ISSN 2319 – 7595
Volume 7, No.5, September - October 2018
International Journal of Information Systems and Computer Sciences
Available Online at http://warse.org/IJISCS/static/pdf/file/ijiscs03752018.pdf
https://doi.org/10.30534/ijiscs/2018/03752018

An Enhanced Text Mining Approach using Dynamic Programming

Desmond Bala Bisandu[1], Nentawe Yusuf Gurumdimma[2], Mammuam Titus Alams[3], Dachollom Dorcas Datiri[4]
[1, 2, 3, 4] Department of Computer Science University of Jos, Plateau State, Nigeria
bisandud@unijos.edu.ng, yusufn@unijos.edu.ng, alamsm@unijos.edu.ng, datirid@unijos.edu.ng
[1]Corresponding author

ABSTRACT— Text mining is a pervasive area of research in Computer Science. This is the process of finding knowledgeable and useful information and patterns from very huge text. It is widely applied in several areas of applications such as information retrievals; computational biology etc. Keyword-based searching has been extensively applied in text dataset considering the keywords as strings. String matching is used in finding all the occurrences of a given pattern P of length m from a given text T of length n, where $m \leq n$. An occurrence of a pattern inside the text can simply be characterized as "exact" or "approximate". This paper proposes a framework for text mining using a fast bit-parallel algorithm for searching exact occurrences of a pattern inside a huge body of text. We evaluate the performance of three algorithms in the literature on different text files and discuss their suitability under different situations.

KEYWORDS— *Algorithms; Bit-Parallelism; Data Mining; String Matching; Text Mining*

1. INTRODUCTION

One of the fundamental problems in information retrieval is searching for key information within huge date sets. Since data on the web is doubling very quickly, this makes knowledge discovery increasingly more challenging and difficult to handle [1]. The act of extracting meaningful knowledge or patterns from a large data set is simply referred to as data mining. Data can be in the form of video, voice, image, document, or text [2, 3].

Data mining has different areas of applications such as text mining, image processing, and artificial intelligence [4]. Information available in the text data-sets can be structured, semi-structured, or unstructured such as in the form of research papers, and electronic mails [5, 42, 43]. Text mining is the technique of mining information from textual data [6].

Searching the text is required to extract knowledge. It is widely applied in numerous areas such as information retrievals, computational biology, text editor, image processing, data science, and search engines [7, 8]. Keywords based searching works by considering series of strings [9].

String matching is used in finding the occurrence of a given pattern P in the given text T [10, 11]. Pattern occurrence can simply be considered as "approximate" or "exact" [12]. Some examples are: exact pattern matching, approximate pattern matching, regular expression searching, and online string searching (used for casual searching) [13]. In order to speed up the searching, there is a high quest for better algorithms [14]. Recently, the bit-parallelism algorithm has been efficiently used to solve pattern matching problems [15]. This kind of algorithm takes advantage of high-speed processor.

This research proposes a framework for text mining using string matching and bit-parallelism with the goal of addressing the problem of polysemy and synonymy in keyword search on textual documents.

The rest of the paper is organized as follows. Section II presents a study of related work. Section III and Section IV discusses the methods, procedures, and results discussion respectively. Finally, Section V is the conclusion and directions for future work.

2. RELATED WORK

The explosive growth of heterogeneous unstructured data on the Internet [16], prompts for efficient ways of exchanging information such as dissertations, and electronic mails through World Wide Web (www) [2, 17]. Generally, frameworks for the exchange of such information are needed, for this purpose data mining mechanism are in high demand [11]. Text mining techniques that are used in converting unstructured data into structured / knowledge discovery have numerous areas of applications such as text searching, string matching, and machine learning [12, 18].

Pattern matching and retrieval of information in text consider the key principles of data mining such as classical string matching [19, 20]. The string matching problem is a pervasive problem widely applied in numerous applications area such as image searching, computational biology, and plagiarism detection; generally using the two classes; exact and approximate string matching [21, 22, 23].

