DOCODE 3.0 (DOcument COpy DEtector): A system for plagiarism detection by applying an information fusion process from multiple documental data sources

Juan D. Velásquez[a], Yerko Covacevich[a], Francisco Molina[a], Edison Marrese-Taylor[a], Cristián Rodríguez[a], Felipe Bravo-Marquez[b]

[a] *Department of Industrial Engineering, University of Chile, Av. República 701*
[b] *Department of Computer Science, The University of Waikato, Private Bag 3105, Hamilton 3240, New Zealand*

Abstract

Plagiarism refers to the act of presenting external words, thoughts, or ideas as one's own, without providing references to the sources from which they were taken. The exponential growth of different digital document sources available on the Web has facilitated the spread of this practice, making the accurate detection of it a crucial task for educational institutions. In this article, we present DOCODE 3.0, a Web system for educational institutions that performs automatic analysis of large quantities of digital documents in relation to their degree of originality. Since plagiarism is a complex problem, frequently tackled at different levels, our system applies algorithms in order to perform an information fusion process from multi data source to all these levels. These algorithms have been successfully tested in the scientific community in solving tasks like the identification of plagiarized passages and the retrieval of source candidates from the Web, among other multi data sources as digital libraries, and have proven to be very effective. We integrate these algorithms into a multi-tier, robust and scalable JEE architecture, allowing many different types of clients with different requirements to consume our services. For users, DOCODE produces a number of visualizations and reports from the different outputs to let teachers and professors gain insights on the originality of the documents they review, allowing them to discover, understand and handle possible plagiarism cases and making it easier and much faster to analyze a vast number of documents. Our experience here is so far focused on the Chilean situation and the Spanish language, offering solutions to Chilean educational institutions in any of their preferred Virtual Learning Environments. However, DOCODE can easily be adapted to increase language coverage.

Keywords:
Plagiarism detection, text patterns information fusion, multi documental data sources.

1. Introduction

Today's scenario shows a significant change in the way of accessing information, emphasizing the use of the Web as one of the main sources of knowledge [48, 49]. However, access to the Web has been cited as one of the main reasons for the perceived decline in academic integrity, particularly in relation to plagiarism [44].

Plagiarism basically consists of taking others' work and labeling it as one's own. Likewise, text plagiarism is defined as the action of copying someone else's writings without the proper citation. When applied to the

Email addresses: jvelasqu@dii.uchile.cl (Juan D. Velásquez), ycovacev@docode.cl (Yerko Covacevich), frmolina@ing.uchile.cl (Francisco Molina), emarrese@wi.dii.uchile.cl (Edison Marrese-Taylor), crodriguezo@wi.dii.uchile.cl (Cristián Rodríguez), fjb11@students.ac.waikato.nz (Felipe Bravo-Marquez)
URL: http://wi.dii.uchile.cl (Juan D. Velásquez)

educational environment, we also find that the term student plagiarism is often used to refer to the incidences of plagiarism committed by students who attend educational institutions [20], which mainly represent cases of text plagiarism. In this context, because there is a vast amount of easy-to-access information, the plagiarism phenomenon has been becoming more popular and easier to resort to. International studies demonstrate the magnitude of this behavior, with a high percentage of students who reported to be using the Web as a major source of plagiarism [29]. In [38], Posner recently estimated that one-third of all high school and college students have committed some kind of plagiarism. The situation in Chile is not different. A 2010 survey carried out by the Department of Industrial Engineering of the University of Chile, showed that about 55% of middle school students and 42% of higher education students declared having copied information without citing the source [31].

Given the large volume of documents and information sources that exist today, originality examination and plagiarism detection are becoming increasingly more complex tasks. While Web search engines can be used to detect Internet plagiarism, the detection process is, by any standards, both tedious and labor-intensive [20]. In today's scenario, a manual examination appears as an extremely time-consuming process and a virtually impossible task; teachers often do not have the necessary time for exhaustive reviews. Also, some students will continue to plagiarize regardless of how hard tutors try to stop them [22]. In the Chilean case, the absence of a suitable plagiarism detection system in Spanish contributes to making the situation we have described above even more alarming.

Plagiarism is an important issue for educational purposes at every level, because it could affect a student's learning process [27]. Teachers and academics abhor plagiarism because it is inconsistent with pedagogical aims. As a result there has been a desire on the part of teachers to attack the problem by developing different measures to detect the originality of the work submitted by the students [44]. Looking at the extent of the problem, [16] concludes that it is quite obvious that academia requires tools to automate and enhance plagiarism detection. These tools, often called plagiarism detection engines, are software that compare documents with possible sources in order to identify similarity and so discover submissions that might be plagiarized [12], making it easier for teachers to analyze a vast number of documents.

A review of the literature about plagiarism in educational institutions shows that many authors have proposed that it is a set of distinct inappropriate behaviors rather than only a single problem. In an effort to tackle this complexity, some of these authors have actually proposed different levels or types of plagiarism, generating subproblems that might be easier to analyze.

From our perspective, when referring to educational purposes, plagiarism detection engines are supposed to offer professors a set of tools to gain insights about the documents that are reviewed, rather than simply checking plagiarism cases, thus tackling the problem of plagiarism from all the perspectives existing in literature. Therefore, our work presents a system that performs automated textual plagiarism detection for educational institutions using a multi-level perspective. Our system, called DOCODE 3.0 (DOcument COpy DEtector 3.0)[1], cooperates with teachers and professors offering them a complete interface with visual tools to discover, understand and handle different plagiarism levels and cases. DOCODE is a full-featured system based on a solid, scalable architecture and implementing a set of algorithms for plagiarism detection that have successfully proven to be effective, in some cases, even outperforming state-of-the-art approaches in literature. These results were validated in multiple previous publications and in international plagiarism detection competitions. Although our experience is so far limited to the Chilean situation and the Spanish language, most of our algorithms are not language dependent, so DOCODE can easily be adapted to increase language coverage.

The rest of this paper is structured as follows. Below in Section 2, we review related work regarding the plagiarism topic and also present some of the most important state-of-the-art plagiarism detection algorithms and frameworks. Then, in Section 3, we explain how DOCODE works and what kind of services it provides. Also, the main algorithms underlying the system are presented. Section 4 shows how DOCODE is structured, explaining its architecture. Later, Section 5 introduces our user interfaces. Finally, Section 6 presents conclusions and proposed future work.

[1] http://www.docode.cl/

2. Related Work

In this section, we give a short review about plagiarism, including most of the important definitions stated by the scientific community, state-of-the-art approaches in automatic plagiarism detection and also a brief review of some the most important copy-detector systems.

2.1. Toward a categorization of plagiarism

Although authors have proposed different definitions of plagiarism over the years, it is possible to state without loss of generality that plagiarizing means to appropriate ideas, passages, etc., from another work or author [18]. However, a closer loot at all these definitions allows us to see that some authors have proposed that plagiarism is a set of different inappropriate behaviors rather than only a single problem, defining categories or types of plagiarism. In fact, one of the first attempts to define plagiarism types was proposed in the early 90s, in [26]. The paper proposes that plagiarism can take six distinct forms, which were also discussed later in [14]. The same paper also says that in education, students may plagiarize to gain a grade, while in academics, the reason may often be to gain popularity and status. However, in both cases, if a plagiarism relationship exists between two texts, it suggests that the texts exhibit some degree of intertextuality, which would not appear between them if independently written.

On the other hand, [1] addresses the problem of student cheating and gives some definitions, categorizing the types of cheating behavior related to plagiarism offenses into copying, exams, collaboration and deception. In [33], authors state that students may use various techniques for disguising plagiarism in their submitted work, regardless of the type of cheating behavior. From our point of view, a classification of plagiarism types is useful in understanding the challenges that appear for automatic plagiarism detection systems. For our case, we have taken the ideas first presented in [28] and developed further in [20], and used them as a framework to later evaluate how DOCODE 3.0 tackles these issues. Basically, we consider the following text plagiarism cases.

(1) *Verbatim copying*: Copy-paste copying from an electronic source, including authorship plagiarism (taking someone else's text and putting one's own name.)
(2) *Paraphrasing*: Adding, replacing or removing characters or words. Adding deliberate grammar or spelling mistakes. Replacing words with its synonyms. Reordering sentences and phrases. Translated plagiarism could also be placed in this category.
(3) *Technical tricks to exploit weaknesses of systems*: Mainly, the insertion of invisible white-colored letters into what seems to be blank space and the insertion of scanned text pages as images into a document, which cannot be processed by systems since they are not recognized as text.
(4) *Deliberate and/or inaccurate use of references*: Providing fake references (those that do not really exist), false references (they exist but do not match text being referenced) and the use of "forgotten" or expired links to sources.

