International Journal of Scientific Research in Computer Science, Engineering and Information Technology
ISSN : 2456-3307 (www.ijsrcseit.com)
doi : https://doi.org/10.32628/CSEIT228516

Comparison of Full-Text Indexing with Metadata Indexing Based Subject Classification Using Graph-Based Index

Soumya George
Assistant Professor Department of Computer Applications St. George's College, