Using Structural Information and Citation Evidence to Detect Significant Plagiarism Cases in Scientific Publications

Salha Alzahrani
Department of Computer Science, Taif University, Taif, Saudi Arabia. E-mail: s.zahrani@tu.edu.sa

Vasile Palade
Department of Computer Science, University of Oxford, Oxford, UK. E-mail: vasile.palade@cs.ox.ac.uk

Naomie Salim
Faculty of Computer Science and Information Systems, Universiti Teknologi Malaysia, Johor Bahru, Johor, Malaysia. E-mail: naomie@utm.my

Ajith Abraham
VSB Technical University of Ostrava, CZ. E-mail: ajith.abraham@ieee.org

In plagiarism detection (PD) systems, two important problems should be considered: the problem of retrieving candidate documents that are globally similar to a document q under investigation, and the problem of side-by-side comparison of q and its candidates to pinpoint plagiarized fragments in detail. In this article, the authors investigate the usage of *structural* information of scientific publications in both problems, and the consideration of *citation evidence* in the second problem. Three statistical measures namely Inverse Generic Class Frequency, Spread, and Depth are introduced to assign a degree of importance (i.e., weight) to structural components in scientific articles. A term-weighting scheme is adjusted to incorporate component-weight factors, which is used to improve the retrieval of potential sources of plagiarism. A plagiarism screening process is applied based on a measure of resemblance, in which component-weight factors are exploited to ignore less or nonsignificant plagiarism cases. Using the notion of citation evidence, parts with proper citation evidence are excluded, and remaining cases are suspected and used to calculate the similarity index. The authors compare their approach to two flat-based baselines, TF-IDF weighting with a Cosine coefficient, and shingling with a Jaccard coefficient. In both baselines, they use different comparison units with overlapping measures for plagiarism screening. They conducted extensive experiments using a dataset of 15,412 documents divided into 8,657 source publications and 6,755 suspicious queries, which included 18,147 plagiarism cases inserted automatically. Component-weight factors are assessed using precision, recall, and *F*-measure averaged over a 10-fold cross-validation and compared using the ANOVA statistical test. Results from structural-based candidate retrieval and plagiarism detection are evaluated statistically against the flat baselines using paired-*t* tests on 10-fold cross-validation runs, which demonstrate the efficacy achieved by the proposed framework. An empirical study on the system's response shows that structural information, unlike existing plagiarism detectors, helps to flag significant plagiarism cases, improve the similarity index, and provide human-like plagiarism screening results.

Introduction

Three types of textual documents have been widely used in document retrieval (DR). The first type is structured documents, which have structural markers defined by a markup language, such as HTML and XML, and can be segmented into blocks. For example, web documents usually have markers, or tags, such as <p>, <br>, <hr>, <h1>, <h2>, etc., that can be used to partition the page into headers, menus, navigation bars, paragraphs, tables, and lists. The second type is semistructured documents, such as scientific artciles, which do not have structural markers, but different units of the document are presented using physical properties, such as the font type or size. The reader of a semistructured document can easily, by looking to its visual appearance, recognize different components, such as titles, sections, subsections,

paragraphs, tables and lists. The third type is free-format documents, which have plain texts without any structural markers and are unlikely to be presented by visual elements.

Structural information in scientific publications, in particular, plays an important role in presenting the content as segments; each with a specific importance (or interest) to the reader. Scientific publications begin usually with a title, authors, abstract, and keywords, and spans into sections. Each section begins with a head title and has a textual body that could be subsections or different elements such as paragraphs, lists, tables, figures, equations, and quotes. These segments of scholarly documents, commonly referred to in the literature as logical structure, can be extracted (Anjewierden, 2001; Bounhas & Slimani, 2010; Burget, 2007; Councill, Giles, & Kan, 2008; Hagen, Harald, Ngen, & Petra Saskia, 2004; K.H. Lee, Choy, & Cho, 2003; Li & Ng, 2004; Luong, Nguyen, & Kan, 2010; Nguyen & Luong, 2010; Ratté, Njomgue, & Ménard, 2007; Stoffel, Spretke, Kinnemann, & Keim, 2010; Wang, Jin, Wang, Wang, & Gao, 2005; Witt et al., 2010; K. Zhang, Wu, & Li, 2006), and can be used to improve document indexing (Bounhas & Slimani, 2010), to represent the semantic content of scientific publications (Luong, Nguyen, & Kan, 2010; Ratté et al., 2007), to extract key phrases and terminologies (Bounhas & Slimani, 2010; Nguyen & Luong, 2010), and to improve document summarization (Teufel & Moens, 2002). To improve indexing of semistructured documents, for instance, a method of terms weighting is applied according to their structural occurrences (or their positions in different segments of the document), instead of using the whole document as in *flat* weighting methods (Bounhas & Slimani, 2010; de Moura, Fernandes, Ribeiro-Neto, da Silva, & Gonçalves, 2010).

Besides structural information, citation evidence in scientific publications should acknowledge previous research work and avoid plagiarism. Citation evidence usually involves three main items: (a) quoting or citing a piece of text that is taken from a previous publication, (b) in-text citation marker using a numerical-numbering style or author-naming style that links that piece of text to one of the references, and (c) a list of references that contains several citation phrases presented normally at the end of the document. Each citation phrase starts in a new line and states the author names, publication title, year, and other information that guides the readers to locate that specific reference. The list of citation phrases can be parsed (Chen, Yang, Chen, & Ho, 2010; Councill et al., 2008), and can be used to search the web for digital resources such as CiteSeer[1] and ParaCit[2], to study the research evolution and trends in particular areas (M. Lee & Chen, 2010), to mine citation information for useful information such as influential scientists in particular areas (Fiala, 2010), and other applications in digital libraries (Chen, 1999; Chudamani & Ilamathi Maran, 2004; Larsen & Ingwersen, 2006; Pentz, 2006; Van & Beigbeder, 2007).

Digital libraries, publishers, and conference management systems have recently employed automatic antiplagiarism tools to ensure academic integrity of published contents. For example, Docoloc[3] is a plagiarism detector integrated into the EDAS conference management system (EDAS Conference Services, Leonia, NJ), and CrossCheck[4] is a tool used by more than 250 publishers including Cambridge University Press, Springer, and Elsevier. The technology behind plagiarism detectors generally works as follows (Butakov & Scherbinin, 2009; Karl, 2008): (a) A collection of scientific publications are compiled into a database, (b) a submitted document is compared with source documents in the database, (c) portions of the submitted document that have a high similarity score are highlighted, and (d) an overall similarity index is calculated in accordance with the percentage of text detected as similar. However, existing tools have some limitations. First, they focus on representing the documents as a "bag-of-words," which may lead to highlighting unnecessary parts as plagiarism.

Figure 1 shows that Docoloc highlights authors' names, affiliations, acknowledgments, bibliographies, and other small matches, although to an investigator this is unlikely to be plagiarism. In such cases, the bag-of-words representation may increase the ratio of plagiarism, or the similarity index, in scientific publications. Second, current tools need further adjustments to fine-tune the detection results and exclude texts with proper citation evidence (Meddings, 2010). Such texts are insignificant to the detection, unless two identical citations are found in two different papers (H. Zhang, 2010). Third, available tools mostly deal with documents as a whole and do not take into account structural information presented by the publication. Structural representation, however, is considered appropriate to represent the semantic ideas or topically related information in the article, and may yield significant improvements in plagiarism detection results.

Thus, this work exploits the logical structural information in scientific publications for better analysis of the text. Structural representation is a good metric to detect nontrivial (or significant) plagiarism cases. Our hypothesis is as follows: the distribution of terms in structural components throughout a scientific publication can indicate the significance of these components, and two publications having global similarity as a whole may be completely different in terms of the semantics and context by looking to their discriminative structural orientation. Subsequently, we propose that the occurrence of plagiarism in certain sections of a scientific article, such as the Methods, Results, or Discussion, is expected to be more significant for the detection algorithm than the occurrence of matching texts in sections such as the Introduction, Acknowledgments, or copyright notation. Using such an assumption can assist in featuring and tracing significant plagiarism cases, such as plagiarizing ideas of other authors.

---

[1]http://citeseer.ist.psu.edu/
[2]http://paracite.eprints.org/

[3]http://www.docoloc.de/
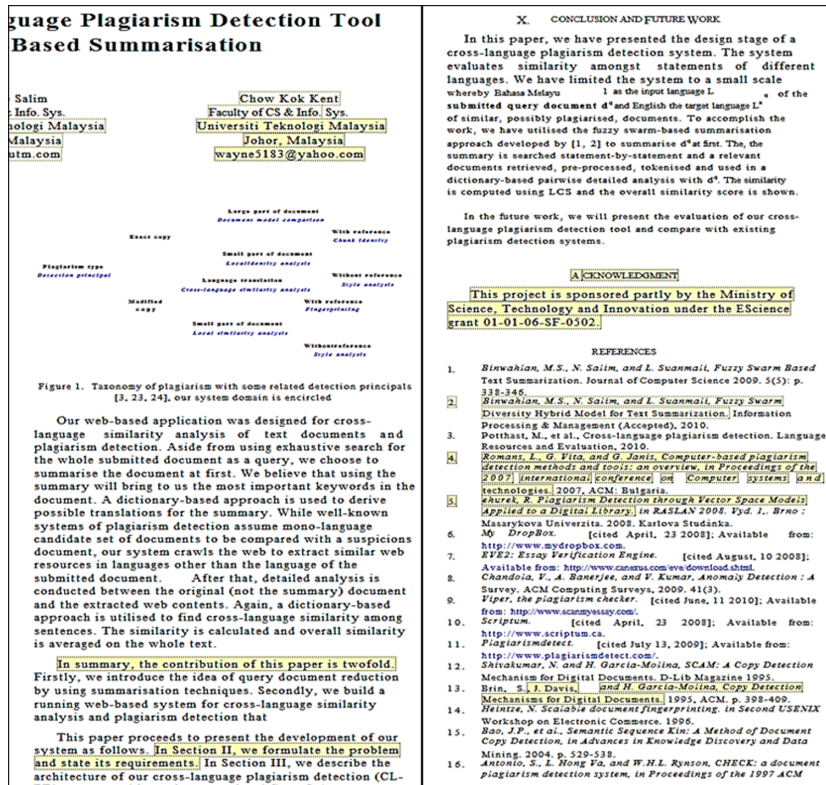[4]http://www.crossref.org/crosscheck.html

FIG. 1.   Sample of plagiarism detection results using a Docoloc tool.

We also presume that the use of citation evidence should considerably improve plagiarism analysis and comprehension of detection results when compiled in a similarity index, as will be discussed later in this article. Therefore, our proposed approach aims to achieve a close resemblance to an individual's way of analyzing, reasoning, and suspecting plagiarism in scientific works.

The rest of this article is organized as follows. In the second section, we provide a literature review in related areas including logical structure extraction (LSE), citation parsing (CP), text type structure (TTS), document retrieval (DR), candidate retrieval (CR), and plagiarism detection (PD). In the third section, we discuss the approach we used for the segmentation of scientific publications including LSE, TTS, and CP. Then we explore different component-based weighting functions of structural components in scientific publications in the fourth section, and present the method we applied for both of CR and PD in the fifth section. In the sixth section we present our experimental results and compare them with two baselines proposed in the literature. Finally, we present our conclusions and suggestions for future work.

Related Work

Structure Extractors, Generic Classifiers, and Citation Parsers

Methods for partitioning scholarly publications consider that the structure is usually presented physically by various visual elements, such as location, position, punctuation, length, and font type/size. Some partitioning methods employ keyword-based strategies to label specific content; for example, using words like "Chapter," "Introduction," I, II, and other numbering styles, to extract section headers. Partitioning scholarly documents comes under a wide research problem know as *logical structure extraction* (LSE) of semistructured documents, and is not the focus of this work. Fortunately, there are efficient LSE solutions addressed in recent literature (Burget, 2007; Luong et al., 2010; Ratté et al., 2007; Stoffel et al., 2010). In this work, we employ Luong's LSE (Luong et al., 2010) developed by the National University of Singapore (NUS), and available for free use or adaptation within other tools under the Lesser GNU Public License (LGPL).[5]

Scholarly documents tend to have a consistent logical structure. Thus, to unify the structure organization, different studies have addressed a so-called text-type structure (TTS), or generic classes in scientific works (Hagen et al., 2004; Siddharthan & Teufel, 2007), also known as *zones* after Teufel (1999). These studies aimed to develop methods (or annotation schemes) that can be used to group (or generalize) different sections in scientific publications under different classes/types. Teufel and Moens (2002) defined seven types of text, or argumentative zones, namely Own, Other, Background, Textual, Aim, Basis, and Contrast, according to the rhetorical status. Hagen et al. (2004) increased the text types to 16 types according to the topics and the problems, namely

---

Research Topic, Background, Others' Work, Rational, Textual, Theory, Concepts, Framework, Method, Data, Data Collection, Data Analysis, Results, Interpretation, Conclusions, and Resource. Luong et al. (2010) developed a generic classifier (GC) able to generalize extracted sections from scientific papers under seven classes according to the problem-solving hierarchy, namely, Abstract, Introduction, Related Work, Method, Evaluation, Conclusions, Acknowledgments, and References. Using a dictionary-based and machine-learning technique, Luong's classifier was able to relate different sections to their generic classes with encouraging results.

Citation parsers (CP), on the other hand, help to identify the intellectual ownership in scientific papers (Teufel & Moens, 2000). Because scientific citation has many styles such as the American Psychiatric Association (APA), Medical Library Association (MLA), Institute of Electrical and Electronics Engineers (IEEE) and others[6], different studies have suggested solutions for parsing different citation styles (Chen et al., 2010; Councill et al., 2008). In this work, we utilize ParsCit; a CP tool developed by Pennsylvania State University and National University of Singapore (Councill et al., 2008). It has the ability to extract two citation contexts: numbering-like styles (e.g. [2,5]) and naming-like styles (e.g., Councill et al., 2008, Councill, Gils & Kan, 2008, Councill, Gils & Kan, 2008). ParsCit is open source and available for free use under the LGPL.

Document Retrieval (DR) and Candidate Retrieval (CR)

Most DR models represent documents as a bag-of-words, wherein redundant and frequent words are excluded. Fingerprinting-based (also called shingling) approaches have been used widely for DR and near duplicate detection in digital libraries (Heintze, 1996; Schleimer, Wilkerson, & Aiken, 2003). In this model, documents are represented as character/word n-grams, and a measure of resemblance, e.g., the Jaccard coefficient, is used to find documents that share considerable n-grams. The vector space model (VSM) is a very popular one (Manning, Raghavan, & Schütze, 2009) that represents documents as vectors of unique and nonfrequent terms called *index terms*, and rank documents based on term frequency and inverse document frequency (IDF-TF) weighting scheme. Latent semantic indexing (LSI) is another weighting scheme based on the reduction of the original VSM (i.e., TF-IDF weighting vectors) using singular value decomposition (SVD; Chris, 1999). These models represent documents as *flat*, and do not take into consideration the *structural* representation of the documents, or the assumption that the occurrence of a term in a specific segment of the document may change its weight. Nonetheless, studies that consider structural representation of web documents for term weighting have yielded comparable or superior results in retrieval and ranking of search results (Bounhas & Slimani, 2010; de Moura et al., 2010; Marques Pereira, Molinari, & Pasi, 2005), in comparison to the normal weighting schemes.

We next review of several studies that have incorporated structural information into term weighting and document ranking.

Pasi (2002) introduced the concept of flexible information retrieval (FIR), also called soft IR, which model vagueness and uncertainty in web search and DR. Methods to represent FIR include the following:

1. Flexible user queries that use fuzzy quantifiers within the query. For example, the user can express the query using words like "at least," "very important," "important" and "alike," in contrast to Boolean queries that allows only "AND," "OR," and "NOT" notions.
2. Flexible indexing that encompasses the user's understanding of a document for terms weighting and document indexing
3. Fuzzy IR, which incorporates thesauruses for retrieval of documents that share global semantic similarity, rather than retrieval of documents that have words in common as in Boolean IR
4. Partial relevance and user feedback for adaptive retrieval

As a practical application of flexible indexing, Marques Pereira et al. (2005) used structural HTML pages to improve indexing and DR. The HTML pages are organized into blocks using tags such as $<p>$ for new paragraphs, $<h1>,<h2>$, $<h3>$ for headers, $<hr>$, $
$, $<div>$ for horizontal splitters, and so on. Then, a numerical weight was assigned to each block manually before the indexing phase to indicate its importance in the page. The block's weight was expresses as $bw_i = (n - i + 1)/n$ where i is the block's rank, and n is total block classes. Table 1 shows the empirical weights used in their work, wherein ranks (f) are successive, and weights (bw_i) are assigned values between 0, which means the least important, and 1 which means the most important. Term weighting was computed thereafter using the formula $F(d, t) = (\sum_{i=1}^{n} bw_i \times TF_i) \times IDF_t$, which indicates the degree of significance of the term by cumulating both term frequencies in different blocks of a web page and block weights.

