# Detection of Near-duplicate User Generated Contents: The SMS Spam Collection*

Enrique Vallés Balaguer
NLE Lab - ELiRF, DSIC
Universidad Politécnica de Valencia
Camino de Vera, s/n
46022 Valencia, Spain
enriquevallesbalaguer@gmail.com

Paolo Rosso
NLE Lab - ELiRF, DSIC
Universidad Politécnica de Valencia
Camino de Vera, s/n
46022 Valencia, Spain
prosso@dsic.upv.es

## ABSTRACT

Today, the number of spam text messages has grown in number, mainly because companies are looking for free advertising. For the users is very important to filter these kinds of spam messages that can be viewed as near-duplicate texts because mostly created from templates. The identification of spam text messages is a very hard and time-consuming task and it involves to carefully scanning hundreds of text messages. Therefore, since the task of near-duplicate detection can be seen as a specific case of plagiarism detection, we investigated whether plagiarism detection tools could be used as filters for spam text messages. Moreover we solve the near-duplicate detection problem on the basis of the CHAMELEON clustering algorithm. We carried out some preliminary experiments on the SMS Spam Collection that recently was made available for research purposes. The results were compared with the ones obtained with CHAMELEON. Althought plagiarism detection tools detect a good number of near-duplicate SMS spam messages even better results are obtained with the clustering approach.

## Categories and Subject Descriptors

I.2.7 [**Artificial Intelligence**]: Natural Language Processing

## General Terms

Experimentation

## Keywords

Near-duplicate detection, plagiarism detection, spam text messages

*Revised version


*SMUC'11,* October 28, 2011, Glasgow, Scotland, UK.


## 1. INTRODUCTION

In recent years, the huge amount of user generated contents have led to great benefits for both users and companies. For a user, they are a great help to find opinions of other users before deciding to purchase a product. And the companies use the user generated contents to identify product problems and to find marketing intelligence information about their competitors [13].

However, there are companies that use the user generated contents for free advertising. This is because promotion is a vital part of any company and depending on the type and quality of promotion, the sale of a product can vary enormously. Unfortunately, there are times that companies opt for promoting their products through spam messages that often are created from templates, for examples:

- SMS spam messages: When companies launch a new product, they need to market it to ensure that the product gains popularity. These companies send SMS marketing campaigns.
- Opinion spam messages: Companies deliberately write fictitious opinions to sound authentic, in order to deceive the reader. These opinions are positives for their products and negatives for the competitors' products [18].

In this paper we focus on the problem of detecting SMS spam messages, since it is a relatively unexplored subject and it begins to be troublesome for mobile users. The main problem with SMS spam is that it is not only annoying because the identification of spam text messages is a very hard and time-consuming task, but it can also be expensive since some people pay to receive text messages. Moreover, there is a limited availability of mobile phone spam-filtering software. Another concern is that important legitimate messages as of emergency nature could be blocked [2].

Due to the fact that mobile phones are a cheap and easy device for communication and is increasingly being used as a source of information, the number of mobile phone users is continuously growing in some countries also with respect to the number of internet users (e.g. in India mobile phone users are nearly 10 times larger than internet users). As a consequence, from the research community there is a growing interest in analysing SMS-like messages also due to the further challenge that analysing noisy and short texts written in "SMS language" implies: they are approx. 160-character long, users compress text by omitting letters, using

slang, etc., and unintended typographical errors are quite frequent due to small size of keypads on mobile phones (as well the poor language skills of the users...). Although some tasks related to the retriaval of near-duplicate SMS messages have been investigated (e.g. "SMS-based Frequently Asked Questions Retrieval" task has been organised in the framework of international fora such as the Forum for Information Retrieval Evaluation FIRE-2011[1]), the detection of near-duplicate SMS messages has not been addressed yet. Just recently the SMS Spam Collection[2] has been released.

