Methodology for the Assessment of the Text Similarity of Documents in the CORE Open Access Data Set of Scholarly Documents

Ivan Kovačič
University of Maribor,
Faculty of Electrical
Engineering and Computer Science,
Koroška cesta 46, 2000 Maribor, Slovenia
ivan.kovacic@um.si

David Bajs
University of Maribor,
Faculty of Electrical
Engineering and Computer Science,
Koroška cesta 46, 2000 Maribor, Slovenia
david.bajs@student.um.si

Milan Ojsteršek
University of Maribor,
Faculty of Electrical
Engineering and Computer Science,
Koroška cesta 46, 2000 Maribor, Slovenia
milan.ojstersek@um.si

Abstract

This paper describes the methodology of data preparation and analysis of the text similarity required for plagiarism detection on the CORE data set. Firstly, we used the CrossREF API and Microsoft Academic Graph data set for metadata enrichment and elimination of duplicates of documents from the CORE 2018 data set. In the second step, we used 4-gram sequences of words from every document and transformed them into SHA-256 hash values. Features retrieved using hashing algorithm are compared, and the result is a list of documents and the percentages of coverage between pairs of documents features. In the third step, called pairwise feature-based exhaustive analysis, pairs of documents are checked using the longest common substring.

Keywords data preparation for text similarity analysis, text similarity, CORE data set, metadata enrichment

1 INTRODUCTION

Text similarity is a measure of the degree of content similarity between two textual documents. It is used as a metric which assists for plagiarism detection. In general, plagiarism is a term that is used to describe acts of intellectual theft and violating the material and moral rights of the authors of intellectual property. However, there are several definitions of plagiarism that may be applied, depending on the circumstance of each individual case of plagiarism. Maurer et. al. [9] have compiled a list of possible definitions of plagiarism. They include:

- presenting other people's work as your own
- copying other people's work (or ideas) without giving credit
- improper quotation
- copying sentence structures with minor modification without giving credit
- copying the majority of your work from others, regardless of giving credit [9]

In order to efficiently assert the existence and degree of plagiarism, electronic detection is necessary along with manual analysis. Using electronic detection, the similarity index [1] is a metric which is reliable for identifying potential cases of plagiarism, but it does not necessarily indicate that plagiarism had occurred in an individual case. Another important step is to obtain a sufficient amount of metadata about any given document which is analyzed. Using a sufficient amount of metadata, it is easier to detect duplicate records of the same document, which allows for a higher quality assessment of the degree of plagiarism on a given document. If a journal article is recorded twice in the same data set with inconsistent metadata, there exists a pair of documents with a high degree of similarity. Without good quality metadata, the pair of documents may appear as totally different journal articles, although they represent the same instance of intellectual work. In the following sections, we describe our methodology of data preparation for text similarity, metadata enrichment and the computation of text similarity metrics on the CORE data set of scholarly documents as well as an analysis of the run time complexity of our methodology.

2 RELATED WORK

Eaton and Crossman did a review of literature in the field of self-plagiarism in social sciences research databases and found that only 5.8 % of a sample of search results consisted of primary research done in this field, which indicates that the field of self-plagiarism is a relevant topic of research in social sciences, we may also infer a higher relevance of this research topic for computational methods of detecting self-plagiarism [2].

DOI https://doi.org/10.18690/978-961-286-516-0.12
ISBN 978-961-286-516-0

In [8], a study was described in which a sample of articles that were published in the Scientific Periodicals Electronic Library (SPELL) was analyzed for plagiarism. A comparison was made between two samples taken from the years 2013 and 2018. The former sample contained literal reproductions in 65.9 % of the total sample size, while the latter sample from 2018 contained evidence of plagiarism in 44 % of the total sample size. A reduction of the similarity index was noticeable, the difference being attributed to the fact that the sample of articles from 2018 contained guidelines related to plagiarism and self-plagiarism. The methodology of this study consisted of random sampling of the data set and inputting the samples into iThenticate plagiarism detector. García-Romero and Estrada-Lorenzo have described the analysis of the Déjà vu database using citation analysis and found that cases of plagiarism are more frequently published in journals with lower visibility and also confirmed that duplicate documents not citing the original document show a higher degree of full text similarity than those citing it [5].

3 METHODOLOGY

In this section, we present our methodology of applying plagiarism detection algorithms and procedures on the CORE data set, including metadata enrichment techniques and the plagiarism detection methods used in this study.

3.1 The CORE data set

CORE is one of the leading aggregation services of open-access scholarly documents publicly available on the web. Currently, CORE has aggregated metadata of over 200 million open-access research articles and other scholarly documents, harvested from over 10.000 open-access repositories. A full list of harvested repositories may be seen in [12]. In our study, we used the 2018-03-01 CORE data set. The data set contains 9.767.152 unique documents with full text. Each individual document is represented as a JSON record containing both the full text of the document as well as the extracted metadata. The schema of the JSON records is described in Table 1.

Metadata is important for plagiarism/self-plagiarism analysis because it allows an expert who analyses an individual case of plagiarism to evaluate the authorship, year of publication and other data of a document which is being analyzed in comparison to a document's potential candidates for document similarity. The CORE data set does provide a rich aggregation of metadata for scholarly documents, but it is also limited by the quality of metadata provided by the individual servers in harvested OAI-PMH repositories. The CORE team actively develops means of enriching/improving the metadata associated with each individual document, since there exist issues with the metadata provided. These issues include:

- **Duplicate document entries** - there exist several duplicates in the CORE data set. These are present because of multiple publications of the same document in several different repositories that are harvested by CORE. The individual repositories have inconsistent metadata, which makes detection of duplicates a non-trivial task.
- **Unstructured metadata** - most documents in the 2018 CORE data set contain insufficiently structured metadata, since the aggregation process makes it difficult to ascertain the data like the journal name, publisher. This information gets missing in the dc:topic and dc:subject fields and mapping the data in these fields to their respective metadata fields is a non-trivial task.
- **Non-defined document types**
- **Non-defined fields of study**
- **Non-normalized author names**

Table 1: CORE data set JSON schema.

JSON attribute	Description
coreId	unique CORE identifier
doi	DOI identifier (*if available*)
title	Title of the document
authors	Authors (array)
contributors	Contributors (array)
datePublished	Date of publication
abstract	The abstract of the document
downloadUrl	URL of the PDF document
language	Language of the document
fullTextIdentifier	Full text unique ID
pdfHashValue	PDF signature hash
journals	List of journals or journal aliases
rawRecordXml	Harvested OAI-PMH XML response
year	Year of publication
relations	Related entities (journals,journal aliases)
topics	OAI-PMH dc:topic value
subjects	OAI-PMH dc:subject value
issn	ISSN (*if available*)
fullText	Full text of the document

The issues can be elaborated with a concrete example. The document with CORE ID 47847252 is a book section written in French. The document type, found in the subjects field, is "Book sections", which is a non-standardized document type, originating from the terminology used by the digital library that hosts the OAI server which provided this document. The authors' names are recorded with the last name written first, followed by the initial of the first name. In contrast, the document with CORE ID 6871221 has authors written with the first name written firstly, followed by the last name. Such inconsistencies, which are the result of different bibliographic notations, increase the difficulty of automated detection of self-plagiarism and/or duplication of documents.

3.2 Microsoft Academic Graph (MAG)

In order to tackle the issue of incomplete and inconsistent metadata, we utilized the mapping to the Microsoft Academic Graph done by the CORE team with the help of their published 2019-04-01 MAG mapping data set. The Microsoft Academic Graph is a document graph that con-

tains metadata records about scholarly documents, citation relationships between them, as well as normalized data about authors, journals, fields of study and other metadata. By connecting CORE to MAG by using the DOI persistent identifier, an intersection of over 1.200.000 million articles was made in 2016 [6]. The MAG data set contains structured metadata of scholarly documents. It is organized the same way a classical relational database is, with individual schemas representing entities and connections between them represented as foreign keys. A full description of the MAG schema is available in [3]. We emphasize some of the more useful metadata entries in MAG which allow for metadata enrichment of CORE:

- **Normalized authors** - the authors in MAG are normalized in a way that makes them uniquely identifiable and distinguishable from other authors. The displayed author name is separated from the author's normalized name. Where available, authors are also associated with an affiliation.
- **Fields of study** - papers documented in MAG have a mapping to fields of study of that particular paper. The fields of study are organized in a hierarchical structure which allows us to construct a tree structure of fields of study for each paper in MAG.
- **Journal data** - in MAG journal articles are associated with journals, which have their names normalized and other metadata associated to them, where available. Namely, a citation count, the publisher of the journal and the ISSN, where available [11].

3.3 CrossREF API

CrossREF is a service that aggregates metadata from publishers and provides tools for several use cases related to metadata, among other things it is an agent of the DOI foundation (DF) and it allows users to register DOIs for their scholarly documents. One of the most useful features of CrossREF is the CrossREF public API, which allows for metadata enrichment by resolving a DOI which is contained in the CrossREF database into the metadata of the associated document. The metadata provided by CrossREF is rich and reliable, with information about the journal and publisher, authors with affiliations (where available), document types (e.g. journal article or conference paper), reference counts, the volume and issue, among others. The API itself is free to use, with no sign-up or authentication required, the client who consumes the REST API must only adhere to throttling mechanisms specified by CrossREF which limit the number of requests per second available to the client. The API returns a requested sleep interval for the client in the `X-Rate-Limit-Interval` HTTP header.

3.4 Metadata enrichment of CORE using CrossREF and MAG

Our approach of metadata enrichment utilized the CORE MAG mapping with some additional enhancements. We imported a published MAG dump available in [10]. When a document in CORE had a DOI available, the document's DOI was used for pairing with MAG, where the mapping was not previously present in the 2019-04-01 MAG mapping data set. In addition, we normalized the document titles and looked for matches in MAG, excluding the pairings where anomalies had been detected (non-matching year of publication, non-matching authors). We further enriched the metadata of these documents by consuming the CrossREF API and acquiring the ISSN, the proper volume, issue and publisher for the documents. Over 30 % of articles in CORE that have a DOI have no ISSN associated with them in the MAG dump provided in [10]. Our method of metadata enrichment is summarized by Algorithm 1.

```
   Data: set of CORE documents
   Result: set of CORE documents with enriched
           metadata
 1 for document in set of CORE documents do
 2     if document in CORE MAG mapping and
         document has DOI then
 3         consume CrossREF API;
 4         map result to metadata properties of the
             CORE document;
 5     else if normalized document title and year
         match then
 6         enrich document using new MAG
             mapping;
 7         consume CrossREF API;
 8         map result to metadata properties of
             CORE document;
 9     else
10         mark document as incomplete;
11     end
12 end
```

Algorithm 1: CORE metadata enrichment with MAG and CrossREF.

3.5 Text matching in the CORE data set

In this section, we describe how plagiarism detection was implemented on the CORE data set. We serialized the entire CORE 2018-03-01 data set of JSON records into a PostgreSQL database, creating a relational model which corresponds to the JSON schema of CORE. To implement a plagiarism detector, we firstly had to implement a candidate search, followed by a pairwise text matching algorithm.

3.5.1 Candidate retrieval

For finding plagiarism candidates for each individual document, we computed n-gram models for each document in CORE. The full text of the documents is tokenized with a minimum sentence length of 40 characters, using rules for mapping ordinal numbers, URLs, email addresses to tokens, removing stop words and computing lexicographically sorted n-gram sequences with $n = 4$. The minimum length of 40 characters was chosen because it is used in the Slovenian open access infrastructure. Sequences of 4-grams have been determined as more efficient for candidate retrieval in previous studies [4]. A space delimited sequence of sentence n-grams is transformed into an SHA-256 hash value which is stored in a database. The

set of all n-gram hashes represents the set of features of the entire full text which can be matched to other documents, thus acquiring lists of potential candidates for plagiarism. In relational databases, this approach can be implemented efficiently by performing a table join of the table containing the SHA-256 hashes with itself, with the table itself being structured as a key-value map, where the key is a unique ID of the document which is mapped to the hashed n-gram value. In order for such a table join query to be time-efficient, indexes are necessary on the column containing the n-grams hashes, which drastically increases the space complexity of the described method. Once the self-join query had been performed, a hash coverage index was computed:

$$hashCoverage = \frac{\text{unique covered hashes in candidate}}{\text{total number of hashes in document}}$$

The candidates are then sorted in descending order by the *hashCoverage* metric and stored into the database as a candidate list. The procedure of candidate search is summarized in Algorithm 2.

```
   Data: fileid - CORE document ID
   Result: set of candidate documents in CORE
1  for document in set of CORE documents do
2  |  perform self-join on the table of hashes
   |    where document ID = fileid;
3  |  compute hashCoverage;
4  |  sort candidate list by hashCoverage
   |    descending;
5  |  store candidate list into database;
6  end
```

Algorithm 2: Candidate retrieval with hashed n-gram sequences.

3.5.2 Text matching of candidates

For each document, a maximum of 100 candidate documents is set as a limit for reasons of reducing the space complexity of the procedure. After lists of candidates are obtained for all documents in CORE, a fine text comparison algorithm is used by Kärkkäinen et. al. which constructs a permuted longest common prefix array efficiently [7]. The output of the algorithm is a sequence of longest common prefixes along with offsets at which a computed longest common prefix occurred in the text. Our implementation of the algorithm is written in C++ and wrapped by a REST API, which accepts two texts with configuration parameters and returns the longest common prefix array. The configuration parameters are the minimum length of the document (*minLen*), the minimum length of the longest common suffix (*lengthBoundary*) and the size of the window in which neighbor prefixes are searched (*windowSize* - for merging adjacent common prefixes into longer sequences). The REST API is bundled into a Docker image, which allows for an implemention of a distributed approach for calculating text matching on the CORE dataset. A workload of documents is generated, on which text matching is performed for each candidateas a pair (document ID, *status*) in a table in a relational database, with *status* being a flag described in Table 2.

Table 2: Distributed text matching workload status flags.

Status flag	Description
-1	unprocessed
0	processing
1	processed

Each node running the distibuted program performs a transactional `UPDATE` query on the workload table, setting the flags of N documents to 0 and simultaneously obtaining the document IDs from the table where the current flag value equals -1. It proceeds by obtaining all the full texts of the document and the candidates from the database and passing them on to a pairwise comparison performed by the Docker REST service which is running on the same node. This approach allows for efficient parallel computation of text matching between documents and their candidates. The parameter N is limited by the amount of free memory available to the node. Given an average number of candidates C per document and an average length L bytes of a document in CORE, the estimated space complexity in bytes iss given by $N * C * L$. If a node failed to perform the comparison, for example in the case of running out of free virtual memory, a transactional update of the status flag is made, setting the flag back to -1. This allows the document to be processed by another node. Using this approach, the procedure is fault-tolerant and partition tolerant. The results of the text matching comparisons are arrays of offsets and length of common prefixes, which are used in the implementation of a plagiarism detector text matching user interface. The prefixes are also used to compute the similarity indices of the documents:

$$similarityIndex = \frac{\Sigma \text{ length of matches}}{\text{total length of document}}$$

3.5.3 Deduplication

After computing pairwise text matching between documents and their candidates for plagiarism, the process of plagiarism detection in the CORE database is complete. We described the problem of duplicate entries in the CORE data set in 3.1. With overall and pairwise similarity indices computed, we were able to conduct deduplication by using the computed indices in conjunction with the enriched metadata we extracted from MAG and CrossREF. The process of removing duplicates begins by acquiring a list of all pairwise matches with a similarity index above 85 %. For each such pair, we compare the normalized titles of the two documents, the year of publication and ISSN. When a full match has been made, the second document is automatically marked as duplicated. When only a partial match has been made (e.g. only the normalized titles match), the document is marked as a potential duplicate. The potential duplicates are submitted for manual inspection, meanwhile the detected duplicates are excluded from candidate lists and the similarity indices of the documents which contained duplicates are corrected, respectively.

4 RESULTS

By utilizing our method for metadata enrichment, we enriched metadata for 3.457.071 documents in the 2018-03-01 CORE data set which contain a DOI persistent identifier value. We also acquired a complete set of metadata for a subset of 618.754 documents. This result is crucial for the goal of data preparation, since text similarity analysis is an instrument of plagiarism detection, which requires a sufficient amount of metadata in order to be useful to the expert conducting the plagiarism analysis. The latter, smallest subset contains ISSN values which allow for further studies on journal plagiarism and self-plagiarism statistics by implementing plagiarism detection on this subset. Using our candidate retrieval method, we have computed over 4.7 billion n-grams hashes for the 9.8 million documents in the 2018-03-01 CORE data set, yielding a total hash table size of 1478 GB in PostgreSQL, using b-tree indexes and document IDs in the form of SHA-256 hashes. On average, we found 87.37 candidates per document for each document in the CORE data set.

