Engineering Applications of Artificial Intelligence 78 (2019) 248–259

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

Can deep learning techniques improve classification performance of vandalism detection in Wikipedia?

Juan R. Martinez-Rico, Juan Martinez-Romo *, Lourdes Araujo

NLP & IR Group, Dpto. Lenguajes y Sistemas Informáticos, Universidad Nacional de Educación a Distancia (UNED), Madrid 28040, Spain

ARTICLE INFO

Keywords:
Vandalism
Wikipedia
Natural language processing
Deep learning
Word embedding

ABSTRACT

Wikipedia is a free encyclopedia created as an international collaborative project. One of its peculiarities is that any user can edit its contents almost without restrictions, what has given rise to a phenomenon known as vandalism. Vandalism is any attempt that seeks to damage the integrity of the encyclopedia deliberately. To address this problem, in recent years several automatic detection systems and associated features have been developed. This work implements one of these systems, which uses three sets of new features based on different techniques. Specifically we study the applicability of a leading technology as deep learning to the problem of vandalism detection. The first set is obtained by expanding a list of vandal terms taking advantage of the existing semantic-similarity relations in word embeddings and deep neural networks. Deep learning techniques are applied to the second set of features, specifically Stacked Denoising Autoencoders (SDA), in order to reduce the dimensionality of a bag of words model obtained from a set of edits taken from Wikipedia. The last set uses graph-based ranking algorithms to generate a list of vandal terms from a vandalism corpus extracted from Wikipedia. These three sets of new features are evaluated separately as well as together to study their complementarity, improving the results in the state of the art. The system evaluation has been carried out on a corpus extracted from Wikipedia (WP_Vandal) as well as on another called PAN-WVC-2010 that was used in a vandalism detection competition held at CLEF conference.

1. Introduction

Wikipedia is an international project with a collaborative nature that aims to maintain an encyclopedia thanks to the generous contribution of thousands of volunteers. At the moment of writing this research work, Wikipedia had more than 35 million articles in 288 languages, the sixth overall web site for daily visits and page views according to Alexa.[1] For this reason, it has become a major source of on-line information used by both end-users and third-party applications. Users can edit any encyclopedia article anonymously and virtually without restrictions and changes are published immediately. This freedom of action, which has encouraged its rapid growth in recent years, is also the cause of one of its main problems: vandalism. According to the definition of Wikipedia, vandalism is any addition, deletion or change of content made deliberately to compromise the integrity of the encyclopedia. Common forms of vandalism are the insertion of obscenities or rude humour, removal of content, introduction of absurd fragments in articles, or the modification of articles so that favour certain points of view or interests. Contributions made in good faith to improve the encyclopedia, even if they are wrong or harmful, are not considered vandalism.

Although the percentage of the total vandalism edits is low (4.64% according to the only study carried out and completed by Wikipedia[2] in 2007, or 7% achieved when the corpus PAN-WVC-10 (Potthast, 2010) was built in 2010) and the most flagrant cases are reversed in a matter of minutes, the mere fact of their existence can undermine its reputation and the trust that users can have in the information it provides. To mitigate this problem, there are a number of automatic systems[3] and filters[4] that detect and correct the most obvious vandalism edits or directly prevent them. However, human intervention to detect the rest of cases is necessary, which is an effort and dedication that could be used in other more constructive tasks.

The main objective of this work is the study of new features and its combination with state-of-the-art features that a machine learning system can use to detect the presence of vandalism on Wikipedia and implementing them in a vandalism detection system. Another challenge is the application of deep neural networks for the construction of features from a bag of words model, obtained from the set of words inserted and removed in an edit of Wikipedia. The main contributions of this work are the following:

[3] http://en.wikipedia.org/wiki/User:ClueBot_NG
[4] http://en.wikipedia.org/wiki/Wikipedia:Edit_filter.

* Corresponding author.
E-mail addresses: jrmartinezrico@invi.uned.es (J.R. Martinez-Rico), juaner@lsi.uned.es (J. Martinez-Romo), lurdes@lsi.uned.es (L. Araujo).

[1] http://www.alexa.com/topsites at Feb 2017.
[2] http://en.wikipedia.org/wiki/Wikipedia:Counter-Vandalism_Unit/Vandalism_studies/Study1.

https://doi.org/10.1016/j.engappai.2018.11.012
Received 13 October 2017; Received in revised form 16 November 2018; Accepted 26 November 2018
Available online 14 December 2018
0952-1976/© 2018 Elsevier Ltd. All rights reserved.

- We have proposed a new method based on word embeddings to automatically extract terminology used by classifiers to identify vandalism and not-vandalism editions.
- We have applied a deep learning technique to automatically obtain features to characterize the vandalism editions reducing the dimensionality of the input data.
- We have also designed new features based on different ranking algorithms applied to a co-occurrence graph. It is constructed by connected terms appearing in the same windows of terms from the considered kind of document (vandalism or not).
- We have constructed a new corpus to be able to compare our work with a recent work in the state of the art that extracts features based on edits and revisions not present in PAN corpus.
- We have checked that the combination of the different sets is able to get significant improvements in the performance.
- We have been able to improve the state of the art results for the problem, even when the level of previous results was very high.

The remainder of the paper proceeds as follows: Section 2 defines the problem of vandalism detection and offers a summary of the approaches that have been explored so far to solve it; Section 3 presents the proposed system and the two strategies used to address this task: techniques based on graphs and semantic similarity for building lists of vandalism terms, and the use of deep neural networks for feature extraction; Section 4 is devoted to describe the evaluation corpus and the methodology adopted for evaluation; Section 5 describes the experiments proposed as well as the results obtained on a public dataset; Finally, Section 6 draws the main conclusions and future work.

2. Problem definition and previous work

Vandalism is any addition, deletion, or change in the content made deliberately to compromise the integrity of the encyclopedia. However these actions can be performed in different ways: inclusion of vulgar words or jargon, easily detectable using regular expressions or a list containing these kind of terms, substitution within an article of certain terms for other similar but with a different meaning, deletion of parts of an article, etc.

Fig. 1 shows an example of vandalism in which much of the page corresponding to an American city has been replaced by an offensive phrase. The previous text offered details about the location, but the phrase "Santa Clarita is a toilet!" has replaced that information.

As it can be seen in Fig. 2, Wikipedia provides a service to show all the edits on an article where the text appears in a special format that highlights the differences between two consecutive changes (an edit). The left side contains various paragraphs of the article before the edit, and the right side shows the new text. In this case, numerous paragraphs have been replaced by an offensive phrase in line 77.

Based on the guidelines used in the CLEF conference (Potthast and Holfeld, 2011), an edit $e = (r_{t-1}, r_t)$ is defined as the transition between two consecutive revisions r_{t-1}, r_t of a Wikipedia article. Let E be the set of all available edits in Wikipedia, the task of vandalism detection consists in determining whether an edit e has been made in bad faith or not. In order to materialize this idea by means of a machine learning system the following is necessary: a corpus $E_c \subset E$ of edits previously labelled with a class value $K \subset \{0, 1\}$, an edit model $\alpha : E \rightarrow E$ that corresponds to each edit e, a vector ϵ of numerical values called features, in which each value identifies certain feature which is indicative of vandalism, and a classifier $K : E \rightarrow \{0, 1\}$ that assigns to each feature vector a class value within the set $\{0, 1\}$ where 0 represents a regular edit and 1 represents a vandalism edit.

2.1. Background

The first tools developed to fight vandalism in Wikipedia used regular expressions to detect offensive terms, heuristics that combined these regular expressions with metadata and statistics extracted from the edits to determine whether they were of vandalism nature. It is not until 2006 when the first machine learning based systems appear, focused on vandalism detection and other closely related tasks such as determining the quality of an edit or assigning a reputation score to an author. The main contributions made in this area of research are reviewed below.

Currently there are three different approaches for the detection of vandalism in Wikipedia: based on the author's reputation, based on the analysis of the metadata edits, and based on text features.

2.1.1. Preservation of contributions and reputation

Within this group, there are proposals that try to identify signs of vandalism from the lifecycle of the fragments of an article, and assign different levels of reputation to authors based on the preservation of their contributions.