Two texts are considered near-duplicates when, although they are not exact duplicates, they are strikingly similar [24]. On the basis of the previous definition, although spam messages texts belonging to the same campaign may look different because spammers need to randomize the messages by adding news paragraphs or obfuscating terms to by-pass a filter [21, 22], they still be considered as near-duplicate texts because they are created from templates and they usually share a certain similarity among them.

Since the task of near-duplicate detection can be seen as a specific case of plagiarism detection, in this paper we study the possibility of using plagiarism detection tools as filters for spam text messages. Moreover, we try to solve the near-duplicate detection problem also on the basis of the CHAMELEON clustering algorithm using CLUTO tool[3].

The rest of the paper is organised as follow. Section 2 describes the difference between near-duplicate and plagiarism detection. Section 3 illustrates the main characteristics of the plagiarism detection tools used. Section 4 shows the results of these preliminary experiments. In Section 5 draw some conclusions and discuss further work.

## 2. NEAR-DUPLICATE DETECTION VS. PLAGIARISM DETECTION

When two texts that are not exactly identical contain nearly the same content, they should be treated as duplicates. In a plagiarism detection scenario, the definition of near-duplicate texts may be even more flexible. When a portion of one text, such as a sentence, is contained in another text, these two texts could be seen as near-duplicates.

In contrast, perhaps the most extreme definition of a near-duplicate exists in an anti-adversarial scenario: "Two texts are near-duplicates if they share more than 80% terminology and their length difference is not more than +20%" [12]. However, as long as the core payload text (e.g., a URL pointing to the spammer's site) is identical, two SMS spam text messages are treated as near-duplicates [17].

Following, we describe the near-duplicate detection and the plagiarism detection tasks.

### 2.1 Near-duplicate detection

Two texts that are strikingly similar although slightly different in some of their parts are not regarded as exact duplicates but as near-duplicates [24]. Typographical errors, versioned, mirrored, or plagiarised documents, multiple representations of the same physical object, email spams or user-generated spams, as SMS spam text messages or opinion spasm, that could have been generated from the same template, are examples of near duplicate documents [26].

Differents methods have been suggested for near-duplicate detection. In [10] the authors have proposed a method for the estimation of the degree of similarity among pairs of texts known as shingling, in which all sequences of adjacent words are extracted. If two texts contain the same shingles set they are treated as equivalent and if the shingles set overlaps, they are considered as exact. In this technique the authors noted that it does not work well on small texts [27]. In [15] the authors have used a five-gram approach as a shingle and sample 84 shingles for each text. Then the 84 shingles are built into six super shingles. The texts having two super shingles in common are considered as nearly duplicate texts.

In [11] the authors have developed an efficient way to determine the syntactic similarity of files and have applied it to every document on World Wide Web. Using their approach, they have clustered all the documents that are syntactically similar.

In [29] the authors presented an approach for the detection of near-duplicate Web pages in Web crawling. Near-duplicate Web pages are detected followed by the storage of crawled Web pages into repositories. The keywords are extracted from crawled pages and on the basis of these keywords the similarity score is calculated taking into account their occurrences in each page. The documents are considered as near-duplicates if its similarity scores are smaller than a threshold value.

Another approach for finding duplicates and near duplicates is based on hashing or fingerprinting. Such methods produce one or more fingerprints that describe the content of a document or fragment. A suspicious document's fragments are compared to the reference corpus based on their hashes or fingerprints. Duplicate and near duplicate passages are assumed to have similar fingerprints. Also one of the first systems for plagiarism detection used this fingerprint-based approach [8]. Although identical duplicates of short texts are easy to detect by standard hashing approaches, the detection of near-duplicate short texts is much more difficult. A single short text contains usually less than 200 characters, which makes more difficult to extract effective features. Moreover, informal abbreviations, transliterations and network languages prevailing in some short text collections [9].

### 2.2 Plagiarism detection

Plagiarism detection can be divided into plagiarism detection with reference and intrinsic plagiarism detection.

