Pattern Recognition 44 (2011) 471–487

Contents lists available at ScienceDirect

Pattern Recognition

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

A coarse-to-fine framework to efficiently thwart plagiarism

Haijun Zhang, Tommy W.S. Chow*

Department of Electronic Engineering, City University of Hong Kong, 83 Tat Chee Avenue, Kowloon, Hong Kong

ARTICLE INFO

Article history:
Received 11 January 2010
Received in revised form
1 June 2010
Accepted 18 August 2010

Keywords:
Document retrieval
Plagiarism detection
EMD
Multilevel matching

ABSTRACT

This paper presents a systematic framework using multilevel matching approach for plagiarism detection (PD). A multilevel structure, i.e. document–paragraph–sentence, is used to represent each document. In document and paragraph level, we use traditional dimensionality reduction technique to project high dimensional histograms into latent semantic space. The Earth Mover's Distance (EMD), instead of exhaustive matching, is employed to retrieve relevant documents, which enables us to markedly shrink the searching domain. Two PD algorithms are designed and implemented to efficiently flag the suspected plagiarized document sources. We conduct extensive experimental verifications including document retrieval, PD, the study of the effects of parameters, and the empirical study of the system response. The results corroborate that the proposed approach is accurate and computationally efficient for performing PD.

1. Introduction

The Internet has, undoubtedly, become an indispensable component of our daily life ranging from restaurant booking to technology research. The online fashion is, however, posing a severe challenge to textual intellectual property because the Internet and computer technology have made disseminating knowledge across the world facile. People can search, copy, save, and reuse online sources in ease. The most flagrant instance of plagiarism is to copy a document from another source without any kind of modifications. But this type of plagiarism is easy to be identified using the plagiarism detection (PD) system. Less obvious examples occur when people integrate an existing work into their work. They attempt to bypass the detection system by conducting substitution of words or sentences within an already existing document, or pasting some phrases from an outside source into a new document. Cut-and-paste PD, at present, has become a growing concern in education system. One of the difficulties of efficiently detecting plagiarism is to search the source with speedy query response because people may copy from one of millions of documents in the Internet, where each document usually involves thousands of words.

Existing techniques for anti-plagiarism include fingerprinting, a method developed specifically for detecting co-derivatives, and ranking, a method developed for document retrieval. Hoad and Zobel [1] investigated the performance of these techniques and demonstrated that the ranking method is superior to the fingerprinting method. Chow et al. [2] also reported promising results by using the ranking approach. Following this line, this paper presents a coarse-to-fine framework to detect plagiarism using multilevel matching (MLM). The proposed approach delivers a number of desirable features that include generality, robustness, and efficiency. Concretely, these features can be described as follows:

- The generality refers to the multilevel-structured document representation and its encoding features. We use document–paragraph–sentence structure to form a coarse-to-fine representation of each document. In document and paragraph level, principal component analysis (PCA), a traditional dimensionality reduction tool, is used to capture the hidden latent semantic topics. Instead of PCA, any other latent semantic analysis or dimensionality reduction techniques can be incorporated into this scheme.
- The proposed system is robust due to its use of signature matching. The signature in document and paragraph level is constructed by involving the length and the histograms of terms in each component. Each sentence is featured by using the index number of each term that indicates the presence of the corresponding term in vocabulary. In this signature encoding, we do not consider the sequence of terms in a sentence, which is reasonable because plagiarists strive to substitute words in each sentence or reorganize the sentence structure so as to bypass the PD system.
- Document modeling and its applications are notoriously computational intensive due to their involvement of at least thousands of words. Our proposed system is based on depth

* Corresponding author.
E-mail address: eetchow@cityu.edu.hk (T.W.S. Chow).

0031-3203/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.patcog.2010.08.023

matching by using a coarse-to-fine strategy to filter out the unpromising searching domain. This pruning capability enables us to bring large computational efficiency. Therefore, the proposed approach can be used for a large dataset and practical online applications.

The main contributions of this paper are threefold. First, we propose a multilevel-structured document representation together with encoding features. Second, we investigate MLM approaches, i.e. histogram based MLM (MLMH) and signature based MLM (MLMS), for relevant document retrieval (DR). Third, two detection algorithms are implemented by setting appropriate thresholds such that undesirable paths are pruned in advance during multilevel matching process.

The remaining sections of this paper are organized as follows. A brief overview of document modeling and its applications are presented in Section 2. The relationship of PD versus document categorization (or classification, and or clustering) and DR are also discussed, respectively. Section 3 introduces a multilevel-structured document representation together with the document segmentation, dimensionality reduction, and feature encoding. Document segmentation is done using HTML tags. In Section 4, we discuss various document retrieval approaches based on histograms and signatures. Two detection algorithms are implemented in Section 5. We conduct extensive experimental verifications in Section 6. Section 7 lists the discussion based on observed results and proposes the system framework from a practical viewpoint. Finally, Section 8 ends the paper with conclusion and future work propositions.

2. Related work

This section briefly reviews the previous work, as partially covered by Tommy et al. [2] and our recent work [3,4]. It involves document modeling and its applications (e.g. categorization, retrieval, and PD). We also discuss the relationship between PD and other applications.

2.1. Document modeling

The earliest work on document modeling is vector space model (VSM) [5], which usually uses the *tf–idf* approach for term weighting. A basic vocabulary of "terms" (or "words") is firstly constructed for feature description. The term frequency (*tf*) is the number of occurrences of each term. The inverse document frequency (*idf*) is a function of the number of document where a term took place. A term weighted vector is then constructed for each document using *tf* and *idf*. Similarity between the two documents is measured using 'cosine' distance or any other distance functions VSM reduces arbitrary length of the term vector in each document to fixed length. But a lengthy vector is required for describing the frequency information of terms because the number of words involved is usually huge. This causes a significant increase in computational burden, making the VSM impractical for large corpus. In addition, VSM reveals little statistical property of a document because of only using low level document features (i.e. term frequency). To overcome these shortcomings, researchers have proposed several dimensionality reduction methods using low dimensional latent representations to capture document semantics. Latent semantic indexing (LSI) [6], an extension from VSM, maps documents associated with terms to a latent space representation by performing a linear projection, singular value decomposition (SVD), to compress the feature vector of the VSM model into low dimension. A step forward in probabilistic models is probabilistic latent semantic indexing (PLSI) [7] that defines a proper generative model of data to model each word in a document as a sample from a mixture distribution and develop factor representations for mixture components. A brief overview of other probabilistic models, such as latent Dirichlet allocation (LDA) [8], exponential family harmonium (EFH) [9], and rate adapting Poisson (RAP) model [10], can be referred to [4]. The above models, however, are only based on the 'bag of words' assumption. Thus some useful semantic information would have been lost because two documents containing similar term frequencies may be contextually different. In our recent work [3], in order to include more semantics in document representation, we proposed a new document representation with multiple features by using different graphs. Term frequency (TF) and term connection frequency (TCF) are extracted from each document by employing different feature extraction methods. We then developed a dual wing harmonium (DWH) model to generate the latent representations of documents by jointly modeling multiple features [4].

2.2. Applications

There is a large body of literature on document applications that mainly include DR, categorization (or classification, and or clustering) (DC), and PD.

DR refers to finding similar documents for a given query. The query can range from a full description of a document to a few keywords. Most of the widely used retrieval approaches are keywords based searching methods, e.g., www.google.com, where untrained users provide a few keywords to the search engine for finding the relevant documents in a returned list. Another type of DR is to use a query document. Using an entire document as a query performs well in improving retrieval accuracy, but it is computationally demanding compared to the keywords based methods. VSM [5], LSI [6], PLSI [7], LDA [8], EFH [9], RAP [10], and DWH [4] can be directly applied to the latter type of DR because they are derived from term vector as the basic feature unit. On the other hand, many existing retrieval systems are based on traditional Boolean logic model [11], where documents and users' queries are hypothetically represented by precise index terms. To overcome the restriction of expressing the inexact and uncertain knowledge of human beings, many fuzzy information retrieval systems are proposed (see a brief review on this respect in [12]). Bear et al. [13] suggested using information extraction (IE) to improve the performance of retrieval systems. According to Gaizauskas and Wilks [14], "Document Retrieval" retrieves relevant documents from a dataset. IE, on the other hand, extracts relevant information from the retrieved documents. Currently IE has been developed through a series of Message Understanding Conferences (MUC). Readers are referred to [14] for a brief review on IE. Self-organizing map (SOM), a versatile unsupervised neural network, has been also studied to perform speedy DR due to its computational efficiency [15]. SOM was used to speed up the retrieval process by automatically formulizing a document map [16]. A flexible multi-layer SOM [2,17] was developed to process generic tree-structured data, for example, document data, that can be represented hierarchically by document features as document–pages–paragraphs. Other than SOM, researchers also suggested that clustering of large document dataset can be used for speeding up the retrieval system [18]. Considering the patterns of the query term occurrence in a document, Park et al. [19] proposed spectral-based DR approaches. It was suggested that documents containing query terms, all of which follow a similar positional pattern, are supposed to be more relevant. These approaches, however, are only applicable to the case of keywords as a query.

DC, apart from DR, has also become important in organizing the massive amount of document data. According to Yang and Pedersen [20], DC is an issue of automatically assigning predefined categories to free text documents. A major characteristic, or difficulty, of DC problems is the high dimensionality of the feature space. Various feature selection methods were investigated [20]. A brief overview of typical document clustering techniques, e.g. agglomerative hierarchical clustering and K-means, can be referred to [21]. On the other hand, Sebastiani [22] conducted a detailed survey examining the main approaches, which have been taken towards automatic text categorization within the general machine learning paradigm. Fuketa et al. [23] introduced a field understanding method using field association words for DC. Others used bigrams [24] or term association rules [25] to enhance the DC accuracy. Neural networks [26] and support vector machines (SVM) [27] were also studied for DC. Other techniques, such as rough sets [28] and statistical approaches [29], have been reported for DC. Artificial intelligent methods [30] have also been investigated for DC and visualization applications. Recently hierarchical DC has also become an important issue in knowledge and content management. It enables interactive data-views at different levels of granularity. Fung et al. [31] introduced a hierarchical document clustering architecture based on association rule mining. It was suggested to some common words, called frequent itemsets, to produce a hierarchical topic tree for clusters. Thus clustered documents can be browsed according to the increasing specificity of topics. Bang et al. [32] then proposed an improved version of k-NN classifier aided by concept based thesauri, which are used to entail structured document categories into hierarchies.

The earlier works on PD only focus on the detection of student plagiarism in computer programs [33]. Compared to DR and DC, related work reported in the literature on PD is scarce, albeit the fact that many commercial tools, e.g., Turnitin,[1] WordCHECK,[2] EVE2,[3] WCopyFind,[4] are available. COPS [34] and SCAM [35] are two well-known PD approaches. The basis of these systems lies in partitioning each document into smaller parts and registering them against a hash table. Thus all documents in the dataset are hashed in a modular form. A query document is then similarly divided into corresponding parts that are orderly matched in the hash table. Monostori et al. [36] introduced a string-matching algorithm using a suffix tree to speed up the PD process. But it is only applicable for a word-by-word matching over a small dataset because suffix trees do not scale well. Scaling up to a large dataset, Heintze [37] proposed a fingerprinting based system. A document fingerprint is firstly formed in character level instead of word level. The detection speed of this approach is largely enhanced but at the expense of losing certain semantic information. Finkel et al. [38] reported a web-accessible text registry system. It extracts a small signature from each registered document. The total matching time increases linearly with the size of the dataset. Thus the overall PD time is rather long when it is required to handle a lengthy document. Recently, Meyer zu Eißen et al. [39] proposed a method to generate suspected passages from a single document in terms of changes in writing style. Those passages can be used for preliminary online search or human inspection. Barrón-Cedeño and Rosso [40] then carried out extensive experiments based on exhaustive comparison of reference and suspicious word-level n-grams. They reported that lower value of n (except unigram) performs better for detection of rewording plagiarism. According to Si et al. [41], most existing PD approaches employ an exhaustive sentence-based matching to compare a suspected plagiarized document with all registered documents. Apparently, this kind of approache is impractical because the number of documents is getting enormous with time and some of the sizes of the documents are also large. The security level of this approach is another major concern because a plagiarized document can easily bypass the detection by including minor modifications on each sentence. Thus a section-based tree-structured document representation was introduced for performing PD [41]. It matches each node using a depth-first search to avoid unnecessary comparison. But this method relies on raw feature units, i.e. keywords, to represent each section, which makes it still unable to handle a large dataset. Recently Tommy et al. [2] proposed using multilayer SOM (MLSOM) for anti-plagiarism. They show that MSOM is computationally efficient to handle a large dataset. But the MLSOM approach requires appropriate preprocessing training time to eliminate the mapping bias of SOM. Moreover, it only uses paragraphs as the lowest feature units to perform PD.

