International Journal of Scientific Research in Computer Science, Engineering and Information Technology
© 2019 IJSRCSEIT | Volume 5 | Issue 5 | ISSN : 2456-3307
DOI : https://doi.org/10.32628/CSEIT195454

An Algorithm Search Engine for Extracting Algorithm From PDF Document

Akshata R. Sanas*, Pallavi S, Patil
Computer Science Engineering, Savitribai Phule Pune University, Pune, Maharashtra, India

ABSTRACT

Algorithms are used to developing, analysing, and applying in computer field and used for developing new application. It is used for finding solutions to any problems in different condition. It transforms the problems into algorithmic ones on which standard algorithms are applied. Day by day Scholarly Digital documents are increasing. AlgorithmSeer is a search engine used for searching algorithms. The main aim of it providing a large algorithm database. It is used to automatically encounter and take these algorithms in this big collection of documents that enable algorithm indexing, searching, discovery and analysis. An original set to identify and pull out algorithm representations in a big collection of scholarly documents is proposed, of scale able techniques used by AlgorithmSeer. Along with this, particularly important and relevant textual content can be accessed the platform and highlight portions by anyone with different levels of knowledge. In support of lectures and self-learning, the highlighted documents can be shared with others. However, different levels of learners cannot use the highlighted part of text at same understanding level. We can solve the problem of guessing new highlights of partially highlighted documents.

Keywords : Algorithms, Pseudo codes, Scholarly big data, Sentence Extractor, Steaming, TFIDF

I. INTRODUCTION

To solve problems, Algorithms are used everywhere in Computer field and over a exact technique. Algorithms have affected every single form of human life. Improvement of the current algorithms and evolution of new algorithms for unsolved problems is being made by researchers. Updating search engines are not optimized to searching any particular algorithm. In addition, they cannot be distinguished which documents contain an algorithm and which is not. They produce results in that contain a mixture and not usable data. For example, user enters a query into the search engine for searching a KNN algorithm. After processing search engine give the result that contain KNN term but it any specific information with regarding to algorithmic aspects of KNNs.

The document that is relevant to the algorithm is not necessarily come up in the results due to inappropriate ranking schemes. For algorithm searchers who are unfamiliar particularly important and relevant textual content can be accessed the platform and highlight portions by anyone with different levels of knowledge. For oral lessons or individual learning, the documents, which are highlighted, may be given to the earning people. However the learners having the different levels of knowledge respectively, highlights are generally incomplete or unsuitable are going to develop the system on its own identifies and pulls out algorithm from the given scholarly input.

This kind of system can be very useful to assist the indexing, finding and a vast quantity of required knowledge of the algorithm and carry out a detail study of evolution of the algorithm and may be increase the output quality of the user. The representation of algorithm is not written in specific format in that include symbols, mathematical expression, various font style etc. Therefore, it becomes a challenge to user for discovery and extraction of algorithms. To overcome this disadvantage we propose this system. In this system, we first detect PCs and APs by using different methods then we find the textual metadata that can be instantly extracted from various documents with the use of Application Program Interface and generate different types of models fitted to various levels from a group of documents that are highlighted of knowledge to forecast new output. We provide linking, indexing of the extracted Meta data and make it search able to the user and it can increase the productivity of user with the help of this system.

II. METHODS AND MATERIAL

A. Related Work

[1] Prasenjit Mitra (2016) et al have studied that to identify and extract algorithm representations from scholarly documents. A novel set of scale able techniques used by AlgorithmSeer. They use hybrid machine learning techniques for algorithm representation. These techniques to pull out meta data for each algorithm are used. The user searching some algorithm on the CiteSeerX data set, this site gives so many documents with his relevant search. On the search result all document index with his best ranking and extract all data related document. This document is in the form of synopsis, on the search keywords and the algorithm and gives out-put to user. Especially they suggested detecting algorithms in scholarly documents. For this purpose they used a group of scale able machine learning based methods. Finally they show how algorithms are indexed and made searchable. All the extracted algorithms and their related textual meta-data are then cataloged using SOLR18, which then makes the algorithms searchable.