Zeng et al. (2006) calculated the certainty of an article using the history of their revisions. This can be extended to the calculation of the certainty of fragments of the article and the trust in their authors. Revisions are viewed as a set of insertions and deletions. Javanmardi et al. (2010) studied the problem of reputation assignment to authors. They only analyse the reputation of administrators and vandals, not from common users, in order to build the authors' models. The reputation is estimated according to the stability (surviving time) of insertions made by users. Contrary to what happened in previous works, in detecting the authorship of a revision, rearrangements of tokens are also considered. Adler et al. (2010) proposed a vandalism detection system based on WikiTrust, a previous work by the same authors. They distinguish two aspects of the problem: the historical vandalism detection and instantaneous vandalism detection. WikiTrust calculates the quality of each revision based on the persistence of this revision, the author's reputation and the reputation of the reviewers. Wu et al. (2010) addressed the problem of elusive vandalism. In this type of vandalism, less obvious than usually seen, very specific changes that are difficult to detect are carried out, such as the change of numerical values by others similar. To detect these changes they use a measure based on the stability of the text. Suzuki and Yoshikawa (2013) developed a system for determining the quality of an article by means of the survival of edits method, iteratively until convergence, and taking into account the editor quality according to the quality of articles he edited (link analysis). The system is able to discriminate vandals edits that remove quality text. Segall and Greenstadt (2013) built a set of features that help to detect reversed edits based on the user history and metadata from those edits. Features from users, articles and edits are used. The final classification is performed using a support vector machine approach.

2.1.2. Spatio-temporal models

Some studies have mainly used the spatio-temporal information present in metadata or infoboxes to detect the presence of certain types of vandalism.

West et al. (2010) used the spatio-temporal information present in the metadata of a revision to detect vandalism. They labelled the corpus consulting the *roll-backs* that were produced and a light classifier that can be used as an initial detector for other anti-vandalism tools. The results were similar to those obtained by other authors that used natural language processing techniques. Alfonseca et al. (2013) proposed a vandalism detection system that used only the information in infoboxes and its temporal evolution. They took advantage of the infoboxes, which typically have a more structured information that the body of the article, in order to extract temporal information from the evolution of the attributes of an article and build a set of features from it that was used to detect vandalism.

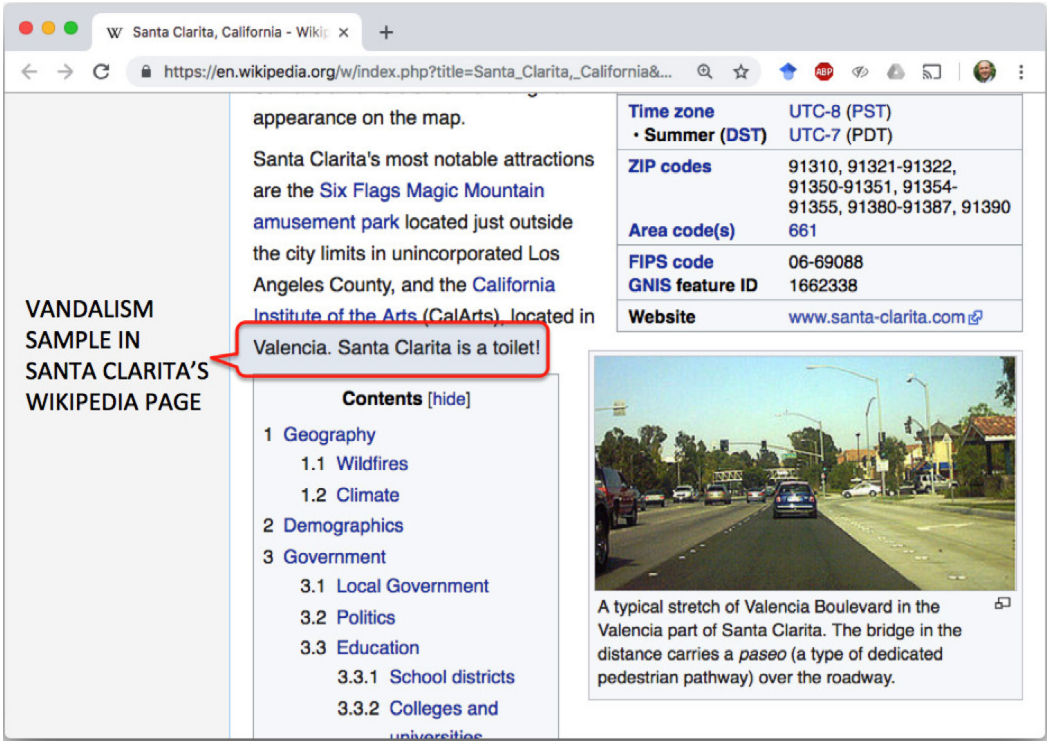

Fig. 1. Example of vandalism: modified text at November 29th, 2009.

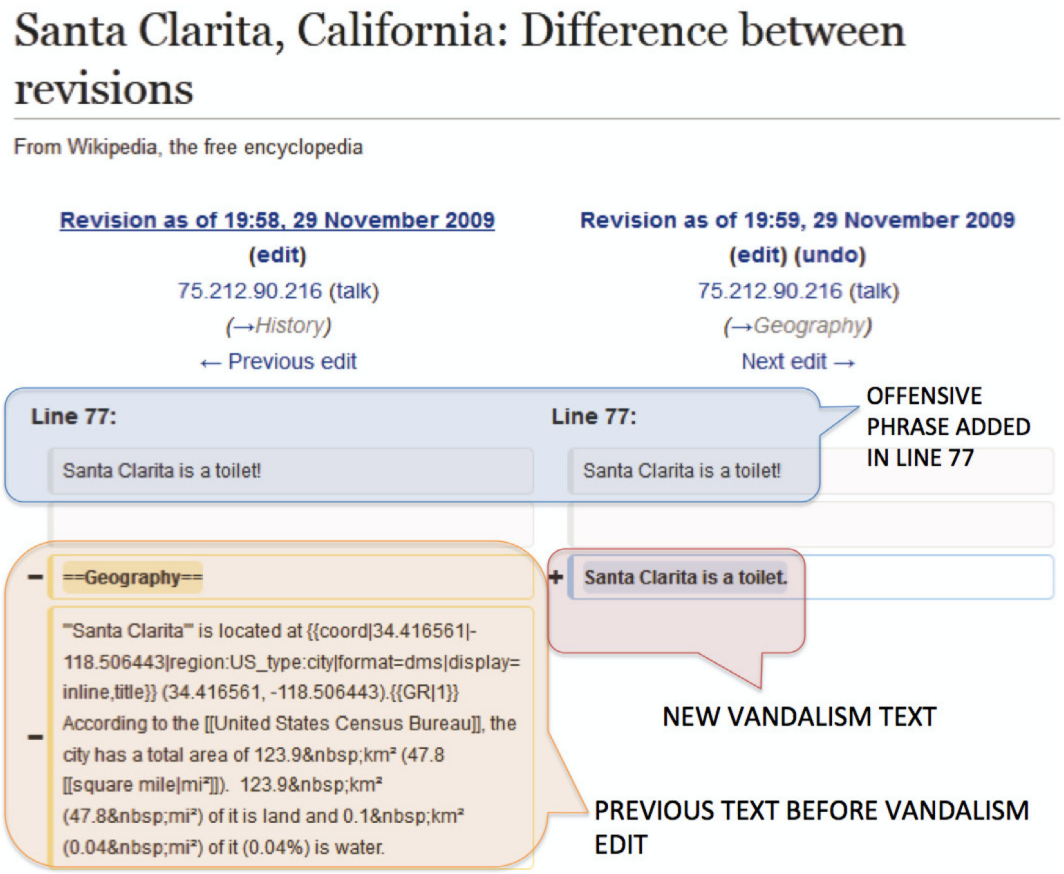

Fig. 2. Example of vandalism: previous paragraph and added vandalized phrase.

2.1.3. Text-based features and combinations

Finally, previous works more similar to the one presented in this article are exposed. First, systems based on features extracted from the text are stated. Then, those systems that have combined features from different nature to build a more efficient classifier are described.

Itakura and Clarke (2009) dealt with vandalism detection using a machine learning approach, based on Markov dynamic compression. To do this, they only consider two types of vandalism: insertion and change. The basic idea is to concatenate regular editions on one hand, and vandalism edits on the other hand, in two separate files. The one with higher compression ratio indicates the nature of the edit to be tested. Wu et al. (2012) developed a system for determining the quality of an article or frequency features based on graphic elements such as the motives. A motive is a graphical representation of patterns that can occur in author–article relationships. They used four machine learning algorithms: random forest, logistic regression, support vector machine and k-neighbours. Tran and Christen (2013) proposed a new set of text-based features. To do this, they analysed the differences between revisions marked as corrections by vandalism and the revisions associated with vandalism. The authors also use a multi-language strategy to analyse how *bots* and humans detect vandalism, and confirm that a classifier trained on a particular language can have a good performance when used with a different language.

Works that present a system based on a single set of features are very interesting to advance in the problem of vandalism detection. However, systems with a better performance usually combine several sets of features to improve the state of the art. Some of such systems are presented below.