For our implementation of text matching comparison, we installed the Docker image containing the text matching REST API implementation on a cluster of 33 machines with Intel i5-8600 and 8 GB internal memory. Our configuration for the parameters equaled $minLen = 20$, $lengthBoundary = 40$ and $windowSize = 350$, respectively, as they are set used in the Slovenian open access infrastructure. The average document length in the CORE 2018-03-01 data set is 73.41 KB, we found setting the parameter N=20 allows for stable and efficient processing of the CORE data set using this approach. Table 3 contains benchmark statistics for a varible number of computing nodes running the text similarity Docker image. A random sample of 1000 documents is selected for a variable amount of nodes which process the workload in parallel. The efficiency of the approach tends to asymptotically decline as we increase the number of nodes, which is a result of the centralized data storage being used for synchronization of the compute nodes. The network infrastructure and the hardware and configuration of the database server represent a bottleneck. The table also contains a reference value for the time necessary to process all the 9.767.152 documents in the CORE 2018 data set for a given number of nodes. The benchmarks were performed on a subset of 10 nodes of the total 33 nodes used to process the entire CORE data set, but the results may be interpolated to a higher number of nodes.

Table 3: Distributed text similarity computation performance benchmarks.

#nodes	duration(s)	est. days (CORE 2018)
1	1590,806	179,83
2	753,917	85,23
3	547,659	61,91
4	425,046	48,05
5	374,564	42,34
6	306,461	34,64
7	294,918	33,34
8	285,888	32,32
9	233,364	26,38
10	217,015	24,53

5 CONCLUSION

The study described in this paper describes the methodology of establishing the largest plagiarism detection dataset in Slovenia. We have developed a framework of processing larger sets of documents into a pipeline for plagiarism detection, with means of metadata enrichment with the help of the CrossREF API and Microsoft Academic Graph. Our output allows for further study in the field of academic integrity, content similarity detection, stylometry and other fields of study. Utilizing our data set, which consists of enriched metadata entries for the 2018-03-01 CORE data set, including the similarity indices for each document in the data set, further research is possible.

6 ACKNOWLEDGMENTS

The authors thank Petr Knoth and the CORE team for providing the 2018-03-01 CORE data set and technical support during our efforts of obtaining data from CORE. Also, we thank David Schmid for publishing the MAG dump on Zenodo from April 2019.

References

[1] Bretag, T., and Mahmud, S. Self-plagiarism or appropriate textual re-use? *Journal of Academic Ethics 7*, 3 (Sep 2009), 193.

[2] Eaton, S. E., and Crossman, K. Self-plagiarism research literature in the social sciences: A scoping review. *Interchange 49*, 3 (Aug 2018), 285–311.

[3] Eide, D., and Huang, C. "microsoft academic graph schema". *Microsoft Academic Graph official website*. Accessed Jun 1, 2021. [Online] Available: `https://docs.microsoft.com/en-us/academic-services/graph/reference-data-schema`.

[4] Fartek, J. *Distributed generation of plagiarism detection reports*. J. Fartek, 2018. Bachelor thesis.

[5] García-Romero, A., and Estrada-Lorenzo, J. M. A bibliometric analysis of plagiarism and self-plagiarism through déjà vu. *Scientometrics 101*, 1 (Oct 2014), 381–396.

[6] Herrmannova, D., and Knoth, P. An analysis of the microsoft academic graph. *D-Lib Magazine 22* (09 2016).

[7] Kärkkäinen, J., Manzini, G., and Puglisi, S. J. Permuted longest-common-prefix array. In *Combinatorial Pattern Matching* (Berlin, Heidelberg, 2009), G. Kucherov and E. Ukkonen, Eds., Springer Berlin Heidelberg, pp. 181–192.

[8] Krokoscz, M. Plagiarism in articles published in journals indexed in the scientific periodicals electronic library (spell): a comparative analysis between 2013 and 2018. *International Journal for Educational Integrity 17*, 1 (Jan 2021), 1.

[9] Maurer, H., Kappe, F., and Zaka, B. Plagiarism - a survey. *Journal of Universal Computer Science 12* (01 2006), 1050–1084.

[10] Microsoft Academic. Microsoft academic graph. *Zenodo*. Accessed Jun 1, 2021. [Online] Available: https://doi.org/10.5281/zenodo.2628216, Apr. 2019.

[11] Sinha, A., Shen, Z., Song, Y., Ma, H., Eide, D., Hsu, B.-J. P., and Wang, K. An overview of microsoft academic service (mas) and applications. In *Proceedings of the 24th International Conference on World Wide Web* (New York, NY, USA, 2015), WWW '15 Companion, Association for Computing Machinery, p. 243–246.

[12] The CORE team. "core data providers". *CORE official website*. Accessed Jun 1, 2021. [Online] Available: https://core.ac.uk/data-providers?q=&size=10.