A FRAMEWORK FOR PLAGIARISM DETECTION IN ARABIC DOCUMENTS

Imtiaz Hussain Khan[1], Muazzam Ahmed Siddiqui[2], Kamal Mansoor Jambi[1] and Abobakr Ahmed Bagais[1]

[1]Department of Computer Science, Faculty of Computing and Information Technology, King Abdulaziz University, Saudi Arabia
[2]Department of Information Systems, Faculty of Computing and Information Technology, King Abdulaziz University, Saudi Arabia
ihkhan@kau.edu.sa, maasiddiqui@kau.edu.sa, kjambi@kau.edu.sa, abobakr.a.2012@gmail.com

ABSTRACT

We are developing a web-based plagiarism detection system to detect plagiarism in written Arabic documents. This paper describes the proposed framework of our plagiarism detection system. The proposed plagiarism detection framework comprises of two main components, one global and the other local. The global component is heuristics-based, in which a potentially plagiarized given document is used to construct a set of representative queries by using different best performing heuristics. These queries are then submitted to Google via Google's search API to retrieve candidate source documents from the Web. The local component carries out detailed similarity computations by combining different similarity computation techniques to check which parts of the given document are plagiarised and from which source documents retrieved from the Web. Since this is an ongoing research project, the quality of overall system is not evaluated yet.

KEYWORDS

Plagiarism Detection, Arabic NLP, Similarity Computation, Query Generation, Document Retrieval .

1. INTRODUCTION

Plagiarism is becoming a notorious problem in academic community. It occurs when someone uses the work of another person without proper acknowledgement to the original source. The plagiarism problem poses serious threats to academic integrity and with the advent of the Web, manual detection of plagiarism has become almost impossible. Over past two decades, automatic plagiarism detection has received significant attention in developing small- to large-scale plagiarism detection systems as a possible countermeasure. Given a text document, the task of a plagiarism detection system is to find if the document is copied partially or fully from other documents from the Web or any other repository of documents. It has been observed that plagiarists use different means to hide plagiarism so that a plagiarism detection system cannot catch plagiarism cases. In an interesting paper [1], Alzahrani and colleagues report different types of plagiarism, including verbatim/exact copy, near copy and modified copy. Whereas verbatim copy can easily be detected by a plagiarism detection system, modified copies pose real challenge

Dhinaharan Nagamalai et al. (Eds) : CCSEA, DKMP, AIFU, SEA - 2015
pp. 01–09, 2015. © CS & IT-CSCP 2015 DOI : 10.5121/csit.2015.50201

to find their original source because in such cases a plagiarist often makes heavy revisions in the original text by making use of structural and semantic changes.

Two approaches have commonly been used in developing such systems: extrinsic or external approach and intrinsic approach. The extrinsic plagiarism detection uses different techniques to find similarities among a suspicious document and a reference collection. In this approach, usually a document is represented as an n-dimensional vector where n is the number of terms or some derived features from the document. A number of measures are available to compute the similarity between vectors including Euclidean distance, Minkowski distance, Mahalanobis distance, Cosine similarity, Simple Matching Coefficient, and Jaccard similarity. This approach effectively detects verbatim or near copy cases, however, with the heavily modified copies the performance of an extrinsic-based plagiarism detection system is greatly reduced. On the other hand, in intrinsic plagiarism detection, the suspicious document is analyzed using different techniques in isolation, without taking a reference collection into account [2-3]. Assuming that a good-enough writing style analysis is available, this approach can effectively detect heavy-revision plagiarism cases or even plagiarism cases from a different language (multi-lingual plagiarism).

The research in automatic plagiarism so far has mostly been confined to English, paying little attention to other languages like Arabic. Research in automatic plagiarism detection for the Arabic language is much demanding and timely. This is because Arabic is the fourth most widely spoken language in the world, and most Arab countries, including Kingdom of Saudi Arabia, have adopted the use of e-learning systems in their educational institutions. In an e-learning environment, where students generally have an access to the World Wide Web, the problem of plagiarism can be very threatening. This calls for the development of state-of-the-art tools to automatically detect plagiarism in Arabic documents.

