International Journal of Innovative Research in Science and Engineering
Vol. No.3, Issue 04, April 2017
www.ijirse.com
ISSN (O) 2454-9665
ISSN (P) 2454-0663

ALGOMINE: A SYSTEM FOR EXTACTING AND SEARCHING FOR ALGORITHMS IN SCHOLARLY BIG DATA

Priyanka More[1], Nishant Bobade[2], Kiran Kshirsagar[3], Suraj Inamdar[4]

B.E, Department of Information Technology, Genba Sopanrao Moze College of Engineering Balewadi, Pune (India)

ABSTRACT

Algorithms are normally published in scholarly articles, especially in the computational sciences and related fields. The ability to mechanically find and retrieve these algorithms in this increasingly wide collection of scholarly digital articles would enable algorithm discovery, indexing, analysis, and searching. In recent time, AlgoMine, a search engine for algorithms, has been investigated as module of CiteSeerX with the intent of serving a large algorithm database. Now, over 200,000 algorithms have been mined from over 2 million scholarly articles. This paper proposes a new set of scalable techniques used by AlgoMine to search and retrieve algorithm representations in a different pool of scholarly articles. Specifically, hybrid machine learning approaches are proposed to discover algorithm presentation. Then, techniques to extract textual metadata for each algorithm are discussed. Conclusively, a demonstration version of AlgoMine that is built on Solr/Lucene open source indexing and search system is presented.

Keywords - Algorithm Search Engine, Ensemble Machine Learning, Scholarly Big Data

I. INTRODUCTION

Computer science and many of its applications are about developing, analyzing, and applying algorithms. Efficient solutions to important problems in various disciplines other than computer science usually involve transforming the problems into algorithmic ones on which standard algorithm seer applied. Furthermore, a thorough knowledge of state-of-the-art algorithms is also crucial for developing efficient software systems.

A significant number of scholarly articles in computer science and related disciplines contain high-quality algorithms developed by researchers. With dozens of new algorithms being reported in these conferences every year, it would be useful to have automated systems that efficiently identify, extract, index and search this ever increasing collection of algorithmic innovations. Such systems could provide an alternative source for researchers and software developers looking for cutting-edge algorithmic solutions to their problems.

Standard algorithms are usually collected and cataloged manually in algorithm textbooks [4] and websites that provide references for computer programmers. While most standard algorithms are already cataloged and made searchable, especially those in online catalogs, newly published algorithms only appear in new articles. The explosion of newly developed algorithms in scientific and technical documents makes it infeasible to manually

ISSN (O) 2454-9665
ISSN (P) 2454-0663

catalog these newly developed algorithms. Manually searching for these newly published algorithms is a nontrivial task. Researchers and others who aim to discover efficient and innovative algorithms would have to actively search and monitor relevant new publications in their fields of study to keep abreast of latest algorithmic developments. The problem is worse for algorithm searchers who are inexperienced in document search. Ideally, we would like to have a system that automatically discovers and extracts algorithms from scholarly digital documents. Such a system could prove to facilitate algorithm indexing, searching, and a wide range of potential knowledge discovery applications and studies of the algorithm evolution, and presumably increase the productivity of scientists.

Identifying and extracting various informative entities from scholarly documents is an active area of research. For algorithm discovery in digital documents, Bhatia, etal, have described a method for automatic detection of *pseudo-codes* (*PCs*) in Computer Science publications [6]. Their method assumes that each PC is accompanied by a caption Such a PC can then be identified using a set of regular expressions to capture the presence of the accompanied caption.

However, such an approach is limited in its coverage due to reliance on the presence of PC captions and wide variations in writing styles followed by different journals and authors. We found that 25.8% (71 out of 275) of the PCs did not have accompanied captions, and would remain undetected by their proposed approach. Furthermore, even though PCs are commonly used in scientific documents to represent algorithms, a majority of algorithms are also represented using *algorithmic procedures* (*APs*). An algorithmic procedure is a set of descriptive algorithmic instructions and differs from a PC in the following ways:

II. RELATED WORK

2.1 Summarizing figures, tables, and algorithms in scientific publications to augment search results

The textual metadata extracted from an algorithm representation is inadequate for effective extract, and propose to use the synopsis creation method. These search engines present a thumbnail view of the articles-elements, some articles metadata like the author title of the paper and title, and in lieu of a article snippet, typically, they represent the caption of the article element. While some authors in some disciplines write carefully tailored captions, generally, the author of a article assumes that the caption will be read in context of the text in the article. When the caption is presented out-of context as in a article-element-search-engine result, it may not contain enough information to help the end-user understand what the content of the article-element is. Consequently, end-users examining article-element search results would want a short “synopsis” of this information represented along with the article- element. Having access privileges to the synopsis allows the end-user to quickly understand the content of the article-element without having to read and download the entire article [1].

2.2 Finding algorithms in scientific articles:

It described a method for automatic detection of *pseudo-codes.* Their method assumes that each PC is accompanied by a caption. Such a PC can then be identified using a set of regular expressions to capture the presence of the accompanied caption. Algorithms are an integral part of computer science literature. However, none of the current search engines offer specialized algorithm search facility. We represent a vertical search engine that searches the algorithms present in articles and retrieves and indexes the related metadata and textual

ISSN (O) 2454-9665
ISSN (P) 2454-0663

description of the identified algorithms. This algorithm specific information is then utilized for algorithm ranking in response to user queries. Experimental results show the superiority of our system on other popular search engines [2].

2.3 generalized topic modeling approach for automatic document annotation:

Automatically textually enriches metadata records, to extract more meaningful textual information for PCs. Ecological and environmental sciences have become more advanced and complex, requiring experimental and observational data from multiple times, places, and thematic scales to verify their hypotheses. Over time, like this data have not only enlarged in amount, but also in diversity and heterogeneity of the data origins that spread throughout the world. This diversity poses a huge challenge for scientists who have to by hand search for desired data [3].

III. IMPLEMENTATION

A prototype of an algorithm search engine, *Algomine*, is presented the high level of the proposed system. First, scholarly articles or documents are processed to identify algorithms then, the textual metadata that provides necessary information about each detected algorithm is extracted. The extracted textual metadata is then indexed, and made searchable.

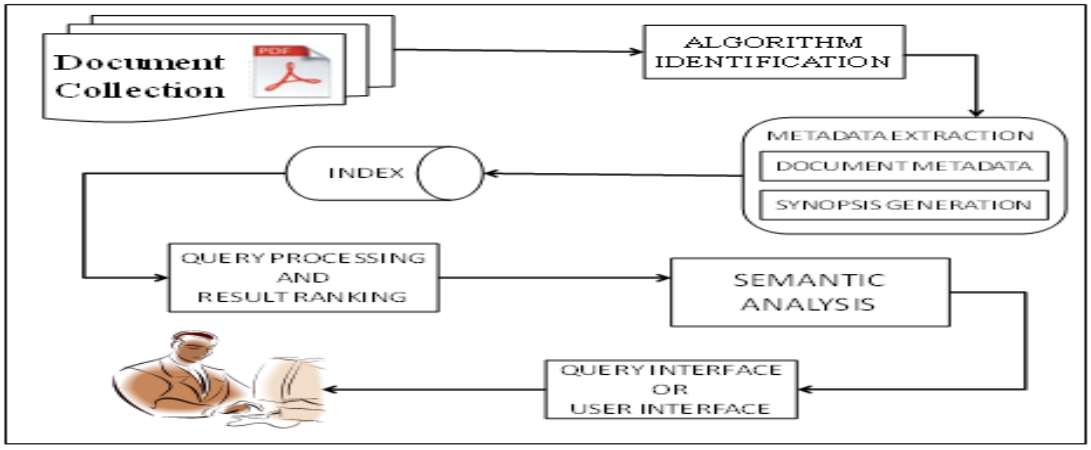

Fig 3: System Architecture

Note that, even though the methods presented in this paper are developed to handle algorithms in scholarly digital articles and papers, the methods could be generalized to other information sources that contain textual representations of algorithms such as pseudo-codes found in web-pages (e.g. Stack Overflow, Wikipedia, etc.). Algorithm extraction in html is different from algorithm extraction in free text, though there are some similarities. For example, WebPages are usually collected in HTML which provides html tags (e.g. <tr>, <td>, etc.) that can be utilized to detect boundaries of article elements, while such functionalities do not exist in free text data. It is also possible that the ideas presented in this algomine paper could be developed for other article elements such as tables and diagrams, textual information from such entities could be extracted.

