression: a Statistical View of Boosting. The Annals of Statistics, **38**(2): (2000).

[25] Yoav Freund and Robert E. Schapire. Experiments with a new boosting algorithm. In Lorenza Saitta, editor, Machine Learning, Proceedings of the Thirteenth International Conference (ICML '96), Bari, Italy, July 3-6, 1996, 148-156. Morgan Kaufmann, (1996).

[26] Judea Pearl and Stuart Russell. Bayesian networks. In M. Arbib, editor, Handbook of Brain Theory and Neural Networks, number R-277. MIT Press, (2001).

[27] Nir Friedman, Dan Geiger, and Moises Goldszmidt. Bayesian network classifiers. Machine Learning, **29**(2-3): 131-163 (1997).

[28] Leo Breiman. Random forests. Machine Learning, **45**(1): 5-32 (2001).

[29] K. Tran, P. Christen, "Cross-language prediction of vandalism on wikipedia using article views and revisions". Proceedings of the 17th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD) (2013).

[30] Si-Chi Chin and W. Nick Street. Enriching wikipedia vandalism taxonomy via subclass discovery. In LDH, 19-24, (2011).