2.3. Relationship

PD, as an important application of document modeling, is different from other applications, i.e. DR and DC. It, on the other hand, much relates to them.

One document is considered to be plagiarized by another document due to simple cut–paste manipulations or minor alternations. The similarity between two documents in PD is more obvious than that in DR and DC. Thus the semantics similarity is more appropriate to be used in DR and DC. DR refers to provide a retrieval list, whereas PD tends to have a document candidate list attached with an overlapping rate, from which users can realize if necessary to further check the returned documents manually. Another option for PD is to directly inform users whether the query document is plagiarized or not. The latter option also can be regarded as a binary categorization problem. But the source category has only very few, maybe one, documents because plagiarists tend to copy some parts of one or two source documents. Thus PD is different from DC because the dataset in DC usually includes many categories, each of which consists of a great number of documents. Thus PD can be seen as a step forward to matching compared to DR and DC. DC can be implemented by either supervised or unsupervised learning. PD, however, has no prior knowledge against the class label. Thus it is, to some extent, only suitable for unsupervised learning. For a large dataset users can first retrieve much relevant documents or automatically assign the query into a relevant category to build a small local dataset, and then perform PD in the narrowed searching space. This somehow motivates the idea of constructing a coarse-to-fine framework to thwart plagiarism. We summarize the relationship (or difference) of PD against DR and DC as follows:

- The condition on the similarity between two documents in PD is stricter than that in DR and DC.
- The evaluation of the performance of PD is based on either a ranking list or a binary decision compared to DR.
- PD can be regarded as a binary categorization problem compared to DC.
- PD has no any prior knowledge compared to DC.
- PD can be regarded as a step forward from DR and DC.

3. Document representation

This section involves the document preprocess and the overall feature extraction procedures. It includes the detailed steps to

[1] http://www.turnitin.com
[2] http://www.wordchecksystems.com
[3] http://www.canexus.com/eve
[4] http://plagiarism.phys.virginia.edu/Wsoftware.html

partition a document into paragraphs and further partition each paragraph into sentences for HTML format documents (see Section 3.1), building two different sizes of vocabularies (see Section 3.2), and construction of multilevel representation (see Section 3.3).

3.1. Document segmentation

We propose a hierarchical multilevel representation of documents that contain text content only. To extract the multilevel structure, a document is segmented into paragraphs that are further segmented into sentences. We only considered HTML documents in this paper and developed a Java platform to implement that kind of segmentation. In HTML format document, we can use HTML tags to easily identify paragraphs that are further partitioned into sentences by marking periods. Before document segmentation, we first filter out the formatted text that appears within the HTML tags. The text is not accounted for in word counts or document features.

The overall document partitioning process can be summarized as follows:

1. Partition the document into blocks using the HTML tags: "<p>", "<br\>", "<li>", "</td>", etc.
2. Merge the subsequent blocks to form a new paragraph until the total number of words of the merged blocks exceeds a paragraph threshold (set at 50 in this paper). The new block is merged with the previous paragraph if the total number of words in a paragraph exceeds the minimum threshold (set at 30).
3. Partition each generated paragraph into sentences using the tag "\.".

For HTML documents, it is noted that there is no rule for minimum/maximum number of words for paragraphs. But the use of a threshold of word counts still enables us to flexibly control the number of paragraphs in each document, and makes the blocks that contain only a few words (e.g. titles) attached to the real paragraph blocks. In this way, we build a hierarchical multilevel structure (or tree structure) to describe the semantic information from global data-view to local data-view. Thus the document contents are structured in a 'document→paragraphs→sentences' hierarchy. This is a simple way to generate a hierarchical structure. It can be further improved by a finer segmentation such as 'document→sections→pages→paragraphs→sentences'. But it needs a more complex algorithm to facilitate this kind of segmentation.

3.2. Vocabulary construction

The main text contents are firstly separated from HTML tags. We then extract words from all the documents in a dataset and apply stemming to each word. Stems are used as basic features instead of original words. Thus 'program', 'programs', and 'programming' are all considered the same word. We remove the stop words (set of common word like 'a', 'the', 'are', etc.) and store the stemmed words together with the information of the term frequency f_t (the frequency of a word in all documents) and document frequency f_d (the number of documents a word appears). In order to form a histogram vector for each document, we need to construct a word vocabulary each histogram vector refers to. Based on the stored term frequency f_t and document frequency f_d information, we use a simple term-weighting measure, which is similar to the *tf–idf*, to calculate the weight of each word

$$W_t = \sqrt{f_t} \times idf, \tag{1}$$

where the inverse-document-frequency $idf = \log_2(N/f_d)$, and N is the total number of documents in the dataset. It is noted that this term-weighting measure can be replaced by other feature selection criteria [20]. The words are then sorted in a descending order according to the weights. Here, we construct two vocabularies denoted as V_1 and V_2, respectively. V_1 is used to form histogram vectors of document and paragraphs, whereas V_2 is used to form signatures of sentences (see Section 3.3). The first N_1 words are selected to construct the vocabulary V_1 and the first N_2 words are selected to construct the vocabulary V_2. The vocabulary V_1 is mainly used for DR (see Section 4), and the vocabulary V_2 is used for further sentence sorting (see Section 5). The vocabulary size N_2 is supposed to be much larger than N_1. According to our empirical study [3,4], using all the words in the dataset to construct the vocabulary V_1 is not necessarily expected to deliver the improvement of the DR accuracy because some words may be noisy features for some topics. Document modeling approaches [5–10], however, almost used all the words to form the basic histogram vectors for DR. Efficient feature selection for DR is still an open problem, which we leave to other researchers. We also conducted detailed experiments to evaluate the performance in terms of different options of the vocabulary sizes (see Sections 6.4.3 and 6.4.4).

3.3. Multilevel representation

After the vocabulary construction, we use the document segmentation procedures (see Section 3.1) to partition each document in the dataset and generate the multilevel representation in the form of the 'document→paragraphs→sentences' structure. The top level contains the histogram of a whole document and the 2nd level is used for paragraphs. Each element of the histogram indicates the number of times the corresponding word in the vocabulary V_1 appears in a document or a paragraph. The 3rd level used for sentences is different from the upper two layers. It uses the index number of words that are included in the vocabulary V_2, instead of using histograms, to indicate the presence/absence of words in a sentence. This architecture has two advantages: saving the storage space (computational efficiency) and improving the detection accuracy (accuracy efficiency) because it examines the document more locally.

Since the document level and paragraph level are mainly used for DR (see Section 4), we apply PCA, a well-known dimensionality reduction tool, to word histogram vector for the whole document and the segmented paragraphs. Here, PCA is employed to project higher dimensional data into lower dimensional latent semantic space without losing much statistical information. We first normalize the histogram vector $H_i^d = [h_t^d]$ $(t = 1, 2, \ldots, N_1)$ of the ith document

$$h_t^d = \frac{n_t}{\sum_{t=1}^{N_1} n_t} \times \log_2\left(\frac{N}{f_t^D}\right), \tag{2}$$

where n_t is the frequency of the tth word in the vocabulary, and f_t^D is the document frequency of the tth word. We then use the normalized histogram to construct the PCA projection matrix B. To save the computational burden, we apply PCA only in the document level. We have used the MATLAB tool [42] to compute the projection matrix. The compressed histogram vector $F_i^d = [f_u^d]$ $(u = 1, 2, \ldots, N_F)$ of the ith document is calculated as

follows:

$$F_i^d = H_i^d \times B, \tag{3}$$

where B is the projection matrix with dimension $N_1 \times N_F$ and N_F is the dimension of the projected feature. Thus the projected features in F_i^d are ordered according to their statistical importance. Likewise, it is similar to use projection matrix B to calculate the compressed histogram vector $F_{ij}^p = [f_v^p]$ ($v = 1, 2, \ldots, N_F$) of the jth paragraph in the ith document. We then save the projection base that is subsequently used to make the features of a new query document.

Fig. 1 illustrates the multilevel representation of a document that hierarchically describes the document content. Owing to this architecture, we can examine the document from coarseness to fineness. Nodes in the document level and paragraph level contain compressed features describing the frequency distribution of different words. It is noted that nodes at different levels include the same word frequency features, but they are extracted from different parts of the document. In terms of the DR applications (see Section 4), two documents having similar word histograms at root nodes can be completely different in semantics, because different spatial distributions of the same set of words can result in different meanings. This is reflected by the finer parts of the multilevel-structured data.

4. Document retrieval

In this section, we present the DR approaches that are different from most existing models, because our proposed methods add local information of a document into the retrieval process by taking advantage of multilevel representation. They also pave the way for the subsequent PD. Currently document modeling methods (e.g. VSM [5], LSI [6], PLSI [7], LDA [8], EFH [9], and RAP [10]) only consider the global information of a document (i.e. term frequency). Two documents, however, containing similar term frequencies may be contextually different when the spatial distribution of terms is very different. For example, *school*, *computer*, and *science* are very different when they appear in different parts of a document compared to the case of *school of computer science* that appear together. Therefore, only using term frequency information from the "bag of words" model is not the most effective way to account for contextual similarity, which includes the word inter-connections and spatial distribution of words throughout the document [2–4]. According to our coarse-to-fine framework, intuitively it is indispensable to include local information to firstly retrieve much relevant documents for a given query. This makes us form a more compact set of suspected plagiarized sources and proceed further matching (i.e. sentence matching). We firstly define a dissimilarity measure (see Section 4.1), and then develop two retrieval methodologies (see Sections 4.2 and 4.3). With a given query, we use the retrieval methods to sort the candidate documents in ascending order. We then build a set Φ, which is constructed by the first predefined number N_{ret} of documents in the retrieval list for the subsequent PD.

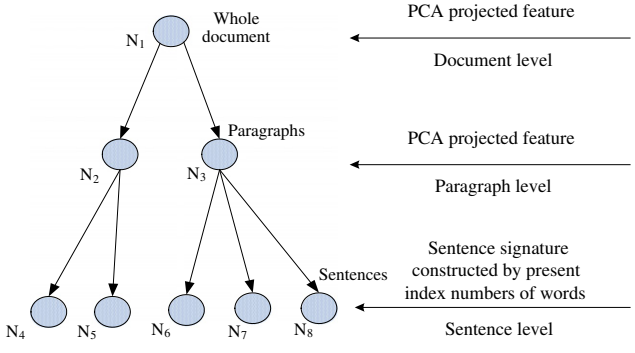

Fig. 1. Multilevel representation of a document.

4.1. Dissimilarity measure

In document applications, *cosine* distance is quite often to be used as a dissimilarity measure. In this paper, we define the following dissimilarity (or distance) measure (called exponential *cosine* distance) in our applications:

$$d(F^{Query}, F^{Candidate}) = 1 - e^{-f(F^{Query}, F^{Candidate})}, \tag{4}$$

$$f(F^{Query}, F^{Candidate}) = 1 - \frac{F^{Query} \cdot F^{Candidate}}{||F^{Query}|| \cdot ||F^{Candidate}||}, \tag{5}$$

where $\cdot$ indicates the dot product operation, F^{Query} represents the compressed PCA features of the whole query document or a paragraph of the query. Likewise, $F^{Candidate}$ denotes the features of a candidate document in the dataset. This measure has the property where for large distances it approaches to 1, whereas for small distances it saturates to 0. This exaggerates the effect that very similar paragraphs can have upon the distance fusion process (see Sections 4.2 and 4.3).