3.1. Mathematical Model

S = {U, F, O}

Where

ISSN (O) 2454-9665
ISSN (P) 2454-0663

U – Set of users

F- Input file.

O – Set of an algorithm which is in file.

Consider a set U consisting of various users,

U= {U1, U2, U3….., Un}

IV. ALGORITHMS

4.3.1 Rule Based Method (PC-RB)

A rule based PC method, which utilizes a grammar for document-element captions to detect the presence of PC captions. We refer their method as our baseline (*PC-BL*) for the PC detection task [3]. A PC caption must contain at least one algorithm keyword, namely *pseudo-code*, *algorithm*, and *procedure*• Captions in which the algorithm keywords appear after prepositions are excluded, as these are not likely captions of PCs.

4.3.2 Machine Learning Based Method (PC-ML)

The Pseudo code rule based yields a high precision, however it still suffers from a low coverage resulting in a poor recall. We found that 25.8% of Pseudo code in our dataset do not have accompanied The accuracy evaluation quantifies how precisely each Pseudo code is cut into a sparse box. For each PC, we measure both the upper boundary delta (the start line number of the actual PC minus the start line number of the sparse box) and lower boundary delta (the end line number of the actual PC minus the end line number of the sparse box). 76.37% and 70.89% of Pseudo code have upper and lower line deltas of ± 2 lines respectively, suggesting that most Pseudo codes are fully and precisely extracted by the proposed sparse box cutting technique.

4.3.3 Combined Method (PC-CB)

Though the *Pseudo code-Machine learning* method can capture PCs even though they do not have accompanied captions, some *Pseudo code*s which are not first captured in a sparse box would still remain undetected. Mostly, such *Pseudo code*s are either written in a descriptive manner, or figures In our dataset DS2, 35 PCs (out of 275 actual *Pseudo code*s) cannot be captured using the sparse box extraction. However, 27 (out of 35) of these undetected *Pseudo code*s have accompanied captions and hence might still be detected using the *Pseudo code –Rule Based* method. We propose a combined method (*PC-CB*) of the *Pseudo code –Rule based* and the *Pseudo code –Machine learning* using a simple as follows:

STEP1 For a given document, run both *PC-RB* and *PC-ML*.

STEP2 For each PC box detected by *PC-ML*, if a PC caption detected by *PC-RB* is in proximity, then the PC box and the caption are combined.

4.3.4 Rule Based Method (AP-RB)

Often, Algorithmic procedure indication sentences have some common properties :The sentences usually end with *follows:*, *steps:*, *algorithm:*, *follows:*, *following:*, *follows.*, *step:*, *below:*, The sentences usually contain at least an algorithm keyword. We create a set of regular expressions to capture sentences according to the rules above.

5.3.5 Machine Learning Based Method (AP-ML)

A set of 26 features is extracted from each sentence. The features can be categorized into two groups such as content based features and context based features. The content based features are extracted from the sentence

ISSN (O) 2454-9665
ISSN (P) 2454-0663

itself, and are designed to learn the characteristics of the Algorithmic procedure indication sentences. The context based features are extracted from the 28 lines below the line where the sentence appears. The number 28 is the average number of lines of Algorithmic procedures observed in our data set.

V. MODULES OF THE SYSTEM

5.1 Algorithm identification in Scholarly Documents:

The system handles a large PDF documents are available in modern digital libraries including CiteSeerX are in PDF format. First, plain text data is extracted from PDF file. We use PDFBox13 to extract text data and modify the package and information such as location and font information from PDF document. Then, three sub processes operate in parallel, such as document segmentation, pc detection, and AP detection. The document segmentation identifies section in document. The PC detection detects pseudo code in parse text file. The AP detector first cleans extracted text and repairs broken sentences then identifies APs. After PCs and APs are identified, the final step is linking this algorithm. The final output would then be a set of unique algorithms.