[2] Elena Baralis (2016) et al have studied that the concept of highlighter. They have introduced about a HIGHLIGHTER is a new technique to inevitably generating focus of learning documents. By using this concept, issue of automatically generating document highlights is resolved by them. Highlights are mark part of the textual content that can use regularly. For example, the most substantial parts of the text can be underlined, colored, or circled. The significance of highlighted points used for learning purpose. Teachers and learners can easily share the highlighted documents through e-learning platforms. Nevertheless, the manual generation of text highlights is time absorbing. So minimizing such problem, they generate classification models. These models are delivered to learners to increase the quality of their learning experience. To start the process of highlighting learning documents, they use text classification techniques. It appraises the capability level of the highlighting users to drive the generation of new highlights.

[3] Saurabh Kataria et al studied that an important source of information that is largely under-utilized are two dimensional plots in digital documents on the web. They explain how data and text can be pulled out inevitably from these 2-D plots. For extracting data and text from two-dimensional plots they advanced automated methods from digital documents and implement it to documents published on the web. This method minimizes the time absorbing manual process of retrieving this data. The algorithm pulls out axes, the ticks on the axes, the text labels associated with the ticks and the labels of the axes. To extract each data point symbol and its textual description from the legend it discovers the legend as a text-dominated box in the figure and pulls out the lines from the legend and segments the lines. To identify their shapes and

records the values of the X and Y coordinates for that point they developed a tool. They to overcome the problem of segmenting can address overlapping data points. The data and text extraction from the 2-D plots are accurate as indicated by experimental results.

[4] Bhatia et al have taken into account an algorithm search engine for software developers. To solve any problem developer first develop algorithms. Algorithms can be crucial and are very important for absolute software projects. In this system, they propose an algorithm search engine that keeps abreast of the latest algorithmic developments. Using a PDF to text converter all the files in the system are first converted to the text file. To sort out the algorithm that is filled out sequentially along with the metadata related to them. The extracted text is then examined cautiously. In the next the engine which is used for query processing then approves the appropriate query given by the user by taking into consideration the query interface and after that it task to search the index for associated similar algorithms. Then finally it shows sorted list by ranks of algorithms to the user.

[5] J.B.Baker et al have studied the methods of analysis of mathematical documents from the particular PDF. It is very challenging job of document analysis of mathematical part of PDF even if the digital document which is available in the standard format. In the context of PDF documents, they suggest the solution for this type of problems. To carry out the character recognition at the same time with the virtual link network generally used for structural analysis they found out OCR approach. To direct extraction of symbol information out from the PDF file, they used another approach with two-stage parser for pulling out layout and expression structure. In the context of mathematical expressions related to first character identification and second structural analysis, they match the efficiency and correctness of these specified to different techniques qualitatively as well as quantitatively in context of layout analysis.

[6] C.L.Giles. et al have studied that finding algorithms in scientific articles. Algorithms are very important part of computer science. To solve any problem first required algorithm. In this system to check whether there is a presence or not of algorithm they first examine documents. After that document text is examine to find out sentences which content the algorithm, if an algorithm is detected. Algorithm a like metadata which is present in document is pulled out and it is arranged I order. To calculate the connection of algorithms with query given by user, the information related with algorithm is used and with decreasing order of connection the algorithms got submitted to the users. In this system a vertical search engine which finds out the algorithm available in that document is delivered and pulled out to form a related metadata of the algorithm.