Our belief is that these categories are suitable for analyzing the problem of automatic plagiarism detection from the perspective of education because each one of them defines a specific detection problem. We do not claim that this categorization is either the only correct one nor the most comprehensive in the literature. Here, we simply propose this categorization for the sake of the analysis of our proposed tool, bearing in mind that an automatic plagiarism detector should tackle as many of these categories as possible, although they are clearly not equally challenging [33].

2.2. Automatic plagiarism detection

A lot of research has been conducted on detecting plagiarism automatically. However, in most of the recent work, as in [41], plagiarism is simply considered as the reuse of someone else's work while pretending it to be one's own. In this context, [43] declares that literature on the subject often puts plagiarism detection on a level with the identification of highly similar sections in texts or other objects. Thus, [39] gives a formal definition of a plagiarism case $s = \langle s_{plg}, d_{plg}, s_{src}, d_{src} \rangle$ as a 4-tuple which consists of a passage s_{plg} in a document d_{plg} that is the plagiarized version of some source passage s_{src} in d_{src}. When given d_{plg}, the task of

a plagiarism detector is to detect s, say, by reporting a plagiarism detection $r = \langle r_{plg}, d_{plg}, r_{src}, d'_{src} \rangle$ which consists of an allegedly plagiarized passage r_{plg} in d_{plg} and its source r_{src} in d'_{src}, and which approximates s as closely as possible.

The same authors also considered that the existing view on plagiarism did not show the whole picture and decided to divide plagiarism detection into two major problem classes, namely external plagiarism detection and intrinsic plagiarism detection. On the one hand, in external plagiarism detection, it is assumed that the source document d_{src} for a given plagiarized document d_{plg} can be found in a document collection D, such as the Web. On the other hand, in intrinsic plagiarism detection, the plagiarism detector attempts to detect plagiarized passages solely based on information extracted from d_{plg} [30]. From our point of view, the intrinsic/external segmentation is interesting because each approach includes a number of subproblems that can be used to define the features (or services) provided by an automatic plagiarism detection system. Furthermore, some of these problems could also be linked with specific levels of plagiarism as defined in the last section.

In order to develop new insights on the topic, annual competitions have been organized under the name of PAN[2] since 2009 [43], [39], [40] and [41]. For the purpose of these competitions, organizers also developed the first corpora explicitly comprising plagiarism text and a set of performance measures for plagiarism detection in [3]. Later, the same authors presented an evaluation framework for plagiarism detection in [42], which included a revised version of their corpora and the formal definition of metrics to evaluate the performance of an automatic plagiarism detector. Again, in 2012, the same authors decided to construct a new corpus comprising long, manually written documents, for the first time emulating the entire process of plagiarizing [41]. Besides the PAN-PC corpora, it is only possible to find the Clough09 corpus, which consists of 57 short answers to one of 5 computer science questions [13].

As a result of the PAN competitions and because of the interest of the scientific community in the problem of plagiarism, several techniques to detect different plagiarism types exist today. Depending on the kind of plagiarism that is being employed, an important set of approaches, based on different characteristics of the text, can be used to proceed with a detection method [12]. Existing literature on each topic is vast, so some authors have already surveyed approaches in automatic plagiarism detection. Here, we merely give reference to the most important studies, including [14], [2], [17] and [28].

2.3. Systems for plagiarism detection

The interest in automated plagiarism detection does not only involve academia. Nowadays, several commercial tools are also available on the market. Each tool offers its own approach. However, in general, plagiarism detection systems can be divided into hermetic and Web. Web detection systems try to find matches for the suspected document in on-line sources while hermetic systems search for instances of plagiarism only within a local collection of documents [33]. Also, some of the existing systems, like Turnitin, are offered as services, whereas others can be directly downloaded and run on a computer or server. A short description of a few of them is given below.

- Turnitin[3]: a commercial company that offers services for plagiarism detection. It checks students' work against continuously updated databases. It currently has more than a hundred million students' papers, over twelve million crawled Web pages and access to magazines and newspapers.
- EVE2[4]: a commercial tool that searches the Web for possible sources of a suspicious student paper. It returns the URLs it finds and a full report to the teacher.
- PlagiarismDetect.com[5]: a commercial tool that searches the Web for possible sources. It works similarly to EVE2.

[2] http://pan.webis.de
[3] http://www.turnitin.com
[4] http://www.canexus.com
[5] http://www.plagiarismdetect.com

- Glatt Plagiarism Services[6]: offers three pieces of software. The first is a tutorial program for help in educating students about what plagiarism is and how to avoid it. The second one is a screening program to detect plagiarism in documents, whereas the third is also a screening program for detection of inadvertent instances of plagiarism.
- Ephorus[7]: Another commercial tool for plagiarism detection that includes support for checking for correct quotations and citations, also giving users the possibility to locate cited sources as used when writing academic papers. The tool is now fusing with Turnitin.
- WCopyfind[8]: An open source windows-based program that compares documents and reports similarities in their words and phrases. It is free and licensed under the GNU Public License.

So far, several publications revise tools and compare them under different criteria, including [23], [28], [10], [20] and [22], among others. To the best of our knowledge, most of the successful commercial tools do not give details on their algorithms and work as black boxes. Clearly, from the perspective of the users, this lack of information may represent an important barrier in understanding how the results are built and shown, making it harder to gain real insights on the documents that are being reviewed. It also makes it more difficult to test, evaluate and compare the performance of these tools. In this sense, DOCODE is different since its related phenomena and inherent algorithms and programs are well known by the scientific community and by our users. Our work is publicly available for the community in the papers that we have published so far.

3. Proposed System

In this and in the following two sections, we present a complete description of our system. In the first place, we will give details about the algorithms that support all our core functionalities. Later, we will explain how DOCODE is structured. Finally, we will explain DOCODE's user interfaces. This organization responds to the fact that our system presents three main contributions.

(1) We developed and implemented several algorithms that tackle all the aspects of the problem of plagiarism, according to what is currently proposed in literature. Some of these algorithms even outperform state-of-the-art approaches. These contributions belong to the field of Information Retrieval (IR).
(2) We provide a robust, efficient and scalable architecture to support all our functionalities and provide high quality services. Our architecture is based on the JEE paradigms, providing a set of developer-friendly interfaces for programmers. Thus, our contributions here are in the field of Software Engineering (SE).
(3) We designed and implemented friendly user interfaces with a set of visualization charts and tools to support the teacher's decision-making process in relation to possible plagiarism cases. Our third contribution is therefore in the field of Human Computer Interaction (HCI).

In the next subsections, we present details on each of the implemented algorithms.

3.1. Cross-document copy-detector: FASTDOCODE

The first service provided by DOCODE is based on our cross-document copy-detector algorithm, called FASTDOCODE. As introduced in [34], FASTDOCODE is based on two main phases which are applied after an initial preprocessing step in each case. In general terms, our algorithm first reduces the search space by using an approximated search of segments of n-grams based on the proposals of [4] and then, using an exhaustive search algorithm within selected pairs of documents, finds the offset and its length for both exact and paraphrased copy[9].

[6] `http://www.plagiarism.com`
[7] `https://www.ephorus.com`
[8] `http://plagiarism.bloomfieldmedia.com/z-wordpress/software/wcopyfind`
[9] Originally referred as obfuscated copy, using the terminology introduced in [43].

In the following, let V be a vector of words that defines the vocabulary to be used. A word will be noted with w, and will be our basic unit of discrete data, indexed by $\{1, ..., |V|\}$. A document d will be a sequence of p words, $p = |d|$, defined by $d = (w_1, ..., w_p)$ where w_p represents the p^{th} word in the document. Finally, a corpus is defined by a collection of n documents denoted by $D = (d_1, ..., d_n)$.