Plagiarism detection with reference is based on comparing a suspicious document or fragment with a set of source documents. There are several approaches for plagiarism detection with reference. The most popular ones are based on word n-grams [25, 28] or also character n-grams [32, 16] comparisons. However, in these approaches the temporal and spatial costs increase exponentially with the number of texts to compare. For this reason, in [7] the authors propose to reduce of the search space on the basis of the Kullback-Leibler distance.

Other plagiarism detection approaches are based on word frequency analysis as [33]. In [19] an approach based on word comparison at sentence level which takes into account

[1]http://www.isical.ac.in/∼clia/faq-retrieval/faq-retrieval.html
[2]http://www.dt.fee.unicamp.br/∼tiago/smsspamcollection
[3]www.cs.umn.edu/∼karypis/cluto

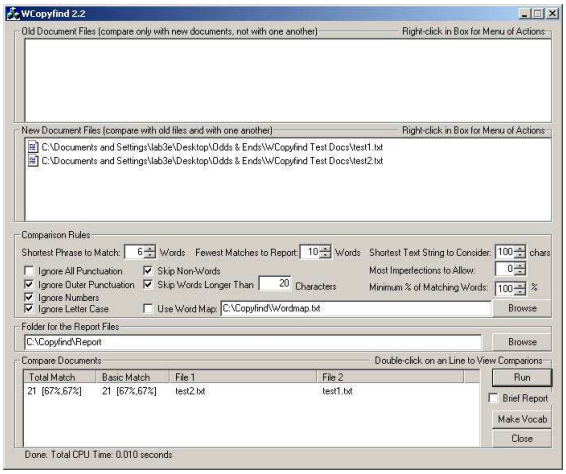


**Figure 1: WCopyFind tool**

vocabulary expansion with Wordnet[4] was described. More recently, some approaches tried to address also the problem of cross-language plagiarism [31, 6].

Intrinsic plagiarism detection bases its analysis on stylistic changes in the suspicious text [35]. The basic principles of intrinsic plagiarism detection are [14]:

- each author has her own writing style;
- the writing style of each author should be consistent throughout the text;
- the features of a style are difficult to manipulate.

Other methods for intrinsic plagiarism detection are the ones described in [34] where character n-gram profiles have been used. A somehow related problem is authorship attribution where linguistic profiles need to be investigated for determining the true author of a text [23].

A system capable to detect any kind of plagiarism, also when there may not be any other reference text to compare with and the linguistic evidence has to be given on the basis of stylistic changes in the document itself, as well cross-language plagiarism, is described in [20].

# 3. PLAGIARISM DETECTION TOOLS

Several are the available tools for plagiarism detection. For our preliminary experiments, we have used the Wcopy-Find, CopyCatch, and the and Pl@giarism tools. Following we describe the three tools.

## 3.1 WCopyFind

WCopyFind[5], developed in 2004 by Bloomfield at the University of Virginia, is a available system for plagiarism detection [37]. The system allows researchers to introduce various parameters as the size of the word n-grams, the minimum number of matching words to report as possible plagiarism, the maximum number of non-matches between perfectly matching portions of a phrase,etc. Although the system allows to introduce the size of the word n-grams, the author proposes to use 6-grams.

Another feature of the WCopyFind is that it allows to introduce a word map (a generalized thesaurus) with the intention of solving the substitution of words for synonyms. The figure 1 shows the WCopyFind tool.

The system tells researchers the percentage of the number of matches with document A versus document B, the percentage of the number of matches with document B versus document A. The system shows a comparation between two documents. In this comparation, WCopyfind shows plagiarised sections marked in red.