Belani (2010) focused on using a standard bag-of-words in which the vocabulary is built on the set of inserted and removed words in different edits. In addition, features based on metadata (for instance, whether the author is anonymous or not) are added to the vectors of this model. In his work this model is evaluated with a logistic regression classifier. Mola-Velasco (2011) developed a vandalism detection system based on a set of new features and others previously implemented by other authors. Some of the new features take into account statistical data and metadata from edits, while others are based on counts by type of words used for that predefined lists of words (vandalism, bias, jargon, etc.). This system won the vandalism detection competition held during the 2010 PAN conference. West and Lee (2011) developed a multilingual vandalism detection system using immediate features (*zero delay*) being language-independent (some reused from previous works by other authors), and *ex post facto* features, i.e. that use known information after the article edit was done as Adler et al. (2010) did (historical detection). Harpalani et al. (2011) proposed a system based on stylometric features. The authors use probabilistic context-free grammars (*PCFG*) to capture deep syntactic regularities in different edits of an article. Then, a system is built in which these features are combined with others based on metadata, lexical clues (offensive vocabulary terms, etc.), and feelings (using subjective terms rather than neutral). Javanmardi et al. (2011) selected a minimal set of features for the detection of vandalism divided into four groups: user, text, metadata, and language models, using a MapReduce (Dean and Ghemawat, 2008) scheme and a Random Forest classifier. Sumbana et al. (2012) proposed an active sampling method that allows reducing the training corpus gathering examples with values of redundant features and thus reducing the effort to be made by annotators. In addition, the authors use a *LAC* (*Lazy Associative Classification*) classifier to build a vandalism detection system. This kind of classifier is based on the premise that if there are strong relationships between feature values and classes, these relationships can be used to predict the class. Shulhan and Widyantoro (2016) apply machine learning techniques by training Cascaded Random Forest classifier on a corpus that has been re-sampled using Local Neighbourhood Synthetic Minority Oversampling Technique (LNSMOTE). Deep Learning has also been used for vandalism detection on Wikipedia as in the work of Kumar et al. (2015). The authors use a set of user editing patterns as features to classify some users as vandals. Their approach uses a set of features derived from a transition probability matrix and then reduces it via a neural net auto-encoder to classify some users as vandals. For this task the authors have built a new dataset consisting of about 33K Wikipedia users (including both a black list and a white list of editors) and containing 770K edits. Adler et al. (2011) integrated three of the approaches to Wikipedia vandalism detection: a spatio-temporal analysis of metadata, a reputation-based system, and natural language processing features. The authors examined in detail the contribution of the three approaches, both for the task of discovering fresh vandalism, and for the task of locating vandalism in the complete set of Wikipedia revisions. Lately there have appeared works like the one of Heindorf et al. (2016) that is based on another platform like Wikidata using some features of previous works. The authors present a new machine learning-based approach to detect vandalism in Wikidata. They propose a set of 47 features that exploit both content and context information, and report on 4 classifiers of increasing effectiveness tailored to this learning task. Recently new works have appeared with an across languages approach as the work of Tran et al. (2015). It proposes a cross-language vandalism detection technique that scales to the size of the full Wikipedia and extends the types of vandalism detectable beyond past feature-based approaches. The authors use word dependencies to identify vandal words in sentences by combining part-of-speech tagging with a conditional random fields classifier. Also the work of Susuri et al. (2016) applies machine learning algorithms for detecting vandalism in two languages as English and Albanian. The authors propose using a list of classifiers in one language, and then evaluate them across languages in two datasets: the hourly count of views of each Wikipedia article, and the used edit history of articles.

3. Vandalism detection system

The main objective of this work is to propose new features, that combined with some previously introduced by other authors, are able to improve the state of the art. In particular, we have applied deep learning techniques to obtain these features, given the potential of these methods and the recent interest showed by the scientific community. For that, three sets of features have been designed which individually focus on different problems of vandalism, and together have a competitive performance.

The extracted features come from three different approaches:

- The first one, is based on the generation of vandalism term lists. It provides 4 features; two of them focus on frequency and impact, and the other two focus on the words that are found in a vandal cluster.
- The second set of features comes from the deep neural network and generates 25 features. These 25 features are obtained from the different combinations used when selecting the number of neurons in the three input layers (X, Y and Z).
- Finally, the set of features from graph-based ranking algorithms generates 8 features. To do this, four different social media measures are used that focus on different aspects in a graph (PageRank, Degree centrality, Eigenvector centrality, and Hits). Over the output of these measures, frequency and impact methods are applied for each of them, generating the final 8 features.

Fig. 3 shows the general scheme of system functioning. First, the system loads the corpus information into a SQL database in which intermediate representations are also stored once generated. From these text representations, and from the metadata present in the editions, the different obtained features are calculated and stored in the database too. Finally, the predictions are performed through a machine learning software. As for the colour coding, the yellow elements represent the processes, the blue ones are the system I/O and databases, and the green ones are the sets of extracted features.

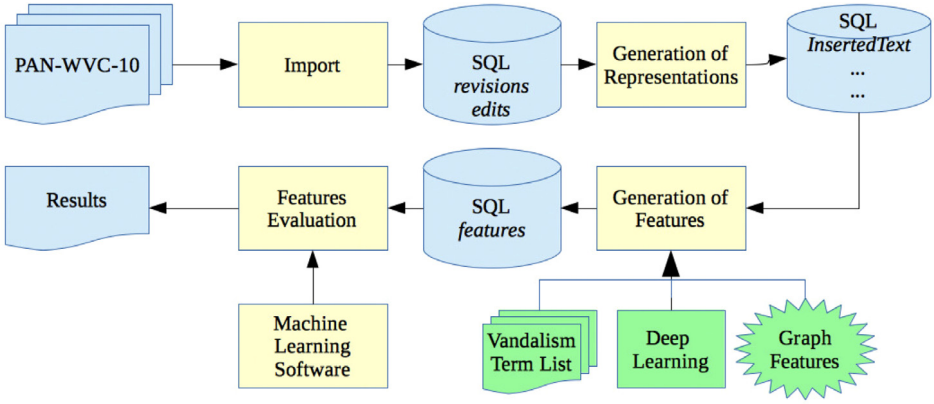

Fig. 3. General scheme of system functioning for the PAN-WCV-10 corpus. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

The evaluation of the system has been done with a corpus extracted from Wikipedia (WP_Vandal) and with the corpus PAN-WVC-10, since it is the reference collection in the field of vandalism detection in Wikipedia. Some previous works in the state of the art extract features that cannot be obtained from the PAN-WVC-10 corpus, so a corpus extracted from Wikipedia has also been used for the evaluation in order to obtain these features. We have built the vandalism corpus WP_Vandal[5] extracted from Wikipedia and constructed according to the methodology followed for the construction of the corpus PAN-WVC-10 (Potthast, 2010), since it is the reference collection in the field of vandalism detection in Wikipedia. This corpus is described in more detail below.

3.1. Automatic extraction of vandalism terms based on word embeddings

In language models based on word or semantic vectors, each word is represented by a vector of real values, unlike traditional vector space models where one word is represented atomically within a document vector. One drawback of the classical model is that two document vectors containing words with close meanings as "car" and "vehicle" will not have any element in common and therefore a relationship between them cannot be easily established based on these two terms.

In this work we will use a representation model of word embeddings that is generated by artificial neural networks. These representation models are usually linked to the generation of a language model. A language model or n-gram model is any probabilistic model that predicts the next word according to the $t-1$ previous ones. There exist several works that use this model of word embeddings as the one of Bengio (2009). However, a simpler variant of this model is the proposal by Mikolov et al. (2013a) that has been used in this work.

3.1.1. Generation of vandalism term lists

Given the observed features in the word embeddings, the possibility of using them as a mechanism to expand lists of terms arises. In the case of vandalism detection, language-dependent features commonly are implemented as two measures (frequency and impact) calculated on a list of words typically generated manually. Frequency (Adler et al., 2011) considers the ratio of vandalism words relative to the size of the edit. Impact (Adler et al., 2011) is the percentage increase that the edit contributes to the overall article size. Both frequency and impact are calculated on inserted words, not on words removed from an edition. Therefore it can be useful to extend these vandalism term lists with others similar obtained with a *CBOW* or *Continuous Skip-gram* model, trained on a set of Wikipedia articles or any other similar source of not annotated information. The models, lists and sets obtained in these experiments have been obtained only from the training set.

[5] https://github.com/juaner4corpora/WP_Vandal.

Table 1
Related words according to the *Continuous Skip-gram* model. Each column corresponds to the term shown in the upper part, appearing their related terms below. The associated score represents the cosine distance.

→ yeah	→ dumb
⟹ yeahs, score = 0.652	⟹ stupid, score = 0.516
⟹ 5ksfe3tez, score = 0.573	⟹ clumsy, score = 0.509
	⟹ dumber, score = 0.501

In order to generate this list, inserted words (InsertedWords set) are examined in each vandalism edit and in every regular edit of *training* generating two separate lists V (vandal) , R (regular). A third list D is generated from them containing the words from V that are not in R, and appear at least twice.