4.2. Histogram based retrieval (MLMH)

Given two documents with multilevel structured representation (see Fig. 1), the global distance H_{Global} can be directly achieved by matching nodes in document level, whilst the local distance H_{Local} can be measured by matching nodes in paragraph level. Documents, however, in the second level contain different numbers of paragraphs. According to our defined dissimilarity measure (see Section 4.1), we simply use the compressed histogram vector of each paragraph as a feature unit to compare each two paragraphs, and then normalize the overall distance

$$H_{Local} = \frac{\sum_{i=1}^{m} \sum_{j=1}^{n} d_{ij}^{Para}}{m+n}, \tag{6}$$

where d_{ij}^{Para} represents the distance between the ith paragraph of the query document and the jth paragraph of the candidate document, m and n denote the number of paragraphs the query document and the candidate document contain, respectively. To jointly include both the global and local information, it is straightforward to define a hybrid distance

$$H_{Hybrid} = \lambda H_{Global} + (1-\lambda) H_{Local}, \tag{7}$$

where λ ($\lambda \in [0,1]$) is the weight used to balance the importance of the global and local distance. Thus the system provides flexibility to change the value of λ to balance this hybrid measure according to the users' expectations. In this work, we also include the study of the effect of parameter λ (see Section 6.4.1).

4.3. Signature based retrieval (MLMS)

Histogram based retrieval only involves the selected terms as features, which are then compressed into latent semantic space

using PCA. It ignores the proportion of selected words to the total words a document or a paragraph contain. On this concern, we propose adding a weight parameter to the histogram of the whole document or each paragraph in the way to generate signature representation. It is straightforward to match signatures at the document level. For paragraph level, however, with the given weights, it is rather computationally demanding to compare documents in paragraph level if one simply relies on exhaustive matching among paragraphs. In addition, finding the optimal matching and appropriately synthesizing the distances of different paragraphs are the other two demanding issues. In this study, we model this problem as the concept of the Earth Mover's Distance (EMD) [43], which finds the optimal distances with the minimum cost of matching signatures by solving the linear programming problem.

4.3.1. Signature generation

Each document consists of two signatures: one is used for document level; the other is used for paragraph level. A signature with the size of K is defined as a set $S=\{s_k=(w_k,F_k)\}_{k=1}^{K}$, where F_k is the compressed PCA features of either a document or a paragraph, and w_k is the weight representing the information capacity delivered by these features. In fact, the size of the signature in document level is 1 because only one node is included. On the other hand, the size of the signature in paragraph level is the number of paragraphs of a document. In this work, the weight w_k is given by

$$w_k=\frac{T_k^s}{\sqrt{T_k^t}}, \tag{8}$$

where T_k^t denotes the total word counts of a document or a paragraph (i.e. the length of a document or a paragraph) and T_k^srepresents the total selected word counts of a document or a paragraph. Apparently, if all the words are selected, the weight w_k equals to the square root of the length of a document or a paragraph. Thus the weight is not only used to deliver the information capacity of selected features, it provides the length information of a document or a paragraph, which is not considered by VSM and other methods.

4.3.2. The Earth Mover's Distance (EMD)

The EMD was firstly introduced by Rubner et al. [43] to evaluate the dissimilarity between signatures, which contain clustered representations of histogram distributions. The EMD has been widely used in image analysis [43–45] because it supports the matching of different distributions, particularly partial matching. The whole set of transportation problem can be efficiently solved using a rather standard optimization technique. The solution of EMD problem is to minimize the amount of work required to transport products from m suppliers to n consumers. Computing EMD can be formalized as solving the following linear programming problem. Let $P=\{(w_i^p,p_i)\}_{i=1}^{m}$ be the supplier set, where w_i^p is the weight representing the amount of products; $Q=\{(w_j^q,q_j)\}_{j=1}^{n}$ be the consumer set, where w_j^q is the weight denoting the demand of products; and define the ground distance matrix as $D=[d_{ij}]_{m\times n}$. The objective is to find a flow matrix $F=[f_{ij}]_{m\times n}$, in which elements indicate the amount of products to be transported from one supplier to one consumer, to minimize the overall transportation cost as follows:

$$Cost^{opt}(P,Q)=\min_{F}\sum_{i=1}^{m}\sum_{j=1}^{n}f_{ij}d_{ij}, \tag{9}$$

which subjects to the following constraints:

$$f_{ij}\geq 0,\quad \text{where}\quad 1\leq i\leq m,\quad 1\leq j\leq n, \tag{10}$$

$$\sum_{j=1}^{n}f_{ij}\leq w_i^p,\quad \text{where}\quad 1\leq i\leq m, \tag{11}$$

$$\sum_{i=1}^{m}f_{ij}\leq w_j^q,\quad \text{where}\quad 1\leq j\leq n, \tag{12}$$

$$\sum_{i=1}^{m}\sum_{j=1}^{n}f_{ij}=\min\left(\sum_{i=1}^{m}w_i^p,\sum_{j=1}^{n}w_j^q\right). \tag{13}$$

The above formulation is a linear programming problem. Once it is solved and the flow matrix is obtained, the EMD is given by

$$EMD(P,Q)=\frac{\sum_{i=1}^{m}\sum_{j=1}^{n}f_{ij}d_{ij}}{\sum_{i=1}^{m}\sum_{j=1}^{n}f_{ij}}. \tag{14}$$

4.3.3. EMD between signatures

Based on the generated signatures (see Section 4.3.1), it is quite straightforward to use the EMD to evaluate the dissimilarity between document signatures because the EMD favors multi-featured sets. On the other hand, there is an analogy between the document comparison and transportation problem. A query document can be regarded as a supplier, whilst a candidate document in the dataset can be viewed as a potential consumer. Thus the issue of measuring the dissimilarity between documents can be regarded as minimizing an objective function (e.g. Eq. (9)).

Apparently, the EMD in document level representing the global distance is a special case of the transportation issue with only one supplier and one consumer, i.e. $m=1$and $n=1$. The global distance between two documents is given by

$$\begin{aligned}S_{Global}=EMD_{Doc}&=\frac{\sum_{i=1}^{m}\sum_{j=1}^{n}f_{ij}^{Doc}d_{ij}^{Doc}}{\sum_{i=1}^{m}\sum_{j=1}^{n}f_{ij}^{Doc}}\sum_{i=1}^{m}\sum_{j=1}^{n}f_{ij}^{Doc}d_{ij}^{Doc}\\&=d_{11}^{Doc}=d(F_{Doc}^{Query},F_{Doc}^{Candidate}),\end{aligned} \tag{15}$$

where $d(F_{Doc}^{Query},F_{Doc}^{Candidate})$ is a ground distance function defined as Eqs. (4) and (5) in document level. On the other hand, the EMD for computing the local distance in paragraph level is given by

$$S_{Local}=EMD_{Para}=\frac{\sum_{i=1}^{m}\sum_{j=1}^{n}f_{ij}^{Para}d_{ij}^{Para}}{\sum_{i=1}^{m}\sum_{j=1}^{n}f_{ij}^{Para}}, \tag{16}$$

$$d_{ij}^{Para}=d(F_{i,Para}^{Query},F_{j,Para}^{Candidate}), \tag{17}$$

where $d(F_{i,Para}^{Query},F_{j,Para}^{Candidate})$ is the distance function about the features of the ith paragraph of query document and the jth paragraph of candidate document. Thus we use a hybrid distance to synthesize the global and local dissimilarity as follows:

$$S_{Hybrid}=\lambda S_{Global}+(1-\lambda)S_{Local}. \tag{18}$$

Of course, the computational efficiency is a major concern of MLM using either histograms (MLMH) or signatures (MLMS) because a speedy query response is always expected by users. The computation complexity of comparing two documents based on MLMH is $O(mn)$, whilst for MLMS it can be solved in

$O(m^3 \log(m))$[43] if two documents have the same number of paragraphs (i.e. $m=n$). There is no explicit expression for the case that two documents have the different sizes of signatures. We empirically studied the time performance of MLMS for DR in the experiment section (see Section 6.5).

5. Plagiarism detection

After retrieving N_{ret} documents, which are regarded as the suspected plagiarized sources, we are now in the position to develop the PD algorithms. It is straightforward to sort the N_{ret} documents again in ascending order by further matching sentences and using distance fusion techniques, and eventually return a short list to the users. This method is called ranking based PD (see Section 5.1). Another way to implement PD is to set an offset value to make the binary decision on the presence of source document, which is an automatic PD (see Section 5.2).

5.1. Ranking based plagiarism detection

Apparently, the performance of the ranking based PD strongly relies on the ranking of source documents. The higher ranking is desirable because it implies the small distance (or dissimilarity) between the query and the source. The basic idea to conduct ranking based PD is to traverse the nodes in paragraph level of each candidate in set Φ and further perform matching in sentence level. Because we have calculated and saved the ground distance matrix d_{ij}^{Para} between the query and the candidate, once the distance, an element in d_{ij}^{Para}, is below a user-defined threshold σ_p, the child sentences are added to the list of the node to be further compared. The distance, on the other hand, is above the threshold, comparison does not explore the sub-sentences. The minimum and maximum of the elements of the ground distance matrix

$$d_{min}^{Para} = \min_{j=1,\ldots,n} (\min_{i=1,\ldots,m} (d_{ij}^{Para})) \quad \text{and} \quad d_{max}^{Para} = \max_{j=1,\ldots,n} (\max_{i=1,\ldots,m} (d_{ij}^{Para})). \tag{19}$$

are used to compute the threshold σ_p as

$$\sigma_p = d_{min}^{Para} + \varepsilon (d_{max}^{Para} - d_{min}^{Para}), \tag{20}$$

where ε ($\varepsilon \in [0,1]$) is a scaling parameter. It is noted that scaling up the parameter ε will result in traversing more paragraphs but at the expense of computational time due to more comparisons. From our empirical study (see Section 6.3), using a small scaling parameter fits better for detecting only a single plagiarized paragraph, whilst a large scaling parameter setting is more effective in detecting multiple plagiarized paragraphs.

Since in sentence level we use the index number of word, which indicates the presence of the corresponding term, in the large vocabulary V_2 (see Sections 3.2 and 3.3) as the signature S^{Sen} for each sentence node. In our applications, the signatures with the number of elements less than three were not considered. We then define the overlap rate between two sentences k (in the query) and l (in the candidate) being proceeded as

$$I_{kl}^{Overlap} = \frac{S_k^{Sen} \cap S_l^{Sen}}{S_k^{Sen} \cup S_l^{Sen}}, \tag{21}$$

and set another threshold τ ($\tau \in [0.5,1]$) to control which sentence will be regarded as a plagiarized part and counted in the distance fusion process.

The overall plagiarism distance between the query document D_q and a candidate document D_c ($D_c \in \Phi$) is defined as follows:

$$d_P(D_q, D_c) = \begin{cases} \dfrac{\sum_{i=1}^{N_q} \sum_{j=1}^{N_c} d_{ij}^{Sen} \delta(d_{ij}^{Para})}{\sum_{i=1}^{N_q} \sum_{j=1}^{N_c} \delta(d_{ij}^{Para})} & \text{if} \quad \sum_{i=1}^{N_q} \sum_{j=1}^{N_c} (d_{ij}^{Para}) \sum_{i=1}^{N_q} \sum_{j=1}^{N_c} \delta(d_{ij}^{Para}) \geq 1, \\ \kappa & \text{otherwise}, \end{cases} \tag{22}$$

$$\delta(d_{ij}^{Para}) = \begin{cases} 1 & \text{if} \quad (d_{ij}^{Para} \leq \sigma_p \quad \text{and} \quad d_{ij}^{Sen} < \kappa), \\ 0 & \text{otherwise}, \end{cases} \tag{23}$$

$$d_{ij}^{Sen} = \begin{cases} \dfrac{\sum_{k=1}^{N_{q,i}^{Sen}} \sum_{l=1}^{N_{c,j}^{Sen}} z(I_{kl}^{Overlap}) \zeta(I_{kl}^{Overlap})}{\sum_{k=1}^{N_{q,i}^{Sen}} \sum_{l=1}^{N_{c,j}^{Sen}} \zeta(I_{kl}^{Overlap})} & \text{if} \quad \sum_{k=1}^{N_{q,i}^{Sen}} \sum_{l=1}^{N_{c,j}^{Sen}} \zeta(I_{kl}^{Overlap}) \geq 1, \\ \kappa & \text{otherwise}, \end{cases} \tag{24}$$

$$\zeta(I_{kl}^{Overlap}) = \begin{cases} 1 & \text{if} \quad I_{kl}^{Overlap} \geq \tau, \\ 0 & \text{otherwise}, \end{cases} \tag{25}$$

$$z(I_{kl}^{Overlap}) = 1 - e^{-(1 - I_{kl}^{Overlap})}, \tag{26}$$

where N_q and N_c represent the number of paragraphs of the query document and candidate document, respectively, $N_{q,i}^{Sen}$ and $N_{c,j}^{Sen}$ denote the number of sentences of the ith paragraph P_i^q ($P_i^q \in D_q$) and the jth paragraph P_j^c ($P_j^c \in D_c$), respectively, and κ is a pre-defined large value for the distance function when no match is found. The complete ranking based PD algorithm is given by *Algorithm 1*.