[4]http://wordnet.princeton.edu

[5]http://www.plagiarism.phys.virginia.edu/Wsoftware.html

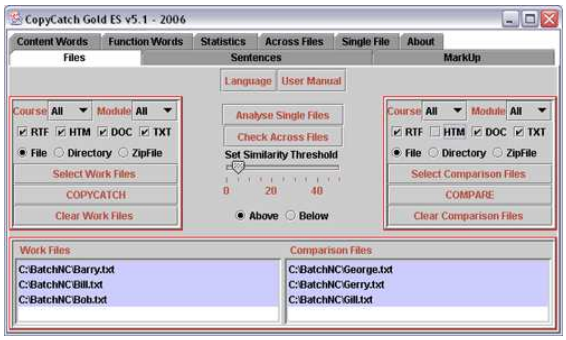


**Figure 2: CopyCatch tool**

## 3.2 CopyCatch

CopyCatch[6], a software developed by Woolls from CFL Software Development. CopyCatch allows researchers to calculate the threshold level of textual similarity. This program incorporates several measurements such as threshold of overlapping vocabulary, unique and exclusive vocabulary and shared-once phrases. It has a visual output, with plagiarised sections marked in red.

The system calculates the comparison from trigrams. However, CopyCatch also obtains a measure of similarity of vocabulary. The figure 2 shows the CopyCatch tool.

The main distinguishing characteristic of this software was that it was much easier to use, especially the submission process: it allows the user to browse and select the files to check. The results are almost immediate. It outputs a list of pairs of files sorted by percentage of match between them. It is then possible to have a list of the vocabulary or phrases shared between the 2 files and it can also mark up the files highlighting the similarities between them. It allows the user to check Word documents as well as text, rtf and HTML files: no previous conversion is needed [5].

## 3.3 Pl@giarism

Pl@giarism[7], developed by the University of Maastricht, is another system for plagiarism detection. It is used extensively by the Law Faculty of Maastricht. Pl@giarism is a simple program that automates the process of determining similarities between pairs of essays by comparing three word phrases in each. An essay is paired with each essay in the folder in turn.

Pl@giarism does not automatically check against sources

[6]http://cflsoftware.com/?page_id=42

[7]http://www.plagiarism.tk

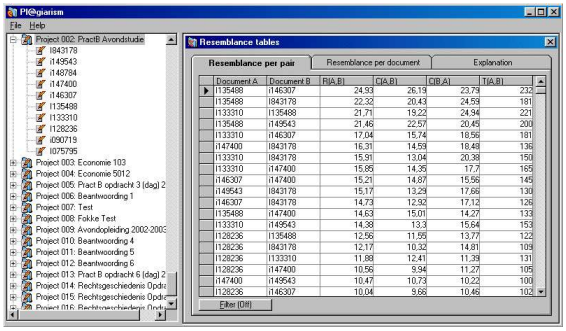


**Figure 3: Pl@giarism tool**

on the Internet, but there is a provision for selecting and submitting one phrase at a time to an Internet search engine for comparison with Internet resources [1]. The figure 3 shows the Pl@giarism tool.

The system returns the percentage of similarity between two documents (A and B), the percentage of the number of matches with document A versus document B, the percentage of the number of matches with document B versus document A and the total amount of matches between the documents A and B. This system performs the comparison based on word trigrams.

# 4. EXPERIMENTAL WORK

To compare the performance obtained employing the plagiarism detection tools and the CHAMELEON clustering algorithm, we employed the following performance measures [4]:

- Spam Caught (SC): The percentage of spam messages detected;
- Blocked Hams (BH): The percentage of ham messages (non-spam messages) considered as spam;
- Accuracy (Acc) %;
- Matthews Correlation Coefficient (MCC) [3]

MCC is used in machine learning as a measure of the quality of binary classifications. It returns a real value between -1 and +1. A coefficient equals to +1 indicates a perfect prediction; 0, an average random prediction; and -1, an inverse prediction.

$$MCC = \frac{(|TP||TN|) - (|FP||FN|)}{\sqrt{(|TP| + |FP|)(|TP| + |FN|)(|TN| + |FP|)(|TN| + |FP|)}} \quad (1)$$

where $|TP|$ is the number of true positives, $|TN|$ is the number of true negatives, $|FP|$ is the number of false positives, and $|FN|$ is the number of false negatives.

## 4.1 Data Collection

As previously said, in order to carry out some preliminary experiments we used the SMS Spam Collection[8] which was recently released for research purposes.