Another study (Marteau, Ménier, & Popovici, 2006) introduced a naïve Bayes model for supervised classification of semistructured documents. Their work exploited structural knowledge from XML trees derived from a set of XML documents. Each element in the tree was weighted according to its contribution to the whole tree, and a weight heuristic was combined with the classification decision as follows:

$$bw_i = \begin{cases} \frac{|V_{i,\omega}|}{|d_i|} & if & d_i(innertext) \neq null \\ 1 & otherwise \end{cases}$$

where $|V_{i,\omega}|$ is the cardinal of the vocabulary associated to node i for the category ω, and $|d_i|$ is the size of inner text of the node i inside document d.

Bounhas and Slimani (2010) represented structural information from web documents as a tree whereby the root is assigned the highest level M, and leaf nodes are assigned

---

TABLE 1. Ranking of different blocks found in HTML pages (Marques Pereira et al., 2005).

f	bw_i	Block class	HTML Tags or parameters	f	bw_i	Block class	HTML Tags or parameters
1	1.00	Title	TITLE, META keywords	7	0.50	Lists	UL, OL, DL, MENUE, DIR
2	0.92	Header 1	H1, FONT SIZE = 7	8	0.42	Emphasized 2	BLOCKQOUTE, CITE, BIG, PRE, CENTER, TH, TT
3	0.83	Header 2	H2, FONT SIZE = 6	9	0.33	Header 4	H4, CAPTION, CENTER, FONT SIZE = 4
4	0.75	Header 3	H3, FONT SIZE = 5	10	0.25	Header 5	H5, FONT SIZE = 3
5	0.67	Linking	A HREF	11	0.17	Header 6	H6, FONT SIZE = 2
6	0.58	Emphasized 1	EM, STRONG, B, I, U, STRIKE, S, BLINK, ALT	12	0.08	Delimiters	P, TD, FONT SIZE = 1, text not included in any tag

the lowest level $N = 1$. Then, the *TF* measure was computed as the summative value of the term frequency in each node multiplies by its level, as stated by the equation: $TF(t, d) = \sum TF(t, n) \times level(n)$ where n indicates the nodes in document tree. The TF-IDF measure was then used for document indexing, where *TF* is defined by the above equation. Their work was applied for mining structural trees and inferring semantic relations between concepts.

An interesting work by de Moura et al. (2010) presented a block weighting (bw_i) approach that "do not require a learning process nor a type of manual investigation to compute blocks ranking as previous research" (p. 2503). The study used HTML tags for webpage segmentation, and exploited DOM tree to describe the layout of webpages and to categorize blocks into classes. Two basic statistical measures namely *Inverse Class Frequency* (*ICF*), and *Spread* were introduced as a basis of bw_i in web documents, where *ICF* defines the contribution of a class (of several blocks) to the document, while *Spread* indicates the contribution of a term to different blocks. Based on these two measures, nine block-weighting functions were evaluated on the term-level, block-level, and class-level. Web retrieval was carried out by modifying Okapi BM25 ranking function to include bw_i block weights. In our work, we use *ICF* and *Spread* measures as well as a newly defined measure for component weighting in a collection of scientific publications different from de Moura's web collections. We also apply component weighting on a new domain—plagiarism detection.

The notion of candidate retrieval (CR), on the other hand, has appeared in plagiarism detection research to indicate the necessity of retrieving a small set of documents that share a global similarity with a suspected document (or query) from large source collections, before conducting a further detailed analysis. Research applied for CR is, therefore, analogous to DR that use bag-of-words-related models such as fingerprinting, VSM, and LSI. Alzahrani and Salim (2010) used word-3-gram fingerprints to retrieve candidates of *q* with Jaccard similarity above a threshold ($\alpha \geq 0.1$). Other research works include using three least-frequent character-4-gram and Jaccard similarity (Alzahrani & Salim, 2009; Yerra & Ng, 2005), using hashed word-5-gram fingerprints and Jaccard similarity (Kasprzak, Brandejs, & Křipač, 2009), and using hashed 50-character chunks with 30-character overlap to retrieve documents that share at least one fingerprint with *q* (Scherbinin & Butakov, 2009). Many research

works have employed VSM for CR. Examples include using word-1-gram VSM and Cosine similarity (Zechner, Muhr, Kern, & Granitzer, 2009), using word-8-gram VSM and custom distance measure (Basile, Benedetto, Caglioti, Cristadoro, & Esposti, 2009), and using character-16-gram VSM and Cosine similarity (Grozea, Gehl, & Popescu, 2009). Ceska (2008) explored the use of LSI model for CR and PD.

Plagiarism Detection

Plagiarism detection (PD) is the successor to CR, which entails in-depth paragraph-to-paragraph (or statement-to-statement) analysis of [query, candidate] pairs to detect local plagiarism cases. Most well-known methods rely on constructing and matching n-gram fingerprints of predefined patterns in the document. The fingerprints could be (a) character-based, which use a sequence of characters, such as 30–45 successive characters from the whole document, or (b) phrase-based, which convert each document to a set of all bigrams or trigrams.

Exact and approximate string matching with various similarity metrics have been used widely for PD. Examples of research works that used exact string matching include character 16-gram matching (Grozea et al., 2009), word 8-gram matching (Basile et al., 2009), and word 5-gram matching (Kasprzak et al., 2009). On the other hand, approximate string matching has been used by a plurality of researchers. For example, Scherbinin and Butakov (2009) used Levenshtein distance to compare word n-grams and combine adjacent similar grams into sections. Su et al. (2008) combined Levenshtein and simplified Smith-Waterman algorithm for identification of local similarities. Elhadi and Al-Tobi (2009) used LCS distance combined with other part-of-speech (POS) syntactical features to identify similar strings locally and rank documents globally.

Vector similarity metrics such as Containment, Cosine, and Jaccard have commonly been applied in different PD research. Examples include using the matching coefficient with a threshold to score similar statements (Daniel & Mike, 2004), using Cosine similarity on document fragments to enable global and partial PD without sharing the documents' content (Murugesan, Jiang, Clifton, Si, & Vaidya, 2010), estimating the similarity between n-gram terms of different lengths using Jaccard coefficient (Barrón-Cedeño, Basile, Degli Esposti, & Rosso, 2010); while in their previous work

(Barrón-Cedeño & Rosso, 2009), a containment similarity measure was used to compare word n-gram, n = {2,3}.

All of the above CR and PD approaches incorporate bag-of-words-related models that emphasize the flat representation of documents and focus on the detection of copied text. Our review on plagiarism linguistic patterns and detection methods (Alzahrani, Salim, & Abraham, 2011) covers other techniques that incorporate fuzzy, syntactic, semantic, and structural features for PD. None of the existing studies, until now, employ CR or PD using structural information in structured or semistructured documents. This work, therefore, exploits the use of structural information in scientific publications to convey different interpretation of plagiarism from different components (or segments). For instance, some plagiarism cases in the introductions, definitions, or general field-based knowledge are of marginal importance comparing to plagiarism cases in the results and evaluations parts of the paper. The introductions normally contain general knowledge and it is legitimate to have general definitions and theories that are redundant across different publications. It is also obvious that some components such as notes, copyrights, and acknowledgments contain texts with no significance to the plagiarism detection. Thus, different structural components in the publication may be assigned a degree of importance (i.e., weight) to highlight significant plagiarism cases.

Segmentation of Scientific Publications

When writing a scientific publication, the author usually organizes the content into titles, sections, paragraphs, and notes to present the ideas in the most understandable way to the audience. It is very conventional that a scientific publication begins with a title, authors' names, abstract, keywords, and sections, which in turn, begin with a header and have a textual body of various components, such as paragraphs, lists, tables, lists, equations, and quotes. When skimming an article, the reader usually looks at the titles, sections, and other structural elements to get the gist of the main ideas. In addition, citation evidence in scientific articles is the proper and the most conventional way to acknowledge previous work in the literature. Scientific publications evolve around citations, which guide the reader to differentiate one's contributions from others' contributions. If a particular part of a scientific publication is taken from others' work and written without proper citation, it is called plagiarism. In the following, we introduce several definitions that are important to characterize the segmentation process, citation evidence, and plagiarism in publications.

Definition 1. A structural component C is a self-contained and self-consistent logical region within a scientific article that has a unique purpose and is visually distinguished from other components. It can be expressed by a pair of two items, as follows:

$$C = (\ell, v) \ : \ \ell \in L$$

where ℓ refers to the label given to the component to indicate its purpose, as will be discussed shortly, and v is the textual value of that component.

Structural components are subject to interpretation by the reader, but also can be identified automatically using LSE methods (Burget, 2007; Luong et al., 2010; Ratté et al., 2007; Stoffel et al., 2010). In this work, we use *SectLabel* tool (Luong et al., 2010) to extract the logical structure of scientific publications. To represent labels, SectLabel uses a rich set of labels L, which is commonly sufficient to label different components in scientific papers, as follows:

$L = \{title, author, affiliation, email, keywords, categories, copyright, sectionHeader, subsectionHeader, subsubsectionHeader, bodyText, equation, construct, figure, figureCaption, footnote, listItem, note, page, table, tableCaption, reference\}$

Notice that *construct* defines any part of text that is separated visually from the *bodyText* such as definitions, lemmas, proofs, algorithms, and pseudo-codes. Figure 2 shows an example of a scientific article with different components and their labels. Each component should be unique within the paper and not overlap with other components in the same article.

Definition 2. A scientific article A can be expressed as a set of all structural components that constitutes the article, as follows:

$$A = \sum_{i=1}^{n} C_i$$

where n is the total number of components, and each component C_i is defined by a pair (ℓ_i, v_i). Notice that labels may be given to more than one component but the pair of label ℓ_i and value v_i should be unique throughout the article.

Definition 3. A generic class G is a text type that a group of structural components may belong to, or can be classified under it. The Methodology section of a scientific paper, for instance, normally expands into multiple structural components, and a generic classifier should be able to group these components that describe the methodology under the same class.

Generic classes are subject to interpretation by analysts who would be experts in the field of that publication. Automatic *generic classifiers* are also available and can be used to classify structural components according to different aspects such as rhetorical status (Teufel, 1999; Teufel & Moens, 2002), rhetorical-problem paradigm (Hagen et al., 2004), or problem-solving hierarchy (Luong et al., 2010). Because the latter is more general than the formers and can be applied to discipline-independent collection of publications, we consider a set of generic classes G that is based on problem-solving hierarchy, as follows:

$G = \{Title, Author data, Abstract, Categories, General terms, Keywords, Introduction, Background, Related work, Method, Evaluation, Discussion, Conclusion, Acknowledgment, Copyright, References\}$
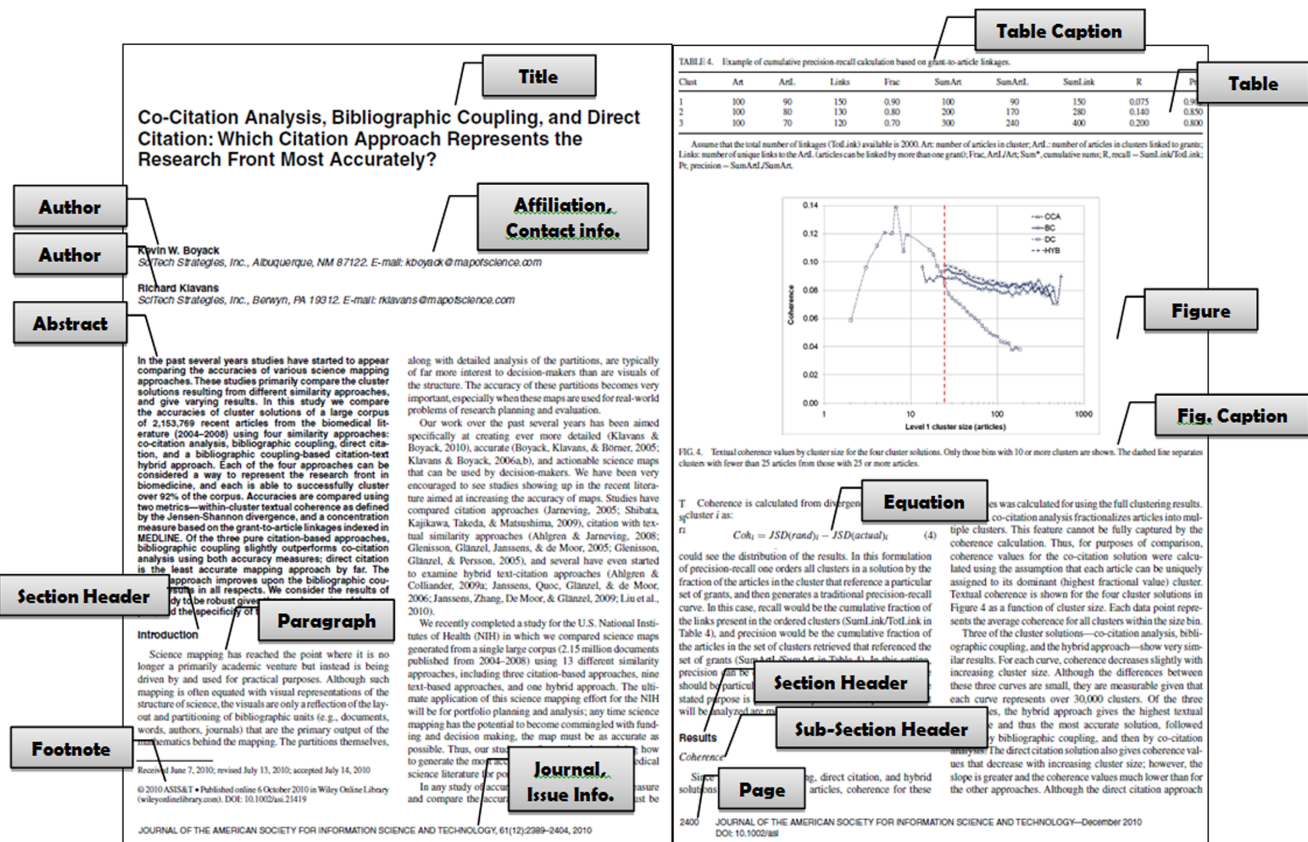
FIG. 2.   A scientific article and its logical structure components. [Color figure can be viewed in the online version, which is available at wileyonlinelibrary.com.]

Under these generic classes, scientific publications can generally be classified into four types: (a) a *full research paper* that explores problem-solving-based research and features components under all generic classes, (b) a *review paper* that explores previous research on a particular problem, but may not include method and evaluation generic classes, (c) a *demo paper* that publishes software or the results of testing specific methods or tools without giving much detail to the literature review, and (c) a *squib*, similar to a demo paper, suggests a new solution or direction of research based on a specific background, but may not give related works and other technical details.

Definition 4. A citation evidence ε is the situation where a component C, or part of it, is quoted or cited to another work in the literature, and the name of the reference is provided in the list of references or bibliography, usually at the end of the article. The citation evidence can be expressed as a triple of three items, as follows:

$$\varepsilon = (\alpha, \beta, \gamma)$$

where α is the citation marker that assigns a text to one of the references, β is the cited (or quoted) string, and γ is the reference phrase in the list of the references.

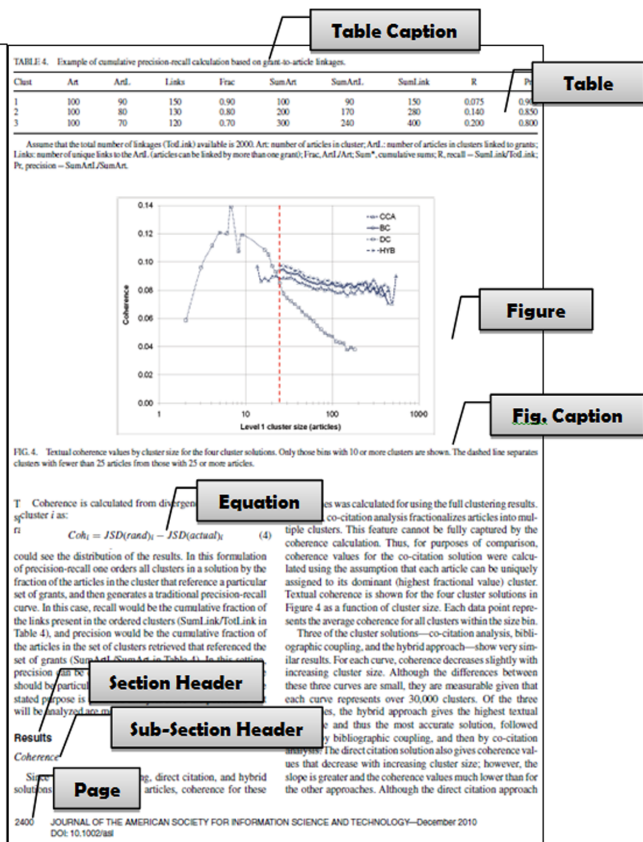To extract the citation evidence in a scientific publication, CP tools should be employed. In this work, we use ParsCit (Councill et al., 2008) which has the ability to (a) extract citation markers of different styles, (b) extract the cited (or quoted) string that indicates the context wherein the citation marker is used to refer to one of the references, and (c) parse citation phrases from the list of references. In some ad hoc experiments that we conducted, ParsCit tool was able to handle cases that the citation marker refers to multiple references (e.g. [2,3,9]) and is also able to handle cases that multiple citation markers refer to the same reference work.