5.2. Automatic plagiarism detection

Automatic PD refers to setting an offset value θ to make the binary decision on the presence of the source document. The similarity between the query and the source is usually much larger than that between the query and other documents if the query is really plagiarized. Thus the users can set an appropriate offset value to make the source flag up and automatically find the source. The procedure of the automatic PD is basically the same with the ranking based PD, except for the source document list we return. In addition, we use

$$z(I_{kl}^{Overlap}) = e^{-(1 - I_{kl}^{Overlap})} \tag{27}$$

instead of Eq. (26) because we use similarity as the measure rather than dissimilarity. If the similarity is larger than the offset value, the document is supposed to be the source. In this way, users will save efforts on further checking the ranking list.

Algorithm1:

Inputs: The query document D_q, the set Φ constructed by the most relevant N_{ret} documents, κ, ε, τ

Output: Ranking list

g ← 0
while g ≤ N_{ret} **do**
repeat
g ← g+1
Calculate the ground distance matrix $[d_{ij}^{Para}]_{N_q \times N_c^g}$ of the query D_q with N_q paragraphs and the candidate D_c^g with N_c^g paragraphs
Calculate the threshold σ_p according to Eq. (20)
Set the accumulative paragraph distance $d_{\text{accum}}^{\text{para}}(D_q, D_c^g) \leftarrow 0$
Set the number of paragraphs being counted in $N_{coun}^{para} \leftarrow 0$
for $i = 1$ to N_q **do**
for $j = 1$ to N_c^g **do**
if $d_{ij}^{Para} \le \sigma_p$ **then**
Set the accumulative sentence distance between the i-th paragraph and the j-th paragraph
$d_{\text{accum}}^{\text{sen}}(P_q^i, P_c^j) \leftarrow 0$
Set the number of sentences being counted in $N_{coun}^{sen} \leftarrow 0$
for $k = 1$ to $N_{q,i}^{Sen}$ **do**
for $l = 1$ to $N_{c,j}^{Sen}$ **do**
Calculate the overlap rate $I_{kl}^{Overlap}$ according to Eq. (21)
if $I_{kl}^{Overlap} \ge \tau$ **then**
Calculate $z(I_{kl}^{Overlap})$ according to Eq. (26)
$d_{\text{accum}}^{\text{sen}}(P_q^i, P_c^j) \leftarrow d_{\text{accum}}^{\text{sen}}(P_q^i, P_c^j) + z(I_{kl}^{Overlap})$
$N_{coun}^{sen} \leftarrow N_{coun}^{sen} + 1$
end if
end for
end for
if $N_{coun}^{sen} \ge 1$ **then**
$d_{ij}^{Sen} \leftarrow d_{accum}^{sen} / N_{coun}^{sen}$
else
$d_{ij}^{Sen} \leftarrow \kappa$
end if
if $d_{ij}^{Sen} \le \kappa$ **then**
$d_{\text{accum}}^{\text{para}}(D_q, D_c^g) \leftarrow d_{\text{accum}}^{\text{para}}(D_q, D_c^g) + d_{ij}^{Sen}$
$N_{coun}^{para} \leftarrow N_{coun}^{para} + 1$
end if
end if
end for
end for
if $N_{coun}^{para} \ge 1$ **then**
$d_P(D_q, D_c^g) \leftarrow d_{\text{accum}}^{\text{para}}(D_q, D_c^g) / N_{coun}^{para}$
else
$d_P(D_q, D_c^g) \leftarrow \kappa$
end if
end while
Sort $\left\{ d_P(D_q, D_c^1), \ ..., \ d_P(D_q, D_c^{N_{ret}}) \right\}$ in ascending order
Return the ranking list

In practice, it is not possible to provide an overall analytical expression in terms of computation complexity for the above PD algorithms because it depends on the actual document data and the parameters we used. For example, the larger the size of the document is, the more time the system expends during the comparison between the query and one document in the candidate set. On the other hand, the scaling parameter ε influences the query response. Scaling up ε, for example $\varepsilon=1$, obviously involves more paragraph comparisons. Increasing the distance threshold, however, results in bringing more noises into the distance fusion, which degrades the detection performance. Usually, the value of ε is lower than 0.5, which is better enough according to our experiments. We also conducted an empirical study on it in terms of the expense of the computational cost (see Section 6.5).

6. Experiments

In this section, we conduct the detailed experiments as efficiency verifications of our proposed PD approach. This section involves the dataset description (see Section 6.1), the performance of relevant DR (see Section 6.2), the performance of PD (see Section 6.3), the impact study of parameters (see Section 6.4), and the empirical study of computational time (see Section 6.5).

6.1. Dataset and experimental setup

We carried out large-scale experiments to show the performance of our proposed PD approach. We have collected a document dataset, "Html_CityU1", which consists of 26 categories [2–4]. Each category includes 400 documents, making a total number of 10,400 documents. In order to provide a more real-life testing platform, we established this database consisting of documents with the size ranged from few hundred words to over 20 thousand words. For each category, 400 documents were retrieved from "Google" using a set of keywords. Some of the keywords are shared among different categories, but the set of keywords for a category is different from that of other categories. The experiments were conducted in two parts. The first part is on the relevant DR for evaluating the retrieval performance, the second is to perform PD. The dataset was firstly split into a candidate set and a test set that is used for query. 1040 test documents were randomly selected from the 26 categories, i.e. 26×40. The remaining 9360 documents were used as a candidate pool. After PCA dimensionality reduction, the test set was used to verify the performance of the relevant DR. To make the plagiarized document set, we compiled four test sets with different plagiarism patterns. Each of them contains 78 documents, i.e. 3×26, among which 3 documents are from each category. The 1st set is that only a small part of a test document was exactly copied from a document of the candidate set called source document. The 2nd set is from the 1st plagiarism document set by changing sentences in the copied text. These minor modifications include change of tense, active to passive, few words, etc. The 3rd set characterizes to delete some sentences in the copied paragraphs, which usually happens in plagiarism. The 4th set is made by splitting exactly plagiarized parts and combining them into different parts in the copied text, which is characterized to be the multiple plagiarism patterns. The entire dataset can be downloaded from 'www.ee.cityu.edu.hk/~twschow/DataSetPD.rar' for other researchers.

6.2. Relevant document retrieval

We first evaluate the performance of the relevant DR aided by local information on the test set. To quantify the retrieval results, we used averaged precision and recall values for each query document. The precision and recall measures are defined as follows:

$$Precision = \frac{No.\ of\ correctly\ retrieved\ documents}{No.\ of\ retrieved\ documents}, \tag{28}$$

$$Recall = \frac{No.\ of\ correctly\ retrieved\ documents}{No.\ of\ documents\ in\ relevant\ category}. \tag{29}$$

We empirically set the parameters: $N_1=5000$, $N_F=100$, $\lambda=0.35$ for the hybrid MLMS (MLMS-Hybrid) and $\lambda=0$ for the MLMH (i.e. it does not include global information) in this subsection. This configuration of the above parameters was observed to deliver good performance. We include the study of these parameters' effects on the results (see Section 6.4). Based on above measures, we compare MLMS and MLMH to VSM, LSI, and our previous work (i.e. SOM – tf+tcf) [3]. The selection of VSM and LSI, two traditional methods, to compare is helpful to investigate the contributions of local information from documents for improving the retrieval performance. The details of VSM and LSI can be found in [5,6]. We investigated the VSM with the original *tf–idf* features without any data reduction techniques. LSI with only the *tf* features performed on 100 dimensional latent semantic representations. It is noted that LSI with the *tf.idf* features is the same with MLM-Global if we use the same term weighting scheme.

The retrieval results of different methods are summarized in Fig. 2 together with the corresponding numerically comparative results listed in Table 1. Fig. 2 shows the precision results when the retrieved documents, the most similar candidate documents from the dataset for each query, vary from 1 to 360. It is observed that MLM methods and LSI model deliver superior precision results than VSM. In general, MLMH, which only considers the local information from paragraph matching, out-performs other approaches when the number of retrieved documents is less than around 180. SOM, our previous work, also delivers satisfactory performance because it directly incorporates the effect of term connections into the document similarity but at the expense of rescanning the whole dataset to extract the evident term connections. Approaches (e.g. MLMH, MLMS-Local, and MLMS-Hybrid) including the local information deliver superior performance than those without the local information when only a few documents are retrieved. It is also noted that MLM-Local has better performance than MLM-Global when the number of retrieved documents is less than 200. On the other hand, owing

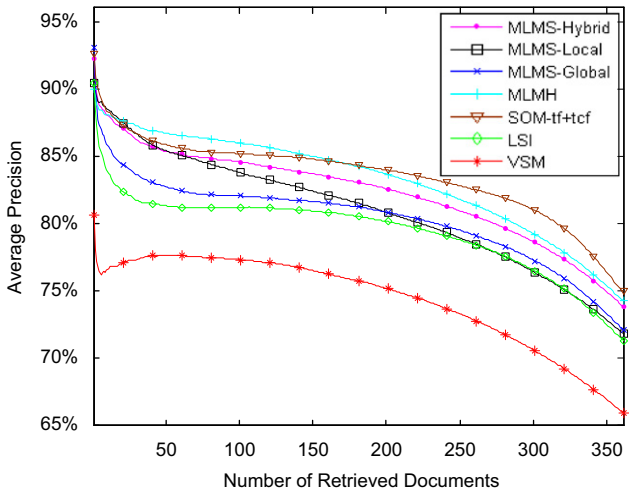

Fig. 2. Retrieval performance of different models.

Table 1
Comparative results of different retrieval methods.

Method	No. of retrieved documents							
	10	40	120	360	10	40	120	360
	Average precision (%)				Average recall (%)			
MLMS-Hybrid	88.05	85.75	84.23	73.93	2.45	9.53	28.08	73.93
MLMS-Local	88.50	85.88	83.25	71.96	2.46	9.54	27.75	71.96
MLMS-Global	85.91	83.06	81.90	72.22	2.39	9.23	27.30	72.22
MLMH	88.06	86.92	85.66	74.36	2.45	9.66	28.55	74.36
SOM − tf+tcf [3]	88.32	86.16	85.07	75.01	2.45	9.57	28.36	75.01
LSI	83.96	81.51	81.15	71.44	2.33	9.06	27.05	71.44
VSM	76.53	77.61	77.10	66.02	2.13	8.62	25.70	66.02

Table 2
Ranks statistics of source document for the 1st set.