In this paper, we describe an ongoing plagiarism detection project which intends to develop an online plagiarism detection system for Arabic documents. The proposed plagiarism detection framework comprises of two main components, one global and the other local. The global component is heuristics-based, in which a potentially plagiarized given document is used to construct a set of representative queries. These queries are then submitted to Google via Google API to retrieve candidate source documents from the Web. Next, the local component carries out detailed similarity computations to detect if the given document was plagiarized from the documents retrieved from the Web or not.

Rest of this paper is organised as follows. In Section 2, related work is discussed. The scope of the proposed project and our approach is described in Section 3. The proposed plagiarism detection framework is outlined in Section 4 followed by discussion in Section 5. Finally, Section 6 concludes the paper.

2. BACKGROUND AND RELATED WORK

A considerable amount of research has focused on automatic plagiarism detection. Here we review some interesting literature on automatic plagiarism detection in natural language text; an in-depth discussion can be found in [4]. Various approaches have been proposed in past two decades to automatically find plagiarism in written documents. Earlier approaches are mainly based on fingerprinting, keyword matching (or term occurrence) and style analysis [5-10]. Brin et al. [5] developed COPS, a system designed to detect plagiarism in research articles using fingerprinting mechanism. Their system works in two phases. In a first phase, they eliminate the most common sentences, and then in a second phase the remaining text is compared to detect plagiarism. A notable limitation of their system is that it is based on exact copy of sentences and

therefore cannot deal with paraphrases, for example. Building on COPS, Shivakumar and Garcia-Molina [6] developed SCAM system for detecting identical documents, based on word level analysis. In SCAM, the original documents are registered to a dedicated server; an attempt to register plagiarised documents can be detected by comparing the latter with the already stored documents. This system works reasonably well for documents with high degree of overlap; however, its performance degrades significantly when there are small overlaps. Si et al. [7] developed CHECK, a plagiarism detection system for documents written in the same domain, for example Physics. Their system works incrementally: first they compare a set of primary keywords in the suspected and source documents, followed by a more fine-grained comparisons only if there was similarity at the top level. This is the kind of approach we aim to adopt in our research, but we also aim to build a plagiarism detection system in a domain independent way. In [8], Broder used document fingerprints to detect the overall similarity between suspected and source documents. He chose the smallest k-gram hashes from the entire document which permits detection of overall similarity between documents for duplicate detection, but not smaller overlaps between documents. Monostori and colleagues [9] built MatchDetectReveal system, which uses algorithms for exact string comparison. They represent the suspected document as a suffix tree data structure, without any loss of information, and then compare this document with other documents represented as strings of texts. The accuracy of their approach is good enough, but this is very time consuming and also requires a lot of space. In [10], Khmelev and Teahan, instead of using suffix trees, adopted the idea of suffix arrays to reduce the memory problem found in suffix trees. However, both Monostori et al. and Khmelev and Teahan do not take paraphrases into account.

Recently, some researchers have proposed to use natural language processing techniques to plagiarism detection [11-15]. Runeson et al. [11] proposed shallow NLP approaches to detect duplicate documents. They used tokenisation, stemming, and stop-word removal. Although the techniques used were simple, the authors reported promising results. Leung and Chan [12] put forward some proposals to apply advanced NLP techniques, including syntactic and semantic analysis to improve automatic plagiarism detection. They proposed to use WordNet to find the synonyms of the keywords used in the document under scrutiny, and compare these synonyms with the documents in the database. If it is suspected that the document under scrutiny contains some contents from the database, the sentences of the document would be further analysed for detailed comparison. In another study [13], Ceska and Fox applied pre-processing techniques to improve automatic plagiarism detection. They used simple heuristics, including numbers' replacement by dummy symbols, removing punctuations, and lemmatisation. Their results suggest a significant impact of applying NLP to plagiarism detection. In yet another study [14], Chong and colleagues applied various NLP techniques, varying from shallow techniques (e.g. simple string matching) to more advanced techniques (e.g. structure analysis of text). They used similarity metrics, including tri-gram similarity and longest common subsequence to measure the similarity scores between suspected and original documents. These similarity scores were then used to train a model which, rather than binary classification, classifies the documents under scrutiny into four levels: exact copy, light revisions, heavy revisions, and no plagiarism. They report promising results.