Given a set D containing a set of possibly suspicious documents, the first phase applies a search space reduction method, which aims at quickly identifying those pairs of documents (d_i, d_j) in the set that potentially have some text in common, possibly one of them having been plagiarized from the other. Preprocessing here includes removing stopwords, obtaining the set of n-grams for the pair of documents d_i and d_j and generating groups of k n-grams. Once both documents are transformed into groups or segments of n-grams, t_i, t_j and k_i, k_j respectively, the n-grams are sorted according to a specified sorting strategy and then a special function selects only the last m n-grams within each segment. This step is proposed as an analogy to a sampling strategy for each segment, thus contributing to minimize the number of comparisons to be executed and enhancing the runtime of the algorithm [34]. Finally, the method's strategy considers an analysis based on word 4-grams, in which if two documents have at least two word 4-gram coincidences close enough as to be in the same paragraph, the documents are marked as suspicious and considered for the next phase [35].

The second phase is an exhaustive search to find plagiarized passages in the documents that were previously detected as suspicious. Conversely, for this phase stopwords are not removed and word 3-grams are used. Also, rather than a full representation of documents, just a subset of l 3-grams is used. Basically, the intersection between the 3-grams of different segments is computed. If the number of common 3-grams in a pair of these segments —each one of a different document— is greater than a parameter r, then a similarity indicator is increased. After finishing comparing every pair of groups, the final similarity indicator is returned. Values of parameters l and r as well as other details of the algorithm are not revealed due here to copyright.

FASTDOCODE was tested during the PAN2010 and PAN2011 competitions on plagiarism detection and was awarded with the 5th (out of 18 competitors) and 3rd place (out of 8 competitors) respectively. Parameters of the algorithms, including the number of n-grams to use for both phases were tuned considering the PAN-PC-2010 corpus. However, they were not selected using an extensive analysis on the algorithms performance due to the size of the corpus. Since it was difficult to run an optimization or grid search strategy over the parameters, we approximated them by iterating and trying on samples. Using the PAN-PC-2010 corpus, the algorithm was able to achieve a precision and recall of 94% and 60% respectively, which lead to an F-measure of 73%. Likewise, using the PAN-PC-2011 corpus, the algorithm obtained 22.58% for recall and 91.17% for precision. More details of these results can be found in the publications that have been released regarding our algorithm so far, particularly in [36].

3.2. Change of Writing Style Detector

Our second functionality is intended to extract evidence of a potential case of plagiarism from a suspicious document only. The approach attempts to find suspicious passages on it using the intrinsic plagiarism detection strategy.

Our algorithm is based on the characterization of the writing style of an author and the following intuition: if some of the words used in the document are author specific, one can think that those words could be concentrated in the paragraphs (or more generally, in the segments) that the mentioned author wrote. In this manner, [36] proposed a model for writing style quantification, aimed at finding significant deviations in a document's writing style; these differing segments could have been plagiarized and are probably useful as a starting point to search for possible source candidates.

Our approach works as follows. First, the document is preprocessed so only alphabetic characters are kept and case folded —that is, spaces, punctuation marks and numbers are discarded. Without removing stopwords, we extract unigrams and create a number of segments of the given documents using a sliding window of length m words over it. The next step consists in the application of a word-frequency-based algorithm to test the self-similarity of the document itself. The general footprint or *style* of a document is represented by the average of all differences of frequencies in terms of the words present for each segment

and in the complete document. After this, we compute the differences in the value of the *styles* between each section and the average *style*. Thus, if certain words are only used in a certain segment, the comparison of that segment against the whole document would lead to a low value in the difference between their *styles*, because the frequency of those words would be the same in both the whole document and the segment. If the variation is significant, the *style* will be lower than the average value minus a predefined threshold δ. Finally, all segments are classified according to their distance with respect to the value of the document's *style*. If the *style* of a segment is lower than the value of the document's *style* minus the threshold, then the segment is classified as suspicious.

During the PAN competitions on plagiarism detection, our algorithm was tested using the PAN-PC-2010 and PAN-PC-2011 corpora. For the parameters, as stated in [35], a sliding window of 400 words and a threshold parameter $\delta = 0.075$ were used. These were iteratively adjusted depending on text length. In the first case, the obtained performance was 38.97% for precision and 31.09% for recall, or an F-measure of 34.58%. For the 2011 corpus, we obtained a precision and recall of 33.98% and 31.23%, which led to an overall score of 32.54%. Compared to state-of-the-art approaches, the algorithm achieved remarkable results, managing to obtain the first place at the PAN-PC-2011, almost doubling the score of the second-place team. It is also important to say that the method does not utilize language-dependent features such as verbs or stopwords, thus providing a starting point to experiment with other languages. More details about these results can be found in the related papers [40], [25] and [36].

3.3. Hidden-Text Detector

As we presented in Section 2, some plagiarism methods include the use of techniques or tricks that try to exploit various weaknesses of existing plagiarism detection systems. Here, we tackle the insertion of invisible white-colored letters as a replacement for blank spaces. Based on our experience in the Chilean case, this behavior appears to be one of the most common strategies employed by students when trying to disguise plagiarism.

Our approach in this context is an algorithm that checks if the average word length of each sentence is in a range of what is considered normal or acceptable. Threshold parameters were calibrated based on linguistic rules, as given to us by Linguistics experts. If an abnormal pattern is detected, the algorithm returns a value that can then be used to trigger alarms to the user.

3.4. Similar Web Document Retriever

Another basic feature that is provided by DOCODE is a similar Web document retriever, which is in charge of obtaining suspicious documents from the Web. Given an initial suspicious document d and a collection D of documents from which d's author may have plagiarized, the first step (so-called heuristic retrieval step) proposes to retrieve a small number of candidate documents $D_x \subseteq D$, which are likely to be sources for plagiarism. This usually considers that D is very large [45].

Based on the suspicious document our algorithm tackles the problem of obtaining a set of similar documents from the Web using search engines. We therefore see this problem from the perspective of Information Retrieval. Although literature usually proposes that search queries can be grouped into three categories, namely informational, navigational and transactional queries, in [37], the authors introduce a new information requirement —retrieving similar documents from the Web. In [8], the authors call this problem the Web document similarity retrieval problem (WDSRP). The key difference between classic query categories and the WDSRP is that, as input, it considers a given document instead of a text-based query.

As stated by [24], given a document d, from which a vocabulary V can be extracted, a language model MD from d is a function that maps a probability measure over strings drawn from V. Language models are used as ranking functions in information retrieval, estimating the probability of generating a query q given a document language model MD, i.e. $P(q|MD)$. In our query generation task, the probabilistic distribution from the language model is used as a randomized term extraction procedure from d. In this context, our algorithm uses the Hypergeometric Language Model (HLM), an extension of language models inspired by the multivalued hypergeometric distribution [19], thus proposing that when obtaining terms from d to create a query, terms should be extracted one by one without replacement. Here, the premise is that

new terms give more information to a search engine than repeated ones in the generated query, considering that search engines allow a maximum length of input queries. The process then starts with the extraction of the vocabulary from d and the assignment of term extraction probabilities, calculated using customizable weighting approaches like tf, tf-idf, among others [36]. Afterwards, we proceed to construct our queries by the concatenation of successive randomized term extractions, without replacing the extracted terms. The length of queries is customizable. In addition to this algorithm, we also proposed another query system aimed to extract a sample of proportionally distributed n-grams ensuring that the terms of a query belong to the same topic [7]. We call this algorithm the random n-gram sample (RNS) fingerprint approach.

Finally, we propose an algorithm to estimate the similarity between document d and each document retrieved from the Web. Our approach combines two main features. The first feature is based on the Zipf-like distribution function over the content of the retrieved documents, modeling the relevance of a given Web search engine answer of a query as a Zipf-like distribution. In this sense, the relevance of the results presented in a Web search engine is inversely related to their rankings. In this manner, we are estimating the relevance of the answer of a query by fusing its ranking and the reliability of search engine results [8]. Our second feature is the result of combining the title and the summary (a.k.a. snippet) of the search engine results into a vector space model, aiming to build an approximated representation of the document's content. Then, we use our two features together to predict similarity assuming that they are strongly related with the similarity between d and the Web document for each result. Our model can be fitted using methods such as Artificial Neural Networks (ANNs) or other related regression techniques [7].