[7] D.M.Blei et al have studied that Latent Dirichlet Allocation (LDA) technique. They developed LDA, relating to probabilistic model for accumulation of distinct data. LSI and pLSI methods are opponent to the LDA method. It is used for setting of reduction in dimension for the given input collection and a basic model. The actual planning for methodic way that includes probabilistic models may be given us to offer circumstantial setup in domain that is made up of different levels of structure. LDA can be easily implanted with very messy model that is not influenced by LSI as a probabilistic module. This permits a given structure in the potential available space and in specific permits a type of document clustering which is unique in the form that is required to get by shared topics. LDA consist of three level hierarchical Bayesian model. In this model, every particular entity of a given collection is designed related to limited combination as compared to underlying group of topic. Every topic in this particular model is designed as very compact combination over different group of various possibilities. By using a various methods and algorithm, they developed efficient inference module

for calculating Bayes parameter. This module is used for showing different representation of given document.

[8] S.Bhatia et al focused on make summary of various items in the published scientific document such as algorithms, figures, tables . For document-elements to help in find out quickly algorithms, tables, figures, by user .The user are using this method for point out the problem of generating summary by them. To find out look alike sentences within given document text with the help of a specific group of features which subsequently uses context and content data relevant to this elements for machine-learning techniques is used by them. To finalize exact content to select in the summary relevant to the main part and original sentences from the elements of documents and uniqueness of the sentence to the original sentence they proposed a simple model. The model attempts to compare the content in the information and range of summary so that the collected information and would be output must be accurate and useful. To pull out useful data from the summary, which includes the elements of document at the same time system, uses the first set of methods. They use two different classifiers. In this first to finalize exact content to select in the summary relevant to the main part and original sentences from the elements of documents and uniqueness of the sentence to the original sentence they proposed a simple model. They study the problem of choosing the advantageous outcome synopsis size that shoots a balance between the information content and the size of the generated synopses.

[9] J.Kittler et al have studied that combining classifiers.

They focus on classifier combination. They develop structure for classifier grouping. Also, make a decision many current schemes can be taken into account where all the representations are used collectively. To make up generally used combination schemes of classifier like sum rule, min rule, max rule, median rule, product rule f or calculating voting by majority they used different types of assumptions and various approximations. Then they equated experimentally different mixture of scheme. Interestingly outcome came out of this is very surprising. This mixture evolved with much different and restrictive assumption; from all classifier mixture schemes the sum rule is the best-performed scheme. They investigate all the mixture schemes to calculate errors in this finding. The sum rule is most flexible to estimation errors as shown by the sensitivity analysis. They follow two steps. In first step, they give theoretical ideas of given mixtures scheme for combining the suggestion of experts, giving a unique pattern representation. In second step to improve the understanding of their properties, they analyse the sensitivity of these schemes to calculate errors.

B. Methodology

Methodology of system includes stop word elimination, stemming, TFIDF and Algorithm identification to extract algorithms from documents.

Stop word elimination: This process is useful to finding the stop words. It discards very common word from language during indexing. Examples of stop words are articles, prepositions, and conjunctions. In this process text is examine then those words are not usable which are rejected.

Stemming: This process minimizes the words to their base or root form. In this process various form of word are reduced and shows in the common form. It increases the performance of Information Retrieval system. This process also used for indexing purpose. For example, nouns, and verbs in general form, and past tenses are re-conducted to a common root form.

TFIDF: It stands for Term Frequency-Inverse Document Frequency. It is used for information retrieval and text mining. This method is used to calculate importance of words. It counts the number of word present in documents.

Algorithm Identification: This method is used for identify an algorithm. Plain text is extracted from the PDF file. For extracting purpose, we use PDFBox. By using this tool, we can pull out text and modify the information from a PDF document. This process is divided into three modules. First module is document segmentation, which is used for find sections in the document; second module is PC detection, which is used for find PC from documents, and third module is AP detection. It first cleans extracted text and repairs broken sentences after that identifies APs. After finding PC and AP, we link relevant algorithm together and give the final output that is unique algorithm to the user.