Definition 5. A collection of scholarly documents D can be expressed as set of publications, as follows:

$$D = \sum_{i=1}^{|D|} A_i$$

where $|D|$ is the total number of publications in the collection. We assume that the collection is free from duplicate documents, and has publications under the same general category, e.g., Information Science and Technology.

Definition 6. A plagiarism case ρ in a scientific article A is when a textual content of a component C, or part of it, gains a high similarity score with another textual content of a component C', or part of it, in a candidate article A', and is not

bounded with proper citation evidence ε. Thus, a plagiarism case can be expressed as a quadruple of four main items, as follows:

$$\rho = (C, C', A, A') : (Sim(C, C') \geq threshold) \wedge ((C \wedge \varepsilon) = false)$$

In this sense, plagiarism checking involves two kinds of evidence: a high similarity score—using some similarity measure—between two components, and the absence of citation evidence. Because current research on plagiarism detection focuses on the first evidence but not the second, part of the experimental work will cover this issue. The rational of using citation evidence is that quoted or cited portions of text will not be screened for plagiarism which may (a) accelerate the detection process, especially for tools that use huge databases; and (b) give more realistic similarity index that express the plagiarism ratio in the submitted document. Besides citation evidence, we explore the use of structural information to screen significant plagiarism cases and present them to the users.

Component-Based Weighting

The logical structure of scholarly publications can be used to improve terms weighting. For example, a single occurrence of a term in the *Title* means that the article fully concerns about that term. Thus, terms weight in these components should be improved to better indicate their significance in the article. To quantify the importance of a term *t* in a structural component *C*, we will use a weighting function *f* that can be defined as follows:

Definition 7. A component-weight factor *f*(*t*, *C*) is a quantitative function that is used to measure the weight of a structural component *C* in an article *A*, based on the relevance between terms in C and other structural components.

In this regard, *f*(*t*, *C*) defines a "qualitative" importance of a component *C* in the article, which can be assigned manually by an expert during the indexing phase of documents. For example, in a scale between 0 (*completely not significant*) and 1 (*completely significant*), one can assume that *f*(*t*, *C*) = 1, if *t* is present in *Title*, and in contrast, *f*(*t*, *C*) = 0 if *t* is present in Acknowledgments and Copyrights. However due to the fact that scientific papers usually have a large number of components of variable lengths, it is almost impossible to assign components weight manually, and immense efforts are required to perform such task individually. To solve this problem, automatic component-weight factors have been introduced in flexible information retrieval by Pasi (2002). Some methods have been developed (Bounhas & Slimani, 2010; de Moura et al., 2010; Marques Pereira et al., 2005; Marteau et al., 2006) that use typical TF-IDF weighting in IR, but with structural components of documents taken into consideration. The following section explores different strategies to weight components in scientific publications.

Strategies to Compute Component-Weight Factors

To compute *f*(*t*, *C*) automatically, we experiment two statistical measures: inverse generic class frequency (*IGF*), and *Spread* (de Moura et al., 2010), as follows:

Definition 8. Given that a generic class *G* has *N* components $\{C_1, C_2, \ldots, C_N\}$, and a term *t* that occurs in $N_{t,C}$ components of *G*, the *IGF* of a term *t* in *C* is defined as:

$$IGF(t, G) = \log \frac{N}{N_{t,C}} \tag{1}$$

Definition 9. The *Spread* of a term *t* in an article *A* is the number of structural components in *A* that contain *t*. The *Spread* can be seen as the structural frequency of a term, which can be expressed as follows:

$$Spread(t, A) = \sum_{C \in A} i \ where \ i = \begin{cases} 1 & if \ t \in C \\ 0 & otherwise \end{cases} \tag{2}$$

The *IGF* and *Spread* reflects the structural information in scientific publications in contrast to the normal term frequency (TF) and inverse document frequency (IDF) in the vector space model (VSM). The *IGF* defines how well a term *t* can discriminate components, and how much a generic class *G* articulates information in a given article, instead of using the whole document collection. For example, the term *citation* in the article shown in Figure 2 has low *IGF* because it is too frequent in that article; hence, it does not discriminate between components. The terms *co* and *direct* that are associated with citation (co-citation vs. direct citation) have high *IGF* in the Method generic class, for instance, because these two terms discriminate very well between components that incorporate co-citation terminology and those that concern direct citation. Spread, on the other hand, diversifies the normal term frequency (TF) of a term *t* as it considers the frequency of structural components that have *t*. Spread implies that the more components that have *t*, the more significant it is in the document. To illustrate, *Spread* of the term citation in the article that discusses different citation approaches as the one shown in Figure 2, is high because this term almost appears in every component, in a sense that it is significant in that article more than, for instance, the term *intercitation*, which appears in few components. Moreover, some terms like co, direct, and bibliographic have high significance based on the *Spread* measure, and a component that has all these terms (e.g., discussion) is likely to be more important than a component that has either term.

In addition to *IGF* and *Spread*, we introduce a new statistical measure called *Depth* as follows:

Definition 10. The *Depth* of a term *t* in a generic class *G* refers to the frequency of *t* in *G* normalized by the maximum frequency in *G* such that we do not underestimate classes with low components. It can be expressed as follows:

$$Depth(t, G) = \frac{TF_{t,G}}{MAX_{t',G}} \tag{3}$$

where $TF_{t,G}$ is the term frequency in generic class G, and $MAX_{t',G}$ is the maximum term frequency gained by a term t' in G. In this regard, *Depth* quantifies how much information associated with a term t in a given generic class G, and will range between 0 and 1. If $t = t'$, $Depth = 1$, which means that this term constitutes much of the information given by G.

To clarify the difference between *Depth*, *Spread*, and *IGF*, let us consider some terms in the sample article shown in Figure 2. The term citation gains a near-to-one *Depth* in nearly all generic classes because it is too frequent, whereas the term intercitation gains zero or near-to-zero *Depth* in all generic classes due to its appearance only once (in one of the paragraphs in the literature section) as a less-frequent alternative terminology to direct citation. Additionally, the *Spread* of citation counts on the whole range of the article, whereas *Depth* of citation counts on generic class G, and a term may gain different *Depths* in different generic classes of the same article. Although *IGF* underestimates classes with low components, as will be discussed shortly, *Depth* and *Spread* do not give legitimate values for low-component generic classes.

Using *IGF*, *Spread*, and *Depth*, we introduce different functions to compute component-weight factors for the logical structural representation of scientific publications. Notice that Equations 4–9 have been developed by de Moura et al. (2010) to weight different blocks in webpages and to improve search results in four web collections. To start with, a component-weight factor that is based on *IGF* can be defined as follows:

$$f_1(t, C) = \begin{cases} \dfrac{\sum_{t' \in C} IGF(t', G_C)}{|C|} & if \quad t \in C \\ 0 & otherwise \end{cases} \quad (4)$$

where t' refers to all distinct terms in a component C, G_C is the generic class that has C, and $|C|$ is the size of C (i.e., number of distinct terms). Another *IGF*-based component-weight factor can be defined based on the contribution of all distinct terms t' in a generic class G_C as follows:

$$f_2(t, C) = \begin{cases} \dfrac{\sum_{t' \in C} IGF(t', V_{G_C})}{|V_{G_C}|} & if \quad t \in C \\ 0 & otherwise \end{cases} \quad (5)$$

where V_{G_C} is the vocabulary composed of all distinct terms from all components under of a generic class G_C and $|V_{G_C}|$ is its size. Notice that the first factor assigns different weights to components under G_C, whereas the second factor assigns the same weight to all components under G_C, which indicate that all components under the Methodology section, for instance, are equally important.

Another two *Spread*-based component-weight factors are defined at the component-level and generic class-level, respectively, as follows:

$$f_3(t, C) = \begin{cases} \dfrac{\sum_{t' \in C} Spread(t', A_C)}{|C|} & if \quad t \in C \\ 0 & otherwise \end{cases} \quad (6)$$

$$f_4(t, C) = \begin{cases} \dfrac{\sum_{C' \in G} \dfrac{\sum_{t' \in G_C} Spread(t', A_C)}{|C|}}{|V_{G_C}|} & if \quad t \in C \\ 0 & otherwise \end{cases} \quad (7)$$

where t' refers to all distinct terms in a component C, A_C is the article that has C, $|C|$ is the size of C, V_{G_C} is the vocabulary composed of all distinct terms from all components under of a generic class G_C and $|V_{G_C}|$ is the size of vocabulary in G_C.

IGF-based and *Spread*-based component-weight factors are combined in the following factors:

$$f_5(t, C) = \begin{cases} \dfrac{\sum_{t' \in C} IGF(t', G_C) \times Spread(t', A_C)}{|C|} & if \quad t \in C \\ 0 & otherwise \end{cases} \quad (8)$$

$$f_6(t, C) = f_2(t, C) \times f_4(t, C) \quad (9)$$

Depth–based factors are similarly defined based on the component-level and generic class-level, as follows:

$$f_7(t, C) = \begin{cases} \dfrac{\sum_{t' \in C} Depth(t', G_C)}{|C|} & if \quad t \in C \\ 0 & otherwise \end{cases} \quad (10)$$

$$f_8(t, C) = \begin{cases} \dfrac{\sum_{C' \in G} \dfrac{\sum_{t' \in G_C} Depth(t', G_C)}{|C|}}{|V_{G_C}|} & if \quad t \in C \\ 0 & otherwise \end{cases} \quad (11)$$

Finally, we combine *Depth* and *Spread* into two new component-weight factors as in Equations 12 and 13, and will experiment their effectiveness on component weighting. Notice that we tend not to combine *Depth* and *IGF* together (even not to combine all of the three measures) due to the problem of *IGF*'s underestimation of some components as will be explained shortly.

$$f_9(t, C) = \begin{cases} \dfrac{\sum_{t' \in C} Depth(t', G_C) \times Spread(t', A_C)}{|C|} & if \quad t \in C \\ 0 & otherwise \end{cases} \quad (12)$$

$$f_{10}(t, C) = f_4(t, C) \times f_8(t, C) \quad (13)$$

Dealing With Classes of Low Components

The previous definition of *IGF* leads to a practical problem with generic classes that have one or only a few components such as Title, Abstract, Introduction, and Conclusion, which leads to the underestimation of the importance, i.e., component-weight, of these classes by *IGF*-based factors.
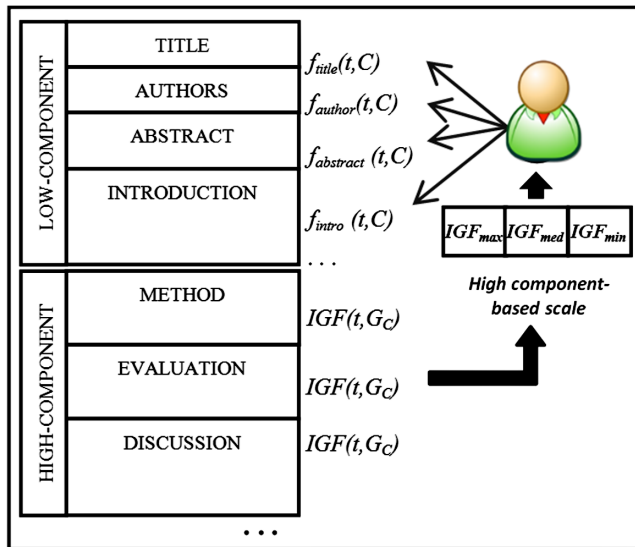
FIG. 3. Strategy for weighting classes with low components. [Color figure can be viewed in the online version, which is available at wileyonlinelibrary.com.]

One proposed solution to deal with generic classes with few components is to use the reader's intuition (or preference) of these classes, as suggested by (Pasi, 2002). The reader, who is usually an expert in the field, can qualitatively judge the importance of different components in a scientific article. To illustrate, the reader may consider some generic classes of a few components such as an Abstract are more important than classes such as an Introduction. Moreover, a component like Title can define the content of the article; terms in this component are considered very significant. However, this qualitative judgment by the reader may not be precisely predicted as quantitative weights.

We propose a procedure that combines the reader's qualitative preference and a quantitative measure, as shown in Figure 3. The steps can be illustrated as follows:

1. All generic classes in a scientific article are classified into low-component generic classes if they have less than three components, and high-component generic classes, otherwise. The following generic classes {*Title, Abstract, Keywords, Conclusion, Copyrights, References*} have always been classified as low component, and the generic classes {*Introduction, Discussion*} have sometimes been classified as low component. Notice that we ignore {*Categories, General Terms*} generic classes because they do not exist in many papers.
2. *IGF*-based component-weight factors are computed only for high-component generic classes (i.e., those with more than three components).
3. Low-component generic classes are presented to the reader (i.e., expert) to qualitatively classify them according to their importance (or in other words, according to their influence or contribution to the body of the paper) into Important, Moderate, or Poor.
4. Then, the scale of *IGF*-based factors for high-component generic classes is used to estimate the component weights

for low-component ones. The assumption is undertaken according to the following formula:

$$f_{low}(t, C) = \begin{cases} IGF_{\max} & if \quad G_{C-\text{IMPORTANT}} \\ IGF_{med} & if \quad G_{C-\text{MODERATE}} \\ IGF_{low} & if \quad G_{C-\text{POOR}} \end{cases} \quad (14)$$

where $IGF_{\max}$, IGF_{med}, and $IGF_{\min}$ are the maximum, median, and minimum *IGF*-based component-weight factors, respectively; as obtained automatically for high-component classes using Equations 4 or 5. We will consider how to adjust the weights of low-component generic classes using this measure in the Parameter Set-Up section.