[8]http://www.dt.fee.unicamp.br/~tiago/smsspamcollection

**Table 1: Basic statistics of the SMS Spam Collection**

| Msg | Amount | % |
|---|---|---|
| Hams | 4,827 | 86.60% |
| Spams | 747 | 13.40% |
| Total | 5,574 | 100% |

**Table 2: Results of the experiments**

| Classifier | SC | BH | Acc | MCC |
|---|---|---|---|---|
| CHAMELEON | 79.51% | 4.62% | 93.25% | 0.781 |
| CopyCatch | 60.04% | 0.00% | 94.94% | 0.751 |
| WCopyFind | 58.07% | 0.00% | 94.51% | 0.743 |
| Pl@giarism | 56.93% | 0.00% | 94.19% | 0.722 |

This collection consists of the union of several SMS spam collections [2]:

- A collection of 425 SMS spam messages which was manually extracted from the Grumbletext[9] Web site;
- A subset of 3,375 SMS randomly chosen ham messages of the NUS SMS Corpus[10] (NSC), which is a dataset of about 10,000 legitimate messages collected for research at the Department of Computer Science at the National University of Singapore;
- A list of 450 SMS ham messages collected from Caroline Tag's PhD Thesis [36];
- And the SMS Spam Corpus v.0.1 Big[11]. It has 1,002 SMS ham messages and 322 spam messages.

In summary, the SMS Spam Collection is composed by a total of 5,574 short messages of which 4,827 are ham messages and 747 are spam messages. Table 1 shows the basic statistics of the collection[12]. The SMS Spam Collection is composed by just one text file, where each line has the correct class followed by the raw message. Below we illustrate an example:

```
ham     Ok lar... Joking wif u oni...
ham     Oh k...i'm watching here:)
spam    Are you unique enough? Find out from 30th
August. www.areyouunique.co.uk
```

## 4.2 Results

As already mentioned, the aim of these preliminary experiments is to investigate whether plagiarism detection tools could be used as filter for SMS spam text messaged. Moreover, we also try to address the near-duplicate detection problem on the basis of the CHAMELEON clustering algorithm.

Table 2 shows the results obtained (results are sorted by descending MCC measure). The best result was obtained using CHAMELEON (MCC 0.781). Moreover, CHAMELEON obtained the highest percentage of SMS spam detected (79.51% vs. 60.04% with CopyCatch, the plagiarism detection tool that obtained the best results). However, with CHAMELEON

[9]http://www.grumbletext.co.uk/
[10]http://www.comp.nus.edu.sg/~rpnlpir/downloads/corpora/smsCorpus/
[11]http://www.esp.uem.es/jmgomez/smsspamcorpus/
[12]For more information on the collection see [2]

**Table 3: Comparation with the fifteen best results obtained in [2]**

| Classifier | SC | BH | Acc | MCC |
|---|---|---|---|---|
| SVM | 83.10% | 0.18% | 97.64% | 0.893 |
| Boosted NB | 84.48% | 0.53% | 97.50% | 0.887 |
| Boosted C4.5 | 82.91% | 0.29% | 97.50% | 0.887 |
| PART | 82.91% | 0.29% | 97.50% | 0.887 |
| MDL | 75.44% | 0.35% | 96.26% | 0.826 |
| **CHAMELEON** | 79.51% | 4.62% | 93.25% | 0.781 |
| C4.5 | 75.25% | 2.03% | 95.00% | 0.770 |
| **CopyCatch** | 60.04% | 0.00% | 94.94% | 0.751 |
| **WCopyFind** | 58.07% | 0.00% | 94.51% | 0.743 |
| **Plagiarism** | 56.93% | 0.00% | 94.19% | 0.722 |
| Bern NB | 54.03% | 0.00% | 94.00% | 0.711 |
| MN TF NB | 52.06% | 0.00% | 93.74% | 0.697 |
| MN Bool NB | 51.87% | 0.00% | 93.72% | 0.695 |
| 1NN | 43.81% | 0.00% | 92.70% | 0.636 |
| Basic NB | 48.53% | 1.42% | 92.05% | 0.600 |
| Gauss NB | 47.54% | 1.39% | 91.95% | 0.594 |
| Flex NB | 47.35% | 2.77% | 90.72% | 0.536 |
| Boolean NB | 98.04% | 26.01% | 77.13% | 0.507 |
| 3NN | 23.77% | 0.00% | 90.10% | 0.462 |
| EM NB | 17.09% | 4.18% | 85.54% | 0.185 |