With this root list D, a fourth list DR is generated,[6] containing the words included in D and also the words that are similar to these ones being in their word embeddings at a distance less or equal than a given threshold d. This DR list is used as the basis for calculating the frequency and impact measures, resulting two features identified as *RelatedFrequency* and *RelatedImpact*. Fig. 4 shows the process schematically.

For instance, in Table 1 some clearly vandalism terms appear in the upper part of each column and their related terms appear below. The score (*score*) is the cosine distance formed by the word embedding from the analysed term regarding to each of the word embeddings associated to the related terms. As it can be seen, both morpho-syntactically related terms (*dumber*) and semantically (*stupid*) are inserted. The shown data were taken from the set of training from the vandalism detection in Wikipedia corpus (WP_Vandal), that will be described later.

In addition to the *RelatedFrequency* and *RelatedImpact* features, the created infrastructure has been used to extract two new features also based on the use of word embeddings.

The first of these new features, which has been called *WordsInvandalCluster1*, represents the number of words that are close to a vandalism cluster generated as it is explained below. Words inserted in edits are group in clusters that are labelled as vandalism or not, as it is explained in the following. In the detection phase, every word inserted in an edit is evaluated, depending on the type of the closest cluster that it has. In more detail, the clusters are generated by the following process (Fig. 5):

- The list of words inserted in each edit of the *training* set is traversed. For every word we search for the cluster to the minimal

[6] The corresponding part to the generation of the word embedding model and the similarity calculation have been adapted from the implementation made in Java by Siegfang (https://github.com/siegfang/word2vec). C.SkipGram is used for the generation of the WordVector model with the following parameters: frequencyThresold = 5, embeddedSize = 200, sentenceLenght = 1000, and windowSize = 5.

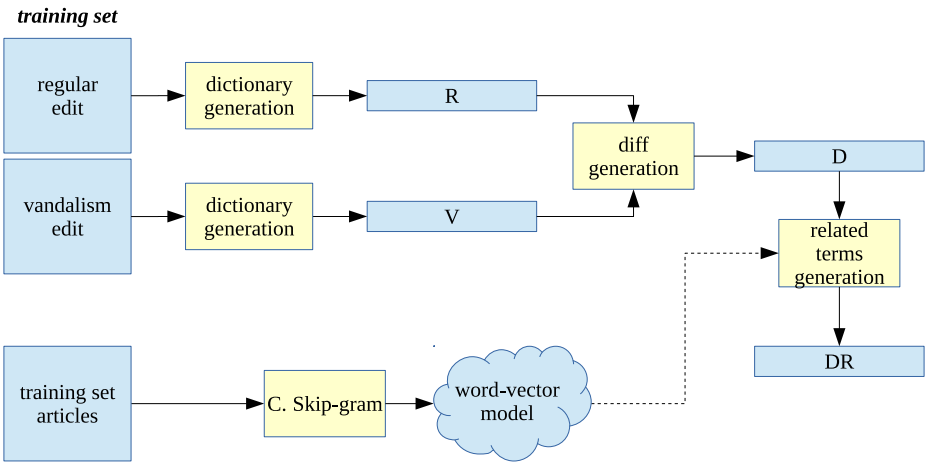

Fig. 4. Generation of related vandalism word lists.

cosine distance to the corresponding word embedding. The initial minimum distance is 0.5.
- If it is found, the word is added to the cluster and the cluster vector is recalculated by adding the word embedding. If a cluster at the minimum distance is not found, a new one is created with the same name as the word and with its word embedding as only member.
- Each cluster has two counters: one (e_v) counting the vandalism edits which have inserted some word and another (e_r) counting the non vandalism edits.
- A cluster is vandal if $e_v - e_r \geq 2$ (this expression avoids recognizing as vandal clusters, those with a single vandalism word, thus preventing their proliferation).

In the evaluation phase on the *test* set, it is determined whether an inserted word is vandal looking for the nearest cluster without setting any minimum distance. If the cluster is vandal according to the above expression then the counter *WordsInvandalCluster1* is increased.

The second feature that has been created is called *WordsInvandalCluster2* and is based on the same idea as the previous one, but the clusters are created in a different way. Vandal clusters are created by taking the words from vandalism edits that are not in the list of exclusive words from vandalism edits (Fig. 6). Thus, unlike *WordsInvandalCluster1* feature, the first group of clusters only groups supposedly vandalism words and the second one only groups regular words.

We have performed a set of experiments to fit the parameters of Word Embeddings used in this work. A vector size of 200 and 400 has been used, being the optimum size 200. Various window sizes (5, 10, and 20) have been used, being the optimal size 5. This is also the configuration that is used by default in many of the existing implementations in the literature.

The word embeddings built in this section from the Wikipedia vandalism edits could be used in a deep learning process like the one described in the next section to obtain other features.

3.2. Deep artificial neural networks for new features

The application of neural network models has brought a new boom in the field of neural networks resulting different architectures among which stand: (1) *Convolutional Neural Networks* (*CNN*) inspired by the visual cortex and suitable especially for tasks of visual and speech recognition; (2) *Deep Belief Networks* (*DBN*) composed of several stacked *Restricted Boltzmann Machines* (*RBM*) that use models based on energy similar to those from classical statistical mechanics; and (3) *Stacked Denoising Autoencoders* (*SDA*).

In this work we have evaluated the last two architectures *DBN* and *SDA*, finally selecting *SDA* for its best performance. The first one (*CNN*) has been discarded since, according to the literature (Britz, 2015) seem less suitable for tasks of natural language processing where large corpus such as vandalism detection are involved.

3.2.1. Feature extraction

The *Stacked Denoising Autoencoders* methods, in addition to classification tasks can be useful as feature extractors since they are able to generate a compact representation of the information that they receive on their input, keeping part of the relevant information and discarding the superfluous. One of the classic methods of document representation widely used in the area of information retrieval is the vector space model, in particular the *bag of words*.

In the task of vandalism detection in Wikipedia it is possible to take advantage of this type of representation as input for a classifier (Belani, 2010), in this case identifying articles (revisions) with documents and discarding both the representation of queries and the similarity measures. Thus, each document vector (article) with dimension $|VC|$ where VC is the vocabulary associated with the set of articles from the corpus, would be considered a feature vector containing information about the presence or absence of certain vocabulary words.

The problem with this approach is that the size of the vocabulary to be handled in a corpus as the two used in this work involves input vectors of high dimensionality, with the addition that these vectors are sparse, i.e., they are mainly composed of zeros and the information associated to each vector is small relative to its size. The consequence of this is the low efficiency in using the memory and long times of training that are required. It is therefore necessary to use some method to reduce this high dimensionality extracting the most relevant information and, *SDA* are initially suitable for this task. To check this, several processes have been implemented in the developed system as it is schematically shown in Fig. 7.

Using this procedure, first, vocabularies belonging to the set of inserted words *InsertedWords* and the set of removed words *DeletedWords* are generated, defining the document vector as the concatenation of these two vocabularies. This makes that some terms appear twice in the document vector, both as added terms and deleted terms. This duplicity is justified by the additional information provided by the appearance of these terms at one or another section of the document vector (a vandalism term in the section of inserted words of the document vector may suggest a vandalism edit, while the same term in the section of deleted words is more likely to appear in a regular edit).

In the construction of these two vocabularies a reduction is performed avoiding the insertion of terms that appear less than n times and have a size smaller than t, using by default a minimum number of appearances of 10 and a minimum length of 3. Requiring this minimum number of appearances, a more generic vocabulary is obtained that can be applied to edits and revisions that were not in the corpus that was used for making this vocabulary. Another form of vocabulary reduction that has been used in any of the tested configurations, is the elimination of stop words.

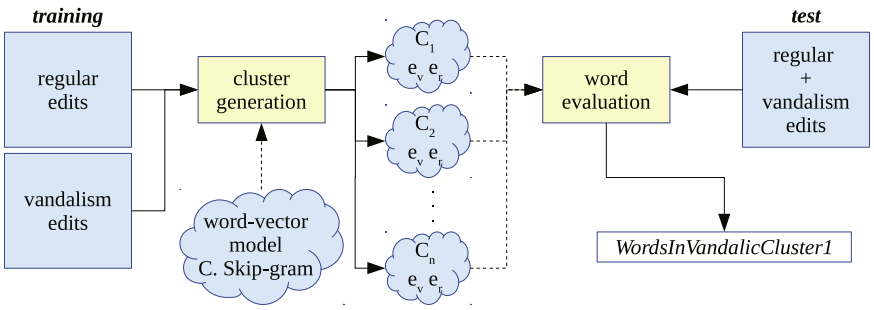

Fig. 5. *WordsInvandalCluster1* calculation.