How to Interpret Component-Weight Factors

We explore the meaning of component-weight factors on a small number of scientific publications chosen randomly from our test collection (details of the test collection are described in a later section). Figure 4 shows an example of a scientific article that contains 65 components, as the first and last pages appear. The components are extracted and classified (Luong et al., 2010) under different generic classes as stated in Definition 3. We employ all component-weight factors, as well as the f_{low} measure, as Table 1 presents the results partially. In our analysis of the sample, as well as the one shown in Figure 4, and we found the following (Table 2):

- *Spread*-based factors estimate high weights for components that the reader would consider important, and normally give the highest weight to the *Title* in comparison to other components because this component contains terms that have high structural frequency within the article. Also, components like *Abstract* are more important than *Author data*, *Copyrights,* and *Acknowledgments*; whereas remaining components like *bodyText, listItem, etc.,* seem to be comparable and have no advantage over each other.
- Although *Spread-f_4* treats all components under a generic class G equally (i.e., *sectionHeader* and *bodyText* are given the same weight), *Spread-f_3* gives low weights to *sectionHeader* when it contains words that are not related to the article's topic (e.g., component 3, 5, and 60), but gives high weights to *sectionHeader, tableCaption,* and *figureCaption* if they contain terms discussed throughout the article thoroughly and mentioned in many other structural components accordingly (e.g., component 7 and 59).
- *IGF*-based factors may give some components that are usually considered important to the readers a zero or near-to-zero weights if they are presented under generic classes with only few components. For instance, the generic class $G = Title$ has only one component which is {*title*}, $G = Abstract$ has two components {*sectionHeader, bodyText*}, and so on. Previous section discusses this problem in detail and proposes a solution to this factor.
- As part of our experimental work investigates the use of f_{low} factor according to Equation 14, we notice that f_{low} yields better component-based weights in comparison to *IGF*-based factors because it incorporates a qualitative measure into its decision. For instance, *title* and *bodyText* in *Abstract* are given the maximum weight, whereas components that are insignificant to the detection process based on an individual's perspective, are given the minimum weight (equal to zero in
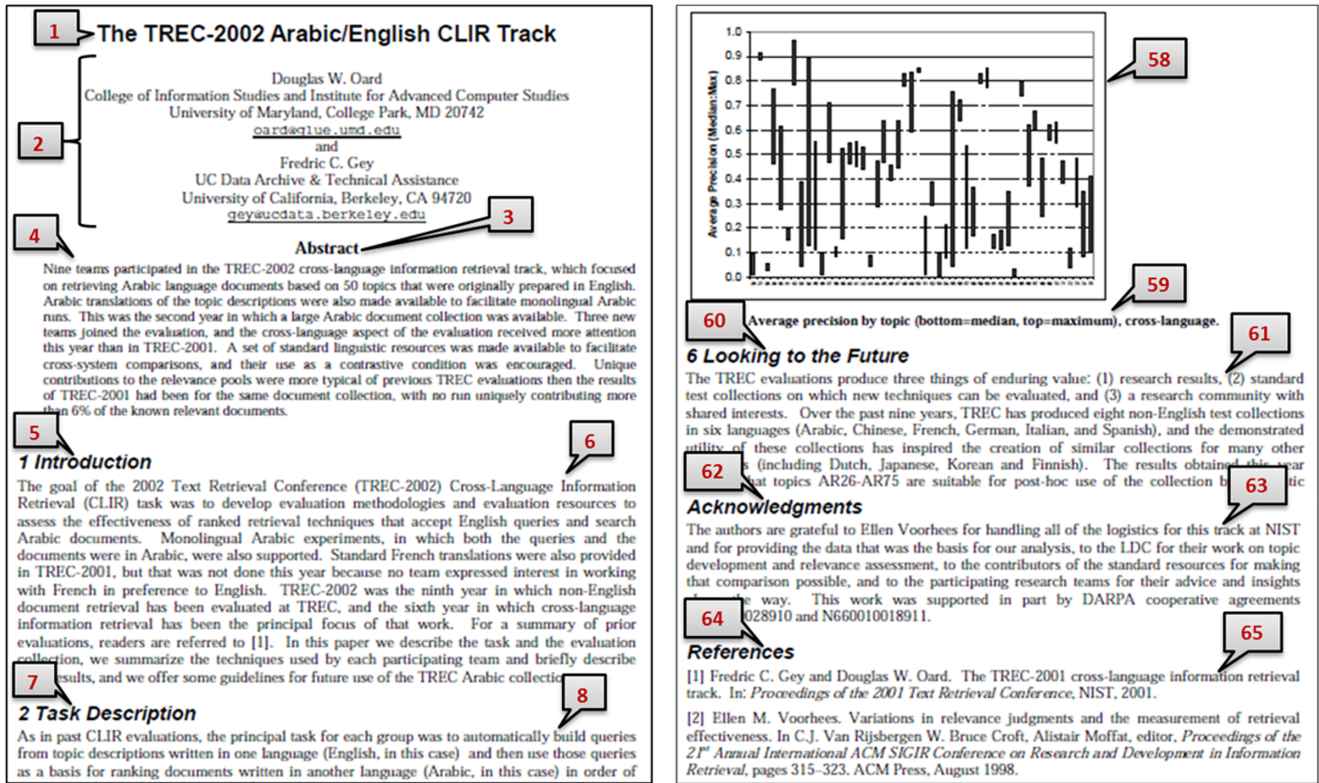
FIG. 4. A sample article from our test collection, with different components shown in the first and last page.

TABLE 2. Examples of component-based weighting factors found in a sample article, as the one shown in Figure 4.

C	G	ℓ	IGF-based		Spread-based		Depth-based		Combined factors				
			f_1	f_2	f_3	f_4	f_7	f_8	f_5	f_6	f_9	f_{10}	f_{low}
1	Title	Title	0.0000	0.0000	9.2500	9.2500	0.3750	0.5500	0.0000	0.0000	3.5000	1.8425	3.2420
2	Author data	Name/affiliation/ e-mail	0.6931	0.0408	0.5000	0.4118	0.2500	0.2588	1.7329	0.0983	0.5000	1.3478	0.0000
3	Abstract	sectionHeader	0.0000	0.6787	1.0000	7.0000	0.0000	0.3802	0.0000	4.7509	0.0000	2.6615	0.0000
4		bodyText	0.6931	0.6787	7.0833	7.0000	0.3750	0.3802	4.9098	4.7509	3.1927	2.6615	3.2420
5	Introduction	sectionHeader	0.0000	0.6803	1.0000	6.1852	0.0000	0.3074	0.0000	4.2079	0.0000	1.9014	0.0000
6		bodyText	0.6931	0.6803	6.2037	6.1852	0.3037	0.3074	4.3001	4.2079	2.4667	1.9014	1.9854
7	Method	sectionHeader	1.9356	3.0031	5.5000	2.5660	0.1429	0.0773	10.5739	7.7059	0.8000	0.1984	1.9356
8		bodyText	2.7944	3.0031	4.8194	2.5660	0.1516	0.0773	10.3199	7.7059	1.4437	0.1984	2.7944
...													
58	Evaluation	figure	1.6975	1.9660	4.7778	4.1282	0.1349	0.1349	8.0850	8.1162	0.6746	0.5570	1.6975
59		figureCaption	1.1478	1.9660	10.3333	4.1282	0.3393	0.1349	10.1621	8.1162	4.0536	0.5570	1.1478
60	Conclusion	sectionHeader	0.0000	0.0000	2.0000	3.4397	0.1875	0.1950	0.0000	0.0000	0.4375	0.6709	0.0000
61		bodyText	0.0000	0.0000	3.4912	3.4397	0.1963	0.1950	0.0000	0.0000	1.0241	0.6709	1.9854
62	Acknowledge	sectionHeader	0.0000	0.6715	1.0000	1.5938	0.0000	0.5156	0.0000	3.0846	0.0000	2.3687	0.0000
63		bodyText	0.6931	0.6715	1.6250	1.5938	0.5000	0.5156	3.2058	3.0846	2.3906	2.3687	0.0000
64	References	sectionHeader	0.0000	0.6850	2.0000	2.8471	0.0000	0.2588	0.0000	1.9502	0.0000	0.7369	0.0000
65		referenceList	0.0000	0.0000	2.8706	2.8706	0.2588	0.2588	0.0000	0.0000	0.9859	0.7430	0.0000

this article), which allows us to exclude these components from the plagiarism detection results.

- *Depth*-based factors have several advantages over all other factors. First, the estimation of the most important components does not exceed one (i.e., range of *Depth* is between 0 and 1), which is more manageable than *Spread* and *IGF*. Second, *Depth*-based factors have shown no problematic behavior with low-component generic classes in contrast to *IGF* factors. Third, *Depth*-based factors have scaled well with large-component generic classes in contrast to *Spread* factors, which could gain big values based on the number of structural components in the article. We also found that *Depth-f_7* works better than *Depth-f_8* in the sense that it is similar to what people usually consider as important.

Plagiarism Detection (PD) System

Plagiarism from different parts of the document may convey different interpretations; for instance, plagiarism in the introduction of the article may be of marginal importance compared to plagiarism in the evaluation and discussion. Besides, there are legitimate reasons that a text may be redundant between different papers especially in the methodology section, such as a proof copied to be extended, a report of self- (or team-) previous work to be expanded, an equation or series of equations applied to a new domain or application, and other situations that might be judged as plagiarism-free by people. This work employs several component-weight factors introduced in the previous section, to give a further decision about significant plagiarism cases in scientific publications. To illustrate, a component that gains high weight and is found to be plagiarized under some similarity measure, may be more significant than a plagiarism instance in a component with low weight. Thus, PD in scientific publications can be assessed by structural information in terms of retrieval of candidate documents, detection of significant cases, and ignorance of non-significant or less important cases. Using citation evidence, additionally, provides a solid base for filtering out cases with proper citation evidence, which may improve the similarity index and reduce false detection results.

FIG. 5.   Plagiarism detection system framework.

General Framework

Figure 5 shows the general framework of the proposed antiplagiarism system including two main phases: source archive preparation and plagiarism revelation. In the first phase, a source collection D composed of scientific publications is passed through a sequence of preprocessing operations as follows:

1. Logical structure extraction and labeling of different components inside the article using *SectLabel* (Luong et al., 2010.
2. Generic class classification of structural components using *SectLabel* (Luong et al., 2010)
3. Citation evidence parsing using *ParsCit* (Councill et al., 2008) wherein the following are extracted: (a) citation marker, which refers to a small phrase of author name or reference number that appeared before, within, or after a reported work; (b) raw text, which indicates the reported text from the literature; and (c) reference string from the list of references.
4. Structural component weighting using one of our component-weight factors listed by Equations 3–16
5. Structural term weighting, which incorporates structural component weights into current term weighting schemes as will be explained in the next section. Results from these steps are usually stored into the archive.

The second phase is actually triggered by a submission of a publication q to be checked against source archive. q is also passed under the previous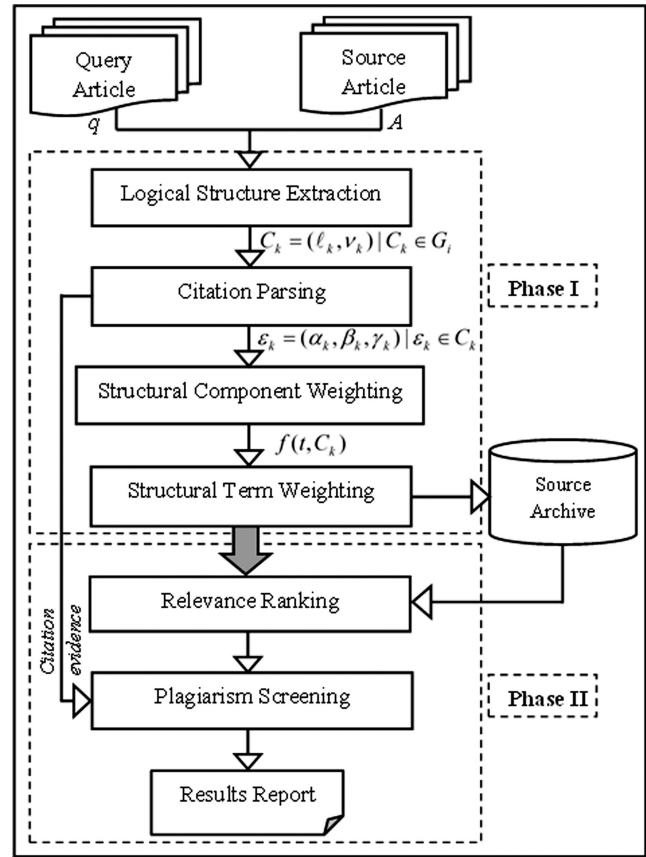 preprocessing steps. Notice that we do not need to store q into the archive, unless it is to be compared with future queries. Two steps are accomplished in this phase: (a) retrieval of candidate documents based on a global similarity measure (as discussed in DR and CR related work), and (b) screening for plagiarism by further comparing q with its candidate documents based on a local similarity measure (as discussed in PD related work) to highlight instances of plagiarism.

Retrieval of Candidate Documents

To incorporate component-based weight factors into CR, we suggest a modification on TF-IDF weighting scheme used in typical VSM. The original TF-IDF incorporates local and global parameters to assign weight to each index term t in document d, according to the formula:

$$w_{t,d} = TF_{t,d} \cdot \log \frac{|D|}{|d \in D : t \in d|} \tag{15}$$

where $TF_{t,d}$ is term frequency in d, $|D|$ is total number of documents in the dataset, and $|d \in D | t \in d|$ is the number of documents that contains t. The above term weighting $w_{t,d}$ is associated with a bag-of-words representation of documents. As suggested by Pasi (2002), we can modify the local parameter to reflect the structural information from the document.

Thus, term weighting in a scientific publication A can be given according to the following formula:

$$w_{t,A} = TF'_{t,A} \cdot \log \frac{|D|}{|A \in D : t \in A|} \qquad (16)$$

where $TF'_{t,A}$ is a new parameter that provides a weighting scheme for the index terms in each logical component under different generic classes that constitutes the article, which in turn can be expressed as:

$$TF'_{t,A} = \sum_{G \in A} \sum_{C \in G} TF_{t,C} \times f_k(t, C) \qquad (17)$$

where $TF_{t,C}$ is the frequency of a term t in a structural component C, and $f_k(t, C)$ is one of our component-weight factors. Then, Cosine similarity can be used to retrieve the most similar source publications to the query article q, which can be calculated as:

$$Sim(A, q) = \frac{\sum\limits_{t \in A \wedge t \in q} w_{t,A} \times w_{t,q}}{\sqrt{\sum\limits_{t \in A \wedge t \in q} (w_{t,A})^2} \times \sqrt{\sum\limits_{t \in A \wedge t \in q} (w_{t,q})^2}} \qquad (18)$$

In this regard, we compare q only with publications $\{A_1, A_2, \ldots, A_n\} \subseteq D$ that satisfy the following condition: $\{t : t \in A_i\} \cap \{t : t \in q\} \neq \theta$. Finally, relevant publications are sorted descendingly as the candidates list of q while publications with $Sim(A, q) \leq threshold$ are excluded, where $threshold$ value will be defined by our experimental set-up.

Screening for Significant Plagiarism After retrieval of a list of candidate publications that share global similarity with q, PD stage involves exhaustive analysis and in-depth comparison of each candidate and q. We use component-based plagiarism screening which measures the degree of overlapping between structural components in q and A. To accomplish this work, we suggest a modification on Jaccard similarity (also known as overlapping distance) whereby all structural components $C_i \in A$, $i = 1, 2, \ldots, n$ and n is the total number of components in A, are compared with structural components $C_j \in q$, $j = 1, 2, \ldots, m$ and m is the total number of components in q, according to the following formula:

$$Overlap(C_i, C_j) = \Delta \cdot \left[\frac{|C_i \cap C_j|}{|C_i \cup C_j|} \right] \quad \text{where}$$
$$\Delta = f_k(C_i, t) \times f_k(C_j, t) \qquad (19)$$

In the abovementioned equation, we introduce a new parameter called *significance factor* Δ that defines to what extent a plagiarism case is significant (or insignificant). To illustrate, the weights of compared components are combined into a single metric, where f_k is one of the component-based factors introduced by the formulas 3–14, and associated with the similarity measure. The effect of the significance factor is to increase/decrease the overlapping similarity between compared components such that components that are highly similar but are not of interest to PD will be discarded or

downgraded to lesser degrees of similarity. For example, *notes*, *copyrights*, *acknowledgments*, *author data*, *references*, and similar components are not important in the detection results. The significant factor Δ may assist the PD algorithm to discard two identical *acknowledgments*. On the other hand, components that are given high weights such as in the *results* and *discussion*, will be upgraded to higher degrees of similarity. In the case of using f_{low} as component-weight factor to compute Δ, a human-based qualitative assessment is integrated to present detected plagiarism cases as degrees according to their occurrence within the document.

Additionally, citation evidence ε is used at this stage to filter out texts that have been cited properly; a feature that is not handled even by well-known plagiarism detectors. Therefore, the decision of a plagiarism case ρ committed by the authors of q from a source article A is true if and only if the overlapping between compared components exceeds a threshold and there exists no citation evidence to mark that component as cited. Thus, the decision about ρ can be undertaken as follows:

$$\rho_{A,q}(C_i, C_j) = \begin{cases} 1 & if\,(Overlap(C_i, C_j) \succ 0) \wedge (|C_j \cap \varepsilon| = 0) \\ 0 & otherwise \end{cases}$$
$$(20)$$

Finally, to present the results of plagiarism screening in a submitted publication, we define three similarity indices as follows:

Definition 11. The Similarity Index (*SI*) of an article q under investigation and a candidate article A is the percentage of text detected as plagiarized from A, as follows:

$$SI(q, A) = \frac{|\sum_j \rho_{A,q}|}{|q|} \times 100 \qquad (21)$$

where $|\sum_j \rho_{A,q}|$ is the total length (in words) of plagiarism cases that are found to be plagiarized from A, and $|q|$ is the total number of words in the suspicious article q.

Definition 12. The Overall Similarity Index (*OSI*) of an article q under investigation is the overall percentage of text in q detected as plagiarized by a PD approach. *OSI* can be expressed as follows:

$$OSI(q) = \frac{\sum_{A \in candidates} |\sum_j \rho_{A,q}|}{|q|} \times 100 \qquad (22)$$

Definition 13. The Structural Similarity Index (*SSI*) of an article q under investigation is the percentage of components that is found (totally or partially) to be plagiarized from one or more components in candidate publications. *SSI* can be expressed as follows:

$$SSI(q) = \frac{\sum_{C_j \in q} i \quad where \quad i = \begin{cases} 1 & if \quad \exists \rho_{A,q} \in C_j \\ 0 & otherwise \end{cases}}{n} \times 100$$
$$(23)$$

where n is the total number of components in q.

```
<sectionHeader confidence="0.909022" genericHeader="method">RECOGNITION</sectionHeader>
<bodyText confidence="0.9381482">A. Description of Data We used the famous training dataset
    used by Slate and Frey [4]; this dataset includes 20000 input vectors with their desired answers.
    Every input vector includes 16 attributes representing features of the picture that it should be
    recognized. We had the choice to do two types of normalization of values of input vectors. The
    goal of this normalization is to reduce the interval of the incoming absolute values between zero
    and one. If we want to normalize data towards the interval [0,1], we will use the following
    formula:</bodyText>
<equation confidence="0.966643444444444">minmax min ' xx xx x − − =</equation>
<bodyText confidence="0.9917245">If we want to normalize data towards the interval [-1,1], we will
    use the formula:</bodyText>
```

FIG. 6.   A sample of a tagged article using SectLabel (Luong et al., 2010).