	Rank statistics					
	AR	SD	MaxR	MinR	FDR (%)	CR
By retrieval						
MLMS-Hybrid	126.45	113.29	476	1	3.85	131.51
MLMS-Local	120.12	106.52	379	1	3.85	124.92
MLMS-Global	141.70	132.02	485	1	8.97	155.68
MLMH	139.88	117.92	478	1	3.85	145.48
By PD						
MLSOM [2]	8.72	23.53	107	1	35.90	13.60
$\varepsilon=0.1$	3.13	5.85	46	1	3.85	3.26

to integrating global information, the results delivered by MLM-Hybrid are better than those achieved by MLM-Local when the number of retrieved documents is larger than 50. Comparing the numerical results in Table 1, we observe that MLM methods achieve around 10% improvement of the retrieval accuracy over VSM when 10 candidate documents are retrieved. They still obtain around 4% accuracy improvement when the number of retrieved documents is increased to 120. Also, MLMS with hybrid distance delivers around 2% improvement over the MLMS with only the global information. Similar results are shown in Table 1 for the recall performance. In summary, adding paragraph information into the relevance performs better using either histograms or signatures, which makes sure that the source document will not mistakenly filter out so as to degrade the results of *Failed Detection Ratio* in the subsequent PD.

6.3. Plagiarism detection

This experimental section investigates the performance of our proposed approaches on PD. We used the same parameter settings with the last section (see Section 6.2). Another three parameters were set as follows: $N_2=20000$, $\tau=0.5$, and $N_{ret}=500$. We first retrieve 500 (i.e. $N_{ret}=500$) documents using different retrieval methods (i.e. MLMH, MLMS-Global, MLMS-Local, and MLMS-Hybrid) when inputting each of the plagiarism documents as a query. We then observe the rank of the source document in the retrieval results. The rank indicates the position of the source document in the retrieved document list. A higher rank value is desirable for easy detection. We then re-rank the candidates using ranking based PD algorithm (i.e. further sentence sorting).

Table 2 shows the average rank statistics[5] of a source document in the retrieval results for 78 queries of plagiarism documents in the 1st set. Table 1 also includes the *Failed Detection Ratio*, indicating the percentage of the retrieved documents, which the source document is not included in. Another criterion called *Composite Rank*[6] is calculated according to *Average Rank* and *Failed Detection Ratio*. Here, the scaling parameter ε is set at 0.1. Its choice is based on Fig. 3. It is clear that using PD algorithm delivers significant improvement of rank performance from all evaluation criteria. MLMS-Local shows better performance than MLMS-Hybrid. It can be seen that including global information may bring noises into the distance fusion function, because plagiarists intend to copy or slightly modify only a few paragraphs from the source document to bypass the PD system. MLMS-Local

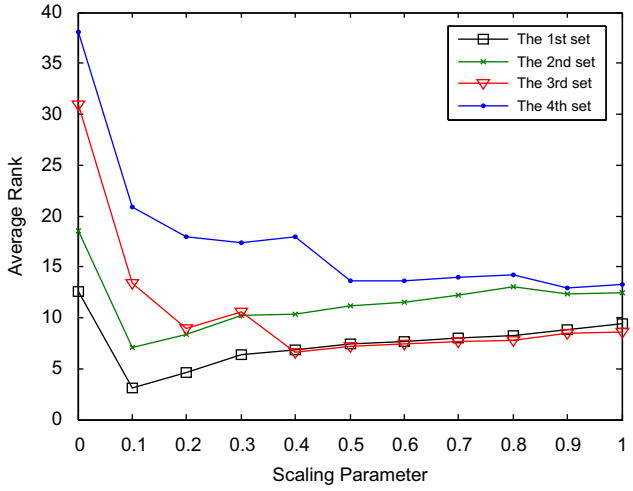

Fig. 3. Average rank against different scaling parameters for all plagiarism sets.

also out-performs MLMH, which can be attributed to the weight $\{w_k\}$ delivering the information capacity of selected features and the length information of a paragraph. MLMH exhibits only slightly better performance than MLMS-Global. In addition, MLMS-Global shows the worst performance in terms of *Failed Detection Ratio*. It indicates that many source documents have not appeared in the retrieval list. We also compare the PD algorithm to the method using multi-layer self-organizing map (MLSOM) [2] for the 1st set. It is unsurprising that our method still delivers superior results because MLSOM only considers paragraph level. We then investigate the results of the presence of the source document in the first N_{ret} number of documents from the final retrieval list for performing PD. Fig. 4(a) plots the presence of source document in the ranking list against the number of documents investigated for the 1st set. It is clear that PD algorithm delivers superior performance over other retrieval approaches. It obtains about 55% percentage of the source document when only one document is retrieved. MLMS-Local significantly out-performs MLMS-Global. In addition, MLMH shows superior performance than MLMS-Global when the number of retrieved documents is larger than around 150. To investigate the effect of scaling parameter ε on the plagiarism results, we summarized the *Average Rank* based on different values of ε from 0.0 to 1.0 at increments of 0.1 (see Fig. 3). It indicates that there is an optimal value to scale the exploration, and $\varepsilon=0.1$ appears to be the best choice for the 1st set. We then consider PD in an automatic way rather than examining probable source document in the ranking list (see Section 5.2). We set

[5] In Tables 2 and 4, AR denotes *Average Rank*, SD denotes *Standard Deviation*, MaxR denotes *Maximum Rank*, MinR denotes *Minimum Rank*, FDR denotes *Failed Detection Ratio*, and CR denotes *Composite Rank*.

[6] Composite Rank = Average Rank/(1 − Failed Detection Ratio).

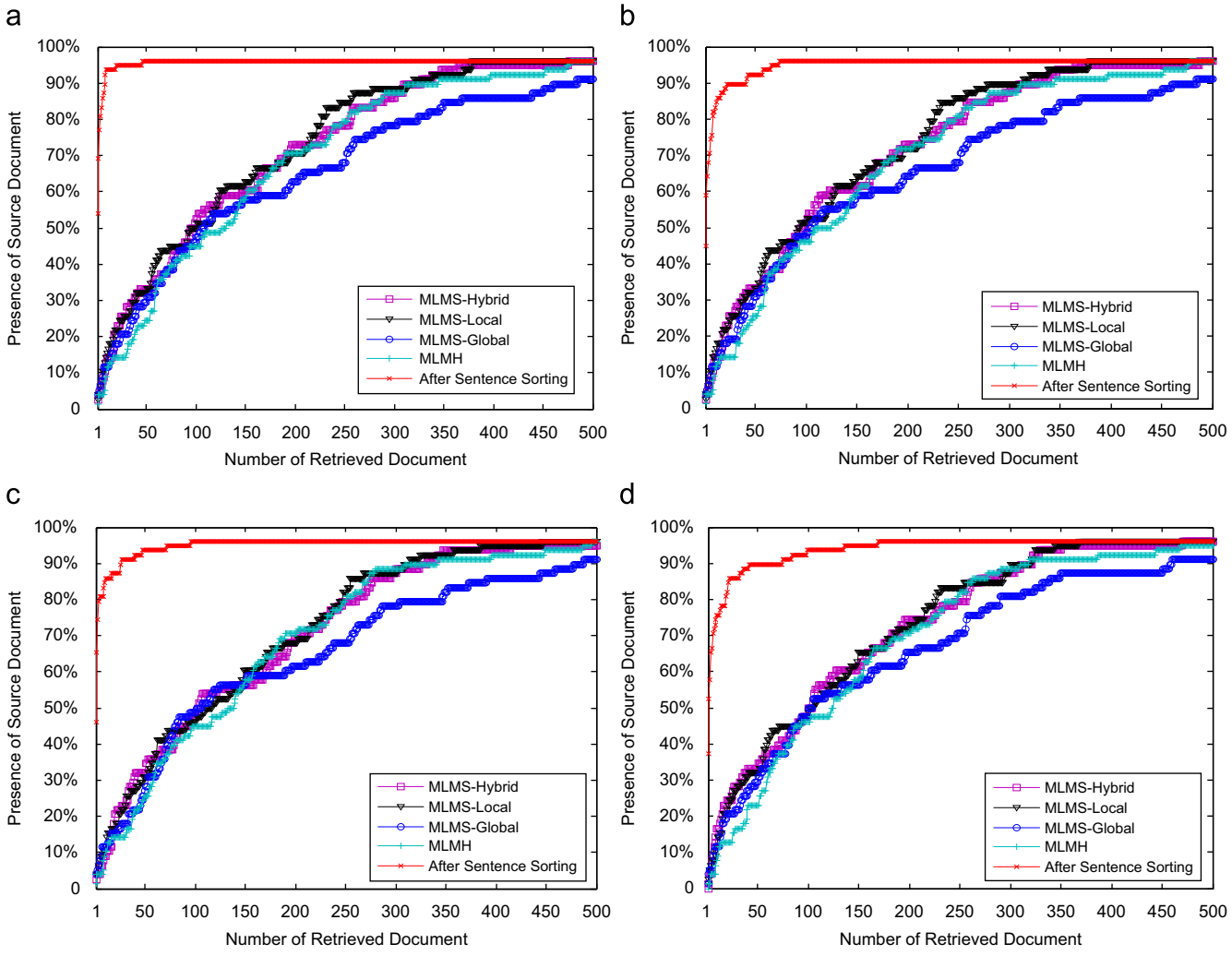

Fig. 4. Presence of source document against number of investigated document for: (a) the 1st set, (b) the 2nd set, (c) the 3rd set, and (d) the 4th set.

different offset values θ that are used for making binary decisions. Fig. 5(a) shows the results of automatic PD against different offset values of θ. It includes the plot of *Source Detection*[7] and *False Detection*. *False Detection* indicates the case when other than source document is detected as plagiarism source. It is observed that a higher value of θ leads to unexpected *False Detection*. With the increase of θ, *Source Detection* drops, whilst *False Detection* drops in an insignificant rate at the early stage and then goes up. An optimal choice of automatic PD is to find an offset value of θ giving *Source Detection* as high as possible, whilst keeping *False Detection* as low as possible. Using a value of θ around 0.1 delivers such a balanced performance. The results on the 2nd set by using ranking based PD are summarized in Fig. 4(b) and Table 3. The results are quite similar to those on the 1st set, because the 2nd set is made from the 1st set by only changing sentences in the copied text. The effects of different offset values of θ (see Fig. 5(b)) are also similar except that the balanced offset value of θ is around 0.05. Compared to the first two sets, PD in the 3rd set becomes more difficult due to deliberately deleting some sentences in the copied text. The results in Table 4 together with Fig. 4(c) also verify the detection difficulties. The average ranks are higher by either retrieval approaches or the PD algorithm. From Fig. 3, it is observed that the optimal value of scaling parameter ε increases to around 0.4, and there is a local optimum around 0.2. In addition, the optimal offset value of θ in automatic PD decreases to around 0.01 as seen in Fig. 5(c). With the increase of θ, *Source Detection* drops sharply. The greatest difficulty of PD is to recognize multiple plagiarism patterns by splitting parts of source document into different parts of the copied text. The 4th set exhibits this kind of an example. Our proposed approaches, however, deliver promising results as seen in Table 5 together with Fig. 4(d). From Fig. 3, it is observed that the value of parameter ε scaling up to 0.9 reaches an optimum, whilst there have many local optima around 0.3 and 0.5. The results of the automatic PD algorithm in terms of different offset values of θ are plotted in Fig. 5(d). The trade-off value of θ is lower than 0.01. PD system fails to detect sources when θ increases to 0.2.

In summary, MLMS-Local consistently out-performs other retrieval approaches. Our proposed PD algorithms deliver significant improvement in terms of the rank statistics. Thus it suggests that the plagiarized sources can be easily identified by users. For different plagiarism patterns, single plagiarism (see examples in the 1st set and the 2nd set) only needs the lower value of scaling parameter ε, which brings large time efficiency

[7] Source Detection = 1 − Failed Detection Ratio.