Our strategy has been tested experimentally to measure the effectiveness of the model at satisfying user information needs, which are related to the WDSRP. We first generated a manually-elaborated corpus[10] of 160 paragraphs, selected from different Web sites in Spanish. Next, the paragraphs were sent to the system as input and the top 15 answers from each paragraph were manually reviewed and classified as relevant or non-relevant results (2,400 Web documents). The criteria to label an answer as relevant was that the retrieved document must contain the given paragraph exactly and the selected evaluation measure was the precision at k, the number of relevant documents retrieved in the top k results divided by the number of documents retrieved in the top k results. Table 1 shows the obtained performance.

k	1	2	3	4	5
Precision	86.9%	70.9%	60.6%	53.0%	46.9%

Table 1: Precision for relevant documents retrieved in the top k results.

As seen, results prove that our proposal is able to satisfy the document similarity retrieval problem. Likewise, they show that the developed meta-search model significantly improved the retrieval capacity over the results of a single search engine [25]. Our algorithm is currently able to connect with the three most important search engines, Google, Yahoo! Boss and Bing.

3.5. Quotation detector for Spanish

Another feature that is provided by our system is a bibliographic quotations detector. In this case, although we designed an algorithm for the Spanish language, similar approaches are also possible for other languages.

According to [32], a quotation is the usage of non-original content of an author, which is indicated in the text as a mark that references the original source. The same work also presents different categories of quotations:

1. Syntax-based categorization: refers to the parts of a sentence with their respective syntactic functions. In this category, it is possible to find specific typographic patterns that denote quotations. These patterns commonly include an entity, namely, the name of the referenced author, a reporting verb (such as indicates, proposes, establishes, etc.) and the quotation itself. It is possible to distinguish two categories: (1) direct quotations, where the words of the referred author are textually used, and (2) indirect quotations, where the words of the author that is being referred are paraphrased.

[10] Our corpus is freely available in `http://dcc.uchile.cl/~fbravo/docode/corpus.xml`

2. Discourse-based categorization: in this case, the quotation is analyzed in relation to the level of relevance that the author gives to it. This is evidenced as the inclusion or absence of the quotation itself in the text. Two categories are defined: (1) integral quotations, where the author that is being referred to is mentioned in the text, and (2) non-integral quotations, in which the author is mentioned outside of the main text, for example, using footnotes or parentheses.

In [46], guidelines of different writing styles are presented for writers, editors and publishers. Authors give several formats to write quotations, all of which were reviewed and grouped considering the categories presented above. Some of the quotation styles were able to be identified and extracted using simple patterns based on regular expressions. Therefore, specific strategies have been developed in order to detect each kind of quotation in the most accurate manner. These strategies include the use of regular expressions but also some other contextual information contained therein. We developed 13 different regular expressions to cover all the guidelines given in [46]. Among these expressions, based on our experience, we also included some patterns that are outside of the official guidelines. In general terms, the procedure to find quotations in a document is the following:

1. Extract the document's text.
2. Mark the text with the coordinates where regular expressions are detected.
3. Search patterns and compare them with a pre-determined tolerance.
4. If there is a detected pattern, it follows that there is a quotation.

In order to test the effectiveness of our approach, we manually elaborated a corpus of bibliographic quotations using undergraduate bachelor theses of the University of Chile Industrial Engineering and Agronomy students as a basis. We reviewed these theses, extracting about 250 and 530 quotations respectively. Once extracted, each quotation was classified using the categories described before. Then, we designed two experiments to test our proposals.

The first experiment was intended to evaluate our quotation-detection algorithm under controlled conditions. In order to do this, we downloaded 484 books from the public domain and selected 5,890 phrases. We then mixed these phrases with the quotations from the agronomy theses, generating a corpus of 6,401 sentences. Our idea was to simulate a collection of documents, in which quotations only represent a small fraction of the sentences. We evaluated different combinations of patterns and our best model obtained an average precision and recall of 85.59% and 55.6% respectively, which led to an F-measure of 48.24%.

Our second experiment aimed to evaluate our algorithm in a real situation. In order to do so, we selected 20 agronomy theses of our corpus and tested our best algorithm on them, using the complete texts as input. Altogether, the documents had 15,169 sentences, from which only 536 were tagged as quotations. This only represents 3.53% of the sentences. Results showed that under real conditions, our algorithm is capable of achieving a reasonable performance in extracting quotations from documents in Spanish. On average, we achieve a precision of 24.45% and a recall of 76.91%. Bearing in mind that our corpus is a highly unbalanced dataset, recall here seems to be more critical. As a detailed inspection of the results showed us, given that our quotation detector is based on regular expressions that include the usage of certain verb patterns, many of the mislabeled cases are simply chunks of text that can be easily recognizable and discarded as non-quotations by a simple visual inspection of the documents. Considering this, even though the algorithm is mislabeling many of the unquoted text cases as such, since our system is intended to provide support to human users on discovering possible plagiarism cases, the obtained value for recall is encouraging. On the other hand, precision can be improved using additional algorithms to filter our extracted results, for instance, using results from search engines. In this sense, our results seem to be effective and very promising.

3.6. Multi-document Thematic Analyzer

Another feature offered by DOCODE is a thematic analyzer of a large set of documents, which allows visualization of the topic closeness of the documents that are being considered. In other words, thematic analysis helps to determine whether the analyzed documents have similar overall content. Our approach is based on LSA (Latent Semantic Analysis.) This method presents a technique that analyzes relationships

between a set of documents and the terms they contain by producing a set of new concepts related to the documents and terms. In order to do this, the algorithm assumes that words that are close in meaning occur in similar pieces of text [15].

Our approach is inspired by the work of [11], where the authors propose a plagiarism detection technique based on LSA. In simple terms, we first compute the tf-idf matrix for the documents, considering as terms all the n-grams of the documents that are going to be analyzed, with different values of n. We then obtain the singular value decomposition or SVD of each matrix. The SVD comprises a new representation of the feature space, where the underlying semantic relationship between terms and documents is revealed, reducing the dimensions of the term by document space [25]. Following what is proposed in [11], for each n-gram where $n > 2$, the terms within it are sorted alphabetically in order to increase the frequency of n-grams in different documents, combining similar concepts into one single dimension. Once we have applied the SVD to all the n-gram-document matrices, we then have a set of low dimensional document matrices M. Since columns from M are concept space representations of each analyzed document, similarity between document columns M_i, as conceived in the vector space model, can be computed using different measures. Thus, the last step computes the mutual pairwise document similarity and generates a symmetric matrix where each pair of documents is evaluated by a score representing the percentage similarity.

3.7. Internal Database Document Retriever

Our system stores all the documents that have been processed in the past and all the Web documents that have been downloaded by the Web retriever in a historic internal database. Given a source document, our database retriever algorithm, based on Apache Lucene[11], is able to collect a set of other similar documents. Lucene provides a ranking model based on the classic vector space model and the boolean model from Information Retrieval [12] which are here used by us to retrieve and select the most similar documents given a query.

4. Architecture

DOCODE is aimed to deliver very different services to a wide number of institutions and individuals. Bearing this idea in mind, functionalities are offered as Web services. Following the SaaS paradigm, applications are accessed by users using a thin client; usually, via Web browsers. In our case, DOCODE features are offered through application programming interfaces (APIs), which are based on two protocols, HTTP or SOAP (Simple Object Access Protocol.) In this manner, clients can consume the services provided by our system using their preferred platforms.

We have designed our system using Java Enterprise Edition or JEE. This platform provides an API and runtime environment for developing and running enterprise software[13], based on distributed and multi-tier architectures. Thus, we have defined an architecture based on multiple layers, bearing in mind the requirements of the platform and trying to conform to the standards and best implementation practices of JEE. Figure 1 shows the layers that we have defined. Below, we give more details about each layer.

- Client Layer: Where the user accesses the application and consumes its services. As we already said, our system requires only a Web service client or a Web browser. For the Web service client, each request is sent to our service with a given URL, using the SOAP protocol over HTTP. Our service receives the request, processes it, and returns a response. As of now, DOCODE is only working in the asynchronous mode. Therefore, after each request is received, the requirement is queued until the system is free to process it. After the answer is ready, it is sent or served to the user.

- Web Layer: Corresponds to the implementation of different Web interfaces that let our users consume the services provided by the system and interact with the provided results. Details on the existing interfaces will be given in Section 5.