5.2 Algorithm Metadata Extraction:

The conventional search engine method handles text documents. For an algorithmic procedure, corresponding text information could be the algorithmic procedure itself. However, detected pseudo-codes have no or small corresponding text. Furthermore, most of the PDF-to-Text tools are not designed for handling mathematical equations and pseudo codes. Due to complex configuration of symbols, font styles. This model describes a set of techniques for generating textual metadata for Pseudo codes.

5.3 Indexing and Searching for Algorithms

The algorithms are presented to the user or person in decreasing order . comparing the performance of our system with other search engine systems have shown supremacy of our approach in terms of precision and ranking performance. We selected a set of 50 popular algorithms as test queries and tested them with our system.

5.4 Semantic Analysis

LSA is an automatic mathematical and statistical technique for extracting and concluding relations of expected contextual usage of words in passages of discussion. It is not a traditional natural language processing or artificial intelligence program. it uses no humanly constructed any dictionaries, knowledge bases, semantic networks, grammars, syntactic parsers, or the like, and takes as its input only raw textual data parsed into words defined as unique character strings and splits into meaningful passages such as sentences or paragraphs. The first step is to represent the textual data as a matrix in which each row contain unique word and each column contain text passage or other contextual information. Each cell include the frequency with which the word of its row appears in the passage specify by its column. Next, the cell entries are tested to a preliminary transformation, whose details we will specify later, in which each cell frequency is weighted by a function that express carries information in the domain of discourse in general.

International Journal of Innovative Research in Science and Engineering
Vol. No.3, Issue 04, April 2017
www.ijirse.com
ISSN (O) 2454-9665
ISSN (P) 2454-0663

VI. RESULT ANALYSIS

6.1SCREENSHOTS

In our project we can develop the Algorithm base searching system. In project we can find algorithm this process can be consume less amount of time. In this picture user Uploaded pdf file can be converted into text format.

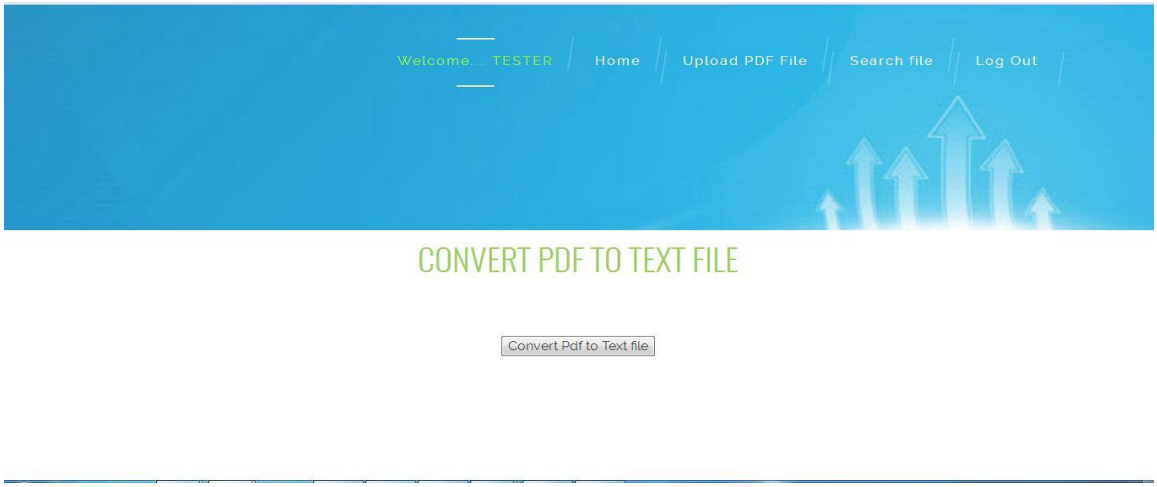

Fig 6.1.1: PDF to Text File

In this picture find the algorithm from the text converted file and display algorithm only in new window.