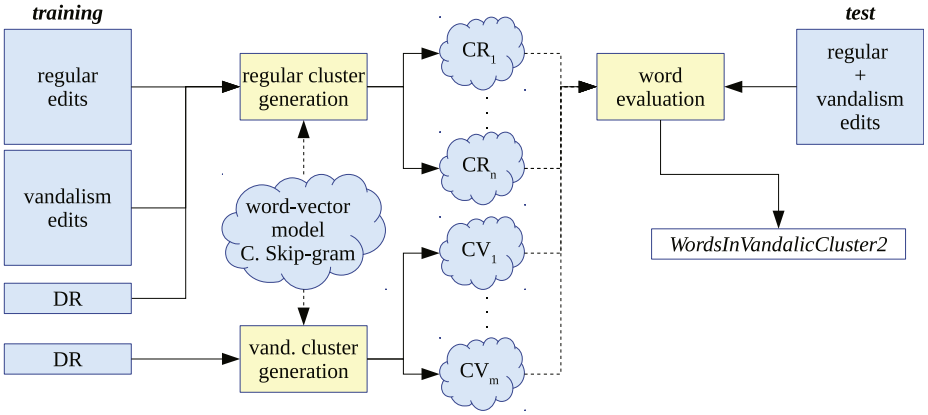

Fig. 6. *WordsInvandalCluster2* calculation.

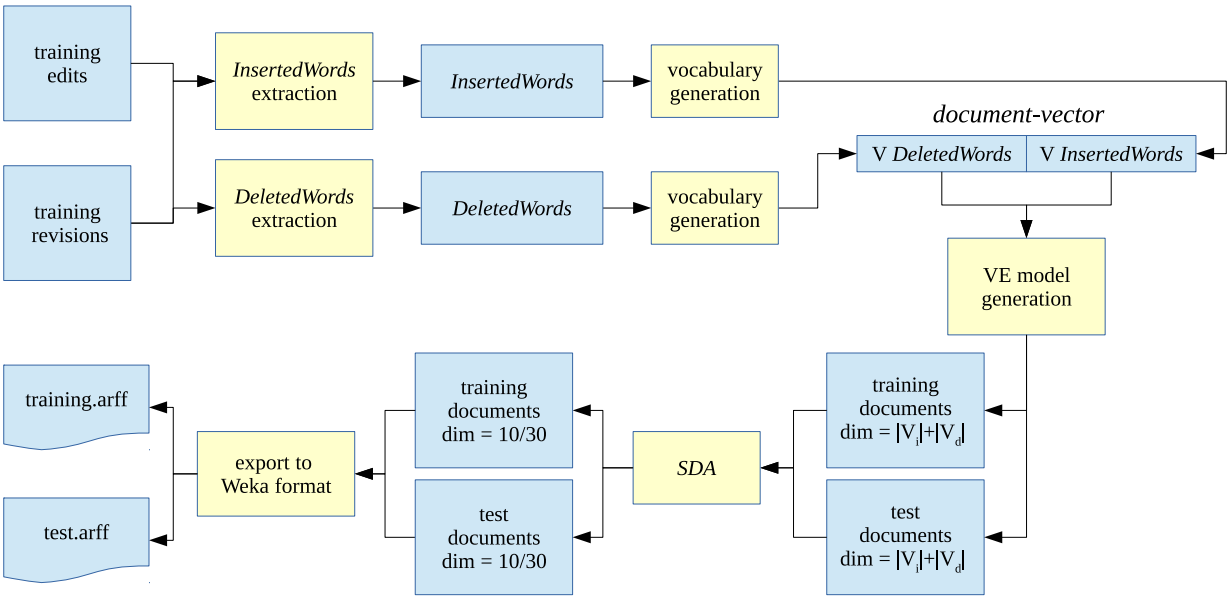

Fig. 7. Features generation by *SDA*.

The next step is the generation of the representations of the bag-of-words model from the edits and revisions in the *training* and *test* sets. Given that *SDA* inputs (used in the implementation in this work) only support binary values, document vectors only show the presence or absence of a term in the *InsertedWords* and *DeletedWords* sets associated with each edit.

The *Stacked Denoising Autoencoder* has been adapted from the available implementation[7] using the *Theano* library (Bergstra et al., 2010; Bastien et al., 2012) written in Python language. With the above mentioned vocabulary restrictions, the input of *SDA* consists of 10,050 units, from which about half belong to the vocabulary of *InsertedWords* set and the rest to the *DeletedWords* set.

We have performed a set of experiments to fit the number of neurons in each layer of a deep network during learning. 875 and 2000 neurons have been used for layer X, 175 and 500 for layer Y, and 2, 10, 20, and 30 for layer Z. Being 875, 175 and 10 the optimum number of neurons for layers X, Y, and Z.

[7] http://deeplearning.net/.

Each of the layers f_i, is actually composed of a *Denoising Autoencoder* (*DA*) layer and a perceptron layer, both with an activation function of sigmoid type.

In the unsupervised training phase only the *DA* components of the layers update the network parameters. After this phase, the supervised learning is performed by the perceptron components of each layer starting from the existing weights in the matrices and bias vectors.

This would be the typical configuration of a classifier with two output classes but, given that the objective is to reduce the input dimensionality, what is done is to extract the input of the last layer and treat these 25 values as the features that gather the codification of the 10,050 inputs in the network. Regarding the number of training iterations, better representations have been obtained when this number is small. In fact, the usual configuration with 20 pre-training iterations and a variable number of iterations of supervised training has been reduced to five pre-training iterations and a single training iteration.

The values present at the input of the last layer are transformed to *Weka* (Witten et al., 2016) format and imported back into the SQL database thereby generating 25 new features to evaluate. It should be noted that unlike the *SDA* inputs, that have a clear meaning being based on a vector space model, the representation obtained in these 25 output features is an abstract coding of the inputs without this correspondence one by one with the vocabulary terms.

3.3. Graph-based ranking algorithms

Given the corpora used in this work where we have a number of revisions, some identified as vandalism and others as regular, the basic idea behind this new set of features is that if we take on the one hand the vandalism revisions and on the other hand the regular revisions, there should be a set of terms that characterize the vandalism edits that are not present in regular edits.

In order to extract these terms, one option is to use strategies based on graphs. The first step is to represent these sets of revisions as a graph and one of the most appropriate graph types for this task is the co-occurrence graph. In this type of undirected graph each vertex identifies a term present in the text and each edge connects two terms that appear in a window of n terms within the same paragraph, in the same document, etc.

In the implementation of the co-occurrence relationship that has been done in this paper, the *InsertedWords* set of each edit without repetition of terms has been used for efficiency reasons and because it contains a more significant information that the full revision. It was considered that two terms co-occur if they appear in a window of 10 consecutive edits of the *training* set and they are in the *InsertedWords* set.

For efficiency issues due to the size of the resulting graph, the *training* set has been further divided randomly into 4 groups of instances with about 4,000 instances per group in order to generate the co-occurrence graphs from the regular edits. Each of these graphs has approximately 35,000 vertices and 8,000,000 of edges.

Over the graphs of regular edits a ranking algorithm is applied to select the r first terms. Then a vocabulary with these four lists of terms is built (joining the four lists with unique terms), and a list of regular terms from it.

The next step is to generate the co-occurrence graph from the vandalism edits. Since in this case the number of instances is much smaller, a division of the *training* set is not performed. The generation of this graph has the particularity that if a word is on the list generated above (regular edits), this is not included in the graph.

Four types of measures have been applied in order to obtain the ranking on the generated vandal graph:

- Degree centrality
- Eigenvector centrality
- PageRank
- Authority (HITS)

Once applied the ranking algorithm to the vandal edits graph according to the measures described above, the first v terms are selected, and the obtained list is used as the basis for calculating frequency and impact measures. In this way, 8 features are obtained, applying frequency and impact measures to the lists of terms obtained from the four measures based on graphs (i.e. pagerank frequency and pagerank impact).

For the calculation of the ranking was used the Jung library (Java Universal Network/Graph Framework)[8] and the graphical analysis environment Gephi.[9] Both tools are interchangeable and produce similar results.

3.4. Other implemented features

As a complement to the features described in the previous sections, another set of 41 features have been implemented that correspond to those described in the works by Potthast et al. (2008), Mola-Velasco (2011), and Tran and Christen (2013). It is necessary to take into account that from these three works, the most recent ones include some features of the previous works and therefore there is an overlap.

These features can be grouped into three categories: based on metadata, textual and language-dependent. Since there is not much information about the implementation of these features, the results obtained for the same features do not exactly match the authors published results but represent a reasonable approximation.