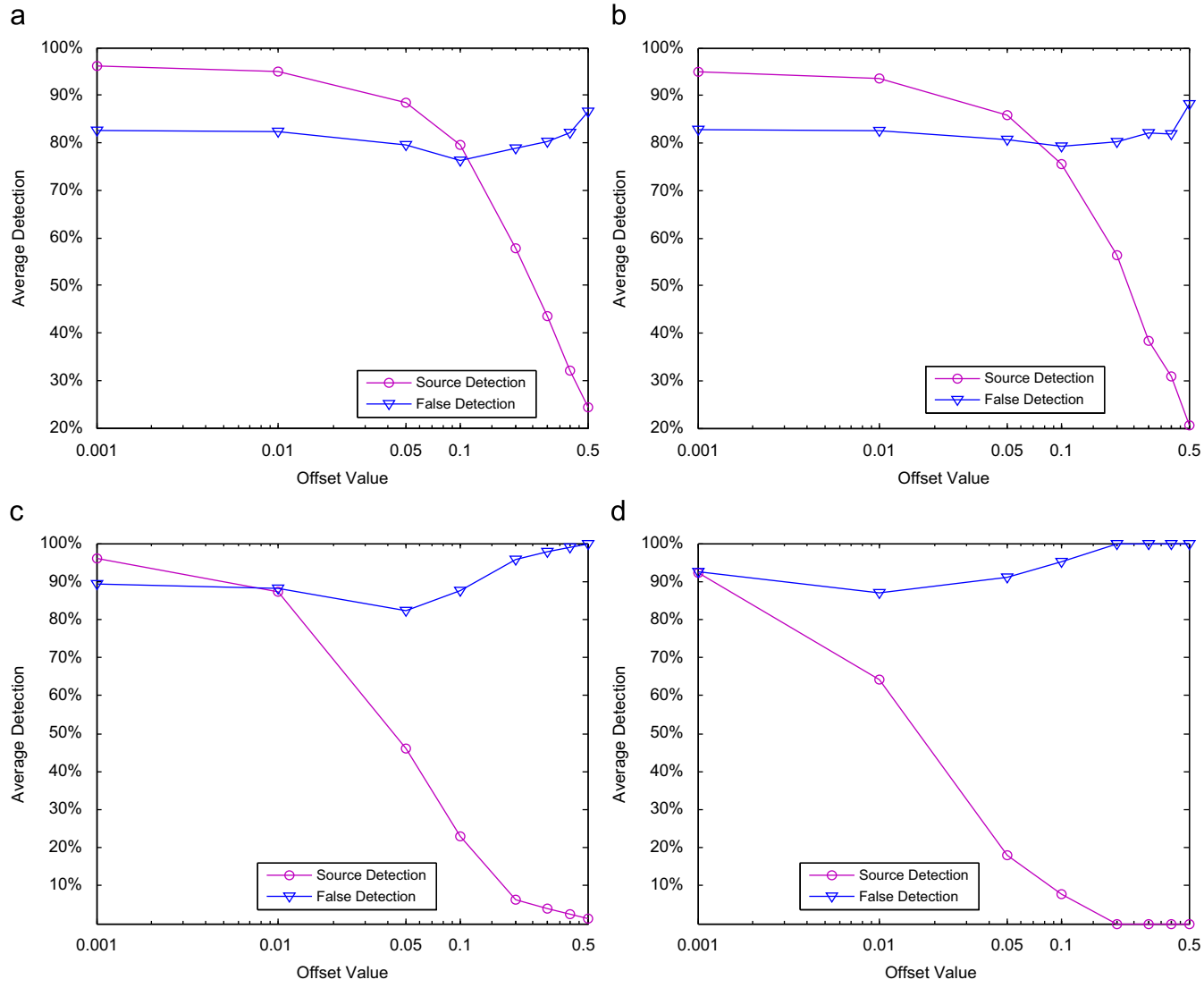

Fig. 5. Detection performance against different offset values for: (a) the 1st set, (b) the 2nd set, (c) the 3rd set, and (d) the 4th set.

Table 3
Ranks statistics of source document for the 2nd set.

	Rank statistics					
	AR	SD	MaxR	MinR	FDR (%)	CR
By retrieval						
MLMS-Hybrid	123.97	112.27	488	1	3.85	128.93
MLMS-Local	117.09	103.04	380	1	3.85	121.78
MLMS-Global	139.62	132.19	485	1	8.97	153.39
MLMH	136.55	116.96	478	1	3.85	142.01
By PD						
$\varepsilon=0.1$	7.12	14.37	74	1	3.85	7.41

Table 4
Ranks statistics of source document for the 3rd set.

	Rank statistics					
	AR	SD	MaxR	MinR	FDR (%)	CR
By retrieval						
MLMS-Hybrid	128.34	108.97	415	1	5.13	135.28
MLMS-Local	129.29	109.74	447	1	3.85	134.47
MLMS-Global	144.72	135.23	488	1	8.97	158.99
MLMH	140.75	117.41	496	1	3.85	146.38
By PD						
$\varepsilon=0.4$	6.64	15.55	96	1	3.85	6.91

gain as well. Multiple plagiarism patterns (see examples in the 3rd set and the 4th set) require scaling the parameter ε up so that copied texts disable to bypass the PD system. But this results in the expense of computational time. Therefore, empirically setting the scaling parameter has much dependency on the plagiarism patterns. On the other hand, determining the optimal offset value of θ depends on different plagiarism patterns. From our observations, detecting multiple plagiarized parts usually requires a lower offset value, whilst a higher offset value is suitable for single plagiarism.

6.4. Study of parameters

In this section we conduct an empirical study on the parameters involved in PD system and their effects on the results.

Table 5
Ranks statistics of source document for the 4th set.

	Rank statistics					
	AR	SD	MaxR	MinR	FDR (%)	CR
By retrieval						
MLMS-Hybrid	122.13	110.34	471	2	3.85	127.02
MLMS-Local	119.27	104.13	368	1	3.85	124.04
MLMS-Global	134.87	124.52	460	1	8.97	148.17
MLMH	132.16	107.25	468	2	5.13	139.31
By PD						
$\varepsilon=0.9$	12.92	29.56	169	1	3.85	13.44

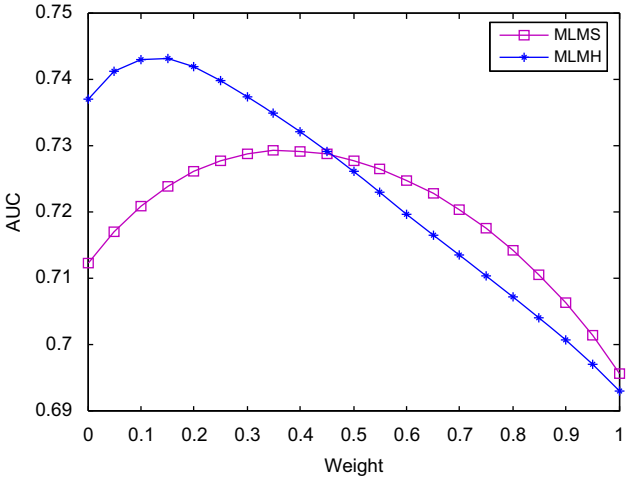

Fig. 6. AUC for MLM approaches against different weights.

It includes the study of the weight λ (see Section 6.4.1), the size of compressed features N_F (see Section 6.4.2), the vocabulary size V_1 and V_2 (see Sections 6.4.3 and 6.4.4), and the overlapping threshold τ (see Section 6.4.5).

6.4.1. *Weight* λ

To balance the effect of the emphasis between the global and local information on the results of relevant DR, we used a weight parameter λ in the design of hybrid MLM approaches (i.e. MLMH and MLMS-Hybrid). We used the measure named "area under the precision-recall curve" (AUC), a function of the weight λ, which can be simply defined as follows:

$$AUC(\lambda) = \sum_{i_A=2}^{n_{max}} \frac{(P_\lambda(i_A)+P_\lambda(i_A-1)) \times (R_\lambda(i_A)-R_\lambda(i_A-1))}{2}, \quad (30)$$

where n_{max} denotes the maximum number of the retrieved documents, $P_\lambda(i_A)$ and $R_\lambda(i_A)$ represent the precision and recall values with i_A documents retrieved corresponding to the weight λ. A higher value of AUC against different weights favors better performance of relevant DR using hybrid MLM. Fig. 6 shows the results of AUC using MLM approaches against the weight values of λ from 0 to 1 at the increments of 0.05. It is observed that there is an optimal weight to balance the importance of the global and local information for either MLMH or MLMS. Thus we can see the contribution of the global and local semantics for improving the retrieval accuracy. In this study, the optimal value of weight λ is around 0.35 for MLMS. That is the reason we set $\lambda=0.35$ for MLMS-Hybrid (see Section 6.2). On the other hand, the optimal value of weight λ is around 0.1 for MLMH. It indicates that we should put much emphasis on local information. For simplicity, we set $\lambda=0$ in the experimental parts for MLMH (see Section 6.2), i.e. only including the local semantics from paragraphs. We also plotted the results of *Failed Detection Ratio* against different weights for all the four sets in Fig. 7, because different weighting

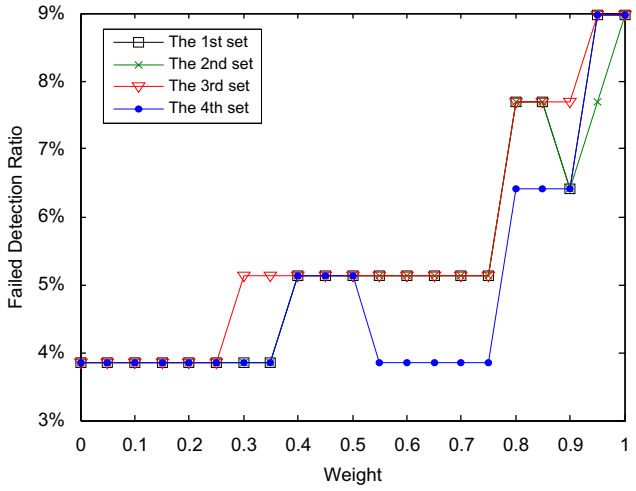

Fig. 7. *Failed Detection Ratio* against different weights for all plagiarism sets.

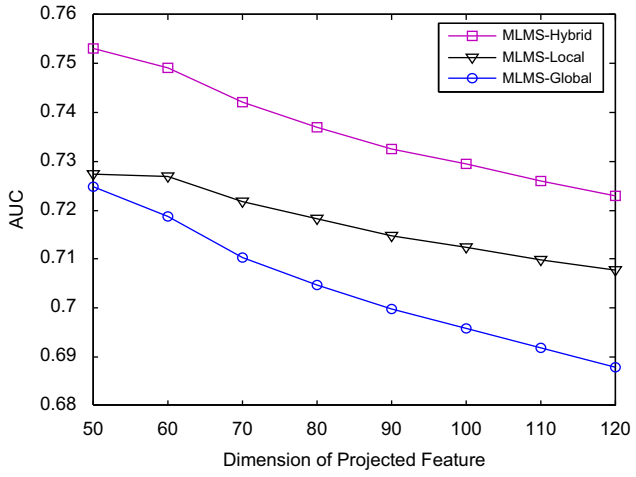

Fig. 8. AUC for MLMS against different dimensions of PCA features.

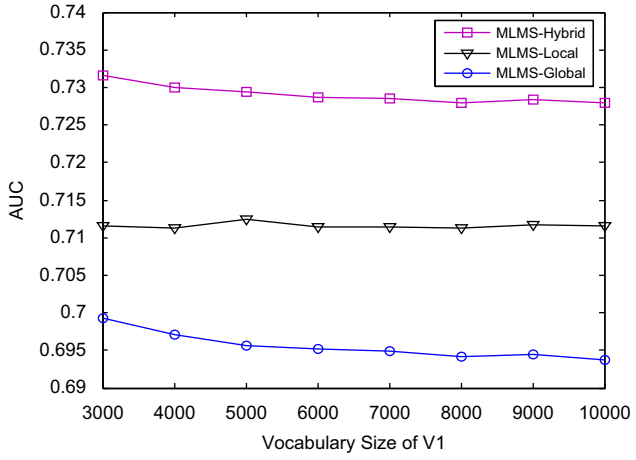

Fig. 9. AUC for MLMS against different vocabulary sizes of V_1.

parameters only have influences on the *Failed Detection Ratio* of MLMS-Hybrid. It is observed that higher value of weight λ usually results in higher value of *Failed Detection Ratio*. With the increase of λ, MLMS-Hybrid only delivers worse or equal (see $\lambda=0.55$–0.75 for the 4th set in Fig. 7) performance compared to MLMS-Local (i.e. $\lambda=0$). Thus it suggests that MLMS-Local is better enough rather than combining extra global information into MLMS with applications to the PD system.