the percentage of false positive (SMS ham considered as spam) is greater than with the plagiarism detection tools (4.62% vs. 0.00%).

However, the results obtained with the CopyCatch plagiarism detection tool in terms of MCC measure are quite similar to the ones obtained with CHAMELEON, and in terms of accuracy are even better. This is because the SMS Spam Collection is an imbalanced corpus (the 88% of the SMS are hams), and plagiarism detection tools seem to be more effective in these cases. Last but certainly not least, the percentage of SMS hams blocked with the plagiarism detection tools is 0.00%. This is an important aspect of plagiarism detection tools versus CHAMELEON, since one of the main problems of SMS spam filters is to avoid blocking SMS hams because this may prevent urgent messages to be blocked.

In Table 3 we compare our results with the ones obtained by the authors that released the SMS Spam Collection [2]. Several well-known machine learning methods have been used in order for automatic spam filtering. Although the results of the top methods are slightly better if compared to the ones obtained with CHAMELEON and the plagiarism detection tools (the SVM classifier obtained 0.893 in MCC measure and 83.10% of percentage of SMS spam detected), plagiarism detection tools showed to be a valid alternative for trying to address the problem from a near-duplicate detection problem especially because they prevent from blocking (potentially important) ham SMS to be considered as spam text messages.

## 5. CONCLUSIONS AND FURTHER WORK

The aim of this paper was to perform some preliminary experiments to investigate whether plagiarism detection tools could be used as filter for SMS spam text messages.

The results show that the plagiarism detection tools detected a good number of near-duplicate SMS spam messages. However, CHAMELEON clustering algorithm was able to detect a greater number of SMS spam messages.

However, CHAMELEON obtained a higher percentage of false positive (non-spam messages detected as spam). This could be a problem because important legitimate messages as of emergency nature could be blocked. On the contrary, the plagiarism detection tools do not block any non-spam message.

As future work it would be interesting to apply a near-duplicate based approach to address the problem of the detection of SMS spam text messages in order to improve the results. Moreover, it would also be interesting to investigate whether these techniques may help to solve the problem of opinion spam detection [30].

## 6. ACKNOWLEDGMENTS

This work has been done in the framework of the VLC/ CAMPUS Microcluster on Multimodal Interaction in Intelligent Systems and it has been partially funded by the European Commission as part of the WIQEI IRSES project (grant no. 269180) within the FP 7 Marie Curie People Framework, and by MICINN as part of the Text-Enterprise 2.0 project (TIN2009-13391-C04-03) within the Plan I+D+i.

We thank Jose Maria Gómez Hidalgo for sharing the SMS Spam Collection. We thank also M. Teresa Turell Julià to allow us to get familiar with CopyCatch in the ForensicLab of IULA, Universitat Pompeu Fabra.

## 7. REFERENCES

[1] H. Ahmad. Plagiarism detection systems : An evaluation of several systems. In *The 6th SEAAIR Annual Conference*, pages 5–7, Langkawi, 2006.

[2] T. A. Almeida, J. M. Gómez Hidalgo, and A. Yamakami. Contributions to the study of SMS Spam Filtering: New Collection and Results. In *Proceedings of the 2011 ACM Symposium on Document Engineering (ACM DOCENG'11)*, 2011.

[3] T. A. Almeida, A. Yamakami, and J. Almeida. Evaluation of approaches for dimensionality reduction applied with naive bayes anti-spam filters. In *Proceedings of the 2009 International Conference on Machine Learning and Applications*, ICMLA '09, pages 517–522, Washington, DC, USA, 2009. IEEE Computer Society.