[11] `http://lucene.apache.org/`
[12] `https://lucene.apache.org/core/3_6_2/scoring.html`
[13] `http://docs.oracle.com/javaee/6/firstcup/doc/gkhoy.html`

- Service Layer: This layer is in charge of making the Web services available to the external net. The first step was to define the objects that are part of the communication. In this case, as established under the JEE paradigm, XSD (XML Schema Definition) files are used to define the structures that are sent within the SOAP messages. We then generated the WSDL (Web Service Description Language) file, which includes all the structures defined in the XSD file as well as the methods that are available for execution. The next step was the implementation of the service itself. We chose to use Java API for XML Web Services (JAX-WS), since this API is included in the JEE plaftorm we selected, GlassFish Server Open Source Edition.

- Business Logic Layer: This layer saves the actions and processes that contain the business rules to DOCODE's correct operation. The business logic rules capture business requirements and analyze them through the different algorithms described in Section 3, finally preparing a display of the results. The business process model is designed to give a big picture of the complete business process [6]. In this way, it is possible to model the whole system using BPMN (Business Process Management Notation).

- Persistence Layer: This layer allows the administration of the relational data model as if it was an object model. To achieve the business objectives, the processes recover information by means of proprietary applications. Each persistence implementation has been developed on its own reference architecture, which provides a view of the defined process in the developed phase [6]. Thus, the persistence layer is implemented using an object-oriented database, based on Entity Beans. These Entity Beans are in charge of representing the relational model as a model of objects, using ad-hoc libraries.

- Data Layer: This layer is a critical component and possibly the main part of the solution [6]. Hence, it is extremely important to take into consideration issues regarding data redundancy, data reuse, access control and frequent backups. In this context, the data layer provides implementations that are completely independent from the rest of the development of the project. The layer is supported by the file system and a relational database. Both structures work together in order to allow the managing and processing of large sets of documents in parallel. Tables disposed in the database are mostly intended to save information about the registered users who are allowed to access DOCODE's services, about the jobs that have been executed and about the documents that have been processed.

We have already developed clients for some of the most important Virtual Learning Environments (VLE), offering integration for Moodle[14], Sakai[15] and also for U-Cursos[16], an academic platform used by more than 30 academic institutions in Chile. Since many different users will be consuming the services provided by DOCODE, security is an essential issue for our system. In the first place, we need to protect our algorithms from external access by avoiding exposing them directly to our users. On the other hand, since some information about our users including request logs or copies of the analyzed documents may be stored on our servers, that data need to be securely stored in order to allay privacy concerns [47]. Considering these issues and in order to structure our system in the most scalable possible manner, we have defined two main components: DOCODE CORE and DOCODE Shield.

The first component, named DOCODE Shield, is in charge of the reception and queue of all the requirements that our users make. In this manner, the Web services always communicate with DOCODE Shield's VPS (Virtual Private Server,) which then communicates the requirement to DOCODE CORE through a firewall. After the Web service gives a requirement, a servlet manages the upload and querying procedures to keep a good VPS performance and to prepare the input for the CORE.

DOCODE CORE is the main engine that implements all the algorithms supporting the functionalities provided. Each one of these algorithms was encapsulated as if it was part of a library. Then, we integrated

[14] `https://moodle.org/`
[15] `https://sakaiproject.org/`
[16] `https://www.u-cursos.cl/`

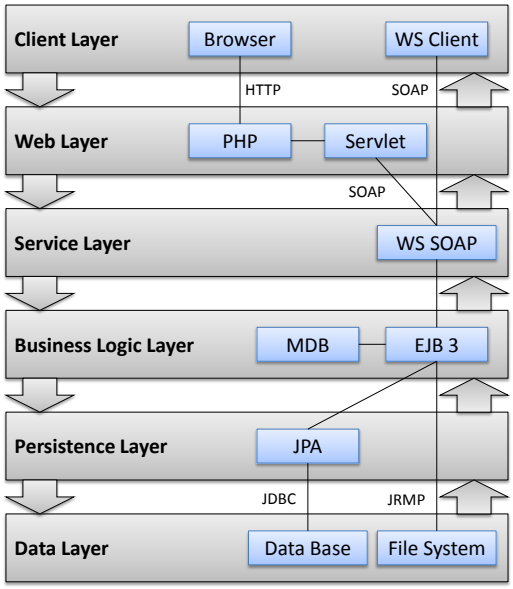

Figure 1: DOCODE system's layers diagram.

each encapsulated algorithm using EJB3 specifications. Whenever a request arrives at DOCODE CORE, an interface is in charge of calling the algorithms that are relevant for the request, according to what is defined in the MDB Topic queue. As Figure 2 shows, the interface is available in a synchronous mode, which directly invokes the EJB, and also in an asynchronous mode, in which a message is left in the algorithm internal queue for later execution. The EJB-based encapsulation paradigm lets the system run different algorithms in parallel, reducing the answer time. It also makes it easier to modify existing algorithms and to add more functionalities.

For the implementation, we took the best practices of the JEE standards and applied them to ensure high quality of service and scalability. Our service-based paradigm allows us to offer high availability and total transparency for the clients that consume the services. In this context, it is possible to place DOCODE in level 4 of the Service-oriented Architecture maturity model, according to OSSIM[17]. The Service-oriented Architecture or SOA maturity level specifies how to measure the service integration levels of an organization and its IT systems and business applications. In this context, level 4 also called service level, indicates that our system services may be invoked using standards and are independent of the underling application technology, also allowing to build new systems based upon these services.

5. User Interface

In this section, we show how DOCODE's user interfaces are designed. In general terms, the services provided by all the clients are the same. However, due to the specifications of each client, the human interface of these features may be a little different in each case. As we said in Section 3, we have already developed clients for Sakai, Moodle and U-Cursos. This means that any institution using these platforms could access all our services. However, since we do not want to limit the access only to these means, we have also developed DOCODE ASP. This software, created by us based on Moodle, is targeted toward those users that are interested in using our services but do not yet possess any of the supported clients. Since it

[17] `http://www.opengroup.org/standards/soa/`

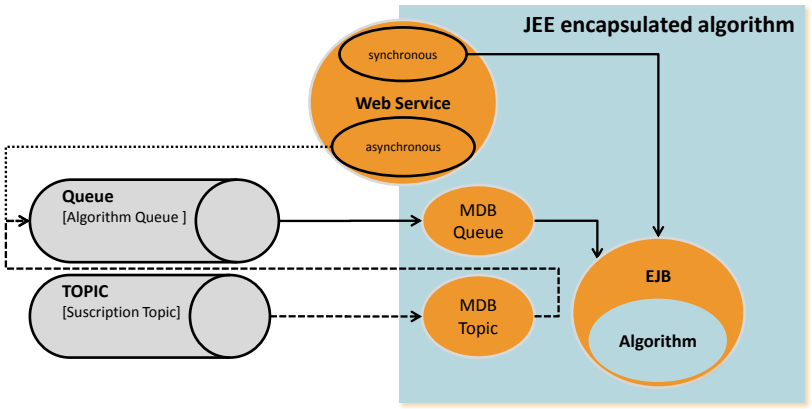

Figure 2: Structure of an encapsulated algorithm.

Document Name	Style Change	Course Similarity	Web Similarity	Options
[illegible].txt	0.0%	32.1%	11	Info
[illegible].txt	0.0%	51.6%	11	Info
[illegible]a.doc	0.0%	99.4%	11	Info
[illegible].docx	0.0%	45.0%	11	Info
[illegible].docx	0.0%	53.3%	11	Info
[illegible].txt	0.0%	57.8%	11	Info
[illegible].docx	0.0%	68.7%	11	Info
[illegible].docx	0.0%	0.0%	11	Info
[illegible].doc	0.0%	0.0%	11	Info
[illegible].docx	0.0%	34.7%	11	Info

Figure 3: Page showing the general results for one processed *corpus*.

is based on Moodle, DOCODE ASP is offered via the Web and therefore can be accessed using any Web browser[18]. Since we implement all the basic features of Moodle as a VLE, we also offer DOCODE ASP to single professors or to institutions that do not currently use any software for supporting their activities.

Before going further into introducing our user interface specifications, we will first define some concepts related to student plagiarism that will help in the explanations. These concepts will be concerned with the fact that most of the educational institutions are currently using VLEs or other learning platforms based on the Web that provide access to classes, homework, grades and so on.