Very recently, Arabic NLP community has shown interest in developing plagiarism detection systems for Arabic language [16-19]. In [16], Alzahrani and Salim reported on an Arabic plagiarism detection system which combines the fuzzy similarity model and semantic similarity model derived from a lexical database. First, they retrieve a list of candidate documents for each suspicious document using shingling and Jaccard coefficient, and then they make sentence-wise detailed comparison between the suspicious and associated candidate documents using the fuzzy similarity model. Their preliminary results indicate that fuzzy semantic-based similarity model can be used to detect plagiarism in Arabic documents. In another study, Bensalem and colleagues [17] have developed a system which uses various stylistic features to account for intrinsic

plagiarism. The system was evaluated on a small corpus, so it is difficult to quantify its effectiveness. In yet another study, Menai [18] used a top-down approach, whereby in a first step a global similarity is measured between a suspicious document and candidate documents. In a second step, a detailed analysis is done at paragraph- and sentence-level.

3. SCOPE AND APPROACH

It is important to mention at the outset what is the scope of our work. We are interested in developing a web-based plagiarism detection system which can detect plagiarism cases in written Arabic documents. The scope of this project is limited to Arabic natural language text and we do not take plagiarism in a programming language code into account. Also we do not consider multi-lingual plagiarism. Rather, we address mono-lingual plagiarism in relatively smaller domain, student assignments and small-scale research papers (whose length is less than 50 pages). Moreover, we assume that the input suspicious document is plain text or a word document only. We do not consider other file formats like .pdf or .eps nor other modalities for example images. These assumptions will help us evaluate our system in a more informed and systematic way.
Our approach is hybrid, that is we incorporate both intrinsic and extrinsic techniques in the single framework. The former is mainly used in this project to generate queries to retrieve candidate documents, whereas the latter is used to thoroughly compute similarity between potential plagiarised and source documents. We consider the problem of plagiarism detection as falling under the general problem of finding similarity among documents.

4. THE PLAGIARISM DETECTION FRAMEWORK

The proposed plagiarism detection framework comprises of two main components, one global and the other local. The global component is heuristics-based, in which a potentially plagiarized given document is used to construct a set of representative queries. These queries are then submitted to Google via Google API to retrieve candidate source documents from the Web. Next, the local component carries out detailed similarity computations to detect if the given document was plagiarized from the documents retrieved from the Web or not. The plagiarism detection framework is depicted in Figure 1.