FIG. 7.   A sample of parsed citations using ParsCit (Councill et al., 2008). (a) Numbering-based citation; (b) author-naming-based citation.

Publications screened for plagiarism are assigned *SI* and *OSI* based on the percentage of texts found as plagiarized, which means the texts have components that assign high weight factors with no citation evidence. However, the SSI provides an indication of the number of structural components that have been plagiarized, which gives an understanding of how well a scientific work contributes to the knowledge base in a specific area. A scientific publication should provide new knowledge based on a reasonable amount of previous works and literature reviews.

Experimental Results

To validate our methods, we conducted experiments on a dataset of scientific publications wherein thousands of plagiarism cases were inserted automatically. We also compared our methods with two different baselines and evaluated their efficiency.

Dataset

The test collection is composed of 15,412 publications[7] downloaded from several open access journals listed in the *Directory of Open Access Journals* (*DOAJ*), and from conference proceedings available on the Web. The publications have been published from 2002–2010, and categorized generally under science and technology. To construct the dataset, all publications were converted from PDF to TXT using *pdftotext* command-line utility in Unix[8], and were randomly divided into 8,657 source publications and 6,755 queries (i.e., documents under plagiarism checking).

We adopted *SectLabel* LSE and GC (Luong et al., 2010) that extracts (i.e., tags or marks) different constructs in the documents such as title, author, address, affiliation, keywords, bodyText, equation, figure, etc., and maps different constructs under the following generic classes: *Title, Author data, Abstract, Introduction, Related work, Method, Evaluation, Conclusions, Acknowledgments, Copyrights, and References*. We also employed *ParsCit* CP (Councill et al., 2008) to extract the citation evidence. Figure 6 shows a screenshot from a tagged article by *SectLabel*, as one of its section entitled "RECOGNITION" is classified under *Method* generic class. Figure 7 shows two snapshots of *ParsCit* results, as part (a) is used for numbering-based citation, and part (b) is used for naming-based citation.

We developed a so-called artificial plagiarism synthesizer[9], which is used to extract a text from a source article, reformulate the extracted text in a way that is similar to what a plagiarist does to obfuscate the text, and insert the new version

---

[7]The dataset is available at www.u2learn.net/plagiarism/corpus/v1/

[8]http://en.wikipedia.org/wiki/Pdftotext

[9]Source code can be distributed for research purposes. Please contact Dr. Alzahrani to obtain the latest version.

of the text into a query document (i.e., suspicious publication). We used an automatic synthesizer to (a) avoid ethical issues of asking people to simulate plagiarism, (b) save people's time and effort when creating multiple revisions of texts as in Clough and Stevenson's corpus (2011), (c) accelerate the creation of the corpus, and (d) automate the annotation of all plagiarism cases. It is proved that constructing plagiarism cases automatically is sufficient to implement the plagiarism concept (Potthast, Stein, Eiselt, Barrón-Cedeño, & Rosso, 2009) and does not make any difference with simulated or real plagiarism cases in many detection algorithms (Potthast, Stein, Barrón-Cedeño, & Rosso, 2010).

The synthesizer was built using PHP (hypertext preprocessor) integrated with various natural language processing (NLP) tools such as stemmer, lemmatizer, POS tagger, WordNet, Google translator, and automatic summarizers. It works according to the following procedure:

1. Choose one of the queries q randomly.
2. For each q:
   a. Choose one of the source publications A randomly.
   b. From a generic class in A, copy or obfuscate part of the text (few sentences to the whole paragraphs) and insert it under a generic class in q. Both classes are chosen randomly to ensure fairness and to investigate efficiency of the proposed methods in detecting significant plagiarism cases.
   c. Repeat (b) until different cases of plagiarism are constructed from A.
   d. Repeat (a)–(c) until different source publications are involved in the process.
   e. Annotate all plagiarism cases in q.
3. Repeat all the above steps until about 60% of queries are chosen as q and annotated with different plagiarism cases of variable lengths.

Different plagiarism synthesizer methods were employed to construct different kinds of plagiarism, which ensure the generality and efficiency of PD. Synthesizer methods include text mappers, manipulators, paraphrasers, and summarizers. All of the methods, except the first, obtain plagiarism-like cases at three obfuscation levels: light when 10–30% of the text is changed, moderate when 30–70% of the text is changed, and heavy when more than 70% of the text is changed. The following provides details of each method.

- Text Mapper: Used to construct nonobfuscated (copy and paste) plagiarism. The mapper works by copying sentences or paragraphs under a generic header in the source document, inserting it under a suspicious generic header.
- Text Manipulator: Used to restructure/reformulate the text by applying several strategies randomly such as sentence shuffling, word shuffling, POS preserving word shuffling, and word inserting/deleting.
- Paraphraser: Used to construct texts with different semantic variations. We used three strategies of automatic paraphrasing:
  ○ A WordNet lexical database wherein the dictionary form (lemma) of a word is used to randomly return one of its synonyms, antonyms, hypernyms, or hyponyms

○ A back-translation method (Jones, 2009) wherein the text is submitted to Google translator, translated to any language, and then retranslated back to English as another strategy to obtain a paraphrased text close to a human's paraphrase
  ○ A double-back-translation method wherein the previous strategy is repeated twice to obtain a more obfuscated paraphrase
- Summarizer: Used to summarize the text while retaining the important ideas. We used three strategies of automatic summarization:
  ○ A word rank-based summarizer[10] that ranks words in the document by their occurrence; ignoring words that are very common such as *a, an, the, this, that, etc.* By listing the top ranked N words (N can be changed by the user), the score of each statement is calculated by summing the word ranks it has. Words in the sentence but not in the list are given rank 0. Top P rated statements are then used to construct the summary, where P is the compression rate (i.e., percentage of the summary to the original text).
  ○ A fuzzy swarm-based summarizer (Binwahlan, Salim, & Suanmali, 2009) that weights sentences based on swarm-based mechanism and a set of fuzzy rules. Top weighted sentences are then chosen to construct the summary.
  ○ A composite paraphraser-summarizer wherein one of the above strategies is applied followed by a paraphraser strategy to obfuscate the summary

All generated cases were annotated in separate XML files for evaluation purposes. Figure 8 shows an example of annotation files created for the dataset. Annotations have several elements namely *features*; each element describes one plagiarism case except the first, which refers to the tagged version of the article. The *feature* element of a plagiarism case indicates the level of obfuscation (i.e., none, light, moderate, and heavy), and the type of plagiarism (i.e., verbatim, restructure, paraphrase, and summary) as constructed by our synthesizer methods. A word counter (i.e., word-start-position and word-end-position) was used to bound each case. We also annotated which context the plagiarism was taken from in the source documents and placed into the suspicious documents using generic class headers from the tagged documents.

Table 3 shows the general statistics about the corpus including the number of documents, sections, paragraphs, words, section headers, tables, figures, and equations. Notice that we considered that 60% of the queries contain plagiarism cases (W plagiarism), while the rest are not (WO plagiarism). Having this variety of queries will ensure that our PD approach is able to avoid false detections in plagiarism-free queries, which is another important aim of any plagiarism detection approach.

Table 4 shows the details of inserted plagiarism cases into the queries. About 60% of the constructed cases were lightly, heavily, or moderately obfuscated. The cases include not only verbatim plagiarism, but also semantically equivalent types

---

[10]http://w-shadow.com/blog/2008/04/12/simple-text-Summariser-in-php/

```
<?xml version="1.0" encoding="utf-8" ?>
- <document reference="suspicious-document-34.txt" language="en">
    <feature name="tagged-document" reference="suspicious-document-34.xml" language="en" />
    <feature name="artificial-plagiarism" type="summary" obfuscation="heavy" thisGenericHeader="discussions"
      thisWordStartPosition="3732" thisWordEndPosition="3990" sourceFile="source-document-3422.xml"
      sourceGenericHeader="method" sourceWordStartPosition="214" sourceWordEndPosition="693" />
    <feature name="artificial-plagiarism" type="verbatim" obfuscation="none" thisGenericHeader="method"
      thisWordStartPosition="759" thisWordEndPosition="1238" sourceFile="source-document-3422.xml"
      sourceGenericHeader="method" sourceWordStartPosition="214" sourceWordEndPosition="693" />
  </document>
```

FIG. 8.   Annotation example.

TABLE 3.   Details of constructed dataset.

	Type	% Ratio	Publications	Generic classes	Structural components						
					Header	Para.	Word	Table	Fig	Eq	List
Source publications		52	8,657	47,008	53,812	162,591	382,347,73	12,022	47,396	57,479	14,129
Queries	w/plagiarism	25	3,955	23,006	25,731	84,439	208,138,95	8,986	17,203	20,302	5,982
	w/o plagiarism	23	2,800	14,202	17,853	49,814	125,440,84	6,537	11,343	12,616	4,969
	Total		15,412	84,216	97,396	296,844	71,592,752	27,545	75,942	90,397	25,080

TABLE 4.   Details of inserted plagiarism cases.

Practice	# Cases	%	Obfuscation	# Cases	%
Verbatim	7648	42	None	7648	42
Restructuring	3686	20	Light	3622	20
Paraphrasing	3375	19	Moderate	3464	19
Summarizing	3438	19	Heavy	3413	19

Length	# Cases	%	Citation evidence	# Cases	%
Short	4537	25	Without ε	18147	–
Moderate	10888	60	Source	6581	9
Long	2722	15	No source	66532	91

| Total cases: 18,147 | | | Total evidences: 73,113 | | |

of plagiarism that were constructed by the synthesizer with comparable percentages. By comparing the length of each plagiarism case to the length of its source document, the length of cases are short (e.g., few sentences) if the case is less than 20% of the source document, moderate if it is between 20–80%, and long (i.e., near duplicate) if it constitutes more than 80% of the source document. As can be seen in the table, about 25% of the plagiarism cases are short, 60% are moderate, and 15% are long. Finally, we look at cases with citation evidence ε against those without citation evidence as follows. First, we presume that all plagiarism cases constructed by our plagiarism synthesizer do not include proper citation evidence, i.e., we do not add a marker that indicates the source article. Second, we consider original contents in the queries (suspicious publications) that have citation evidence, and their source publications happen to be in the collection. Third, we consider original contents with proper citation evidence, but source publications are not present in the collection. Table 4 shows that the first consideration referred to as "without ε" includes 18,147 cases, while the total citations

in the second and third considerations are 73,113 in our queries. A small percentage (9%) of the citation evidence has its "source" publications in the collection; the rest have "no source" article in our collection.

Baselines

To evaluate the impact of using the proposed approach on the plagiarism retrieval and similarity index, we implemented two baselines that are based on *flat* document representation.

The first baseline is a method of document fingerprinting (or shingling) and Jaccard similarity for CR stage, which we implemented in a previous work (Alzahrani & Salim, 2010). This baseline, which we referred to as FLAT-SHING in our experiments, has been applied widely in plagiarism detection research (Kasprzak et al., 2009; Manning, Raghavan, & Schütze, 2008; Scherbinin & Butakov, 2009; Schleimer et al., 2003). We implemented a k-shingle (or word k-gram) document representation scheme, and computed a Jaccard coefficient to obtain a list of candidate publications

for a query article q, as follows:

$$Sim(A, q) = \frac{|Shingles_A \cap Shingles_q|}{|Shingles_A \cup Shingles_q|} \quad (24)$$

The second baseline uses typical TF-IDF weighting as stated previously in Equation 15, and Cosine coefficient shown in Equation 18 to find similar publications. Because the approach is a modification of this baseline, typical TF-IDF weighting versus structural TF-IDF weighting schemes of scientific publications can be clearly compared using this setting. We referred to the second baseline as FLAT-TFIDF in our experiments, in contrast to the proposed STRUC-TFIDF shown by Equations 16, 17, and 18.

In both baselines, we adopted three string matching schemes for PD from the literature namely word 5-gram (Kasprzak et al., 2009), word 8-gram (Basile et al., 2009) with 3-word overlapping, and sentence-to-sentence matching (Alzahrani & Salim, 2010). These comparison schemes have been commonly used in existing plagiarism detectors, and have yielded good results. In our experiments, we referred to these methods as W5G, W8G, and S2S, respectively.

Parameters Set-Up

Part of our experimental work explores structural weighting of low-component generic classes using the f_{low} measure according to Equation 14. We conducted a qualitative study to investigate the influence (or importance) of different parts in scientific publications. We focused on the parts that have very few components and hence classified as low-component generic classes. A questionnaire was distributed via the postgraduates' mailing list which includes more than 200 e-mails, at the Faculty of Computer Science and Information Systems, University of Technology Malaysia. The questionnaire included a list of 15 titles of scientific publications taken from our dataset, and three simple instructions that ask volunteers to choose one of the publications, read its content (all publications are attached in the e-mail), and indicate the significance/importance/influence of the following parts in the selected article: {*Title, Author data, Keywords, Abstract, Introduction, Conclusion, Copyrights, References*}. If some volunteers were willing to contribute, but none of the papers suited their research interests, they could contact us to request another article. The respondents could choose one of the options—Important, Moderate, and Poor—to indicate the influence of each part in the selected article. Seven volunteers responded to the questionnaire; we found this sample was sufficient to reflect the impression of different components of selected publications on a variety of readers. The majority of the respondents reported the same impressions. We conducted follow-up questions for some volunteers that have reported answers different from the others. For instance, one volunteer reported that {*Author data*} has Important impact, but when we asked about the reasons, the volunteer replied that the author's data may help to search for more related publications by the same author. The main findings of this qualitative study show that the following parts {*Title,*

Abstract} have Important influence on all the respondents, {*Conclusion*} have also Important influence on the majority, {*Introduction, Keywords*} have Moderate significance, while {*Author data, Copyrights, References*} have Poor influence on the majority of the respondents. Therefore, we use these findings as a basis to compute the f_{low} measure.

In FLAT-SHING baseline, we initialized k to 3 to indicate publications that share considerable $k-$shingles as similar. After computing the similarity based on Equation 24, we needed to set a threshold such that publications with Jaccard similarity above this threshold are retrieved as candidates. We conducted an ad hoc experiment on 15% of the queries (i.e., 1013 publications) chosen randomly, and varied threshold values from 0 to 2 with 0.05 incremental step. In each time, a threshold value was used to select candidates. The optimal precision at the optimal recall was obtained at point 0.85. Higher threshold values result in better precision, but at the expense of lower recall due to the increase of false-negatives (FN). On the other hand, lower than 0.85 threshold values yield better recall, but at the expense of decreasing precision. We also manually checked some publications with similarity below 0.85, and we could see that they had either none or very few similar phrases, while those with similarity above this value had between 1 and 16 candidates.

Similarly, in FLAT-TFIDF baseline, we conducted an ad hoc experiment to set a threshold for candidate publications that satisfy the condition $Sim(A,q) \geq threshold$ whereby $Sim(A,q)$ was calculated from Equation 18. We used the same queries and assigned threshold values from 0 to 1 incremented by 0.02 at each run. We found that the optimal value of the threshold in this baseline was 0.92, because values below 0.92 resulted in decreasing the number of true-positives (TP), while values above increased the false-negatives (FN). More interesting, using the threshold in both baselines allows the assumption that the number of candidates of each q is dynamic and may be small, which saves computation time, in contrast to having a fixed number of candidates to each q. Notice that a query q with no candidates means that it may not contain plagiarism at all.

Unlike previous settings, using the threshold with STRUC-TFIDF could be inappropriate because term weights vary based on component weights, as stated by Equations 16 and 17, which may cause significant changes in the calculation of $Sim(A,q)$ in Equation 18. Therefore, all publications that gain similarity above zero are considered.