Gope and Behera in [24] presented a novel algorithm for pattern matching in the area of bioinformatics. The approach finds a unique and exact sequence occurrence from the given DNA gene dataset using a genetic algorithm (GA). The drawback of the genetic algorithm is in the process of choosing a prime number (q), it has a worst-case complexity time of O((n-m+1)m). Similarly, Faro in [14] also presented a string matching algorithm for the exact online pattern matching on the DNA sequences of the genome. With a large amount of biological data, for instance looking at the length of the given pattern which constitutes a four character set {A, C, G, T} is making the problem more and more difficult in providing efficient solutions due to the volume of the dataset.

The authors in [25] proposed a two-way shift-OR (TSO) algorithm which is the enhancement of the traditional Shift-And algorithm, with a running time of O(n/m).

Islam and Talukder in [26] proposed a pattern discovery algorithm framework; using the Index Based Shift (IBS) a word based pattern matching algorithm in the area of computational biology to analyze textual data either using the exact (error-free) or approximate (limited error) pattern matching.

Jangra & Nagpal in [28] proposed a searching algorithm for pattern matching based on sequence token. The advantages of the algorithm are that the time consumption is reduced. Secondly, patterns are represented in a sequential form which performs the matching using backtracking techniques. Gupta and Vasgi in [17] proposed an algorithm for relevant text document retrieval. The algorithm considered searching in a pattern that performed by passing through some major process techniques such as deploying patterns and evolving patterns (noisy removal). Analogously Anujna and Ushadevi in [29] proposed an algorithm using Porter D algorithm to remove stop words and stemming (tail part removal in the text) using regular expression algorithm.

3. METHODS AND PROCEDURES

One of the techniques used in machine learning and data mining for extracting information from text is called text mining. Text mining is the process of extracting meaningful knowledge from a text data; it is also known to be the act of discovering knowledge from the text [11]. Some major challenges researchers face in text mining knowledge extraction stems from the lack of frameworks that can efficiently extract knowledge in data with noise, word ambiguity, sensitive context, synonymy, and polysemy (automobile = vehicle = car = Honda). Application of text mining is not only limited to bioinformatics (DNA and RNA), but also in text searching, string matching, soft computing, machine learning, and artificial intelligence [30]. Text data is said to be in a string if it is in a raw format, which represents a sequence of characters. String searching and retrieval of information in text is done using text mining techniques based on string matching [31, 32].

A. Exact String Matching

The exact string matching problem (ESM) can be solved using a linear time algorithm when all patterns are completely identified [12, 19]. Exact string matching *(without allowing errors*) is to find the positions of a text where a given pattern occurs; it does not allow any number of "errors" in the matches [33]. Numerous algorithms have been developed for solving exact string matching problems such as *Naïve Approach Algorithm, Knuth-Morris-Pratt Algorithm, and Boyer-Moore Algorithm etc* [34]. The ESM problem in Computer Science is pervasive due to its different areas of applications such as: data compressions, image processing, computational biology, signal processing, and retrieval of information [19, 35, 36].

B. Backward Directed Acyclic Word Graph (DAWG) Matching

Backward DAWG matching (BDM) uses the suffix automaton techniques by constructing a Directed Acyclic Word Graph (DAWG) using the reversed patterns for finding all suffixes [17, 37]. The suffix of the reverse pattern is same as the prefix of the original pattern. The pattern is shifted when a mismatch occurs, while the automaton continues reading text from the previous shift position BDM Algorithm **[**38].

C. Shift-And Algorithm

The bit-parallel Shift-And algorithm was invented by [39]. Its original idea was to match a pattern inside a body of huge text. The algorithm becomes important as it speed up the operation by cutting the number of steps to $\lceil n/w \rceil$, where n is the size of the text and w is word length of the computer used. The algorithm encodes the pattern in the form of bit-sequence (G) and constructs a finite automaton (with state vector K) to process the character of the text. Here the length of G is $\geq$ m (m being the length of the pattern) and the state vector is updated by $K \leftarrow (K<<1) \mid G[T[i]]$, where T is text [40, 41].