We have implemented several features from previous works in vandalism detection on Wikipedia since we expect that an appropriate combination of them and the ones we have proposed are able to overcome the current state of the art. Some of these known features are language-dependent and others not. Thus, given the good performance of these measures and their simple calculation, we decided to implement them in order to compare the system performance using all the features or only those language-independent.

4. Evaluation corpus and methodology

Previous works Tran and Christen (2013) in the state of the art extract some features from edits and reviews that cannot be obtained from the PAN-WVC-10 corpus and for this reason these authors have decided to generate a new corpus from Wikipedia (WP_Vandal) but with more information than the original corpus.

In this work we have decided to use both corpus to evaluate our system against all the works of the state of the art. Using a corpus extracted from Wikipedia (WP_Vandal) like the previous work we have been able to extract more features and evaluate our system compared to the previously published works. We will also use the PAN-WVC-10 corpus in order to evaluate our work with the unique reference corpus in this area. It is necessary to mention that in the experiments carried out on the PAN-WVC-10 corpus no information of the other corpus was used and therefore the conditions in which we worked were the same as those of the participants in the competition.

4.1. WP_Vandal corpus

This section details the process followed to generate the WP_Vandal[10] corpus, used later in the evaluation of our vandalism detection system. We have used the first five xml files from the full history of revisions of the English version of Wikipedia present in the dump made in May 2016.[11] These files also contain in addition to encyclopedic articles (namespace 0), another type of information[12] that is not relevant for

[8] http://jung.sourceforge.net/.
[9] http://gephi.github.io.
[10] https://github.com/juaner4corpora/WP_Vandal.
[11] https://dumps.wikimedia.org/enwiki/20160501/.
[12] https://en.wikipedia.org/wiki/Wikipedia:Namespace.

this work, so it has been ignored in the construction of the dataset. From these files a total of 7,460 articles and 8,662,332 revisions have been extracted.

Once these revisions have been uploaded, we have marked those that have been identified as vandalism by bots or by human reviewers. To do this, the "comment" field has been queried for any of the following substrings: "vandal", "rv", "rev" or "revert", usually used to indicate this circumstance, marking as vandalism the immediately previous revision which belongs to the same article. Following this method, 1,190,649 vandal revisions have been identified, that is a 13.74% of the total revisions.

The next objective in this construction process has been to balance the dataset, that is, to get the number of regular and vandal revisions to be approximately the same. This goal has been achieved by marking the vandal revisions whose immediately previous revision is regular, and the regular revisions whose immediately posterior revision is vandalism, obtaining in this way 2,309,296 revisions (1,154,649 vandalism). Finally, all the revisions prior to the year 2016 have been filtered, obtaining 36,315 revisions of which 18,506 were marked as vandalism. This latter dataset has been split in two parts in order to create the training and test sets used in the evaluation of the system. The first one contains the edits whose "revision_timestamp" corresponds to the months of January, February and March 2016 (27,724 revisions), and the test set contains the revisions made in April 2016 (8,591 revisions). The content of the article, except metadata, present in the "text" field of these revisions has been tokenized and converted into plain text in order to be used in the calculation of any of the features developed in this work.

4.2. PAN-WVC-10 corpus

PAN-WVC-10 corpus was the corpus used in the vandalism detection competition held in CLEF 2010 conference. The main difference of this corpus with WP_Vandal is that this corpus do not use information after the modification of an article.

This corpus was created in 2010 under the direction of Potthast (2010) and with the collaboration of 753 annotators that using the Amazon's Mechanical Turk[13] tool, examined 32,452 edits from 28,468 articles on the English version of Wikipedia of which 2,391 were annotated as vandalism. The corpus consists of revisions and edits. The former are full versions of Wikipedia articles while edits represent the transition from one revision to the next. In the edits, the number of the original review and the revision number which replaces it, are identified together with other associated metadata (author, size, date–time, etc.).

The result are two evaluation collections distributed in separate comprised archives. The first contains the training corpus composed of 15,000 edits of which 921 are identified as vandalism. In total there are 29,922 reviews on this training corpus. The second collection contains the test corpus with 317,443 edits over a total of 595,082 reviews. 17,443 were annotated and 1,481 identified as vandalism.

The evaluation of the learning schemes used in all the predictions of this paper was performed by a training–test evaluation. The corpus is split into two sets, the first used for training and second used as test for getting the results of the system. We have adopted a set of well-known (Potthast et al., 2010) performance measures in Vandalism Detection: *F-Measure*, area under the *ROC* curve (*AUC-ROC*) and area under the Precision–Recall Curve (*AU-PRC*).

4.3. Classification algorithms

Table 2 shows the results of the classification process using all the features introduced in this work on WP_Vandal and using several classification algorithms included in the Weka data mining tool. Results for the PAN-WVC-10 corpus are similar, being Random Forest the algorithm with a best performance. Thus Random Forest (Breiman, 2001) will be used in the following experiments. A large number of classification algorithms from different families have been analysed being "RandomForest" the algorithm that works best. However, other algorithms such as "ClassificationViaRegression" and the "MultiLayer Perceptron" have obtained a similar performance.

Table 2
Results using different classification algorithms, based on their F-Measure, *AUC-ROC*, and *AU-PRC*. Best results appear in boldface.

Features	F-measure	*AUC-ROC*	*AU-PRC*
RandomForest	**0.791**	**0.960**	**0.859**
ClassificationViaRegression	0.786	0.951	0.847
Bagging + RandomForest	0.780	0.953	0.856
Bagging + REPTree	0.774	0.947	0.850
AdaBoostM1	0.770	0.948	0.829
IBK	0.702	0.808	0.650
SimpleLogistic	0.779	0.941	0.846
MultilayerPerceptron	0.775	0.940	0.842
Logistic	0.773	0.939	0.838
NaiveBayesMultinomial	0.390	0.612	0.521
VotedPerceptron	0.389	0.614	0.522
J48	0.755	0.857	0.713
DecisionStump	0.766	0.861	0.702
MLP	0.785	0.920	0.816
SVM	0.754	0.836	0.785
EXTRA Tree	0.731	0.812	0.695

Table 3
Results in WP_Vandal corpus using Random Forest on the different sets of features implemented in this work. Set A corresponds to features extracted from vandalism terms based on vandalism word lists, Set B to Deep Artificial Neural Networks features, Set C to features extracted from Graph-based ranking algorithms, and Set P to features proposed in previous works.

WP_Vandal corpus			
Features	F-Measure	*AUC-ROC*	*AU-PRC*
Set A	0.512	0.718	0.601
Set B	0.549	0.761	0.635
Set C	0.425	0.605	0.542
Set P (previous works)	0.764	0.935	0.818
A + B	0.590	0.792	0.684
A + C	0.571	0.769	0.652
B + C	0.579	0.778	0.682
A + B + C	0.623	0.839	0.722
A + B + C + P	**0.791**	**0.960**	**0.859**

5. Results

In this section the results obtained by the system developed in this work are shown for the different sets of implemented features. First, each group of features is analysed separately. Then, the aggregated performance of the three groups of features is checked. The results obtained by the features from the works of other authors are displayed later and the system performance using these features together with the others presented in this paper are also illustrated. Finally, the performance of this system is compared to results from other previous published works.

Tables 3 and 4 illustrate the different combinations of features tested in Wikipedia and PAN-WVC-10 corpora using the Random Forest classifier, trained on the training set and evaluated on the set test based on their F-Measure, *AUC-ROC*, and *AU-PRC*. In these Tables 3 and 4 results identified as A + B + C + ... represent a configuration of the developed system in which the sets of features A,B, and C are used simultaneously, where A are based on word embeddings, B are drawn with *SDA*, C are based on ranking on graphs, and P are the implemented features from other authors.

First, it is important to note in Table 3 that the set B, relative to the deep learning features, is the one that obtains the best performance. It can be seen that combining any two sets of features presented in

[13] https://www.mturk.com/.

Table 4
Results in PAN-WVC-10 corpus using Random Forest on the different sets of features implemented in this work. Set A corresponds to features extracted from vandalism terms based on vandalism word lists, Set B to Deep Artificial Neural Networks features, Set C to features extracted from Graph-based ranking algorithms, and Set P to features proposed in previous works.

PAN-WVC-10 Corpus			
Features	F-Measure	*AUC-ROC*	*AU-PRC*
Set A	0.519	0.686	0.489
Set B	0.553	0.865	0.696
Set C	0.396	0.632	0.345
Set P (previous works)	0.612	0.931	0.755
A + B	0.681	0.882	0.738
A + C	0.569	0.726	0.566
B + C	0.622	0.858	0.689
A + B + C	0.702	0.903	0.766
A + B + C + P	**0.725**	**0.959**	**0.804**

this work (A, B or C), the values obtained are higher respectively than the maximum values obtained separately for any set. In addition, the combination of the three sets of features (A + B + C) is better than any combination of just two sets (A + B, A + C, or B + C). Within the combination of two sets, the set A + B is the one that obtains the best results, showing in this way that features based on deep learning have a great influence on the final result.