Evaluation Measures

To evaluate the CR stage of the proposed approach, we used the $F_{harmonic}$ measure that defines the harmonic mean of two complementary measures known as *Precision (P)* and *Recall (R)*, as follows:

$$F_{harmonic} = 2\frac{P \times R}{P + R}, \quad where \quad P = \frac{TP}{TP + FP} \quad and$$

$$R = \frac{TP}{TP + FN}$$

where *TP* refers to true-positives (i.e., the number of candidate documents retrieved as candidate), *FP* refers to false-positives (i.e., the number of documents retrieved as candidate but they are not), and *FN* refers to false-negatives (i.e., the number of candidate documents that are not retrieved). In the experimental work, we used micro-averaged *Precision* (P_{micro}) and *Recall* (R_{micro}), as follows:

$$P_{micro} = \frac{\sum_{q \in Q} TP(q)}{\sum_{q \in Q} TP(q) + \sum_{q \in Q} FP(q)} \quad and$$

$$R_{micro} = \frac{\sum_{q \in Q} TP(q)}{\sum_{q \in Q} TP(q) + \sum_{q \in Q} FN(q)} \quad (25)$$

where *Q* is the set of queries (suspicious publications that we used to insert plagiarism).

On the other hand, we used micro-averaged *Precision* (P_{plag}), *Recall* (R_{plag}), $F_{harmonic}$, and *Granularity* (G_{plag}) to evaluate PD results (Potthast et al., 2010). We presumed that each plagiarism case (denoted as ρ previously) is bounded by a start word and an end word in the source article and the query (suspicious) article, as shown in the annotation example of Figure 8. Therefore, P_{plag} and R_{plag} can be calculated as follows:

$$P_{plag} = \frac{\sum_{q \in Q} \sum_{\rho \in q} TP(\rho)}{\sum_{q \in Q} \sum_{\rho \in q} TP(\rho) + \sum_{q \in Q} \sum_{\rho \in q} FP(\rho)} \quad and$$

$$R_{plag} = \frac{\sum_{q \in Q} \sum_{\rho \in q} TP(\rho)}{\sum_{q \in Q} \sum_{\rho \in q} TP(\rho) + \sum_{q \in Q} \sum_{\rho \in q} FN(\rho)}$$
$$(26)$$

where $\sum_{\rho \in q} TP(\rho)$ is the number of correct plagiarism cases as defined in *q*'s annotation file, and $\sum_{\rho \in q} FP(\rho)$ is the number of wrong plagiarism cases (or cases detected as plagiarism but are not defined in *q*'s annotation file). Besides, detection granularity, G_{plag} of ρ indicates the ability of the detection algorithm to detect that case at once (Potthast et al., 2010). For example, in PD methods that compare statements or n-grams, the detected similar statements or n-grams should be combined as paragraphs or larger segments to ignore small detections (few n-grams that do not constitute much of the text), and to display coherent plagiarism cases to the end users of PD systems. In the proposed approach, we use component-based comparison; hence, granularity can measure the ability of PD to combine subsequent components into sections. We simplify the mathematical computation of G_{plag} as follows:

$$G_{plag} = \frac{N\rho_{detected} : \rho_{detected} \subseteq \rho_{annotated}}{N\rho_{annotated}} \quad (27)$$

where $N\rho_{detected}$ denotes the number of detected plagiarism cases that are TP, i.e., intersects (partially or totally) with one of the plagiarism cases annotated in *q*'s annotation file, and $N\rho_{annotated}$ is the total number of annotated plagiarism cases in *q*. Finally, PD evaluation measures are combined into a single score called $Score_{plag}$, that allows us to compare different PD methods (Potthast et al., 2010) as follows:

$$Score_{plag} = \frac{F_{harmonic}}{\log_2(1 + G_{plag})} \quad (28)$$

Statistical Analysis

Before statistical analysis, the results were first obtained using *k*-fold cross-validation rather than leave-one-out cross-validation because the latter is much more time-expensive; especially with more than 18,000 plagiarism samples contained in the dataset. A stratified 10-fold cross-validation was performed in three stages of this study: component-weight factors comparison, candidate retrieval, and plagiarism screening. The dataset contains two parts: source publications and suspicious publications (i.e., documents under investigation). The suspicious publications part was equally stratified before the cross-validation was performed, whereas the source publications part was kept for comparison with each fold. In this setting, we obtained 10 folds with equal number of documents and equivalent number of plagiarism cases as well as plagiarism-free documents. We repeated the experiment 10 times. Each time, we fine-tuned the algorithm on 9 folds such that better precision and recall could be obtained, while the remaining fold was used to report the final results. The same folds were used across all CR and PD algorithms.

The final performance of the component-factor comparison and candidate retrieval (i.e., first two stages) was generally assessed by precision, recall, and *F*-measure—averaged over the 10 folds—on ground-truth annotated data (see the Dataset section for more details), and by comparison with two previous baselines (see the Baselines section). Further, the performance of plagiarism screening (i.e., the third stage) was evaluated on a ground-truth annotated data using averaged precision, recall, *F*-measure, and granularity over the 10 folds. Plagiarism detection measures were furthermore combined into a single metric called $Score_{plag}$ shown in Equation 28, and compared with three plagiarism detection methods proposed in the literature (see Baselines section).

We examined the statistical significance of the proposed approach using *t* hypothesis testing as follows. We set a null hypothesis—flat-based CR and structural-based CR perform equally (i.e., the true mean difference is zero)—and worked to gather evidence against this null hypothesis. Because cross-validation yielded 10-fold pairs of $F_{harmonic}$ results during the flat-based and structural-based CR algorithms, a paired-*t*-test (Leech, Barrett, & Morgan, 2008) was used to reject/do not reject the null hypothesis. Similarly, flat-based PD and structural-based PD yielded 10-fold pairs of $Score_{plag}$; therefore, a paired-*t*-test can be used to test the null hypothesis—flat-based PD and structural-based PD have no significant difference.

To carry out a paired-*t*-test on *k*-fold cross-validation results ($k = 10$), we calculated the difference in the results obtained from each algorithm in each fold as $d_i = x_i - y_i$, where $i = 1, 2, \ldots, k$. In the CR stage, x_i refers to $F_{harmonic}$ value obtained from the flat-based algorithm on the *i*th fold, and y_i refers to $F_{harmonic}$ value obtained from the structural-based algorithm on the *i*th fold. In the PD stage, x_i refers to $Score_{plag}$ value obtained from the flat-based PD algorithm

TABLE 5. Results from proposed component-weight factors organized into five categories; IGF-based, spread-based, depth-based, combined and qualitative.

Structural factors	P_{micro}	R_{micro}	$F_{harmonic}$	*SD*
IGF-based				
f_1	0.3760	0.2541	0.3002	0.0386 (0.0141)
f_2	0.3374	0.0815	0.1298	0.0450 (0.0836)
Spread-based				
f_3	0.4006	0.9288	0.5596	0.0150 (0.0198)
f_4	0.3914	0.9533	0.5548	0.0125 (0.0185)
Depth-based				
f_7	0.4057	0.8701	0.5531	0.0166 (**0.0080**)
f_8	0.4007	0.8913	0.5527	0.0119 (0.0179)
Combined				
f_5	0.3802	0.2449	0.2940	0.0481 (0.0156)
f_6	0.3405	0.0821	0.1308	0.0448 (0.0830)
f_9	0.4115	0.6973	0.5163	0.0310 (**0.0095**)
f_{10}	0.4053	0.7656	0.5297	0.0246 (0.0123)
Qualitative				
f_{low}	0.3949	0.6371	0.4868	0.0192 (**0.0046**)

Note. The first three columns give the mean precision, mean recall and mean *F*-measure over all folds. The last column shows the standard deviation over 10 runs of cross-validation in each component-weight factor, as well as the standard deviation of the means over all component-weight factors, in parentheses.

on the ith fold, and y_i refers to the $Score_{plag}$ value obtained from the proposed PD algorithm on the ith fold. Then, we computed the mean difference $\bar{d} = (\sum_{i=1}^{k} d_i)/k$, and used that to determine the standard deviation of the mean differences across the k folds $\alpha = \sqrt{\sum_{i=1}^{k} (d_i - \bar{d})^2/(k-1)}$. We used α to compute the standard error $SE(\bar{d}) = \alpha/\sqrt{k}$, and the t-statistic $T = \bar{d}/SE(\bar{d})$, which under the null hypothesis, follows a normal distribution with $k-1$ degrees of freedom. Using the Student's t distribution table[11], we compared T to the t_{k-1} distribution to obtain the probability p-value, which answers the alternative hypothesis—structural-based algorithms make significant changes in comparison to flat-based algorithms.

Before we conduct a paired-t-test for CR and PD methods, it is important to compare different component-weight factors proposed in this study. For this purpose, we used a statistical test called an analysis of variance (ANOVA; Leech et al., 2008), which generalizes the t-test in a way that examines whether or not the means of several algorithms are equivalent (paired-t-test compares two algorithms). Therefore, we set a null hypothesis that "All component-weight factors f_1, $f_2, \ldots, f_10$, and f_{low} perform equally," and work to gather evidence towards the alternative hypothesis—At least one of the means of the component-weight factors is significantly different.

Results and Discussion

This section demonstrates the results obtained from different structural-based and flat-based algorithms. In the first subsection, we present the results obtained from different component-weight factors proposed in this article. In the second subsection, we show the results obtained from the

structural CR approach denoted as STRUC-TFIDF, and compare them with the results obtained from typical flat-based CR baselines denoted as FLAT-SHING and FLAT-TFIDF. We discuss the results from the proposed component-based overlapping measure in PD stage referred to as STRUC-C2C, and compare them with three PD methods from the literature denoted as FLAT W5G, W8G, and S2S in the third subsection.

Comparing Component-Weight Factors

First we aimed to define the effects of different structural component-weight factors on the retrieval of candidate publications. In this regard, 10 component-weight factors as well as the f_{low} factor were compared and evaluated on a fraction of the dataset. Factors that show good retrieval results will be considered for the whole dataset in the remaining experiments. For this purpose, we used the same query publications (15% of the total queries) that were used previously to setup different parameters for the baselines.

Table 5 presents the results obtained when we ran the CR stage using the modified term weighting algorithm stated by Equations 16–18. For each factor, 10-fold cross-validation data was used for testing, and the means and standard deviations are reported in the table. The results are assessed using precision, recall, and F-measure averaged over the 10 folds. The first important observation is that *Spread*-based, *Depth*-based factors and their combinations yielded superior results, especially recall, to *IGF*-based factors. We also found that the proposed qualitative factor denoted as f_{low} was significantly effective compared with the results obtained from typical *IGF*-based factors.

Figure 9 demonstrates precision-recall curves displayed for one factor from each category (a), for combined factors (b), and for all factors (c). Unlike recall, precision results (i.e., the ability of the method to avoid false-positives) are obviously comparable (<0.5) in all component-weight factors.
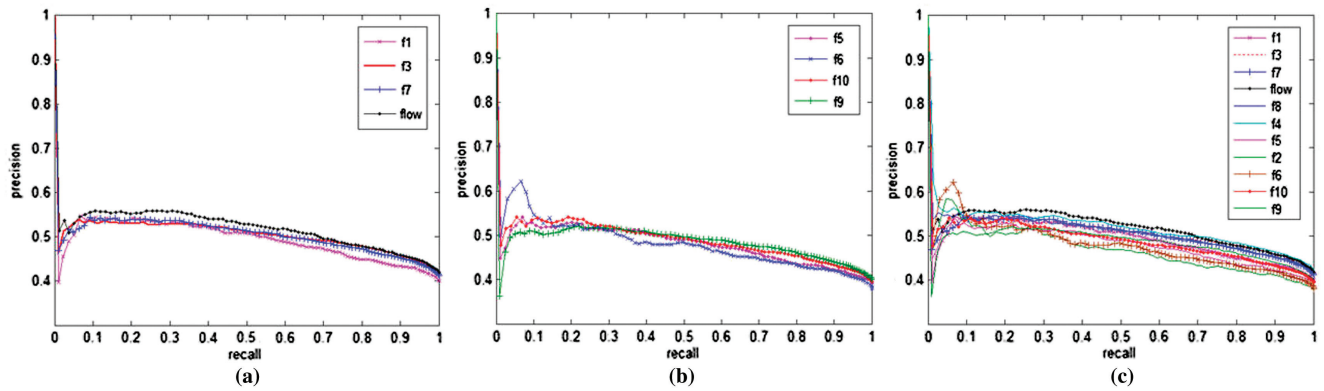
FIG. 9.   Precision-recall curve for proposed component-weight factors. (a) A component-weight factor is chosen from each category; (b) combined component-weight factors; (c) all component-weight factors. [Color figure can be viewed in the online version, which is available at wileyonlinelibrary.com.]

In fact, we consider all source publications annotated as "sourceFile" in the annotation files as true-positives and false-positives, which enables us to avoid manual judgment as "relevant" or "nonrelevant" as in typical IR systems. However, two scenarios may cause such comparable, somewhat low precision in the retrieval results. The first scenario is that many publications share global similarity (i.e., true-positive) with the query, but do not contain plagiarism, and may be retrieved during this stage. The second scenario is that there are about 6,581 texts (see Table 4) taken from different source publications, but cited properly (i.e., with citation evidence) and more likely to be retrieved at this stage. Similar texts, but with proper citation evidence and publications that share global similarity, but do not contain plagiarism are not bounded by the annotation files, and are considered as false-positives at this stage. Nonetheless, documents that bypass the CR stage, but do not contain plagiarism are very likely to be thwarted during the next stage because the PD algorithm is designed to flag parts that have only true plagiarism cases.

Table 5 also shows that the standard deviation for each component-weight factor over 10 runs of cross-validation is relatively small, which apparently means that there is a slight variance between the results gained from each fold. This indicates that the dataset used for this experiment is equally stratified and the algorithm behaves in a similar way over the 10 runs. On the other hand, the standard deviation of the means over all component-weight factors, shown in parentheses, indicates the performance variance among different component-weight factors. The highest standard deviation indicates that the results may possibly be unreliable, as it is the case with f_2 and f_6, while the lowest mean over all factors occurs with f_{low}, f_7, and f_9, respectively.

We used the simplest form of ANOVA to verify whether or not there exists a significant difference between the means of the component-weight factors (see last paragraph in the Statistical Analysis section). We applied a "conservative test" that was not likely to reject the hypothesis, as the results are shown partly[12] in Table 6. As can be seen, the ANOVA test reveals a statistical difference for each factor. We conclude

_____
[12]Please e-mail Dr. Alzahrani to obtain the full list of ANOVA results.

that *Spread*-based factors, which incorporate structural frequency information of index terms within the components, and *Depth*-based factors, which investigate the normalized term frequency within the generic classes, are significantly different from *IGF*-based factors. More precisely, the performance of *IGF*-based factors (and their combinations) are slightly worse compared to others. On the other hand, the ANOVA fails most of the time to reveal a statistically reliable difference between the means of *Depth*-based factors and *Spread*-based factors. There is no evidence which *Depth*-based, *Spread*-based, or combined factor performs the best [other nonparametric tests, such as Kruskal-Wallis (Kruskal & Wallis, 1952) may be used]. For simplicity, factors that have stable performance (i.e., least standard deviations) of the means over all component-weight factors are considered in the remaining experiments. The selected factors include f_7, f_9, and f_{low}.

Comparing Structural and Flat CR

We next explored the structural-based retrieval approach denoted as STRUC-TFIDF, and compared it with typical flat-based retrieval baselines denoted as FLAT-SHING and FLAT-TFIDF, respectively. Table 7 presents the mean precision, recall, and $F_{harmonic}$, and the standard deviation obtained from 10-fold cross-validation data. Because the CR stage focuses on the retrieval of a candidate pool that possibly contains the sources of plagiarism, recall results get more attention in this stage. As we can see in the table, recall results suggest that STRUC-TFIDF with selected component-weight factors was consistently superior to both flat baselines. The highest recall was obtained with the f_7 factor. However, the precision of STRUC-TFIDF was less than in FLAT-SHING and comparable with FLAT-TFIDF. We explained two possible scenarios in the previous section that may cause documents to be retrieved when they have global similarity with the query document, but with no evidence of the occurrence of plagiarism.

Table 8 presents the statistical results of a paired-*t*-test wherein we set a null hypothesis that FLAT-SHING and STRUC-TFIDF perform equally. Table 9 shows the statistical

TABLE 6. Statistical results from ANOVA parametric test for different component-weight factors.