D. Backward Non-deterministic Matching (BNDM)

Navarro & Raffinot, (1998) state reason behind bit-parallelism implementation. It is simply to speed up the searching operations, as this technique takes advantages of intrinsic parallelism of the bit operations inside a computer word. BNDM is implemented using the Shift–And Algorithm [35] and BDM, firstly by building a G table which for each and every z character a bit mask $g_m \ldots g_1$ is stored.

The mask in G [z] has the i^{th} bit set if and only if $x_i = z$. The state of the search is kept in a machine word $K=k_m \ldots k_1$, where k_i is set whenever $x_1 x_2 \ldots x_i$ matches the end of the text read up to now (State k_i is set if and only if it is active). Therefore, we report a match whenever k_m is set. We set $K=0^m$ originally and, for each new character s_j, updates K using the formula [38].

$K' \leftarrow ((K<<1) \mid 0^{m-1}1) \,\&\, G[s_j]$

The beauty of this BNDM algorithm is that, it works only when m (pattern length) $\leq$ w (machine word size). The algorithm is as shown in Figure 1.

BNDM ($P=p_1p_2\ldots p_m$, $T=t_1t_2\ldots t_n$)

1. for $c \in \Sigma$ do $B[c] \leftarrow 0^m$
2. for $i \in 1\ldots.m$ do $B[P_{m-i+1}] \leftarrow B[P_{m-i+1}] \mid 0^{m-i}\,1\,0^{i-1}$
3. pos$\leftarrow$0
4. while pos $\leq$ n - m
5. do j $\leftarrow$ m, last $\leftarrow$ m, D$\leftarrow 1^m$
6. while D$\neq 0^m$
7. do D$\leftarrow$D& $B[t_{pos+j}]$
8. j$\leftarrow$j-1
9. if D& $10^{m-1} \neq 0^m$
10. then if j>0 then last $\leftarrow$j
11. else report an occurrence at pos+1
12. D$\leftarrow$D<<1
13. pos $\leftarrow$ pos + last

Figure 1: BNDM Algorithm [38]

E. Proposed Framework

The newly proposed framework consists of the following modules: text collection, preprocessing stages which include: removal of punctuations, extra-whitespaces removal, text stemming and lastly the stop words removal from the text, exact string matching using the Bit-parallelism: Backward Non-deterministic DAWG Matching (BNDM) discovered patterns and analysis. Figure 2 shows the proposed framework used in this research.

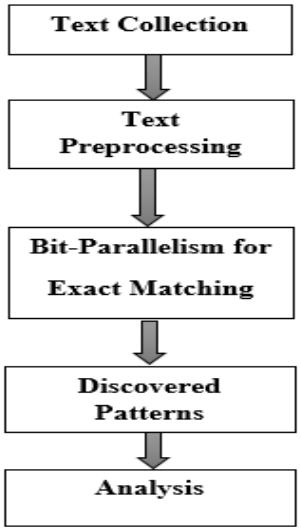

Figure 2: Proposed framework

4. RESULTS DISCUSSION

A. Presentation of Results

A framework for the text mining using string matching and bit-parallelism has been implemented using the R programming language version 3.4.3 and C development environment. R programming is used during the preprocessing phase which includes removal of all punctuations, all extra-whitespaces, stemming of the text, and lastly stop words. The C language is used in the string matching phase, fundamentally on the techniques of exact (without any error) string matching algorithm for finding all the occurrences of a pattern from a given text using the Naive, Shift-And and Backward Non-deterministic DAWG Matching (BNDM) algorithms. Reason behind choosing above two languages is that the R provides very good tools for preprocessing and C is highly efficient for bit-parallel algorithms.

B. Presentation of Performance

Performance-based on execution time and speed was recorded using all the three algorithms (Naïve, Shift-AND, and BNDM). For our preprocessing the compilation time and the running time for all the process (punctuation removal, extra-whitespaces removal, stemming of text, and lastly the stop words removal) was generally the same since we are using the R programming language for the preprocessing tasks. Search Time was recorded using the compilation + running time of the same text file after preprocessing using the Dev-C.