[4] T. A. Almeida, A. Yamakami, and J. Almeida. Filtering spams using the minimum description length principle. In *Proceedings of the 2010 ACM Symposium on Applied Computing*, SAC '10, pages 1854–1858, New York, NY, USA, 2010. ACM.

[5] E. Atwell, P. Gent, J. Medori, and C. Souter. Detecting student copying in a corpus of science laboratory reports: simple and smart approaches, 2002.

[6] A. Barrón-Cedeño, P. Rosso, E. Agirre, and G. Labaka. Plagiarism detection across distant language pairs. In *Proc. of the 23rd International Conference on Computational Linguistics, COLING-2010*, pages 37–45, Beijing, China, 2010.

[7] A. Barrón-Cedeño, P. Rosso, and J. M. Benedí. Reducing the plagiarism detection search space on the basis of the Kullback-Leibler distance. *Proceedings of the 10th International Conference on Computational*

*Linguistics and Intelligent Text Processing, (CICLing'09).*, pages 523–534, 2009.

[8] S. Brin, J. Davis, and H. Garcia-Molina. Copy detection mechanisms for digital documents. In *ACM International Conference on Management of Data (SIGMOD 1995)*, 1995.

[9] S. Brin, J. Davis, and H. Garcia-Molina. Message text clustering based on frequent patterns. In *M.S. thesis, Institute of Computing Technology, Chinese Academy of Sciences. Beijing, China*, 2006.

[10] A. Z. Broder. On the resemblance and containment of documents. *Compression and Complexity of Sequences*, pages 21–29, 1997.

[11] A. Z. Broder, S. C. Glassman, M. S. Manasse, and G. Zweig. Syntactic clustering of the web. In *Selected papers from the sixth international conference on World Wide Web*, pages 1157–1166, Essex, UK, 1997. Elsevier Science Publishers Ltd.

[12] J. Conrad and C. P. Schriber. Constructing a text corpus for inexact duplicate detection. In *Proceedings of ACM SIGIR'04*, pages 25–29, South Yorkshire, UK., 2004.

[13] K. Dave, S. Lawrence, and D. M. Pennock. Mining the peanut gallery: opinion extraction and semantic classification of product reviews. In *Proceedings of the 12th international conference on World Wide Web*, WWW '03, pages 519–528, New York, NY, USA, 2003. ACM.

[14] J. Dierderich. Computational methods to detect plagiarism in assessment. *Proceedings of the 7th International Conference on Information Technology Based Higher Education and Training (ITHET '06)*, pages 147–154, 2006.

[15] D. Fetterly, M. Manasse, and M. Najork. On the evolution of clusters of near-duplicate web pages. In *Proceedings of the First Conference on Latin American Web Congress*, volume 2, pages 37–45, Washington, DC, USA, 2003. IEEE Computer Society.

[16] C. Grozea, C. Gehl, and M. Popescu. ENCOPLOT: Pairwise sequences matching in linear applied to plagiarism detection. In B. Stein, P. Rosso, E. Stamatatos, M. Koppel, and E. Agirre, editors, *SEPLN 2009 Workshop on Uncovering Plagiarism, Authorship, and Social Software Misuse (PAN 09)*, pages 10–18, San Sebastian, Spain, 2009. CEURWS.org. http://ceur-ws.org/Vol-502.

[17] H. Hajishirzi, W.-t. Yih, and A. Kolcz. Adaptive near-duplicate detection via similarity learning. In *Proceeding of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '10, pages 419–426, New York, NY, USA, 2010. ACM.

[18] N. Jindal and B. Liu. Opinion spam and analysis. In *Proceedings of the international conference on Web search and web data mining*, WSDM '08, pages 219–230, New York, NY, USA, 2008. ACM.

[19] N. Kang, A. Gelbukh, and S. Y. Han. PPChecker: Plagiarism pattern checker in document copy detection. *Lecture notes in computer science*, (4188):661–668, 2006.