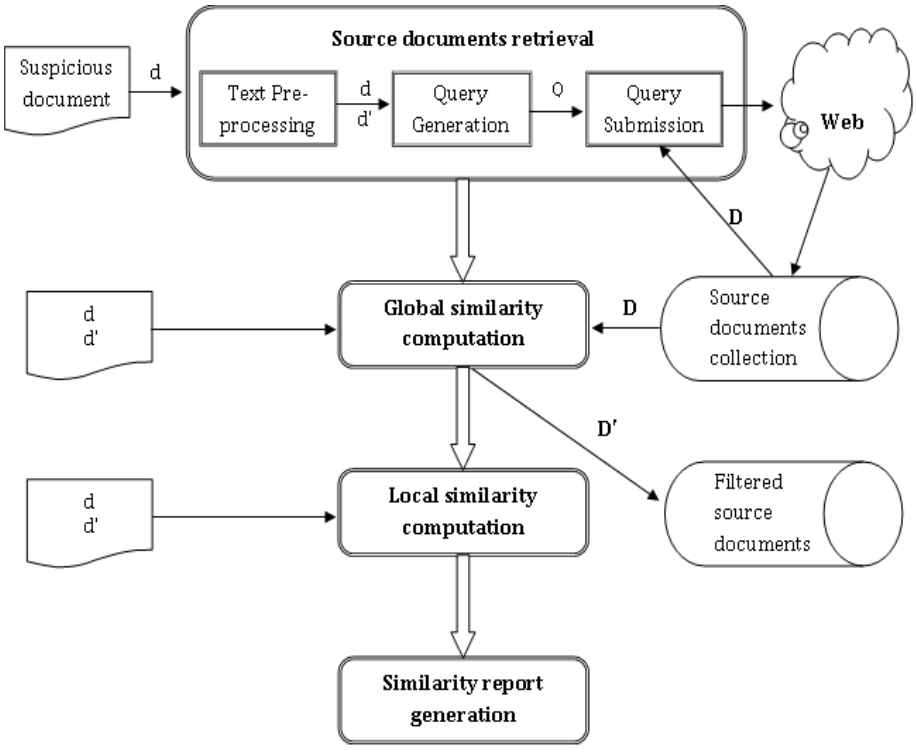

Figure 1: Plagiarism detection framework

In what follows, each component of the proposed framework is discussed in turn. (It is important to mention here that source document retrieval (4.1) is fully implemented and thoroughly evaluated; global and local similarity components (4.2, 4.3) are partially implemented and tested.)

4.1. Source Document Retrieval

We developed an information retrieval system which attempts to retrieve source documents from the Web against a given suspicious document. The system takes the suspicious document *d* as an input and goes through the following steps to find the potential source documents.

a) **Text pre-processing step:** The system starts by pre-processing the input suspicious document *d*. First of all, the document *d* is converted into a standard UTF-8 file format. Next, it is tokenized into words using a custom-built Java tokeniser. The resulting words are then converted to their base form using Khoja stemmer [20]. Then, the document is segmented into sentences which allows line-by-line processing in the subsequent phases. Finally, the stopwords (functional words which are common across documents, for example في meaning in) and other punction marks are removed to generate a pre-processed document *d*'.

b) **Query generation step:** Different query generation heuristics are used to generate a set of queries Q from the given suspicious document d. The query generation module takes the document d, the pre-processed document d' and the query generation heuristic as input and returns a set of queries Q as output. We developed different document retrieval heuristics, including key-phrase based heuristic, variance in readability score accross sentences and first sentence in every paragraph of the document. These heuristics were

thoroughly evaluated in terms of precision, recall and f-measure on a sizeable corpus [21] and selected the top three best performing heuristics. The evaluation results also showed that a combination of those three select heuristics was significantly better than the each individual heuristic that is why we combine them for the source document retrieval. In is instructive to briefly describe here one heuristic, namely key-phrases based heuristic (for details, see [18]).This heuristic takes the pre-processed document d'. We sampled a set of top N (N = 5 in this study) distinct keywords, based on the frequency of each word in the entire document. Then, for each keyword we constructed a phrase (henceforth key phrase) by taking two preceding and two succeeding words, at its first appearance in the original document (i.e., without preprocessing). If the keyword appeared at the beginning (or end) of a sentence, four preceding (or four succeeding words) words were used to construct the key phrase. An example, key phrase is "جيل من الطلاب قادر على" ("A generation of students capable of"), in which keyword is underlined. It is important to mention here that we developed different heuristics and thoroughly evaluated their performance in terms of precision, recall and f-measure on the corpus [21] developed as part of our project.