Consider an average educational institution that actively uses a VLE software. It is possible to recognize certain similarities in the way students are organized and evaluated. Under our notation, a written assignment or task given to a student or group of students will be called *homework*. The file containing the answers submitted by one student will be called *submitted document* or simply document. On the other hand, the person (or group of persons) in charge of assigning a particular *homework* will be called *teacher*. Then, we define a *course* as a set of students that share one same *homework*. Finally, we define a *corpus* as a collection of submitted documents for one *homework*. From this it follows that in case the source and copy documents are both within a *corpus*, we will be facing an intra-corporal plagiarism or collusion case. Conversely, when the copy is inside the corpus and the source is outside of it (for instance in a textbook or

[18] `http://asp.docode.cl/`

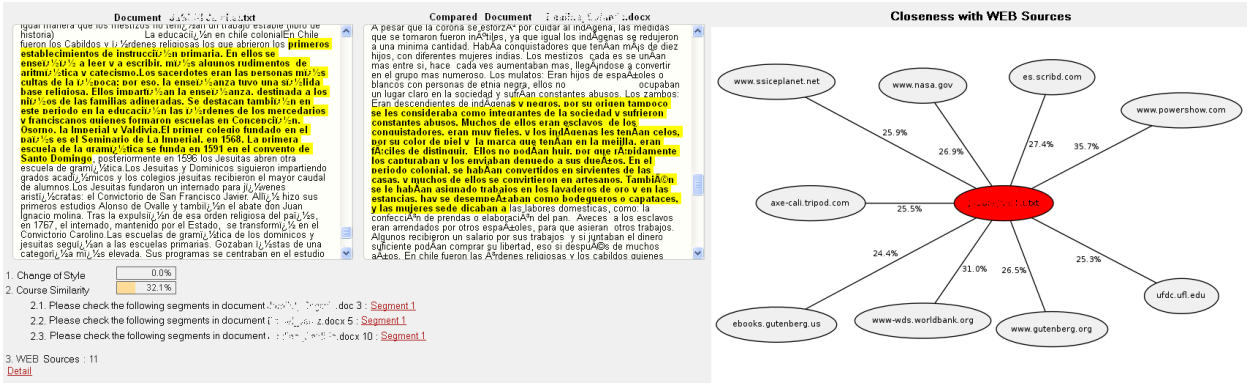

Figure 4: (Left) Interactive documents highlighting the suspicious passages in the source and original documents inside a *corpus*. (Right) Chart showing the summary of Web sources detected for a single document.

on the Web,) this is then a case of extra-corporal plagiarism [22].

DOCODE operates on the basis of *courses*, *homework* and the corresponding *corpora*, which may contain one or more documents. Results offered by DOCODE for a *corpus* can be grouped in two categories, which we proceed to explain below.

5.1. For single documents inside one corpus

Results for each document inside a *corpus* are first presented in a general screen, which is intended to show the big picture of all the submitted documents in the *corpus* (see Figure 3). By clicking on each row on the screen, users can access the detailed results in that document, which are displayed on a different page. The basic setting for this new page is an interactive on-line version of the document that will be used to show the suspicious passages of the text. Below, we explain the offered results for single documents as they appear in both the general and detailed pages.

- Style Change: In the general page, we present a measure that evaluates the relative amount of passages that seem to present style change in each document. In the page of detailed results, we offer a list with all the passages that present style deviations according to our algorithms. By clicking on each passage on the list, the system highlights the selected passage in the interactive document mentioned above. This feature is based on our change of writing style detector algorithm.

- Hidden-text alarm: On the general page, we show an indicator that lights up when we detect an unusual and possibly deceptive behavior in one document. Of course, this feature uses our hidden-text detector as a basis.

- Web Sources Identification: Using our similar Web document retriever algorithm, we collect possible Web sources related to each processed document and count them, offering the resulting number on the general page. On the detailed page for a particular document, we offer the list of the URLs of each detected Web source. By clicking each URL, the corresponding suspicious passages are highlighted in the interactive document. In addition, as shown in Figure 4, we offer a chart showing all the Web sources and the degree of similarity with the document.

- Course Similarity: On the general page, we show a measure of how close each document is to the rest of the documents in the *corpus*. On the detailed results page, we offer a list of all the documents in the corpus that may be part of an intra-corporal plagiarism case. For each document, we present the list of passages that are similar. Upon clicking, each passage will be highlighted in the interactive document. At the same time, the system will also highlight the similar passage in a new interactive display for the other document. In this manner, as shown in Figure 4, we offer a comparative view of

the documents involved in a possible intra-corporal plagiarism case. All of these results are provided by our cross-document copy-detector algorithm.

- Extracted Quotations: In the detailed page for a document, we offer a list with all the extracted quotations. With this feature, *teachers* can check if *students* are writing references in a proper manner. They also can check more specific facts. As an example that is worth mentioning, teachers can check if the given recommended bibliography for a *homework* was considered by students when writing it or not.

5.2. For all the documents inside the corpus together

Besides results for single documents, we also offer some results of processing all the documents inside a corpus together. The outputs of our analysis are different charts that are intended to summarize the general situation of the submitted documents. These charts, which we proceed to explain below, are all presented on the general page.

- Course Similarity: Using FASTDOCODE, we compare all the documents inside the *corpus* to each other and offer a chart to summarize these results, showing the degree of similarity between them (see Figure 5). The similarity degree is based on the number of passages that are similar, as a percentage of the total length of documents.

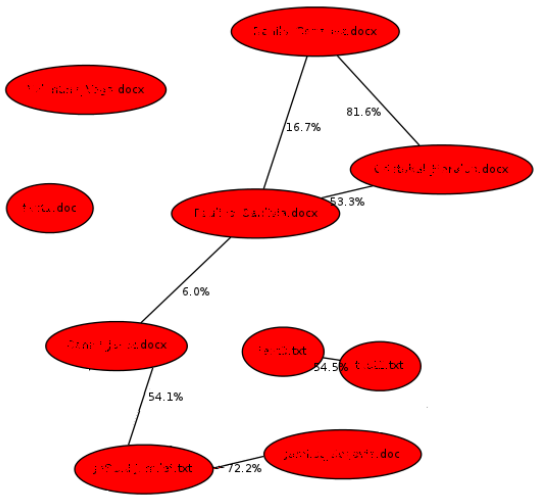

Figure 5: *Course* similarity chart showing the relationships among documents inside a *corpus*.

- Thematic analysis: This chart , based on the results of our multi-document thematic analyzer algorithm, is intended to show the similarity in topics between documents inside a corpus. In the chart, as shown in Figure 6, each document is drawn like a bubble. If two documents discuss a common topic, we draw a straight line between them. The more lines a document has in common with others, the higher number of topics they will have in common. Since it is a common practice to request students in a class to produce essays about the same topic, it is likely that all the documents in the *corpus* for one *homework* have many topics in common. However, since those documents presenting plagiarism cases share more pieces of text among them —either because the author has performed verbatim or copy paraphrasing, they might either present a higher number of topics in common or they may have special topics in common that other documents do no have. Therefore, by detecting the groups of documents that have any of these irregular behaviors, it is possible to discover possible plagiarism cases.

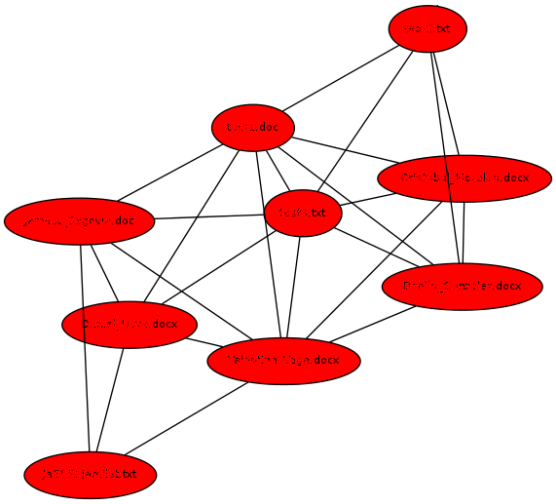

Figure 6: Chart of the thematic analysis showing relations between topics inside documents of a *corpus*.

- Web Sources Analysis: Based on the results of our similar Web document retriever of all the documents inside the *corpus*, we select the more frequent sources and generate a chart to visualize them. This figure is very similar to the chart that we offer for Web sources of a single document, so we are not including a picture in this case.

5.3. DOCODE Lite

Finally, we also offer DOCODE Lite, a free service where users can analyze a single document by comparing its content with different Web sources [21]. This special service is offered via the Web and can be accessed by any user upon registration. This software is intended mostly for users interested in trying some of the functionalities provided by DOCODE.