[20] J. Kasprzak and M. Brandejs. Improving the reliability of the plagiarism detection system. In *Notebook Papers of CLEF 2010 LABs and Workshops*, University of Padova, Padova, Italy, 2010.

[21] A. Kolcz and A. Chowdhury. Hardening fingerprinting by context. In *CEAS*, 2007.

[22] A. Kolcz and A. Chowdhury. Lexicon randomization for near-duplicate detection with i-match. *The Journal of Supercomputing*, 45:255–276, 2008.

[23] M. Koppel, J. Schler, and S. Argamon. Computational methods in authorship attribution. *Journal of the American Society for Information Science and Technology*, 60(1):9–26, 2009.

[24] J. P. Kumar and G. P. Duplicate and near duplicate documents detection: A review. *European Journal of Scientific Research*, 32:514–527, 2009.

[25] C. Lyon, R. Barrett, and J. Malcolm. A theoretical basis to the automated detection of copying between texts, and its practical implementation in the Ferret plagiarism and collusion detector. In *Plagiarism: Prevention, Practice and Policies Conference*, Newcastle, UK, 2004.

[26] G. S. Manku, A. Jain, and A. D. Sarma. Detecting near-duplicates for web crawling. *Proceedings of the 16th international conference on World Wide Web*, pages 141–150, 2007.

[27] M. Mathew, S. N. Das, T. R. L. Narayanan, and P. K. Vijayaraghavan. Article: A novel approach for near-duplicate detection of web pages using tdw matrix. *International Journal of Computer Applications*, 19(7):16–21, April 2011.

[28] M. Muhr, R. Kern, Z. M., and M. Granitzer. External and intrinsic plagiarism detection using a cross-lingual retrieval and segmentation system. In *Notebook Papers of CLEF 2010 LABs and Workshops*, University of Padova, Italy, 2010.

[29] V. A. Narayana, P. Premchand, and A. Govardhan. Fixing the threshold for effective detection of near duplicate web documents in web crawling. In *Proceedings of the 6th international conference on Advanced data mining and applications: Part I*, ADMA'10, pages 169–180, Berlin, Heidelberg, 2010. Springer-Verlag.

[30] M. Ott, Y. Choi, C. Cardie, and J. T. Hancock. Finding deceptive opinion spam by any stretch of the imagination. *ACL'11*, pages 309–319, 2011.

[31] M. Potthast, A. Barrón-Cedeño, B. Stein, and P. Rosso. Cross-language plagiarism detection. *Languages Resources and Evaluation. Special Issue on Plagiarism and Authorship Analysis*, 45(1), 2011. doi: 10.1007/s10579-009-9114-z.

[32] S. Schleimer, D. S. Wilkerson, and A. Aiken. Winnowing: Local algorithms for document fingerprinting. In *Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data*, New York, NY, 2003.

[33] N. Shivakumar and H. Garcia-Molina. SCAM: A copy detection mechanism for digital documents. *Proceedings of the 2nd International Conference on Theory and Practice of Digital Libraries*, 1995.

[34] E. Stamatatos. Intrinsic plagiarism detection using character n-gram profiles. *Proceedings of the SEPLN'09 Workshop on Uncovering Plagiarism, Authorship and Social Software*, pages 38–46, 2009.

[35] B. Stein, N. Lipka, and P. Prettenhofer. Intrinsic plagiarism analysis. *Language Resources and Evaluation*, 2010.

[36] C. Tagg. A corpus linguistics study of sms text messaging. PhD Thesis. The University of Birmingham, Birmingham, UK, July 2009.

[37] E. Vallés Balaguer. Putting ourselves in SME's shoes: Automatic detection of plagiarism by the WCopyFind tool. In B. Stein, P. Rosso, E. Stamatatos, M. Koppel, and E. Agirre, editors, *SEPLN 2009 Workshop on Uncovering Plagiarism, Authorship, and Social Software Misuse (PAN 09)*, pages 34–35, San Sebastian, Spain, 2009.