In addition to these three groups of new features, features described in previous works as Potthast et al. (2008), Mola-Velasco (2011), and Tran and Christen (2013) have also been implemented (Set P). Although the results obtained with this set of features (P) are high in relation to the sets proposed in this work, it is necessary to take into account that these features are a selection of the best features proposed in the state of the art. Moreover, what is more important is that the combination of these features of previous works with the features proposed in this work obtain a higher performance when combined than being used in isolation. Therefore answering the question of the title of this work, deep learning techniques can improve classification performance of vandalism detection in Wikipedia.

Regarding Table 4, results show a similar behaviour for the different sets of features. Set B, relative to the deep learning features, obtains the best performance. In addition, the best combination of features is the one that joins the A, B and C sets. However, sets A and B, both relative to deep learning techniques, get a performance close to the best combination. In the same way, it is also concluded that the combination of all the features provides the best result, therefore it is a set of features that focus on different aspects of vandalism. In the case of this corpus, the results are slightly lower, probably due to the fact that it contains less information than the WP_Vandal corpus.

5.1. State-of-art comparison

In this section we analyse whether the new features developed in this work can complement those developed by other authors comparing the values obtained with different combinations of features for measures *F-Measure* (Not Available in PAN), *AUC-ROC*, and *AUC-PRC*.

For that, we take the results obtained by different vandalism detection systems presented in the competition held at CLEF 2010 (Potthast et al., 2010) as reference, training on the training set and evaluating on the test set. These results are shown in Table 5 (obtained from Potthast et al. (2010)). As can be seen, the system proposed in this work overcome the results obtained by any participating system. However, our system did not participate in the competition.

For that, we have implemented the features described in the two of best works in the state of the art to the best of our knowledge as well as those proposed in this work. For the implementation of each feature we have followed every one of the steps provided by the authors in their works. Once these features have been implemented, each set of features has been applied an evaluation process on the PAN and WP_Vandal corpora presented in this paper, divided into a training and a test set. These results are shown in Tables 5 and 6.

Table 5
Final ranking of systems presented in CLEF 2010 (Potthast et al., 2010). Best results corresponding to the proposed system appear in boldface.

System	*AUC-ROC*	*ROC* rank	*AUC-PRC*	*PRC* rank
Mola Velasco	0,922	1	0,665	1
Adler et al.	0,903	2	0,492	3
Javanmardi	0,898	3	0,447	4
Chichkov	0,893	4	0,562	2
Seaward	0,879	5	0,413	7
Hegedus et al.	0,876	6	0,422	5
Harpalani et al.	0,858	7	0,414	6
White and Maessen	0,843	8	0,393	8
Iftene	0,654	9	0,122	9
Random detector	0,500	10	0,084	10
Our system	**0,959**		**0.804**	

Table 6
Results in WP_Vandal corpus using Random Forest with the different sets of features implemented in this work, based on their F-Measure, *AUC-ROC*, and *AU-PRC*. Best results corresponding to the proposed system appear in boldface.

Features	F-Measure	*AUC-ROC*	*AU-PRC*
Mola-Velasco (2011)	0.772	0.840	0.819
Tran and Christen (2013)	0.780	0.848	0.845
Our system	**0.791**	**0.960**	**0.859**

These results do not try to compare directly the system presented in this work with the systems that participated in the competition, but it is a sample of the advance that we try to propose with a new set of features.

Results shown in 6 reflects that the features proposed in this work, combined with some of the features proposed by other authors, improve the results achieved in the state of the art independently of the evaluation measure used. Thus, as in the work by Tran (Tran and Christen, 2013), where some of the features introduced by Mola-Velasco (Mola-Velasco, 2011) are used to obtain an advance in the state of art, the features proposed in this work may be used in the future to continue to progress in an area where there is still room for improvement. There is a significant improvement in the AUC-ROC evaluation measure, obtaining a score of 0.96.

5.2. Statistical significance

We performed a simple two-tailed, paired sample t-test at significance level p = 0.05 to determine whether the difference between the evaluation measures from the previous works features and each of the combination of our sets of features on the test corpus is statistically significant. We found that at the 95% significance level (p = 0.05), 6 out of 9 combinations in Table 7 performed better than previous works features *AUC-ROC*, other 2 combinations (A + C + P + B and B + P) obtained a same score and only the A + B + C set and the single sets A, B, and C got a worse score. Using the same significance level, 7 out of 9 combinations performed better than previous works features *AUC-PRC*, the B + P set got a same score and the same 4 sets than in the *AUC-ROC* case, got a worse score.

Our result is encouraging since it indicates that a deep learning approach using relatively few features can achieve scores comparable to those of systems built using a big amount of features.

6. Conclusions

In this paper, we have introduced new features able to overcome some of the best works published in the state-of-art. New developed features can be framed into three groups.

Table 7
Paired sample t-test at significance level p = 0.05 for *AUC-ROC* and *AUC-PRC* measures using set P as test base for different combinations of sets of features in Tables 3 and 4. B indicates a better statistical score, w worse, and — the same score.

Set	Table 3		Table 4	
of Features	*AUC-ROC*	*AUC-PRC*	*AUC-ROC*	*AUC-PRC*
A	w	w	w	w
B	w	w	w	w
C	w	w	w	w
A + P	**B**	–	**B**	**B**
B + P	**B**	**B**	**B**	**B**
C + P	–	–	–	–
A + B + P	**B**	**B**	**B**	**B**
A + C + P	**B**	**B**	–	**B**
B + C + P	**B**	**B**	**B**	**B**
A + B + C + P	**B**	**B**	**B**	**B**

The first group of features takes advantage of the semantic similarity relationships observed in dense representations of words, known as word embeddings. Given that manual preparation of these lists of vandalism terms is costly, a mechanism to extend these lists automatically may be useful to improve the performance of the features based on these lists. For that, we have implemented a process that, from a word embedding extracted from the set of both vandalism and regular revisions of the corpus and a list of vandalism terms, extends this list with other related terms.

The second group of features takes a different approach to the above. Instead of trying to create or expand lists of terms to be used for frequency or impact type features, the features developed in this group are directly used by the classifiers. The classical vector space model is not very appropriate for two corpora as the used in this work given the vocabulary size that is handled. In this way we propose the use of deep neural network architectures that have been recently developed as a mechanism to reduce the dimensionality and build a compact representation of the input. A network *SDA* is built with four layers and a decreasing number of units as they approach the output, extracting the 25 inputs in the last layer as features.

The last group is composed of the features created by ranking algorithms on graphs. It has been proven that these algorithms, especially PageRank, are able to extract a list of relevant words from a co-occurrence graph generated over the set of terms inserted into an edit, using as document a window of *n* edits.

The joint use of these three groups of features reveals that there is an important complementarity among them, since a clear increase was obtained in the area under the *ROC* curve and the area under the precision–recall curve, in relation with results obtained with any of these groups separately. In the case of the PAN-WVC-10 corpus, the combination of the three sets of features presented in this paper improve the results of the other features from previous works.

On the other hand, if we combine these new features with a set of features that have been reproduced from gathered information in previous works, the resulting system gets the best score compared with the previous works, obtaining an area under the *ROC* curve of 0.960, and an area under the precision–recall curve of 0.859 in the Wikipedia corpus (WP_Vandal) and obtaining an area under the *ROC* curve of 0.959, and an area under the precision–recall curve of 0.804 in the PAN-WVC-10 corpus.

6.1. Future work

First, in order to generate term rankings, the most of ranking algorithms based on graphs suitable for this task have been explored, but in the construction of co-occurrence graphs it is possible to vary both the source of the information, for instance using the set of the inserted text (*InsertedText*) instead of the set of the inserted words (*InsertedWords*), and the window size of edits that determines the document boundaries.

Regarding the features based on semantic similarity of word embeddings, there are also parameters such as the minimum distance between word embeddings that can be studied further. In addition, we have experimented with a concrete implementation of distributed representations of words (Mikolov et al., 2013b), but others are available (Pennington et al., 2014) that could provide better results.

Feature extraction or classification itself by deep networks is perhaps the area where more research lines exist, in the use of different models (*CNN*, *DBN*, *SDA*), in the architecture of the models varying the number of layers and the size of each of them, thereby generating more or less compact representations, in studying the numerous parameters with which can be configured such networks, or in the application of different types of inputs alternative to document vectors from the vector space model. For instance, it is common to see this type of networks using word embeddings as inputs so it would be interesting to study a vector space model of a reduced vocabulary in which each term in the document vector is not an atomic input but a word embedding.

Finally it might be useful to explore the generation of a set of features based on user profiles.