c) **Query submission step:** Queries Q are submitted (one at a time) to the Web via Google's search API to retrieve source documents. Google's search API attempts to find relevant documents from the Web and returns the results including URL of the source document. Subsequently, these URLs are extracted from the returned results and the respective documents (at the URL) are downloaded and saved locally. The query submission procedure works as follows. The first query is submitted to the Web and top 10 matching documents are downloaded, maintaining a set D of documents. Subsequently, a query is only submitted to the Web if its *extension*, denoted as [[q]], does not contain a document in the local document collection D. Extension of a query, [[q]], is a set of documents which contains q. Our query submission approach is similar in spirit to Haggag and El-Beltagy [22], but we compute [[q]] in a different way. In Haggag and El-Beltagy's case, a document $d \in D$ is in the [[q]] set if 60% or above tokens in q are also present in d. They do not take position of those tokens into account though. We compute the extension of a query [[q]] by using Ferret tri-gram model [23]. Accordingly, a document $d \in D$ is in the [[q]] set if a sequence of three query words appear in d. It is important to remember here that we use 5-words long queries, generated in the previous step (see above).

4.2. Global Similarity Computation

After downloading the source documents from the Web in a local repository, the next step is the detailed similarity analysis to find which parts of the suspicious document are plagiarised from which documents in D. However, before carrying out this task, the source document collection D needs some necessary pre-processing. This is because the documents in D may contain some unnecessary HTML tags, which need to be cleaned up to extract the actual text. We implemented an HTML clean-up module which does the necessary clean up. Moreover, the source documents are converted into one single file format UTF-8, which is also the format of the given suspicious document.

Before computing the detailed similarity between suspicious document and documents in D, it is important to incorporate some filtration process to discard some documents from D which may have very little similarity with the suspicious document. This is important to avoid some unnecessary computation, which may degrade the overall efficiency of the system. However, this step should provide a reasonable balance between computational cost and accuracy of the system. That is, only some unwanted documents from D should be filtered out with minimum computational cost. To achieve such a balance, we employed a simple document-level similarity

heuristic which computes the similarity between the suspicious document *d* and a source document *s* as follows (equation 1).

$$sim(d,s) = \left(\frac{|d \cap s|}{\min(|d|,|s|)}\right) \quad \text{----------} \quad (1)$$

We discard the document *s* from D (resulting a new document collection D', cf. Figure 1) if the similarity score *sim* is less than 0.2: experts suggest that around 20% similarity between two documents may not be considered as plagiarism. A preliminary investigation of our own corpus reveals that this similarity threshold (i.e. 0.2) is reasonable.

4.3. Local Similarity Computation

As mentioned earlier, we use both extrinsic and intrinsic plagiarism detection techniques to compute similarity between two documents. The detailed similarity computation module combines different similarity measures, including Euclidean distance, Minkowski distance, Mahalanobis distance, Cosine similarity, Simple Matching Coefficient, and Jaccard similarity, to find one final similarity score. The similarity between two documents (*d* and *s*) will be computed across two dimensions, precision and recall. Recall will indicate how much of *d* matches *s*, and precision will indicate the level of similarity, e.g. exact or near copy.

The local similarity module will also be spotting which sentence (or phrase of at least 5 consecutive words) is plagiarised from which source document on the Web. Such a pairing will be shown in the similarity report generated in the next step.

4.4. Similarity Report Generation

Finally, a similarity report for the suspicious document *d* will be generated, where the plagiarized parts of *d* will be highlighted with different colors indicating the source as shown in iThenticate and other well known plagiarism detection systems like Turnitin.