The textual data with file name **testcrude.txt** from (http://staff.utia.cas.cz/vomlel/data/TEST_CRUDE_DAT.GZ **)** has been loaded on the R platform; the program is coded to read in all the text line by line and displays 4176 as the total number of characters in the entire text. Table I show that BNDM algorithm has the best performance in general with 1.01 sec. as compile time and 0.99 sec. running time, and cumulative time of 2.0 sec. Fig. 3 is the bar plot of the performance of all the three algorithms based on execution time.

On increasing the file size, we used the 16 bit size, pattern = "crude". The results are shown in Table II. We noticed that the number of the pattern occurrences increases as soon as we increase the file size. From Table II, BNDM has the best performance in terms of speed with both compilation and running time. This table also shows that Naïve performs better on smaller text size but drops in performance behind the shift-and as text size increases. The study has been conducted on another data set which has been collected from the repository https://archive.ics.uci.edu/ml/datasets/Car+Evaluation. Table III shows the results of running BNDM algorithm on this data set.

Data set on car evaluation has the same objectives on finding the rate on how many people that uses car based on all the attributes: *unacc, med, small, low, high, vhigh, big, more*. After preprocessing our textual data set on the same R programming language environment, we recorded a number of 53,516 character sets, and after running the program several time on Dev-C environment in searching based on the individual attributes, BNDM algorithms still gives best result as shown in Table 3 and Figure 4.

Our findings show that a total number of 2500 people buy cars based on the attribute "high", which displayed a very simple and straightforward bar chart plotted from the R environment.

5. CONCLUSION AND FUTURE WORK

We concluded our research findings by demonstrating a better performance in results and discussions. The study has been classified into two distinct aspects of preprocessing (removal of punctuations, extra-whitespace removal, stemming of the text, and removal of stop words) the textual data set and then performing the matching on the pattern from the given text after preprocessing. Naïve, Shift-And, and bit-parallel BNDM algorithm have been applied for the implementation of finding the number of occurrences of pattern from the textual data sets. Performance was measured using the execution time and speed, which all comprises of both compilation and running time in seconds.

Looking into another data set on car evaluation with same objectives on finding the rate on how much number of people that uses car based on all this attributes *(unacc, med, small, low, high, vhigh, big, more).* After preprocessing our textual data set on the same R programming language environment we recorded a number of 53,516 character sets, and after running the program several time on Dev-C environment in searching based on the individual attributes. Bit-parallel BNDM algorithms still give a much better results in recording both the compilation and running time in seconds.

A. Future Work

As future work, this research study can further be experimented on the approximate string matching algorithm as well as more classes of string matching. Huge data set can be considered for the study and C can be integrated into R / Python programming language. The technique can be used to convert unstructured data to structured data which can be more helpful in decision making.

Table 1: Execution Time of the three Algorithms

Execution Time (sec)	Naïve		Shift-AND		BNDM	
	Compile	**Running**	**Compile**	**Running**	**Compile**	**Running**
Reading Text Files	0.00	0.03	0.00	0.03	0.00	0.03
Punctuations Removal	0.00	0.01	0.00	0.01	0.00	0.01
Extra-whitespaces Removal	0.00	0.03	0.00	0.03	0.00	0.03
Stemming of Text	0.00	0.79	0.00	0.79	0.00	0.79
Stop words Removal	0.00	0.05	0.00	0.05	0.00	0.05
Search Time	1.47	0.38	1.92	0.26	1.01	0.08
Total	**1.47**	**1.29**	**1.92**	**1.17**	**1.01**	**0.99**
	2.76		**3.09**		**2.0**	