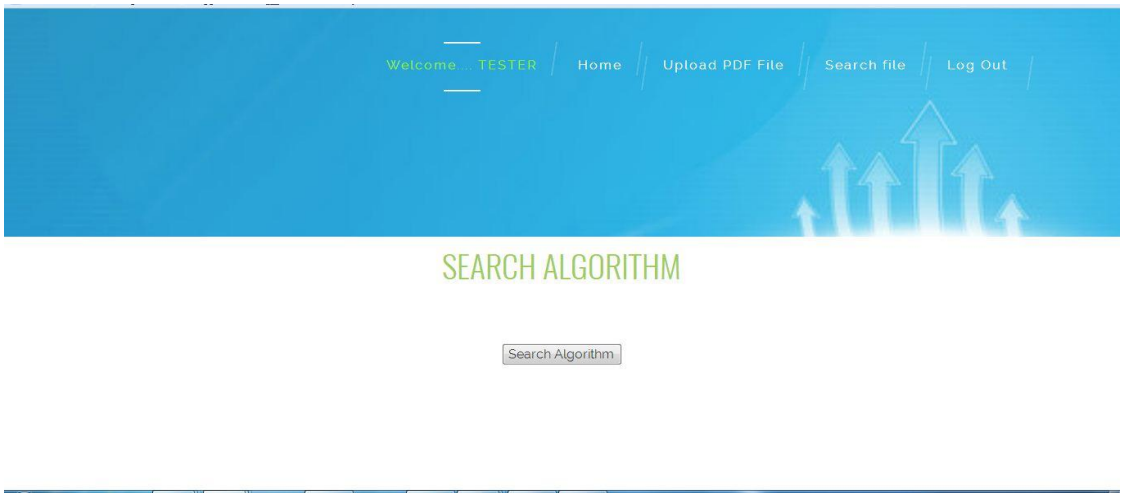

Fig 6.1.2: Search Algorithm

In this picture final output is display.

Algorithm 1 Upload Algorithm 1: procedure UPLOADFILE(D, PK, DKE, A, TTL) 2: encrypt the input data with DKE 3: buffer ? DataEncrypt(DKE,D) 4: if connect(DataSever)=fail then 5: return fail 6: end if create a ?le in the storage server 7: and write buffer into it 8: DataServer.?le? buffer 9: CM ? DEKEncrypt(PK,DKE,A) 10: use Shamir's algorithm to get key shares 11: k is count of data servers in CloudSky 12: sharedkeys[1...k] ? ShamirSecShare(n, k, CM) 13: for ?i ? [1...k] do 14: if connect(DataServer[i])=success then 15: create object(sharedkeys[i],TTL) 16: else 17: for ?j ? [1...i] do 18: delete key shares created before this one 19: end for 20: return fail 21: end if 22: end for 23: return success 24: end procedure key shares generated by Shamir's secret sharing algorithm,
****