Acknowledgements

This work has been partially supported by the Spanish Ministry of Science and Innovation, Spain within the projects EXTRECM (TIN2013-46616-C2-2-R) and MAMTRA-MED, Spain (TIN2016-77820-C3-2-R).

References

Adler, B., de Alfaro, L., Pye, I., 2010. Detecting wikipedia vandalism using wikitrust. Notebook Papers of CLEF, Vol. 1, pp. 22–23.

Adler, B.T., De Alfaro, L., Mola-Velasco, S.M., Rosso, P., West, A.G., 2011. Wikipedia vandalism detection: Combining natural language, metadata, and reputation features. In: International Conference on Intelligent Text Processing and Computational Linguistics. Springer, pp. 277–288.

Alfonseca, E., Garrido, G., Delort, J.-Y., Peñas, A., 2013. WHAD: Wikipedia historical attributes data. Lang. Resour. Eval. 47, 1163–1190. http://dx.doi.org/10.1007/s10579-013-9232-5.

Bastien, F., Lamblin, P., Pascanu, R., Bergstra, J., Goodfellow, I., Bergeron, A., Bouchard, N., Warde-Farley, D., Bengio, Y., 2012. Theano: new features and speed improvements. arXiv preprint arXiv:1211.5590.

Belani, A., 2010. Vandalism detection in wikipedia: a bag-of-words classifier approach. arXiv preprint arXiv:1001.0700.

Bengio, Y., 2009. Learning deep architectures for AI. Found. Trends Mach. Learn. 2, 1–127. http://dx.doi.org/10.1561/2200000006.

Bergstra, J., Breuleux, O., Bastien, F., Lamblin, P., Pascanu, R., Desjardins, G., Turian, J., Warde-Farley, D., Bengio, Y., 2010. Theano: a CPU and GPU math expression compiler. In: Proceedings of the Python for Scientific Computing Conference (SciPy), Vol. 4. Austin, TX, p. 3.

Breiman, L., 2001. Random forests. Mach. Learn. 45, 5–32.

Britz, D., 2015. Implementing a cnn for text classification in tensorflow. In: REINFORCEMENT LEARNING. pp. 1–5, http://www.wildml.com/2015/12/implementing-acnn-for-text-classification-in-tensorflow/.

Dean, J., Ghemawat, S., 2008. MapReduce: simplified data processing on large clusters. Commun. ACM 51, 107–113.

Harpalani, M., Hart, M., Singh, S., Johnson, R., Choi, Y., 2011. Language of vandalism: Improving wikipedia vandalism detection via stylometric analysis. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2. Association for Computational Linguistics, pp. 83–88.

Heindorf, S., Potthast, M., Stein, B., Engels, G., 2016. Vandalism detection in wikidata. In: Proceedings of the 25th ACM International on Conference on Information and Knowledge Management. In: CIKM '16, ACM, New York, NY, USA, pp. 327–336. http://dx.doi.org/10.1145/2983323.2983740, URL http://doi.acm.org/101145/29833232983740.

Itakura, K.Y., Clarke, C.L., 2009. Using dynamic markov compression to detect vandalism in the wikipedia. In: Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM, pp. 822–823.

Javanmardi, S., Lopes, C., Baldi, P., 2010. Modeling user reputation in wikis. Stat. Anal. Data Min. 3, 126–139.

Javanmardi, S., McDonald, D.W., Lopes, C.V., 2011. Vandalism detection in wikipedia: a high-performing, feature-rich model and its reduction through lasso. In: Proceedings of the 7th International Symposium on Wikis and Open Collaboration. ACM, pp. 82–90.

Kumar, S., Spezzano, F., Subrahmanian, V., 2015. Vews: A wikipedia vandal early warning system. In: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. In: KDD '15, ACM, New York, NY, USA, pp. 607–616. http://dx.doi.org/10.1145/2783258.2783367, URL http://doi.acm.org/101145/27832582783367.

Mikolov, T., Chen, K., Corrado, G., Dean, J., 2013a. Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J., 2013b. Distributed representations of words and phrases and their compositionality. In: Burges, C., Bottou, L., Welling, M., Ghahramani, Z., Weinberger, K. (Eds.), Advances in Neural Information Processing Systems 26. Curran Associates, Inc., pp. 3111–3119, URL http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf.

Mola-Velasco, S.M., 2011. Wikipedia vandalism detection. In: Proceedings of the 20th International Conference Companion on World Wide Web. ACM, pp. 391–396.

Pennington, J., Socher, R., Manning, C.D., 2014. Glove: Global vectors for word representation. In: Proceedings of the Empiricial Methods in Natural Language Processing (EMNLP 2014), Vol. 12. pp. 1532–1543.

Potthast, M., 2010. Crowdsourcing a wikipedia vandalism corpus. In: Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM, pp. 789–790.

Potthast, M., Holfeld, T., 2011. Overview of the 2nd international competition on wikipedia vandalism detection. In: Petras, V., Forner, P., Clough, P.D. (Eds.), CLEF (Notebook Papers/Labs/Workshop). pp. 1–14.

Potthast, M., Stein, B., Gerling, R., 2008. Automatic vandalism detection in wikipedia. In: European Conference on Information Retrieval. Springer, pp. 663–668.

Potthast, M., Stein, B., Holfeld, T., 2010. Overview of the 1st international competition on wikipedia vandalism detection. In: Braschler, M., Harman, D., Pianta, E. (Eds.), CLEF 2010 LABs and Workshops, Notebook Papers, 22-23 September 2010, Padua, Italy. In: CEUR-WS.org, vol. 1176, pp. 1–12, CEUR Workshop Proceedings.

Segall, J., Greenstadt, R., 2013. The illiterate editor: metadata-driven revert detection in wikipedia. In: Proceedings of the 9th International Symposium on Open Collaboration. ACM, p. 11.

Shulhan, M., Widyantoro, D.H., 2016. Detecting vandalism on english wikipedia using lnsmote resampling and cascaded random forest classifier. In: 2016 International Conference on Advanced Informatics: Concepts, Theory and Application (ICAICTA). pp. 1–6. http://dx.doi.org/10.1109/ICAICTA.2016.7803106.

Sumbana, M., Gonçalves, M.A., Silva, R., Almeida, J., Veloso, A., 2012. Automatic vandalism detection in wikipedia with active associative classification. In: Theory and Practice of Digital Libraries. Springer, pp. 138–143.

Susuri, A., Hamiti, M., Dika, A., 2016. Machine learning based detection of vandalism in wikipedia across languages. In: 2016 5th Mediterranean Conference on Embedded Computing (MECO). pp. 446–451. http://dx.doi.org/10.1109/MECO.2016.7525689.

Suzuki, Y., Yoshikawa, M., 2013. Assessing quality score of wikipedia article using mutual evaluation of editors and texts. In: Proceedings of the 22nd ACM International Conference on Conference on Information and Knowledge Management. ACM, pp. 1727–1732.

Tran, K.-N., Christen, P., 2013. Cross language prediction of vandalism on wikipedia using article views and revisions. Adv. Knowl. Discov. Data Min. 268–279.

Tran, K.-N., Christen, P., Sanner, S., Xie, L., 2015. Context-aware detection of sneaky vandalism on wikipedia across multiple languages, pp. 380391.

West, A.G., Kannan, S., Lee, I., 2010. Detecting wikipedia vandalism via spatio-temporal analysis of revision metadata. In: Proceedings of the Third European Workshop on System Security. ACM, pp. 22–28.

West, A.G., Lee, I., 2011. Multilingual vandalism detection using language-independent & ex post facto evidence. In: PAN-CLEF '11: Notebook Papers on Uncovering Plagiarism, Authorship, and Social Software Misuse. pp. 1–10.

Witten, I.H., Frank, E., Hall, M.A., Pal, C.J., 2016. Data Mining: Practical Machine Learning Tools and Techniques. Morgan Kaufmann.

Wu, G., Harrigan, M., Cunningham, P., 2012. Classifying wikipedia articles using network motif counts and ratios. In: Proceedings of the Eighth Annual International Symposium on Wikis and Open Collaboration. ACM, p. 12.

Wu, Q., Irani, D., Pu, C., Ramaswamy, L., 2010. Elusive vandalism detection in wikipedia: a text stability-based approach. In: Proceedings of the 19th ACM International Conference on Information and Knowledge Management. ACM, pp. 1797–1800.

Zeng, H., Alhossaini, M.A., Ding, L., Fikes, R., McGuinness, D.L., 2006. Computing trust from revision history. In: Proceedings of the 2006 International Conference on Privacy, Security and Trust: Bridge the Gap Between PST Technologies and Business Services. In: PST '06, ACM, New York, NY, USA, http://dx.doi.org/10.1145/1501434.1501445, pp. 8:1–8:1.