5. DISCUSSION

Building automatic plagiarism detection systems has gain much popularity and attention over past 20 years. Different plagiarism detection systems have been developed, however, the challenge still remains how to effectively identify the plagiarism cases. The challenge is even worse for Arabic language because of its complex and morphologically rich nature. In this paper, we proposed a plagiarism detection framework for Arabic. This research raised some interesting questions some of them were unexpected:

- Performance of our system is partially dependant on the accuracy of Google search results. This is because in a first step, we retrieve potential source documents from the Web using Google's search API. We believe that with the improvement of Google search techniques, particularly use of synonymy and other related techniques accuracy of our system would increase significantly. This is important, because if potential source documents are not retrieved in this initial stage, accuracy of subsequent stages would degrade accordingly.
- Google has limitation on maximum number of submissions per day: 100 queries per free-subscription account. Also, it places limits on query length. Moreover, Google results contain HTML tags in the returned documents, so HTML cleanup is necessary to extract the actual text.
- Global similarity computation may exclude some potential source documents. Care must be taken in selecting an appropriate threshold value. A preliminary value 0.2 seems

reasonable but more and thorough experimentation is needed to adjust this threshold value.

- One important aspect of Arabic writing came to fore during the corpus analysis. In Arabic documents, sentence length vary pretty unpredictably: we found sentences of length 3 words only or as maximum as 250 words. This may greatly affect intrinsic techniques to plagiarism detection which are mainly based on readability score.

6. CONCLUSION

This paper described the proposed plagiarism detection framework for Arabic documents. The proposed plagiarism detection framework comprises of two main components, one global and the other local. The global component is heuristics-based, in which a potentially plagiarized given document is used to construct a set of representative queries by using different best performing heuristics. These queries are then submitted to Google via Google's search API to retrieve candidate source documents from the Web. Next, the local component carries out detailed similarity computations to detect if the given document was plagiarized from the documents retrieved from the Web or not. The global component is thoroughly evaluated, whereas the local component is partially implemented so far.

In future, we intend to integrate the different components of the system to build one final web-based plagiarism detection system. We will be thoroughly investigating the performance of different similarity measures before incorporating them in the final similarity computation model. The implemented system would then be thoroughly evaluated using our own corpus [21] before deploying.

ACKNOWLEDGEMENTS

This work was supported by a King Abdulaziz City of Science and Technology (KACST) funding (Grant No. 11-INF-1520-03). We thank KACST for their financial support.

REFERENCES

[1] Alzahrani, S.M., Salim, N.& Abraham, A.(2012). Understanding plagiarism linguistic patterns, textual features, and detection methods. IEEE transactions on systems, man, and cybernetics—part c: applications and reviews, 42(2), pp. 133-149.

[2] Eissen, M., Stein, B. & Kulig, M.(2007). Plagiarism detection without reference collections. In Proceedings of the advances in data analysis, pp. 359–366.

[3] Benno, S., Moshe, K. & Efstathios, S.(2007). Plagiarism analysis, authorship identification, and near-duplicate detection. In Proceedings of the ACM SIGIR Forum PAN07, pp 68–71, New York.

[4] Clough, P. (2003). Old and new challenges in automatic plagiarism detection. National Plagiarism Advisory Service, (February edition).

[5] Brin, S., Davis, J., & Garcia-Molina, H.(1995). Copy detection mechanisms for digital documents. In proceedings of the ACM SIGMOD annual conference.

[6] Shivakumar, N., & Garcia-Molina, H.(1996). Building a scalable and accurate copy detection mechanism. Proceedings of the first ACM international conference on digital libraries.

[7] Si, Leong, H.V., & Lau, R.W.(97). CHECK: A document plagiarism detection system. In Proceedings of ACM symposium for applied computing, pp. 70-77.

[8] Broder, A.Z. (1997). On the resemblance and containment of documents. In compression and complexity of sequences , pp. 21-29.

[9] Monostori, K., Zaslavsky, A., & Schmidt, H. (2000). MatchDetectReveal: Finding overlapping and similar digital documents. In proceedings of information resources management association international conference, pp. 955-957.