Following Fig.1 shows the architecture of proposed system:

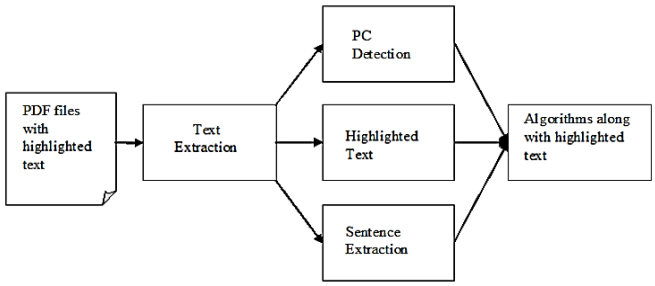


Figure 1. Architecture of proposed system

III. RESULTS AND DISCUSSION

In this section, we evaluate the Effectiveness and efficiency of different search engine. We selected a set of 20 popular algorithms as test queries and tested them with our proposed system, Google Scholar and Google Web Search. Then it shows that the time required for getting result in our system is less as compared to other search system. Our proposed system achieves a precision of 81% at top 10 ranks as other.

Following figure shows the screenshot of the result page for the query DFS.

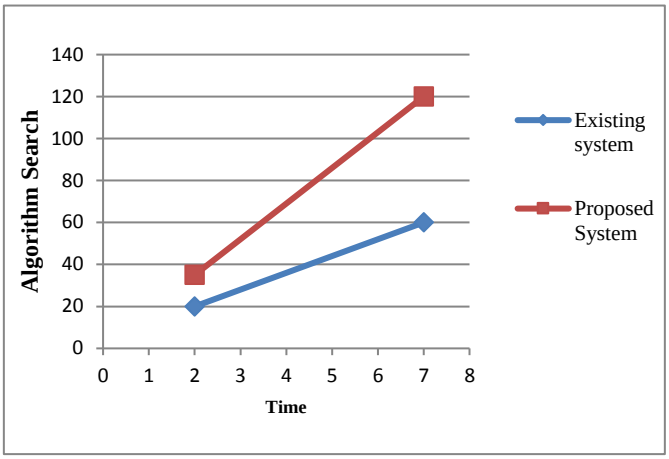
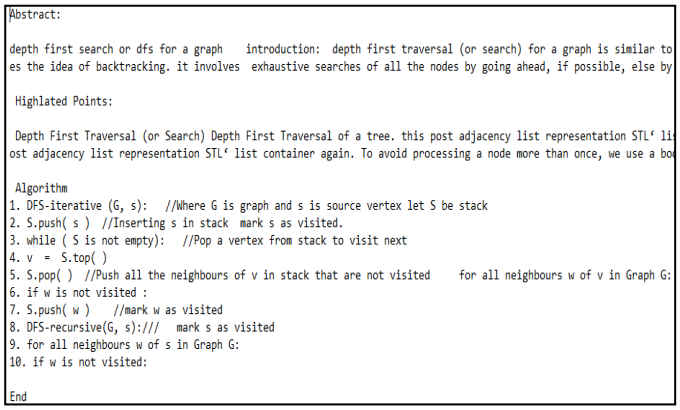


Figure 2. Screenshot of showing the result in Proposed System

A. Evaluations:

In this section, we show the comparison between proposed system and existing system. Following figure shows that time required for searching algorithms in proposed system is lesser than existing system. User getting faster output in this system


Abstract:

depth first search or dfs for a graph introduction: depth first traversal (or search) for a graph is similar to
es the idea of backtracking. it involves exhaustive searches of all the nodes by going ahead, if possible, else by

Highlated Points:

Depth First Traversal (or Search) Depth First Traversal of a tree. this post adjacency list representation STL‘ li
ost adjacency list representation STL‘ list container again. To avoid processing a node more than once, we use a bo

Algorithm
1. DFS-iterative (G, s): //Where G is graph and s is source vertex let S be stack
2. S.push(s) //Inserting s in stack mark s as visited.
3. while (S is not empty): //Pop a vertex from stack to visit next
4. v = S.top( )
5. S.pop( ) //Push all the neighbours of v in stack that are not visited for all neighbours w of v in Graph G:
6. if w is not visited :
7. S.push(w) //mark w as visited
8. DFS-recursive(G, s):/// mark s as visited
9. for all neighbours w of s in Graph G:
10. if w is not visited:

End

Figure 3. Time graph

B. Performance metrics:

Precision and Recall

- A dataset contains 100 pdf.
- A search was conducted on that 100 pdf.
- 80 Algorithms were retrieved.
- Out of the 80 Algorithms retrieved, 65 were relevant.
- Calculate the precision and recall scores for the search.