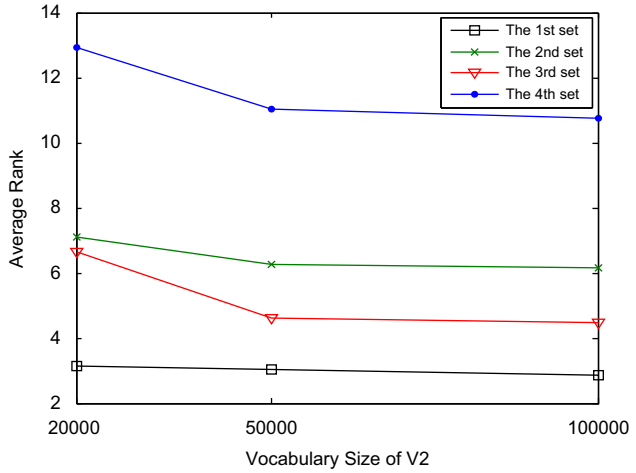

Fig. 10. Average ranks against different vocabulary sizes of V_2 for all four sets.

6.4.2. *Dimension of projected feature N_F*

In this sub-section, we study the impact of different dimensions of PCA features on the results of relevant DR. AUC measure (see Eq. (30)) is used to evaluate the performance of DR with different values of N_F. With the setting of $N_1=5000$, Fig. 8 shows the results of the AUC against the dimension of PCA features that varies from 50 to 120 at the increments of 10. It is observed that, with the increase of dimensions, AUC values of all three MLMS approaches drop slowly. Thus, it indicates that higher dimension of PCA features does not perform well in terms of DR. The optimal choice depends on the dataset.

6.4.3. *Vocabulary size of V_1*

Term selection is always an important issue in language processing and its applications. In this study, we use the VSM weighting scheme (see Eq. (1)) to rank the importance of terms, and select the first N_1 terms as the vocabulary V_1 for the relevant DR. In this sub-section, we investigate the effects of different

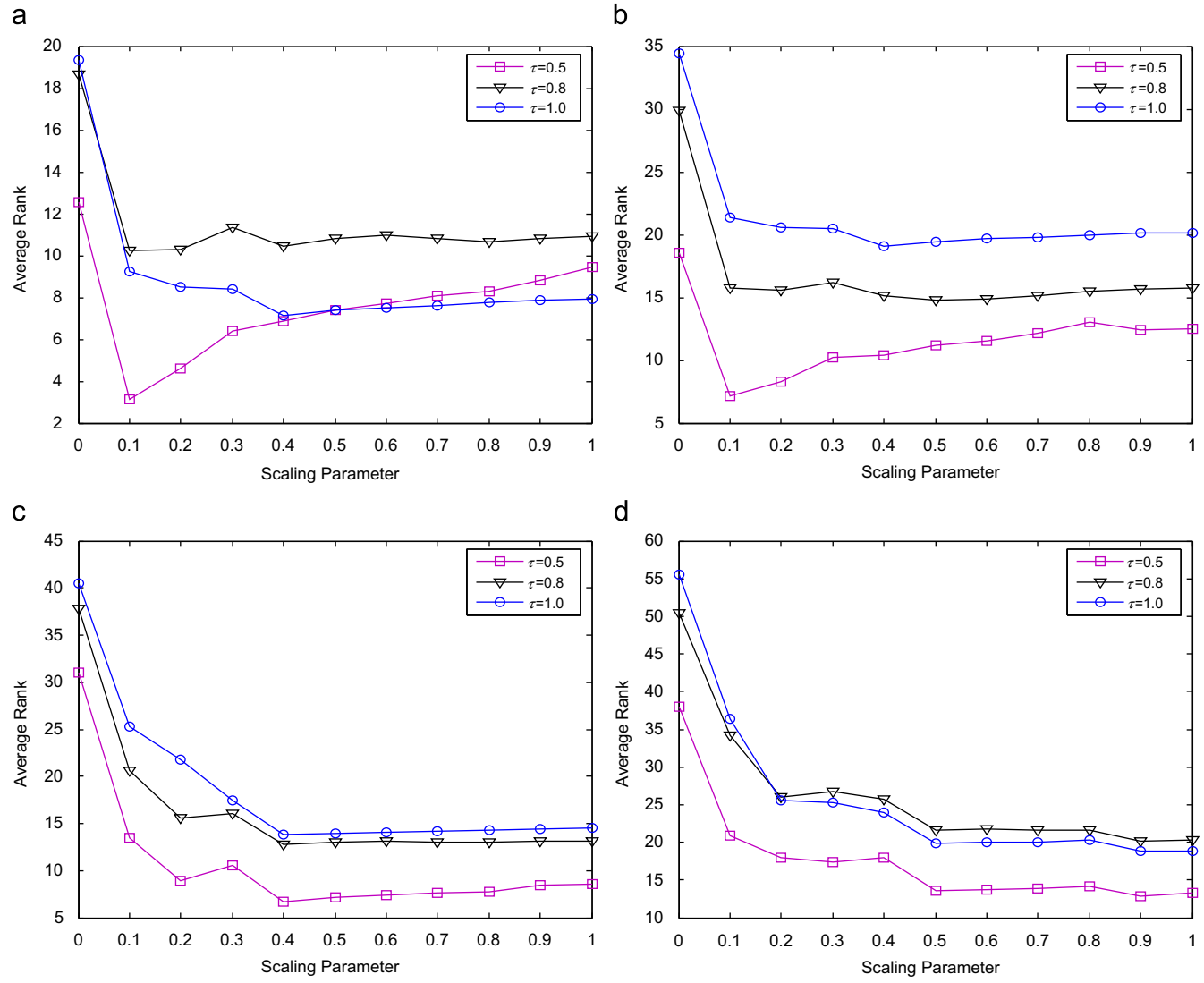

Fig. 11. Results with different values of threshold τ: (a) the 1st set, (b) the 2nd set, (c) the 3rd set, and (d) the 4th set.

vocabulary sizes of V_1 on the AUC performance of DR. With the setting of $N_F=100$, we plotted the results in Fig. 9, where the vocabulary size N_1 varies from 3000 to 10,000 at the increments of 1000. The results suggest that different values of N_1 almost have no impacts on the performance of DR in our dataset. This attributes to the weighting scheme that make the important words rank higher in the vocabulary list.

6.4.4. Vocabulary size of V_2

In PD, sentence signatures are constructed by the vocabulary V_2 accordingly. We investigated the results of PD with different sizes of V_2 in Fig. 10. The same parameter settings were used as Section 6.3. It is noted that the increase of the size of V_2 delivers similar results to the case of $N_2=20{,}000$ for the first two sets, whilst it performs slightly better for the last two sets. It indicates that large vocabulary size for sentence signature favors the improvement of the PD accuracy though at the expense of storage space.

6.4.5. Threshold τ

Threshold τ is used to control which sentence will be regarded as a plagiarized part and counted in the distance fusion process (see Section 5.1). It provides the beliefs of plagiarism for the sentence distance function. We empirically studied the effects of different values of threshold τ on the results of *Average Rank* along with the variations of the scaling parameter ε for all the four sets. Fig. 11(a) shows the results of the 1st set. It indicates that, with the increase of the threshold value, the performance degrades. The best choice of the scaling parameter ε also shifts to around 0.4. Fig. 11(b) for the 2nd set exhibits similar results. For the last two sets, higher values of τ deliver similar results from Fig. 11(c) and (d), which are also worse than that of a lower value of τ (for example, $\tau=0.5$).

6.5. Computational time

Computational time is the major concern of the PD system due to usually involving a large-scale database. Since our proposed system involves two parts the relevant DR and PD algorithm, we empirically conducted the study of query response by handling the DR and PD separately. All the experiments were performed on a PC with Intel Core-2 2.13 GHz and 2 GB memory. The DR programs were written in Visual Studio C++ 6.0, and the PD algorithms were tested in MATLAB 7.01. We first plotted the query time against the number of retrieved documents based on our dataset in Fig. 12. It is observed that the computational cost noticeably increases when the number of retrieved documents is from 1000 unto 10,000. In practice, we, however, only first retrieve a few documents (usually under 1000, here, we used $N_{ret}=500$ in the PD) to facilitate further sentence matching in PD. Since the scaling parameter ε directly influences the time performance of the PD, in Fig. 13 we plotted the average time performance against the scaling parameter varying from 0 to 1 with the increments of 0.1. It is observed that the query time elapses exponentially along with the increase of ε. We, however, usually set ε with less than 0.5, which is better enough to deliver acceptable PD performance (see Section 6.3). For the case of $\varepsilon=0.5$, the query time is around 133 s. Also, if we use Visual Studio C++ to implement the PD algorithms rather than MATLAB, it will significantly shorten the query time with only a few seconds. Thus the total query time including the DR and PD will be completed within 1 min or less. Therefore, it indicates that our proposed method is applicable to the real PD applications with an acceptable system response.

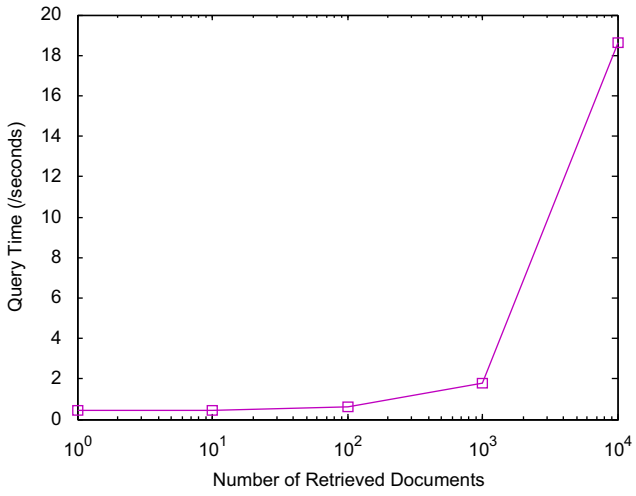

Fig. 12. Query time against number of retrieved documents in DR.

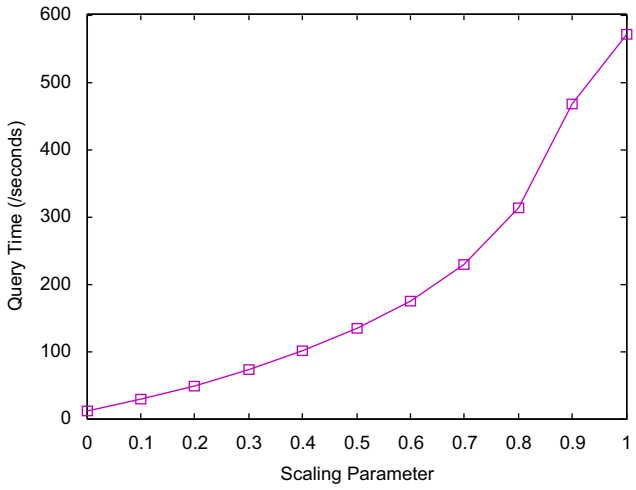

Fig. 13. Query time against different scaling parameters in PD.

7. Discussion and extension

Currently developing an efficient PD system is a very demanding work because plagiarism easily occurs in the information age. Although we conducted experiments in a simulation platform, many interesting results can be observed:

- The usage of local information from sections or paragraphs significantly enhances the performance of the DR because it explores the spatial distributions of words.
- Histogram based DR approach with an appropriate distance fusion performs better in the DR than in the PD.
- Signature based DR approach aided by the EMD performs better in both the DR and PD, because it considers the information capacity in paragraphs and converts the MLM into the linear programming problem.
- The best choice of weight parameter λ in MLMS depends on the dataset, whist it approximates to zero for MLMH (i.e. only including local information).
- The best choice of the size of compressed features depends on the dataset, but usually it can be set at around 100.
- The vocabulary size used for the DR usually has no effects on the results, but given the storage space and computational burden, it can be set at a few thousands.

- Larger vocabulary size used for performing PD (i.e. constructing the sentence signature) favors accuracy efficiency of the PD.
- The choice of the scaling parameter ε used to control the exploration in the paragraph level relies on different plagiarism patterns. Lower value of ε is suitable for single plagiarism, whilst higher value of ε can be used for multiple plagiarism. Empirically, it can be set at around 0.5.
- The threshold τ in the sentence level can be set at around 0.5. Higher value of τ, however, degrades the PD performance.
- In handling a large-scale database, the computational time of our coarse-to-fine system will not significantly increase, because it restricts its plagiarism searching to a limited retrieval document set. Thus the query will be responded within acceptable computation time.

For the practical implementation of the PD system, it is easy to extend our coarse-to-fine framework to an online application. In fact, many Internet-based commercial tools[1–4] for anti-plagiarism have been available. It is difficult to directly compare our method to these tools because their datasets are Internet-based. We selected some queries from our dataset and inputted them into the Turnitin[1] to investigate the performance. It is observed that our method delivers similar and comparable results. It is worth noting that our method in this paper makes clear decision controlled by a threshold, whilst commercial tools usually provide a lengthy report listing the similarity percentage, which requires the users to manually check out the suspected sources. It is easy to implement our method in the fashion of generating an originality report like commercial tools by eliminating the threshold. Extending current PD system, we can first develop a document analyzer to extract a few key words (i.e. important words that indicate the relevant topics) with the given query, input them into the online search engine to construct a local database including thousands of relevant documents, and then use our proposed approach to implement PD (i.e. document segmentation→vocabulary construction→multi-level representation→relevant DR→PD). In addition, a document can be represented by a finer structure. Other techniques rather than PCA also can be used for dimensionality reduction.

8. Conclusion

A coarse-to-fine framework to efficiency thwart plagiarism is proposed in this study. Each document is represented by a multilevel structure, i.e. document–paragraph–sentence. Different signatures are constructed to represent components in different levels. Relevant DR approaches by adding or only using local information to explore rich semantics from documents are introduced to retrieve the suspected sources. Two PD algorithms by further sentence matching are designed and implemented to identify the plagiarized sources. Extensive experiments are conducted on DR, PD, the study of the effects of parameters, and the empirical study of query time. The results indicate that our proposed approach is capable of improving the precision in terms of DR and effectively identifying the plagiarized sources. It is also suggested that our proposed PD system can be used as a practical tool for real-time applications. However, currently we only achieved the results of our proposed method relying on simple patterns and we did not consider the sequence of terms in PD. In fact, it is straightforward to incorporate the sentence matching strategy based on n-grams [40] into our framework. In the future work, we plan to investigate our technique in other competitive tasks[8] to detect more sophisticated plagiarism patterns, and explore more efficient searching algorithms to multi-level document matching. We also need to study other representations of document features for saving storage space and computational time. On the other hand, the issue of automatically setting the parameters (e.g. the scaling parameter and thresholds) can be another future work. We are working on using Bayesian and other probabilistic approaches to estimate these thresholds.

[8] http://www.webis.de/research/corpora

Acknowledgments

The authors would like to express many thanks to anonymous reviewers for helpful comments to improve the standard of this paper.

References

[1] T.C. Hoad, J. Zobel, Methods for identifying versioned and plagiarized documents, Journal of the American Society for Information Science and Technology 54 (3) (2003) 203–215.
[2] Tommy W.S. Chow, M.K.M. Rahman, Multi-layer SOM with tree structured data for efficient document retrieval and plagiarism detection, IEEE Transactions on Neural Networks 20 (9) (2009) 1385–1402.
[3] Tommy W.S. Chow, Haijun Zhang, M.K.M. Rahman, A new document representation using term frequency and vectorized graph connectionists with application to document retrieval, Expert Systems with Applications 36 (2009) 12023–12035.
[4] Haijun Zhang, Tommy W.S. Chow, M.K.M. Rahman, A new dual wing harmonium model for document retrieval, Pattern Recognition 42 (11) (2009) 2950–2960.
[5] G. Salton, M. McGill (Eds.), Introduction to Modern Information Retrieval, McGraw-Hill, 1983.
[6] S. Deerwester, S. Dumais, Indexing by latent semantic analysis, Journal of the American Society of Information Science 41 (6) (1990) 391–407.
[7] T. Hofmann, Probabilistic latent semantic indexing, in: Proceedings of the Twenty-Second Annual International SIGIR Conference, 1999.
[8] D. Blei, A. Ng, M. Jordan, Latent Dirichlet allocation, Journal of Machine Learning Research 3 (2003) 993–1022.
[9] M. Welling, M. Rosen-Zvi, G. Hinton, Exponential family harmoniums with an application to information retrieval. In: Advances in Neural Information Processing Systems, vol. 17, MIT Press, Cambridge, MA, 2004, pp. 1481–1488.
[10] P. Gehler, A. Holub, M. Welling, The rate adapting Poisson model for information retrieval and object recognition, in: Proceedings of the 23rd International Conference on Machine Learning, Pittsburgh, PA, 2006.
[11] R.B. Yates, B.R. Neto, in: Modern Information Retrieval, Addison Wesley Longman, 1999.
[12] S.M. Chen, Y.J. Horng, C.H. Lee, Document retrieval using fuzzy-valued concept networks, IEEE Transactions on Systems, Man, and Cybernetics B 31 (1) (2001) 111–118.
[13] J. Bear, D. Israel, J. Petit, D. Martin, Using information extraction to improve document retrieval, in: Proceedings of the Sixth Text Retrieval Conference (TREC 6), Gathiersburg, MD, November 1997, pp. 367–377.
[14] R. Gaizauskas, Y. Wilks, Information extraction: beyond document retrieval, Journal of Documentation 54 (1) (1998) 70–105.
[15] A. Georgakis, C. Kotropoulos, A. Xafopoulos, I. Pitas, Marginal median SOM for document organization and retrieval, Neural Networks 17 (3) (2004) 365–377.
[16] K. Lagus, Text retrieval using self-organizing document maps, Neural Processing Letters 15 (2002) 21–29.
[17] M.K.M. Rahman, P.Y. Wang, T.W.S. Chow, S.T. Wu., A flexible multi-layer self-organizing map for generic processing of tree-structured data, Pattern Recognition 40 (5) (2007) 1406–1424.
[18] N. Rooney, D. Patterson, M. Galushka, V. Dobrynin., A scalable document clustering approach for large document corpora, Information Processing and Management 42 (5) (2006) 1163–1175.
[19] L.A.F. Park, K. Ramamohanarao, M. Palaniswami., A novel document retrieval method using the discrete wavelet transform, ACM Transactions on Information Systems 23 (3) (2005) 267–298.
[20] Y. Yang, J. O. Pedersen, A comparative study on feature selection in text categorization, in: Proceedings of the International Workshop on Machine Learning, 1997.
[21] M. Steinbach, G. Karypis, V. Kumar, A comparison of document clustering techniques, in: Proceedings of the KDD Workshop on Text Mining, 2000.
[22] F. Sebastiani, Machine learning in automated text categorization, ACM Computing Surveys 34 (1) (2002) 1–47.
[23] M. Fuketa, S. Lee, T. Tsuji, M. Okada, J. Aoe., A document classification method by using field association words, Information Sciences 126 (1–4) (2000) 57–70.
[24] C.M. Tan, Y.F. Wang, C.D. Lee., The use of bigrams to enhance text categorization, Information Processing and Management 38 (4) (2002) 529–546.

[25] M.L. Antonie, O.R. Zaiane, Text document categorization by term association, in: Proceedings of the IEEE International Conference on Data Mining (ICDM2002), 2002, pp. 19–26.
[26] A. Selamat, S. Omatu, Web page feature selection and classification using neural networks, Information Sciences 158 (2004) 69–88.
[27] L.M. Manevitz, M. Yousef., One-class SVMs for document classification, Journal of Machine Learning Research 2 (2001) 139–154.
[28] S. Singh, L. Dey., A new customized document categorization scheme using rough membership, Applied Soft Computing 5 (4) (2005) 373–390.
[29] N. Bouguila, Clustering of count data using generalized Dirichlet multinomial distributions, IEEE Transactions on Knowledge and Data Engineering 20 (4) (2008) 462–474.
[30] X. Cui, J. Gao, T.E. Potok., A flocking based algorithm for document clustering analysis, Journal of Systems Architecture 52 (8–9) (2006) 505–515.
[31] B. C. M. Fung, K. Wang, M. Ester, Hierarchical document clustering using frequent itemsets, in: Proceedings of the Third SIAM International Conference on Data Mining, 2003, pp. 59–70.
[32] S.L. Bang, J.D. Yang, H.J. Yang, Hierarchical document categorization with k-NN and concept-based thesauri, Information Processing and Management 42 (2006) 387–406.
[33] A. Parker, J.O. Hamblen, Computer algorithms for plagiarism detection, IEEE Transactions on Education 32 (2) (1989) 94–99.
[34] S. Brin, J. Davis, H. Garcia-Molina, Copy detection mechanisms for digital documents, in: Proceedings of the of The ACM SIGMOD Annual Conference, 1995, pp. 398–409.
[35] N. Shivakumar, H. Garcia-Molina, Building a scalable and accurate copy detection mechanism, in: Proceedings of the First ACM Conference on Digital Libraries (DL'96), Bethesda, MD, 1996, pp. 160–168.
[36] K. Monostori, A. Zaslavsky, H. Schmidt, MatchDetectReveal: finding overlapping and similar digital documents, in: Proceedings of the 2000 Information Resources Management Association International Conference on Challenges of Information Technology Management in the 21st Century, Anchorage, Alaska, United States, 2000, pp. 955–957.
[37] N. Heintze, Scalable document fingerprinting, in: Proceedings of the Second USENIX Workshop on Elctronic Commerce, Oakland, CA, November, 1996, pp. 18–21.
[38] R.A. Finkel, A. Zaslavsky, K. Monostori, H. Schmidt, Signature extraction for overlap detection in documents, in: Proceedings of the 25th Australasian Conference on Computer Science, Melbourne, Victoria, 2002, pp. 57–64.
[39] Sven Meyer zu Eißen, Benno Stein, Marion Kulig, Plagiarism detection without reference collections, in: Reinhold Decker, Hans J. Lenz (Eds.), Advances in Data Analysis, 2007, pp. 359–366.
[40] Alberto Barrón-Cedeño, Paolo Rosso, On automatic plagiarism detection based on n-grams comparison, in: Boughanem et al. (Eds.), ECIR 2009, Lecture Notes in Computer Science, vol. 5478, pp. 696–700.
[41] A. Si, H.V. Leong, R.W.H. Lau, CHECK: a document plagiarism detection system, in: Proceedings of the ACM Symposium for Applied Computing, February 1997, pp. 70–77.
[42] Esben Høgh-Rasmussen, in: BBTools—a Matlab Toolbox for Black-Box Computations, Neurobiology Research Unit, Copenhagen University Hospital, 2005 URL: 〈http://nru.dk/software/bbtools/〉.
[43] Y. Rubner, C. Tomasi, L.J. Guibas., The Earth Mover's Distance as a metric for image retrieval, International Journal of Computer Vision 40 (2) (2000) 99–121.
[44] Francesc Serratosa, Alberto Sanfeliu, Signatures versus histograms: definitions, distances and algorithms, Pattern Recognition 39 (5) (2006) 921–934.
[45] X. Gao, B. Xiao, D. Tao, X. Li, Image categorization: graph edit distance+edge direction histogram, Pattern Recognition 41 (10) (2008) 3179–3191.

Haijun Zhang received his B.Eng. degree in the Department of Civil Engineering and Master degree in the Department of Control Theory and Engineering from the Northeastern University, Shenyang, PR China in 2004 and 2007, respectively. He worked as a research assistant at the City University of Hong Kong in April–September 2007. He is currently working towards his Ph.D. degree at the City University of Hong Kong, Hong Kong. His research interests are evolutionary computation, structure optimization, neural networks, machine learning, data mining, pattern recognition and their applications.

Tommy W.S. Chow (IEEE M'93–SM'03) received his B.Sc. (First Hons.) and Ph.D. degrees from the University of Sunderland, Sunderland, U.K. He joined the City University of Hong Kong, Hong Kong, as a Lecturer in 1988. He is currently a Professor in the Electronic Engineering Department. His research interests include machine fault diagnosis, HOS analysis, system identification, and neural network learning algorithms and applications.