Fig 6.1.3: Extracted Algorithm

ISSN (O) 2454-9665
ISSN (P) 2454-0663

6.2 GRAPH

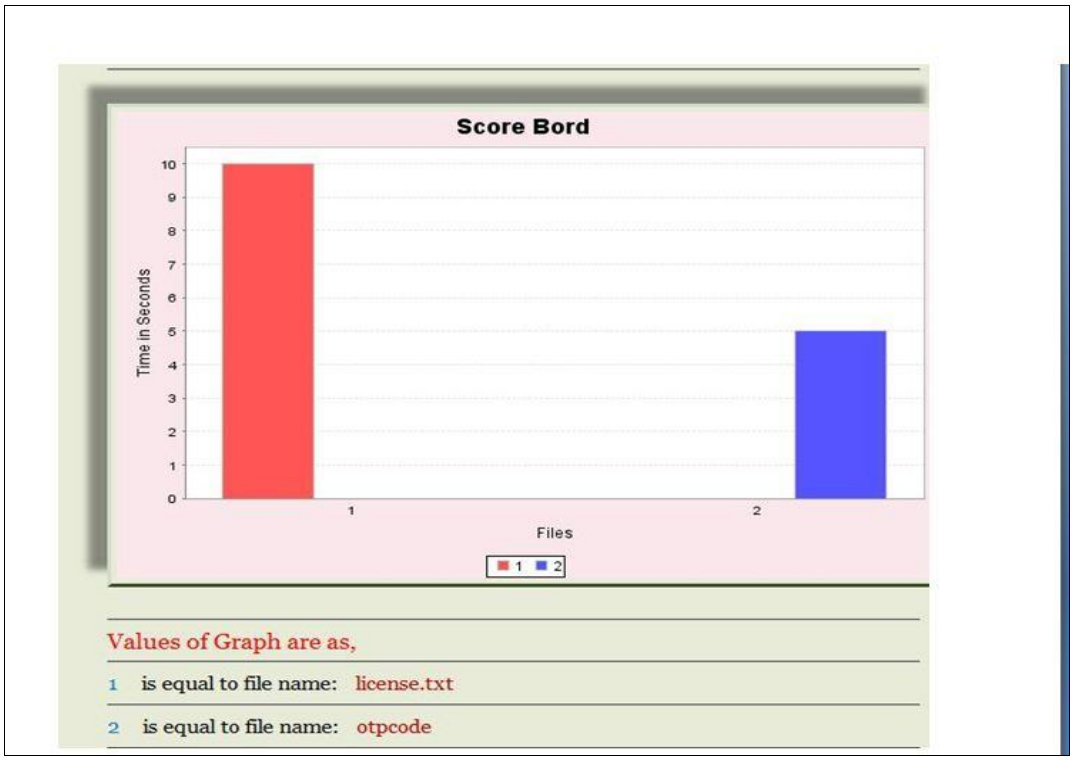

Fig: 6.2 Result of Algomine Search

VII. CONCLUSION

Algorithms plays an important role in solving research problems In digital libraries, being able to extract and index these algorithms would introduce a number of exciting applications including algorithm searching, discovering and analyzing. We are using the synopsis generation and document annotation methods to extract textual metadata for pseudo-codes, and finally we explained how algorithms are indexed and made searchable.

VIII. ACKNOWLEDGMENTS

We would like to take this opportunity to thank a few who were closely involved in the completion of this endeavor. Through this acknowledgement, we express our sincere gratitude to all those people who have been associated with this paper and have helped us with it. We sincerely thank Prof. Priyanka More, Head of IT Department, G.S. Moze College of Engineering and Research, Prof. Priyanka More, Guide who have cooperated with us at different stages during the preparation of the paper.

REFERENCES

[1.] S. Bhatia and P. Mitra. Summarizing figures, tables, and algorithms in scientific publications to augment search results. *ACM Trans. Inf. Syst., 30(1):3:1–3:24, Mar. 2012.*

[2.] S. Bhatia, P. Mitra, and C. L. Giles. Finding algorithms in scientific articles. WWW ’10, pages 1061–1062, 2012 Algorithms play an important part in solving research problems. Scientific publications host a tremendous amount of such high-quality algorithms developed by professional researchers. In digital

ISSN (O) 2454-9665
ISSN (P) 2454-0663

libraries, being able to extract and catalog these algorithms would introduce a number of exciting applications including algorithm searching, discovering, and analyzing. We discussed the prototype of Algomine, a search system for algorithms in large scale scholarly documents, along with providing an illustration of the actual demo system. Specifically, we proposed a set of scalable machine learning based methods to detect algorithms in scholarly documents, we disc0.

[3.] S. Bhatia, S. Tuarob, P. Mitra, and C. L. Giles. An Algorithm Search Engine for Software Developers. 2011.

[4.] J. M. Kleinberg and E. Tardos. Algorithm Design, volume 30.Addison Wesley, 2005.

[5.] Bhatia, P. Mitra, and C. L. Giles. Finding algorithms in scientific articles. WWW '10, pages 1061–1062, 2010