Solution:

Using the designations above:

A = The number of relevant Algorithms retrieved,

B = The number of relevant Algorithms not retrieved

C = The number of irrelevant Algorithms retrieved.

In this example

A = 65, B = 35 i.e. (100-65) ,C = 15 i.e. (80-65).

Recall = (65/(65+35))*100% => 65/100 * 100% = 65%

Precision = (65/(65+15))*100% => 65/80 * 100% = 81%

Following table shows the precision and recall value

TABLE I. PRECISION AND RECALL VALUE

No.of Algorithm	PRECISION	RECALL
Algorithms	86	78
Abstract	90	86
Highlighted	86	76

IV. CONCLUSION

Professional researchers developed an enormous amount of high-quality algorithm. We have made prototype like AlgorithmSeer. It is a search engine for finding algorithms from PDF file. This engine is extracted algorithm and giving the unique algorithm to the user with highlighted document. User can store this algorithm on his email. Traditional prototype is improved to get better results. It reduces `the time for reading full document. So, this system helps user to find best algorithm.

In future, we will study the fault tolerance after this system failure; explore the semantic analysis of algorithms and how algorithms influence each other over time.

V. REFERENCES

[1] Sumit Bhatia, Prasenjit Mitra and C. Lee Giles.2016 "*AlgorithmSeer: A System for Extracting and Searching for Algorithms in Scholarly Big Data",* IEEE Transactions On Big Data 2332-7790 (c) IEEE 2016.

[2] Elena Baralis, and Luca Cagliero. 2017. "*Highlighter: Automatic highlighting of electronic learning documents",* IEEE Transactions on Emerging Topics in Computing 2168-6750 (c) IEEE 2017.

[3] S. Kataria, W. Browuer, P. Mitra, and C. L. Giles. 2008 "*Automatic extraction of data points and text blocks from 2-dimensional plots in digital documents*", Proceedings of the 23rd national conference on Artificial intelligence - Volume 2,AAAI08, pages 11691174. AAAI Press, 2008

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

[5] J. B. Baker, A. P. Sexton, V. Sorge, and M. Suzuki. 2011, "*Comparing approaches to mathematical document anal-ysis from pdf*", ICDAR 11, pages 463467, 2011.

[6] S. Bhatia, P. Mitra, and C. L. Giles. 2010. "*Finding algorithms in scientific articles",* 2010.

[7] D. M. Blei, A. Y. Ng, and M. I. Jordan. 2003. "*Latent dirichlet allocation*", Journal of Machine Learning Research 3 (2003) 993-1022, Mar. 2003.

[8] J.Kittler, M. Hatef, R. P. W. Duin, and J. Matas. 1998. "*On combining classifiers*", IEEE Trans. Pattern Anal. Mach. Intell., 20(3):226239, Mar. 1998.

[9] T.A. Asuncion, M. Welling, P. Smyth, and Y. W. Teh. 2009. "*On smoothing and inference for topic models",* In Proceedings of the Twenty-Fifth Conference on Uncertainty in Articial Intelligence, UAI.2009.

Cite this article as :
Akshata R. Sanas, Pallavi S, Patil, "An Algorithm Search Engine for Extracting Algorithm From PDF Document ", International Journal of Scientific Research in Computer Science, Engineering and Information Technology (IJSRCSEIT), ISSN : 2456-3307, Volume 5 Issue 5, pp. 10-16, September-October 2019.
Available at doi :
https://doi.org/10.32628/CSEIT195454
Journal URL : http://ijsrcseit.com/CSEIT195454