Test critical value =	17.87035467	Decision	CI for difference between factors	
Test statistics for f_1 VS f_9	253.0382881	Reject Ho	0.1685	To 0.2904
Test statistics for f_2 VS f_9	768.3699832	Reject Ho	0.3389	To 0.4608
Test statistics for f_3 VS f_9	4.292587601	Do not reject Ho	−0.0311	To 0.0909
Test statistics for f_4 VS f_9	266.9650333	Reject Ho	0.1747	To 0.2967
Test statistics for f_5 VS f_9	764.5747512	Reject Ho	0.3379	To 0.4598
Test statistics for f_6 VS f_9	2.646130141	Do not reject Ho	−0.0375	To 0.0844
Test statistics for f_7 VS f_9	2.546652093	Do not reject Ho	−0.0380	To 0.0840
Test statistics for f_8 VS f_9	0.857092199	Do not reject Ho	−0.0476	To 0.0743
Test statistics for f_9 VS f_{10}	8.817925903	Do not reject Ho	−0.0181	To 0.1038
. . .	. . .			
Hypothesis =	Means of all component-weight factors are equal			
Alternative hypothesis =	At least one mean is significantly different			
Alpha level =	0.05	$df = 9$		
Total mean =	0.405306043	Final decision:		
F Test statistic =	295.2802436	reject hypothesis		
Critical F Value =	1.985594964			
P Value =	8.58017E-63			

Note. CI = Confidence interval. The top part shows the t-critical value, t statistics for f_{10} with other component-weight factors, and the decision taken is either to reject the hypothesis if t-statistic $>$ t-critical, or do not reject; otherwise. The last column shows the confidence interval (C.I.) between different pairs of factors. All other factors are compared similarly but not shown due to space limitations. The bottom part of the table summarizes the ANOVA test statistics between 11 component-weight factors wherein F-statistic is greater than the F-critical with 9 degrees of freedom.

TABLE 7. Results from CR using FLAT-SHING and FLAT-TFIDF baselines, and proposed STRUC-TFIDF with three component-weight factors.

CR	P_{micro}	R_{micro}	$F_{harmonic}$	SD
FLAT-SHING	0.5660	0.6492	0.5992	0.0345 (0.0059)
FLAT-TFIDF	0.3707	0.5948	0.4541	0.0459 (0.0047)
STRUC-TFIDF				
f_7	0.3949	0.8807	0.5450	0.0139 (0.0005)
f_9	0.3980	0.7641	0.5232	0.0149 (0.0000)
f_{low}	0.3887	0.6663	0.4908	0.0171 (0.0010)

Note. The first three columns show the mean precision, mean recall and mean F-measure over all folds. The last column gives standard deviation over 10 runs of cross-validation in each algorithm, as well as standard deviation of the means over all CR methods, in parentheses.

TABLE 8. Statistical results from paired t-test of FLAT-SHING CR and STRUC-TFIDF CR.

Hypothesis Test for the difference of two means: dependent sample (Paired t-test)

Statistics	Two-tailed test	FLAT-SHING	STRUC-TFIDF	Difference
Hypothesis =	FLAT-SHING = STRUC-TFIDF	0.5868	0.5552	0.03154
Alternative hypothesis =	FLAT-SHING ≠ STRUC-TFIDF	0.6078	0.5546	0.05315
Alpha level =	0.05	0.6267	0.5406	0.08609
Mean differences =	0.0542	0.6020	0.5628	0.03922
SD =	0.0426	0.5996	0.5378	0.06187
Sample size =	10	0.6057	0.5492	0.05654
Test t statistic =	4.0219	0.6328	0.5392	0.09358
t-Critical value =	±2.2622	0.6119	0.5354	0.07648
P-Value =	0.0030	0.6100	0.5164	0.09362
Decision =	Reject hypothesis	0.5082	0.5585	−0.05026
Confidence interval for paired difference:				
Confidence level =	0.95			
Confidence interval =	0.0237< μd <0.08466			

TABLE 9. Statistical results from paired *t*-test of FLAT-TFIDF CR and STRUC-TFIDF CR.

Hypothesis Test for the difference of two means: dependent sample (Paired *t*-test)

Statistics	Two-tailed test	FLAT-TFIDF	STRUC-TFIDF	Difference
Hypothesis =	FLAT-TFIDF = STRUC-TFIDF	0.4514	0.5552	−0.1038
Alternative hypothesis =	FLAT-TFIDF ≠ STRUC-TFIDF	0.4299	0.5546	−0.1248
Alpha level =	0.05	0.4369	0.5406	−0.1037
Mean differences =	−0.0909	0.4346	0.5628	−0.1281
SD =	0.0431	0.4483	0.5378	−0.0894
Sample size =	10	0.4414	0.5492	−0.1078
Test *t* statistic =	6.6770	0.4477	0.5392	−0.0914
t-Critical value =	±2.2622	0.4336	0.5354	−0.1018
P-Value =	0.0001	0.4337	0.5164	−0.0827
Decision =	Reject hypothesis	0.5831	0.5585	0.02461
Confidence interval for paired difference:				
Confidence level =	0.95			
Confidence Interval =	$-0.1217 < \mu d < -0.06011$			

TABLE 10. Results for PD using W5G, W8G, S2S in both baselines, and proposed STRUC-C2C with three selected component-weight factors and two candidate selection strategies.

CR	PD	P_{plag}	R_{plag}	G_{plag}	$Score_{plag}$	SD
FLAT-SHING	W5G	0.8178	0.3792	1.6770	0.3672	0.1507 (0.0542)
	W8G	0.8109	0.4203	1.7071	0.3907	0.1930 (0.0740)
	S2S	0.7261	0.5490	3.1420	0.3114	0.2810 (0.0335)
FLAT-TFIDF	W5G	0.8171	0.3965	1.6959	0.3733	0.1281 (0.0300)
	W8G	0.8284	0.4152	1.8188	0.3766	0.2752 (0.0625)
	S2S	0.7212	0.5793	3.2061	0.3130	0.4619 (0.0462)
STRUC-TFIDF	C2C –f_7	0.8633	0.6125	1.0664	0.6821	0.0140 (0.0217)
(Same-factor selection)	C2C –f_9	0.8666	0.6145	1.0644	0.6855	0.0154 (0.0308)
	C2C –f_{low}	0.8668	0.6387	1.1169	0.6740	0.1186 (0.0196)
STRUC-TFIDF	C2C –f_7	0.8664	0.6426	1.1161	0.6789	0.1352 (0.0567)
(Optimum-factor selection)	C2C –f_9	0.8663	0.6424	1.0620	0.6996	0.0125 (0.0600)
	C2C –f_{low}	0.8663	0.6424	1.1751	0.6598	0.2501 (0.0363)

Note. The first four columns give the mean precision, recall, granularity and score of plagiarism over all folds. The last column shows the standard deviation over 10 runs of cross-validation in each approach, as well as the standard deviation of the means over all approaches, in parentheses.

results of the same test runs over paired samples in FLAT-TFIDF and STRUC-TFIDF. Notice that we choose STRUC-TFIDF with f_7 to run both statistical tests. We conclude from both tables that a paired-*t*-test reveals a statistically reliable difference (i.e., rejecting the null hypothesis) between the mean *F*-measure in structural-based CR method and flat-based CR methods. Therefore, different structures of scientific publications could make significant changes into terms weighting according to which components the term has occurred.

Results for Plagiarism Screening

This section covers the experimental work that we carried out during the PD stage. A component-based overlapping approach with the "significance" factor Δ denoted as STRUC-C2C was implemented based on Equation 19, and compared with flat PD methods referred to as W5G, W8G, and S2S. Results are assessed using *precision,* which

indicates the ability of the plagiarism system to avoid false detections, and *recall,* which refers to the ability of the algorithm to reveal different plagiarism instances. Table 10 shows the averaged precision, recall, granularity, and $Score_{plag}$ obtained from the 10-fold cross-validation data using different PD methods. The results are discussed in the following paragraphs.

Flat-based PD methods obtained positive precision results. This is not a surprising outcome because we know that these methods have been widely implemented in PD research (Alzahrani et al., 2011), and have obtained good detection results principally for "cut and paste" plagiarism (Barrón-Cedeño, Basile, Degli Esposti, & Rosso, 2010; Barrón-Cedeño & Rosso, 2009). W5G and W8G obtained slightly better results than S2S, which means that n-gram-based methods have more of a chance to catch plagiarism when it is committed by combining/splitting sentences/phrases than in statement-by-statement methods. Recall results indicate that W5G, W8G, and S2S were able to

TABLE 11. Statistical results from paired t-test of FLAT-TFIDF (W5G) PD and STRUC-TFIDF (C2C-f_7) PD in the optimum-factor selection strategy.

Hypothesis test for the difference of two means: dependent sample (Paired t-test)

Statistics	Two-tailed test	FLAT-TFIDF (W5G)	STRUC-TFIDF (C2C-f_7)	Difference
Hypothesis =	FLAT-TFIDF (W5G) = STRUC-TFIDF (C2C-f_7)	0.3749	0.6451	−0.2703
Alternative hypothesis =	FLAT-TFIDF (W5G) ≠ STRUC-TFIDF (C2C-f_7)	0.3761	0.6717	−0.2956
Alpha level =	0.05	0.3598	0.6814	−0.3215
Mean differences =	0.0542	0.3440	0.6817	−0.3377
SD =	0.0426	0.3613	0.6723	−0.3110
Sample size =	10	0.3559	0.6721	−0.3162
Test t statistic =	4.0219	0.3828	0.6739	−0.2911
t-Critical value =	±2.2622	0.3757	0.6911	−0.3154
P-Value =	0.0030	0.3516	0.7072	−0.3556
Decision =	Reject hypothesis	0.4510	0.7244	−0.2733
Confidence Interval for paired difference:				
Confidence level =	0.95			
Confidence Interval =	$-0.1217 < \mu d < -0.06011$			

detect 30–50% of the annotated plagiarism cases. It corroborates that such methods are not designed to detect obfuscated plagiarism cases as discussed in the literature (Alzahrani et al., 2011).

Structural PD methods, on the other hand, obtained positive results. It is important to specify a candidate pool from available candidate sets that were obtained with three component-weight factors. We proposed two strategies to select a candidate set before applying the STRUC-C2C PD approach, which in turn, was implemented using four chosen component-weight factors. The first strategy is called *same-factor selection*, which proposes the use of the candidates list obtained from the same factor f_x that will be used with STRUC-C2C. The other strategy is called *optimum-factor selection*, which makes use of the factor that "recall" the majority of candidates. The optimum factor was obtained by f_7 (see Table 7).

Table 10 shows that the results from both selection strategies are comparable. An important observation is that $Score_{plag}$ results from STRUC-C2C are in general better than flat-based methods because of higher precision and recall, and near-optimal granularity gained by this method. Superior precision and recall results in the proposed STRUC-C2C PD approach may be because (a) weighting of structural components in scientific publications helps to avoid parts that are not important to the detection algorithm such as copyrights, acknowledgments, and introductory sentences that are acceptable to be redundant between papers (Figure 1 exemplifies such texts in the early parts of this paper); (b) using citation evidence in the PD algorithm dismisses texts with proper citation evidence from the results; and (c) we used the rich candidate pool obtained from structural-based CR stage rather than the one obtained from flat-based CR. On the other hand, the near-optimal granularity in our proposed method might be related to the comparison of components rather than n-grams or sentences. To illustrate, we compare structural components and if two components acheive high overlapping scores, the plagiarism is bounded

and recorded at once. W5G, W8G, and S2S methods, however, compare smaller pieces of texts and they need extra post-processing steps to combine adjacent n-grams or sentences into paragraphs, for example.

To prove that our results from structural PD methods are statistically significant compared with flat PD methods, we used a paired-t-test to compare the $Score_{plag}$ obtained from 10-fold cross-validation runs. We compared STRUC-TFIDF (C2C-f_7) PD in the optimum-factor selection strategy with the FLAT-TFIDF (W5G) PD approach as shown in Table 11. Statistical results from a two-tailed test showed that the alternative hypothesis, i.e., there is a significant difference between the two approaches, is true. The C2C-f_7 PD approach obtained significant results in comparison with the W5G PD approach: a t-statistic = 4.02, t-critical = ±2.2622, and $p = 0.003$ with confidence level = 0.95 and 9 degrees of freedom.

A Case Study

An empirical case study of the system's response well illustrates significant plagiarism screening and the structural similarity index. Figure 10 simulates part of the detection results for a query publication chosen from our dataset. Such results may also be obtained from existing antiplagiarism tools such as Turnitin, Docoloc, and CrossCheck. It is important to note that marked plagiarism cases were inserted as dummy text, and they do not reflect the real text within that article. As the figure shows, OSI is relatively high because it includes all sequences of matching words even though the journal's copyrights and cited texts should not be highlighted as plagiarism. Moreover, cases are marked according to their appearance in the article under investigation; the user determines the seriousness of the plagiarism.

Unlike the "crude" checking model, Figure 11 visualizes plagiarism for the same article, but incorporates structural information and citation evidence into the results. The proposed framework pinpoints cases that are significant to a
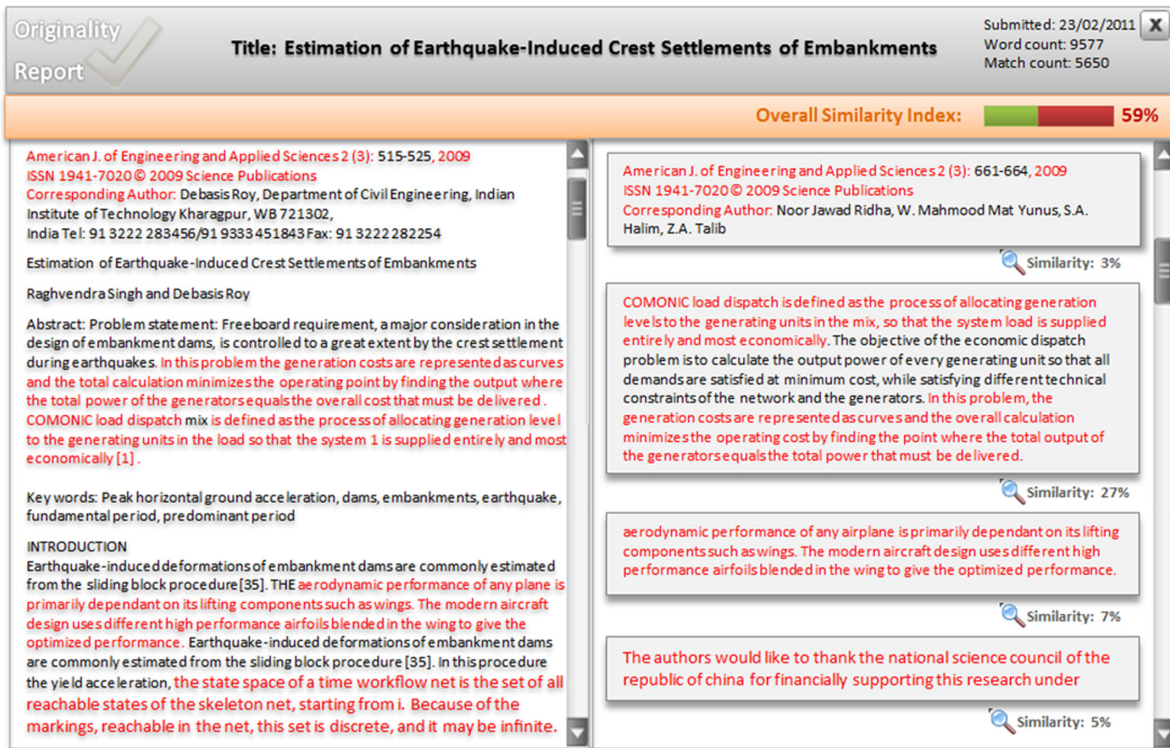
FIG. 10. Plagiarism visual screening results using traditional word n-gram matching methods, and their effects on similarity index (SI), and overall similarity index (OSI). [Color figure can be viewed in the online version, which is available at wileyonlinelibrary.com.]
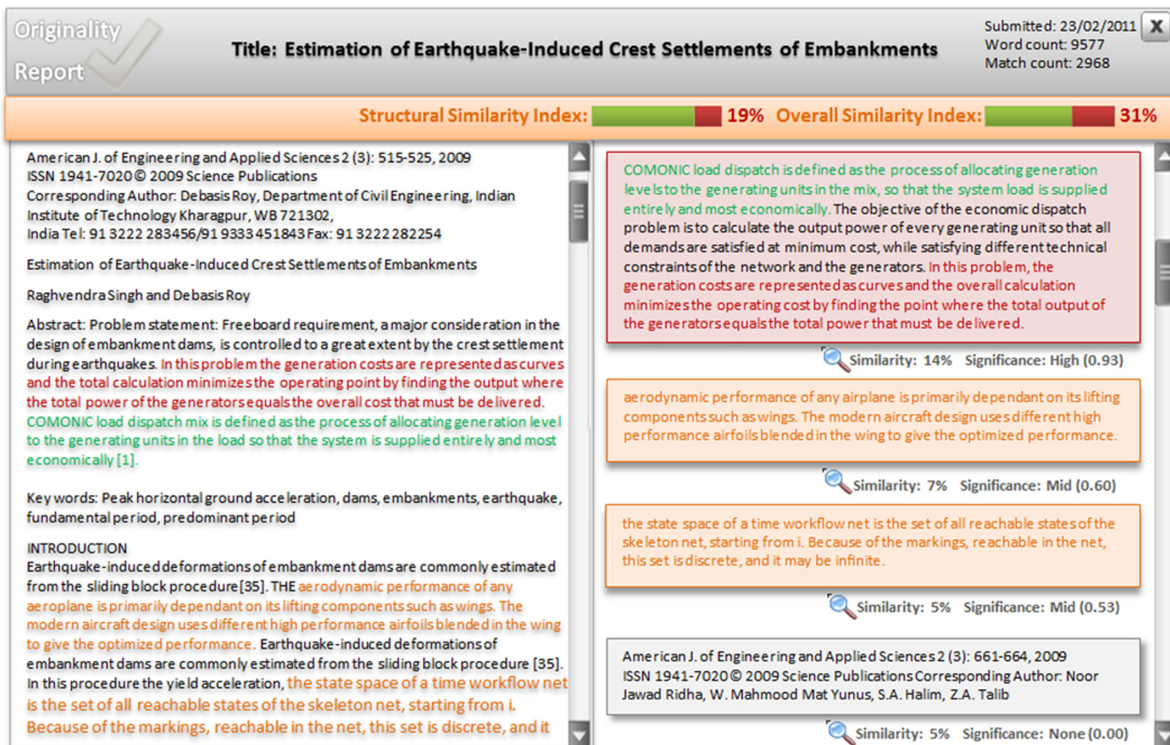

FIG. 11. Plagiarism visual screening results using component-based structural weighting methods and their effects on similarity index (SI) and overall similarity index (OSI), highlighting the significant cases within different components and structural similarity index (SSI). [Color figure can be viewed in the online version, which is available at wileyonlinelibrary.com.]

decision maker. Structural component-weight factors have influenced the conclusion about plagiarism such that some cases of little importance are suppressed, while significant cases are ranked and displayed accordingly. For instance, a plagiarism case in the *Abstract* was given a significance of 0.93 using a *Depth*-based factor in contrast to other cases in the *Introduction*. Further, using citation evidence for quoted and paraphrased texts (shown in green color) makes the computation of *OSI* more realistic based on real and significant plagiarism cases than in flat-based PD.

Moreover, the *SSI* in Figure 11 illustrates the amount of the work that is taken from other sources without acknowledgment. The difference between *OSI* and *SSI* is that the former works based on a word-matching count, whereas the latter is based on structural components that have been found to be plagiarized (in full or in part) from other sources. It also indicates that 19% of the semantic parts that constitute the article were taken from elsewhere; the quality of the work can thus be judged accordingly.

Conclusions and Future Work

Current plagiarism detectors are text-matching systems that do not determine plagiarism precisely. These detectors may yield a high similarity index for a submitted article because every matching, but not necessarily plagiarized text is included in the calculation of *SI* and *OSI*. Text that is acceptably redundant and text that is cited properly are all incorporated into the *SI*, and the final determination of plagiarism is left up to the user. Higher values of *SI* and *OSI*, however, may indicate unexpected false detections. Therefore, the approach presented in this article seeks to develop similarity indices to reflect true plagiarism cases and to filter out parts with proper citation evidence. The proposed approach reduces the human input in validating the detection results, and gives more trust to a plagiarism detector. Our results show that using structural information influences the performance of plagiarism detection by (a) better retrieval of candidate publications than near-duplicate and TFIDF retrieval methods; (b) better precision, recall, and granularity of plagiarism detection results than flat-based PD techniques; and (c) marking the degree of significance and ranking significant cases in different structural components accordingly. Our future work will include combining structural information and semantic-based PD methods to go beyond "component" plagiarism as there are other types of plagiarism with different semantic variations (e.g., plagiarism by paraphrasing a text or summarizing an idea).

Acknowledgments

References

Alzahrani, S., & Salim, N. (2009, August). Statement-based fuzzy-set IR versus fingerprints matching for plagiarism detection in Arabic documents. Paper presented at the 5th Postgraduate Annual Research Seminar (PARS '09), Johor Bahru, Malaysia.

Alzahrani, S., & Salim, N. (2010, September). Fuzzy semantic-based string similarity for extrinsic plagiarism detection: Lab report for PAN at CLEF '10. Paper presented at the 4th International Workshop on Uncovering Plagiarism, Authorship, and Social Software Misuse (PAN-10) in conjunction with CLEF '10, Padua, Italy.

Alzahrani, S.M., Salim, N., & Abraham, A. (2011). Understanding plagiarism linguistic patterns, textual features and detection methods. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, PP(99).

Anjewierden, A. (2001, September). AIDAS: Incremental logical structure discovery in PDF documents. Paper presented at the 6th International Conference on Document Analysis and Recognition (ICDAR '01), Seattle, WA.

Barrón-Cedeño, A., Basile, C., Degli Esposti, M., & Rosso, P. (2010). Word length n-grams for text re-use detection. In Proceedings of the 10th International Conference on Computational Linguistics and Intelligent Text Processing (CICLing-2010) (pp. 687–699). Heidelberg: Springer-Verlag.

Barrón-Cedeño, A., & Rosso, P. (2009). On automatic plagiarism detection based on n-grams comparison. In Proceedings of the 31th European Conference on IR Research on Advances in Information Retrieval (ECIR '09) (pp. 696–700). Heidelberg: Springer-Verlag.

Basile, C., Benedetto, D., Caglioti, E., Cristadoro, G., & Esposti, M.D. (2009, September). A plagiarism detection procedure in three steps: Selection, matches and "squares." Paper presented at the XXV Annual Congress of the Spanish Society for Natural Language Processing 2009 (SEPLN '09), Donostia, Spain.

Binwahlan, M.S., Salim, N., & Suanmali, L. (2009). Fuzzy swarm based text summarization. Journal of Computer Science, 5(5), 338–346.

Bounhas, I., & Slimani, Y. (2010, March). A hierarchical approach for semi-structured document indexing and terminology extraction. Paper presented at the International Conference on Information Retrieval and Knowledge Management (CAMP '10), Selangor, Malaysia.

Burget, R. (2007). Automatic document structure detection for data integration. In W. Abramowicz (Ed.), Business information systems (Vol. 4439, pp. 391–397). Heidelberg: Springer-Verlag.

Butakov, S., & Scherbinin, V. (2009). The toolbox for local and global plagiarism detection. Computers and Education, 52(4), 781–788.

Ceska, Z. (2008). Plagiarism detection based on singular value decomposition. In Lecture notes in computer science (Vol. 5221 LNAI, pp. 108–119). Heidelberg: Springer-Verlag.

Chen, C. (1999). Visualising semantic spaces and author co-citation networks in digital libraries. Information Processing & Management, 35(3), 401–420.

Chen, C., Yang, K., Chen, C., & Ho, J. (2010). BibPro: A citation parser based on sequence alignment. IEEE Transactions on Knowledge and Data Engineering, PP(99), 1.

Chris, H.Q.D. (1999). A similarity-based probability model for latent semantic indexing. In Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. New York: ACM Press.

Chudamani, K.S., & Ilamathi Maran, S. (2004, January). Citation analysis as a tool for determining contextually relevant information in libraries in the digital environment. Paper presented at the Conference on Digital Information Exchange: Pathways to Build Global Information Society, Chennai, India.

Clough, P., & Stevenson, M. (2011). Developing a corpus of plagiarised short answers. Language Resources and Evaluation, 45(1), 5–24.

Councill, I.G., Giles, C.L., & Kan, M.-Y. (2008, May). ParsCit: An open-source CRF reference string parsing package. Paper presented at the Language Resources and Evaluation Conference (LREC '08), Marrakesh, Morocco.

Daniel, R.W., & Mike, S.J. (2004). Sentence-based natural language plagiarism detection. ACM Journal on Educational Resources in Computing, 4(4), 2.

de Moura, E.S., Fernandes, D., Ribeiro-Neto, B., da Silva, A.S., & Gonçalves, M.A. (2010). Using structural information to improve search in Web collections. Journal of the American Society for Information Science and Technology, 61(12), 2503–2513.

Elhadi, M., & Al-Tobi, A. (2009, November). Duplicate detection in documents and webpages using improved longest common subsequence and documents syntactical structures. Paper presented at the 4th International Conference on Computer Sciences and Convergence Information Technology, Seoul, Korea.

Fiala, D. (2010). Mining citation information from CiteSeer data. Scientometrics, 86(3), 553–562.

Grozea, C., Gehl, C., & Popescu, M. (2009, September). ENCOPLOT: Pairwise sequence matching in linear time applied to plagiarism detection. Paper presented at the XXV Annual Congress of the Spanish Society for Natural Language Processing 2009 (SEPLN '09), Donostia, Spain.

Hagen, L., Harald, L., Ngen, & Petra Saskia, B. (2004, July). Text type structure and logical document structure. Paper presented at the 2004 ACL Workshop on Discourse Annotation, Barcelona, Spain.

Heintze, N. (1996, November). Scalable document fingerprinting. Paper presented at the 2nd USENIX Workshop on Electronic Commerce, Oakland, CA.

Jones, M. (2009, September). Back-translation: The latest form of plagiarism. Paper presented at the 4th Asia Pacific Conference on Educational Integrity (4APCEI), Wollongong, Australia.

Karl, O.J. (2008). Practical issues for academics using the Turnitin plagiarism detection software. In Proceedings of the 9th International Conference on Computer Systems and Technologies and Workshop for PhD Students in Computing. New York: ACM Press.

Kasprzak, J., Brandejs, M., & Křipač, M. (2009, September). Finding plagiarism by evaluating document similarities. Paper presented at the XXV Annual Congress of the Spanish Society for Natural Language Processing 2009 (SEPLN '09), Donostia, Spain.

Kruskal, W.H., & Wallis, W.A. (1952). Use of ranks in one-criterion variance analysis. Journal of the American Statistical Association, 47(260), 583–621.

Larsen, B., & Ingwersen, P. (2006). Using citations for ranking in digital libraries. In Proceedings of the 6th ACM/IEEE Joint Conference on Digital Libraries (p. 370). New York: ACM Press.

Lee, K.H., Choy, Y.C., & Cho, S.B. (2003). Logical structure analysis and generation for structured documents: A syntactic approach. IEEE Transactions on Knowledge and Data Engineering, 15(5), 1277–1294.

Lee, M., & Chen, T. (2010). Visualising intellectual structure of ubiquitous computing. In Lecture notes in computer science (including subseries Lecture notes in artificial intelligence and Lecture notes in bioinformatics) (Vol. 6232 LNAI, pp. 261–272). Heidelberg: Springer-Verlag.

Leech, N.L., Barrett, K.C., & Morgan, G.A. (2008). SPSS for Intermediate statistics use and interpretation (3rd ed.). Mahwah, NJ: Erlbaum.

Li, Z., & Ng, W.K. (2004). WICCAP: From semi-structured data to structured data. In Proceedings of the 11th IEEE International Conference and Workshop on the Engineering of Computer-Based Systems (ECBS '04) (pp. 86–93). Piscataway, NJ: IEEE.

Luong, M.-T., Nguyen, T.D., & Kan, M.-Y. (2010). Logical structure recovery in scholarly articles with rich document features. International Journal of Digital Library Systems (IJDLS), 1(4), 1–23.

Manning, C.D., Raghavan, P., & Schǐtze, H. (2008). Web search basics: Near-duplicates and shingling. In Introduction to Information Retrieval (pp. 437–442). New York: Cambridge University Press.

Manning, C.D., Raghavan, P., & Schütze, H. (2009). Scoring, term weighting and the vector space model. In Introduction to Information Retrieval (pp. 109–133). New York: Cambridge University Press.

Marques Pereira, R.A., Molinari, A., & Pasi, G. (2005). Contextual weighted representations and indexing models for the retrieval of HTML documents. Soft Computing, 9(7), 481–492.

Marteau, P.-F., Ménier, G., & Popovici, E. (2006, October). Weighted naïve Bayes model for semi-structured document categorization. Paper presented at the 1st International Conference on Multidisciplinary Information Sciences and Technologies (InSciT2006), Merida, Spain.

Meddings, K. (2010). Credit where credit's due: Plagiarism screening in scholarly publishing. Learned Publishing, 23, 5–8.

Murugesan, M., Jiang, W., Clifton, C., Si, L., & Vaidya, J. (2010). Efficient privacy-preserving similar document detection. The VLDB Journal, 19(4), 457–475.

Nguyen, T.D., & Luong, M.-T. (2010, July). WINGNUS: Keyphrase extraction utilizing document logical structure. Paper presented at the ACL 2010 Workshop on Evaluation Exercises on Semantic Evaluation (SemEval 2010), Uppsala, Sweden.

Pasi, G. (2002). Flexible information retrieval: Some research trends. Mathware and Soft Computing, IX(9–1), 107–121.

Pentz, E. (2006). CrossRef at the crossroads. Learned Publishing, 19(4), 250–258.

Potthast, M., Stein, B., Barron-Cedeno, A., & Rosso, P. (2010, August). An evaluation framework for plagiarism detection. Paper presented at the 23rd International Conference on Computational Linguistics (COLING 2010), Beijing, China.

Potthast, M., Stein, B., Eiselt, A., Barrón-Cedeño, A., & Rosso, P. (2009, September). Overview of the 1st international competition on plagiarism detection. Paper presented at the XXV Annual Congress of the Spanish Society for Natural Language Processing 2009 (SEPLN '09), Donostia, Spain.

Ratté, S., Njomgue, W., & Ménard, P. (2007). Highlighting document's structure. World Academy of Science, Engineering and Technology, 31, 34–36.

Scherbinin, V., & Butakov, S. (2009, September). Using Microsoft SQL server platform for plagiarism detection. Paper presented at the XXV Annual Congress of the Spanish Society for Natural Language Processing 2009 (SEPLN '09), Donostia, Spain.

Schleimer, S., Wilkerson, D., & Aiken, A. (2003). Winnowing: Local algorithms for document fingerprinting. In Proceedings of the ACM SIGMOD International Conference on Management of Data (pp. 76–85). New York: ACM Press.

Siddharthan, A., & Teufel, S. (2007, April). Whose idea was this, and why does it matter? Attributing scientific work to citations. Paper presented at the 2007 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL HLT 2007), New York, NY.

Stoffel, A., Spretke, D., Kinnemann, H., & Keim, D.A. (2010). Enhancing document structure analysis using visual analytics. In Proceedings of the 2010 ACM Symposium on Applied Computing (pp. 8–12). New York: ACM Press.

Su, Z., Ahn, B.R., Eom, K.Y., Kang, M.K., Kim, J.P., & Kim, M.K. (2008, December). Plagiarism detection using the Levenshtein distance and Smith-Waterman algorithm. Paper presented at the 3rd International Conference on Innovative Computing Information and Control (ICICIC '08), Dalian, Liaoning.

Teufel, S. (1999). Argumentative zoning: Information extraction from scientific articles (Unpublished doctoral dissertation). University of Edinburgh, Scotland.

Teufel, S., & Moens, M. (2000, October). What's yours and what's mine: Determining intellectual attribution in scientific text. Paper presented at the 2000 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora, Hong Kong.

Teufel, S., & Moens, M. (2002). Summarizing scientific articles: Experiments with relevance and rhetorical status. Computational Linguistics, 28(4), 409–445.

Van, T.-T., & Beigbeder, M. (2007). Citation-based methods for personalized search in digital libraries. In Lecture notes in computer science (LNCS 4832) (pp. 362–373). Heidelberg: Springer-Verlag.

Wang, L., Jin, Y., Wang, Z., Wang, Y., & Gao, K. (2005). A new model of document structure analysis. In Fuzzy systems and knowledge discovery (Vol. 3614, pp. 658–666). Heidelberg: Springer-Varlag.

Lüngen, H., Bärenfänger, M., Hilbert, M., Lobin, H., & Puskas, C. (2010). Discourse relations and document structure. In A. Witt, & D. Metzing (Eds.), Linguistic modeling of information and markup languages (Vol. 40, pp. 97–123). Heidelberg: Springer-Verlag.

Yerra, R., & Ng, Y.-K. (2005). A sentence-based copy detection approach for web documents. In L. Wang & Y. Jin (Eds.), Fuzzy systems and knowledge discovery (pp. 557–570). Heidelberg: Springer-Verlag.

Zechner, M., Muhr, M., Kern, R., & Granitzer, M. (2009, September). External and intrinsic plagiarism detection using vector space models. Paper presented at the XXV Annual Congress of the Spanish Society for Natural Language Processing 2009 (SEPLN '09), Donostia, Spain.

Zhang, H. (2010). CrossCheck: An effective tool for detecting plagiarism. Learned Publishing, 23, 9–14.

Zhang, K., Wu, G., & Li, J. (2006, May). Logical structure based semantic relationship extraction from semi-structured documents. Paper presented at the 15th International Conference on World Wide Web, Edinburgh, Scotland.