The design of DOCODE Lite follows the same ideas that we have already discussed, with the difference that it does not need any VLE client or DOCODE ASP to operate. Basically, after creating a free account, users can upload a file less than 3 MB size and send it using DOCODE Lite's Web platform. After the file is processed and the results are ready, the user receives an e-mail with the results of the analysis based on Web sources, showing all the detected sources and the corresponding suspicious passages. For each result, we also provide a score measuring the probability of being a real plagiarism case.

6. Conclusions

In this paper, we have introduced DOCODE 3.0, a plagiarism detection engine that is intended to offer professors and educators a set of tools to gain insights about students' work by information fusion from multi-document data sources, making it easier to analyze a vast number of documents. We have shown that DOCODE is a full-featured system that implements state-of-the-art algorithms for external and intrinsic plagiarism detection that also has a Web-based user interface to display the results of the analysis in a simple and intuitive manner. Since we wanted to offer DOCODE to virtually any person or educational institution, we offer our system as a service and hence have also created an ad-hoc architecture to support it.

According to [22], in order to describe plagiarism detection engines, it is necessary to catalogue the types of academic misconduct that they are intended to detect. In this sense, a review of our implemented algorithms shows that DOCODE 3.0 covers all the levels of plagiarism; DOCODE is intended to detect all the existing categories of such academic misconduct in educational institutions. In particular, taking the proposals of [28] and [20], we see that FASTDOCODE is capable of detecting verbatim copying (or

copy-paste), while our quotation detector for Spanish helps to discover the deliberate and/or inaccurate use of references. We also have implemented a technique for detecting one of the most common technical tricks to exploit weaknesses of plagiarism detector engines, namely, replacing spaces for white-colored characters. Moreover, we use our LSA-based algorithm for thematic analysis, whose effectiveness for the problem of plagiarism was already demonstrated by the SAIF system [33], [9].

In the same context, [20] proposes that an effective automatic plagiarism detector has to be able to identify the plagiarism involved in copying from another student's work and also the plagiarism involved in copying without acknowledgment from on-line reference materials. Again, it is easy to see that our system covers these cases. Indeed, we have already seen that our metrics and results explicitly embrace the problems of intra-corporal plagiarism, using the documents inside a *corpus*, and also extra-corporal plagiarism, based on the results of our similar Web document retriever algorithm. In this sense, DOCODE can be then categorized as a hermetic and external plagiarism detection system.

As we have seen so far, none of the proposed algorithms in this article addresses the problem of paraphrase detection in an explicit fashion. However, considering that paraphrasing can take multiple forms e.g., sentence rearrangement, or synonym replacement, its detection is implicitly performed by DOCODE across some of its modules. For instance, as FASTDOCODE relies on word n-grams as a comparison criterion, it can successfully detect reorders of sentences or phrases across documents. In a similar way, paraphrased Web sources from which words or sentences were reordered could be identified by the query generation mechanisms of the Web document retriever. On the one hand, the detection of word reorders is addressed by the randomized extraction of words provided by HLM. Conversely, the detection of phrase reorders is tackled by the random n-gram sample fingerprint method. Finally, considering that the LSA-based thematic analyzer module can detect the latent semantic associations between document pairs [11], it could potentially detect paraphrased documents in which similar words were replaced, such as synonyms. More complex forms of paraphrase detection, including lexical substitution or negation switching are still difficult to detect, even with the use of state-of-the-art plagiarism detectors, and therefore are not currently implemented in DOCODE. For further information regarding the paraphrase mechanisms behind plagiarism we refer the reader to [5].

In conclusion, DOCODE is a full-featured tool that is designed to help teachers and professors tackle the complex problem of plagiarism in educational institutions. Having obtained the first place in the intrinsic and external plagiarism detection tasks in the PAN2011 competition and having been reviewed by the scientific community in several past publications, our algorithms have proven to be very effective at tackling plagiarism detection from different perspectives, sometimes even outperforming existing state-of-the-art approaches. By the means of a detailed user interface based on the results of these consolidated algorithms, our system is able to support educator decision-making processes when faced with possible plagiarism cases among their students. In this sense, our ultimate goal is to encourage learning and improve the quality of education, taking the Chilean case as a starting point.

For future work, we plan to perform a series of qualitative analyses of our platform based on feedback from our users. We are already working on a set of new visualization tools based on RIA (Rich Internet Application). However, we first want to know which of the current features the users liked or disliked most so far in order to make our new interface better. At the same time, we intend to evaluate the efficiency of DOCODE in order to prepare our physical systems for larger scales. We also plan to modify our algorithms in order to make them work for multi-language purposes. Finally, as our internal database continues to grow, we also intend to design better algorithms for internal similar document retrieval and fully integrate this algorithm into the user interfaces, a task that is still in progress.

Acknowledgements

The authors would like to acknowledge the continuous support of the Chilean Millennium Institute of Complex Engineering Systems (ICM: P-05-004-F, CONICYT: FBO16); the INNOVA CORFO project (13IDL4-24315) entitled, DOCODE: DOcument COpy DEtector (`www.docode.cl`).

References

[1] Ashworth P., Bannister, P., Thorne, P. & Students on the Qualitative Research Methods Course Unit (1997). Guilty in whose eyes? University students' perceptions of cheating and plagiarism in academic work and assessment. *Studies in Higher Education*, *22*, 187–203.

[2] Barrón-Cedeño, A., Gupta, P., & Rosso, P. (2013). Methods for cross-language plagiarism detection. *Knowledge-Based Systems*, *50*, 211 – 217.

[3] Barrón-Cedeño, A., Potthast, M., Rosso, P., Stein, B., & Eiselt, A. (2010). Corpus and evaluation measures for automatic plagiarism detection. In N. C. C. Chair), K. Choukri, B. Maegaard, J. Mariani, J. Odijk, S. Piperidis, M. Rosner, & D. Tapias (Eds.), *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*. Valletta, Malta: European Language Resources Association (ELRA).

[4] Barrón-Cedeño, A., Rosso, P., & Benedí, J.-M. (2009). Reducing the plagiarism detection search space on the basis of the kullback-leibler distance. In A. Gelbukh (Ed.), *Computational Linguistics and Intelligent Text Processing* (pp. 523–534). Springer Berlin Heidelberg volume 5449 of *Lecture Notes in Computer Science*.

[5] Barrón-Cedeño, A., Vila, M., Martí, M. A., & Rosso, P. (2013). Plagiarism meets paraphrasing: Insights for the next generation in automatic plagiarism detection. *Computational Linguistics*, *39*, 917–947.

[6] Binildas, C. A., Barai, M., & Caselli, V. (2008). *Service Oriented Architecture with Java*. Packt Publishing.

[7] Bravo-Marquez, F., L'Huillier, G., Ríos, S., & Velásquez, J. D. (2011). A text similarity meta-search engine based on document fingerprints and search results records. In *Web Intelligence and Intelligent Agent Technology (WI-IAT), 2011 IEEE/WIC/ACM International Conference on* (pp. 146–153). IEEE Computer Society volume 1.

[8] Bravo-Marquez, F., L'Huillier, G., Ríos, S. A., & Velásquez, J. D. (2010). Hypergeometric language model and zipf-like scoring function for web document similarity retrieval. In E. Chavez, & S. Lonardi (Eds.), *String Processing and Information Retrieval* (pp. 303–308). Springer Berlin Heidelberg volume 6393 of *Lecture Notes in Computer Science*.

[9] Britt, M. A., Wiemer-Hastings, P., Larson, A. A., & Perfetti, C. A. (2004). Using intelligent feedback to improve sourcing and integration in students' essays. *International Journal of Artificial Intelligence in Education*, *14*, 359–374.

[10] Bull, J., Collins, C., Coughlin, E., Sharp, D., & Square, P. (2001). Technical review of plagiarism detection software report. University of Luton. JISC publication.

[11] Ceska, Z. (2008). Plagiarism detection based on singular value decomposition. In B. Nordström, & A. Ranta (Eds.), *Advances in Natural Language Processing* (pp. 108–119). Springer Berlin Heidelberg volume 5221 of *Lecture Notes in Computer Science*.

[12] Clough, P. (2000). Plagiarism in natural and programming languages: an overview of current tools and technologies. In *Research Memoranda: CS-00-05, Department of Computer Science. University of Sheffield, UK* (pp. 1–31).