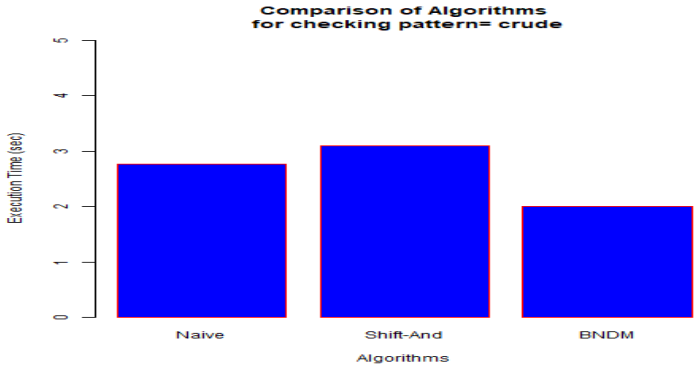

Figure 3: Results showing the comparison of all the algorithms

Table 2: Comparison of the Speed of three Algorithms on Increasing the File Size

Speed (sec)	**Naive Average time (running + compilation)**	**Shift-and Average time (running + compilation)**	**BNDM Average time (running + compilation)**	**No. of occurrences**
File Size: 10243	**1.85**	**2.18**	**1.09**	**76**
File Size: 25528	**3.64**	**3.08**	**2.43**	**631**
File Size: 68508	**4.58**	**3.36**	**2.89**	**645**
File Size: 99833	**5.49**	**4.01**	**3.06**	**659**

Table 3: Comparison of the Speed of the BNDM Algorithm on Car Evaluation

Speed (sec)	**BNDM Average time (running + compilation)**	**No. of people**
unacc	2.9	**1210**
Med	4.0	**2016**
small	1.4	**576**
Low	3.5	**1440**
High	5.0	**2304**
vhigh	2.1	**864**
big	1.3	**576**
more	2.3	**1008**

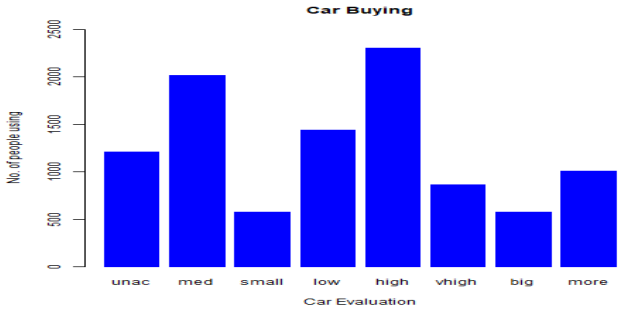

Figure 4: Results showing the comparison of the BNDM algorithm

REFERENCES

1 S. Hussain, "Survey on Current Trends and Techniques of Data Mining Research" *Expert Systems with Applications*, vol. 39, no. 4, pp. 4609-4617, 2017.

2 B. Singh and H. K. Singh, "Web data mining research: a survey," In *Computational Intelligence and Computing Research (ICCIR), IEEE International Conference on*, IEEE, 2010, pp.1-10.

3 D. K. Sharma and A. K. Sharma, "The Dark Web: Breakthroughs in Research and Practice: Breakthroughs in Research and Practice," *Deep Web Information retrieval Process*, vol. 5, no. 1, pp. 1-22, 2017.

4 K. S. Deepashri and A. Kamath, "Survey on Techniques of Data Mining and its Applications," *ACM*, vol. 6, no. 2, pp. 198-201, 2017.

5 K. V. Prasad, S. K. Saritha, and D. Saxena, "A Survey Paper on Concept Mining in Text Documents," *International Journal of Computer Applications*, vol. 166, no. 11, pp. 32-38, 2017.

6 S. Shehata, F. Karray, and M. Kamel, "An efficient concept-based mining model for enhancing text clustering," *IEEE Transactions on Knowledge and Data Engineering*, vol. 2, no. 10, pp. 1360- 1371, 2010. https://doi.org/10.1109/TKDE.2009.174

7 P. Patil, *Application for Data Mining and Web Data Mining Challenges*. Springer, 2017.

8 E. M. Tzanakou, *Supervised and unsupervised pattern recognition: feature extraction and computational intelligence*. CRC Press, 2017.

9 R. Sudhir, "A survey on image mining techniques: Theory and applications," *Computer Engineering and Intelligent Systems*, vol. 2, no. 6, pp. 44-52, 2011.