[10] Khmelev, D., & Teahan, W. (2003). A repetition based measure for verification of text collections and for text categorization. In Proceedings of the 26th annual international ACM SIGIR conference on research and development in information retrieval, pp. 104-110.
[11] Runeson, P., Alexandersson, M., & Nyholm, O. (2007). Detection of duplicate defect reports using natural language processing. In proceedings of 29th international conference on software engineering, pp. 499-510.
[12] Leung, C.-H., & Chan, Y.-Y. (2007). A natural language processing approach to automatic plagiarism detection. In proceedings of the 8th ACM SIGITE conference on information technology education, (pp. 213-218).
[13] Androutsopoulos, I., & Malakasiotis, P.(2009). A Survey of paraphrasing and textual entailment methods. Technical report, Athens University of Economics and Business, Greece.
[14] Ceska, Z., & Fox, C.(2009). The influence of text pre-processing on plagiarism detection. In recent advances in natural language processing, RANLP'09 .
[15] Chong, M., Specia, L., & Mitkov, R. (2010). Using natural language processing for automatic detection of plagiarism. In proceedings of 4th international plagirism conference.
[16] Alzahrani, S.M. & Salim, N. (2009) Fuzzy semantic-based string similarity for extrinsic plagiarism detection. In Proceedings of the 2nd international conference on the applications of digital information and Web technologies., London, UK.
[17] Bensalem, I.Rosso, P. & Chikhi, S. (2012). Intrinsic plagiarism detection in Arabic text: preliminary experiments. In Proceedings of the 2nd Spanish conference on information retrieval, Spain.
[18] Menai, M.(2012) Detection of plagiarism in Arabic documents. International journal of information technology and computer science (IJITCS), 4(10).
[19] Khan, I.H.,Siddiqui, M. Jambi, K. M., Imran, M & Bagais, A. A. (2014). Query optimization in Arabic plagiarism detection: an empirical study. To appear in International Journal of Intelligent Systems and Applications.
[20] Khoja, S.(1999). Stemming Arabic Text. Online available: http://zeus.cs.pacificu.edu/shereen/research.htm.
[21] Siddiqui, M.A., Elhag, S.,Khan, I.H., & Jambi, K. M. Building an Arabic plagiarism detection corpus. To appear in language resources and engineering.
[22] Haggag, O. & El-Beltagy, S. (2013). Plagiarism candidate retrieval using selective query formulation and discriminative query scoring. In proceedings of PAN, CLEF.
[23] Ferret (2009). Online available at University of Hertfordshire: http://homepages.feis.herts.ac.uk/~pdgroup/.

AUTHORS

Imtiaz Hussain Khan is an assistant professor in Department of Computer Science at King Abdulaziz University, Jeddah, Kingdom of Saudi Arabia. He received his MS in Computer Science from the University of Essex UK in 2005 and PhD in Natural Language Generation from the University of Aberdeen UK in 2010. His areas of research are Natural Language Processing and Evolutionary Computation.

Muazzam Ahmed Siddiqui is an assistant professor at the Faculty of Computing and Information Technology, King Abdulaziz University. He received his BE in electrical engineering from NED University of Engineering and Technology, Pakistan, and MS in computer science and PhD in modeling and simulation from University of Central Florida. His research interests include text mining, information extraction, data mining and machine learning.

Kamal Mansoor Jambi is a professor in Department of Computer Science at King Abdulaziz University, Jeddah, Kingdom of Saudi Arabia. He received his Masters (Computer Science) degree from Michigan State University, USA in 1986. He earned his PhD (Artificial Intelligence, OCR) degree from the Illinois Institute of Technology, Chicago, USA in 1991. His areas of research are Natural Language Processing, Speech Recognition, OCR and image processing.

Abobakr Ahmed Bagais received his BSc degree in Computer Science from King Abdulaziz University, Saudi Arabia. He is currently pursuing M.E. in Network optimization from King Abdul-Aziz University. His areas of interest include Arabic natural language processing, bioinformatics and optimization network.