[13] Clough, P., & Stevenson, M. (2011). Developing a corpus of plagiarised short answers. *Language Resources and Evaluation*, *45*, 5–24.

[14] Clough, P. & Department Of Information Studies (2003). Old and new challenges in automatic plagiarism detection. In *National Plagiarism Advisory Service, 2003* (pp. 391–407).

[15] Dumais, S. T. (2004). Latent semantic analysis. *Annual review of information science and technology*, *38*, 188–230.

[16] Fialkoff, F. (1993). There's no excuse for plagiarism. *Library Journal*, *118*, 56.

[17] Flores, E., Barrón-Cedeño, A., Moreno, L., & Rosso, P. (2015). Uncovering source code reuse in large-scale academic environments. *Computer Applications in Engineering Education*, *23*, 383–390.

[18] Hanks, P., Wilkes, G., Urdang, L., & McLeod, W. (1986). *Collins Dictionary of the English Language: An Extensive Coverage of Contemporary International and Australian English*. Collins.

[19] Harkness, W. L. (1965). Properties of the extended hypergeometric distribution. *The Annals of Mathematical Statistics*, *36*, 938–945.

[20] Kakkonen, T., & Mozgovoy, M. (2010). Hermetic and web plagiarism detection systems for student essays an evaluation of the state-of-the-art. *Journal of Educational Computing Research*, *42*, 135–159.

[21] Kudelka, M., Snasel, V., Horak, Z., Ella Hassanien, A., Abraham, A., & Velásquez, J. D. (2014). A novel approach for comparing web sites by using microgenres. *Engineering Applications of Artificial Intelligence*, *35*, 187–198.

[22] Lancaster, T., & Culwin, F. (2005). Classifications of plagiarism detection engines. [Online; accessed 09-Nov-2014].

[23] Lukashenko, R., Graudina, V., & Grundspenkis, J. (2007). Computer-based plagiarism detection methods and tools: An overview. In *Proceedings of the 2007 International Conference on Computer Systems and Technologies* CompSysTech '07 (pp. 40:1–40:6). New York, NY, USA: ACM.

[24] Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to information retrieval* volume 1. Cambridge University Press Cambridge.

[25] Marrese-Taylor, E., & Velásquez, J. D. (2014). Tools for external plagiarism detection in DOCODE. In *Web Intelligence (WI) and Intelligent Agent Technologies (IAT), 2014 IEEE/WIC/ACM International Joint Conferences on* (pp. 296–303). "IEEE Computer Society" volume 2.

[26] Martin, B. (1994). Plagiarism: a misplaced emphasis. *Journal of Information Ethics*, *3*, 36–47.

[27] Maurer, H., & Kulathuramaiyer, N. (2007). Coping with the copy-paste-syndrome. In *World Conference on E-Learning in Corporate, Government, Healthcare, and Higher Education* (pp. 1071–1079).

[28] Maurer, H. A., Kappe, F., & Zaka, B. (2006). Plagiarism - A Survey. *Journal of Universal Computer Science (J.USC)*, *12*, 1050–1084.

[29] McCabe, D. (2005). Cheating: Why students do it and how we can help them stop. *American Educator*, *25*, 38–43.

[30] Meyer zu Eissen, Sven and Stein, Benno (2006). Intrinsic plagiarism detection. In M. Lalmas, A. MacFarlane, S. Rüger,

A. Tombros, T. Tsikrika, & A. Yavlinsky (Eds.), *Advances in Information Retrieval* (pp. 565–569). Springer Berlin Heidelberg volume 3936 of *Lecture Notes in Computer Science*.

[31] Molina, F., & Velásquez, J. D. (2011). The digital document plagiarism phenomenon: An analysis of the current citation in the chilean educational system. *Revista Ingeniería de Sistemas*, *25*, 5–28.

[32] Moya, P. (2013). *Plagiarism and integration of multiple sources in first year college students' texts: Approach from the theory of understanding and latent semantic analysis*. Master's thesis University of Chile.

[33] Mozgovoy, M., Kakkonen, T., & Cosma, G. (2010). Automatic student plagiarism detection: future perspectives. *Journal of Educational Computing Research*, *43*, 511–531.

[34] Oberreuter, G., L'Huillier, G., Ríos, S. A., & Velásquez, J. D. (2010). FASTDOCODE: Finding approximated segments of N-Grams for document copy detection - Lab Report for PAN at CLEF 2010. In *Working Notes Papers of the CLEF 2010 Evaluation Labs*. Padua, Italy.

[35] Oberreuter, G., L'Huillier, G., Ríos, S. A., & Velásquez, J. D. (2011). Approaches for intrinsic and external plagiarism detection - notebook for PAN at CLEF 2011. In *Working Notes Papers of the CLEF 2011 Evaluation Labs*. Amsterdam, The Netherlands.

[36] Oberreuter, G., & Velásquez, J. D. (2013). Text mining applied to plagiarism detection: The use of words for detecting deviations in the writing style. *Expert Systems with Applications*, *40*, 3756–3763.

[37] Pereira, Á., & Ziviani, N. (2003). Retrieving similar documents from the web. *Journal of Web Engineering*, *2*, 247–261.

[38] Posner, R. A. (2009). *The little book of plagiarism*. Random House Digital, Inc.

[39] Potthast, M., Barrón-Cedeño, A., Eiselt, A., Stein, B., & Rosso, P. (2010). Overview of the 2nd international competition on plagiarism detection. In M. Braschler, D. Harman, & E. Pianta (Eds.), *Working Notes Papers of the CLEF 2010 Evaluation Labs*. Padua, Italy.

[40] Potthast, M., Eiselt, A., Barrón-Cedeño, A., Stein, B., & Rosso, P. (2011). Overview of the 3rd international competition on plagiarism detection. In V. Petras, P. Forner, & P. Clough (Eds.), *Working Notes Papers of the CLEF 2011 Evaluation Labs*. Amsterdam, The Netherlands.

[41] Potthast, M., Gollub, T., Hagen, M., Grassegger, J., Kiesel, J., Michel, M., Oberländer, A., Tippmann, M., Barrón-Cedeño, A., Gupta, P., Rosso, P., & Stein, B. (2012). Overview of the 4th international competition on plagiarism detection. In P. Forner, J. Karlgren, & C. Womser-Hacker (Eds.), *Working Notes Papers of the CLEF 2012 Evaluation Labs*. Rome, Italy.

[42] Potthast, M., Stein, B., Barrón-Cedeño, A., & Rosso, P. (2010). An evaluation framework for plagiarism detection. In C.-R. Huang, & D. Jurafsky (Eds.), *Proceedings of the 23rd International Conference on Computational Linguistics (COLING 10): Posters* (pp. 997–1005). Association for Computational Linguistics Stroudsburg, Pennsylvania: Association for Computational Linguistics.

[43] Potthast, M., Stein, B., Eiselt, A., Barrón-Cedeño, A., & Rosso, P. (2009). Overview of the 1st international competition on plagiarism detection. In B. Stein, P. Rosso, E. Stamatatos, M. Koppel, & E. Agirre (Eds.), *Proceedings of The 25th edition of the Annual Conference of the Spanish Society for Natural Language Processing (SEPLN 09), Workshop on Uncovering Plagiarism, Authorship, and Social Software Misuse (PAN 09)* (pp. 1–9). CEUR-WS.org.

[44] Scanlon, P. M., & Neumann, D. R. (2002). Internet plagiarism among college students. *Journal of College Student Development*, *43*, 374–385.

[45] Stein, B., Meyer zu Eissen, S., & Potthast, M. (2007). Strategies for retrieving plagiarized documents. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* SIGIR '07 (pp. 825–826). New York, NY, USA: ACM.

[46] University of Chicago Press (1982). *The Chicago manual of style*. University of Chicago Press.

[47] Velásquez, J. D. (2013). Web mining and privacy concerns: Some important legal issues to be consider before applying any data and information extraction technique in Web-based environments. *Expert Systems with Applications*, *40*, 5228–5239.

[48] Velásquez, J. D., & Jain, L. C. (2010). *Advanced Techniques in Web Intelligence - 1*. Studies in Computational Intelligence. Springer.

[49] Velásquez, J. D., Palade, V., & Jain, L. C. (2013). *Advanced Techniques in Web Intelligence - 2*. Studies in Computational Intelligence. Springer.