10 G. Kesavaraj and S. Sukumaran, "A study on classification techniques in data mining," In *Computing, Communications and Networking Technologies (ICCCNT), Fourth International Conference on*, IEEE, 2013, pp. 1-7.

11 Y. Zhang, *Image Understanding*. Walter de Gruyter GmbH & Co KG, 2017. https://doi.org/10.1515/9783110524130

12 V. Hugot, A. Boiret, and J. Niehren, "Equivalence of Symbolic Tree Transducers," In *International Conference on Developments in Language Theory*. Springer, Cham, 2017, pp. 109-121. https://doi.org/10.1007/978-3-319-62809-7_7

13 Y. Cheng, I. Izadi, and T. Chen, "Pattern matching of alarm flood sequences by a modified Smith–Waterman algorithm," *Chemical engineering research and design*, vol. 91, no. 6, pp. 1085-1094, 2013. https://doi.org/10.1016/j.cherd.2012.11.001

14 S. Faro and M. O. Külekci, "Efficient algorithms for the order preserving pattern matching problem," In *International Conference on Algorithmic Applications in Management*, Springer, Cham, 2016, pp. 185-196. https://doi.org/10.1007/978-3-319-41168-2_16

15 T. Chhabra, S. Faro, M. O. Külekci, and J. Tarhio, "Engineering order of preserving pattern matching with SIMD parallelism," *Software: Practice and Experience*, vol. 47, no. 5, pp. 731-739, 2017. https://doi.org/10.1002/spe.2433

16 N. Kofahi and A. Abusalama, "A framework for distributed pattern matching based on multithreading," *Int. Arab J. Inf. Technol.*, vol. 9, no. 1, pp. 30-38, 2012.

17 S. Gupta, R. Prasad and S. Yadav "Fast and Practical Algorithms for Searching the Gapped Palindromes," *Current Bioinformatics*, vol. 12, no. 3, pp. 225-232, 2017. https://doi.org/10.2174/1574893610666150828193203

18 S. Mitra and T. Acharya, *Data mining: multimedia, soft computing, and bioinformatics*. John Wiley & Sons, 2005.

19 G. Navarro and A. O. Pereira, "Faster compressed suffix trees for repetitive collections," *Journal of Experimental Algorithmics (JEA)*, vol. 2, no. 11, pp. 1-8, 2016. https://doi.org/10.1145/2851495

20 R. S. Boyer and J. S. Moore, *A computational logic handbook: Formerly notes and reports in computer science and applied mathematics*, Elsevier, 2014, pp. 32-44.

21 R. Prasad, A. K. Sharma, A. Singh, S. Agarwal, and S. Misra, "Efficient bit-parallel multi-patterns approximate string matching algorithms," *Scientific Research and Essays*, vol. 6, no. 4, pp. 876-881, 2011.

22 M. Aldwairi, A. M. Abu-Dalo, and M. Jarrah, "Pattern matching of signature-based IDS using Myers algorithm under MapReduce framework," *EURASIP Journal on Information Security*, vol. 2017, no. 1, pp. 9-15, 2017. https://doi.org/10.1186/s13635-017-0062-7

23 R. Singh, D. Rai, and R. Prasad, "A review on parameterized string matching algorithms," *Journal of Information and Optimization Sciences*, vol. 39, no. 1, pp. 275-283, 2018.

24 A. P. Gope and R. N. Behera, "A Novel Pattern Matching Algorithm in Genome Sequence Analysis," *IJCSIT) International Journal of Computer Science and Information Technologies*, vol. 5, no. 4, pp. 5450-5460, 2014.

25 T. Hirvola, H. Peltola, and J. Tarhio, "Improved Two-Way Bit-parallel Search⋆," *In Prague Stringology Conference*, 2014, pp. 71-75.

26 T. Islam and K. H. Talukder, "An improved algorithm for string matching using index based shifting approach," In *Computing, Analytics and Security Trends (CAST), International Conference of Computer and Information Technology (ICCIT)*, IEEE , 2017, pp. 138-143. https://doi.org/10.1109/ICCITECHN.2017.8281772

27 H. Peltola and J. Tarhio, "Alternative algorithms for bit-parallel string matching," In *International Symposium on String Processing and Information Retrieval* Springer, Berlin, Heidelberg, 2003, pp. 80-93. https://doi.org/10.1007/978-3-540-39984-1_7

28 A. Jangra & S. Nagpal, "Enhancement of Pattern Matching In Data Mining And Comparative Analysis With Knuth-Morris-Pratt, Boyer-Moore Algorithm". *International Journal of Research in Management, Science & Technology. Vol.* 5, no. 1, pp. 137-139, 2017.

29 M. Anujna and A. Ushadevi, "Converting and Deploying an Unstructured Data using Pattern Matching," *American Journal of Intelligent Systems*, vol. 7, no. 3, pp. 54-59, 2017.

30 M. Allahyari and K. Kochut, *Semantic tagging using topic models exploiting Wikipedia category network*, In Semantic Computing (ICSC), IEEE Tenth International Conference on. IEEE, 2016, pp. 63-70.

31 R. Baeza and G. Navarro, "New and faster filters for multiple approximate string matching," *Random Structures and Algorithms*, vol. 20, no. 1, pp. 23-49, 2002. https://doi.org/10.1002/rsa.10014

32 C. C. Aggarwal. *Data Mining*. Cham: Springer International Publishing, 2015.

33 K.. Fredriksson and S. Grabowski, "Efficient algorithms for pattern matching with general gaps, character classes, and transposition invariance," *Information Retrieval*, vol. 11, no. 4, pp. 335-357, 2008. https://doi.org/10.1007/s10791-008-9054-z

34 G. Cormode, and S. Muthukrishnan, "The string edit distance matching problem with moves," *ACM Transactions on Algorithms (TALG)*, vol. 3, no. 1, pp. 2, 2007. https://doi.org/10.1145/1186810.1186812

35 S. Faro and T. Lecroq, "The exact string matching problem: a comprehensive experimental evaluation," *ArXiv preprint arXiv: 1012.2547*. pp. 1-22, 2010.

36 R. Cánovas and G. Navarro, "Practical compressed suffix trees," In *International Symposium on Experimental Algorithms*. Springer: Berlin, Heidelberg, 2010.pp. 94-105. https://doi.org/10.1007/978-3-642-13193-6_9

37 G. Navarro, "A self-index on block trees,". In *International Symposium on String Processing and Information Retrieval*. Springer, Cham, 2017, pp. 278-289. https://doi.org/10.1007/978-3-319-67428-5_24

38 G. Navarro and M. Raffinot, "Fast and flexible string matching by combining bit-parallelism and suffix automata," *Journal of Experimental Algorithmics (JEA)*, vol. 5, no. 4, pp. 20-28, 2000. https://doi.org/10.1145/351827.384246

39 R. A. Baeza-Yates, "Improved string searching," *Software: Practice and Experience*, vol. 19, no. 3, pp. 257-271, 1989. https://doi.org/10.1002/spe.4380190305

40 K. Fredriksson and S. Grabowski, "Practical and optimal string matching," In *International Symposium on String Processing and Information Retrieval*. Springer, Berlin, Heidelberg. 2005, pp. 376-387. https://doi.org/10.1007/11575832_42

41 I. Hussain, S. Kausar, L. Hussain, and M. A. Khan, "Improved approach for exact pattern matching," *Int. J. Computer Science vol*. 10, no. 2, pp. 59-65,2013.

42 B. D. Bisandu, R. Prasad, M. M. Liman "Clustering news articles using efficient similarity measures and N-grams" *Int. J. of Knowledge Engineering and Data Mining vol.5, no.4,* pp.333-348, 2018. https://doi.org/10.1504/IJKEDM.2018.095525

43 Jain, Rekha, Rupal Bhargava, Sulochana Nathawat, and G. N. Purohit. "Retrieval of Web Pages Using Integrated Content and Structured Exploration." *International Journal of Information* 2, no. 2, pp. 9- 13, 2013.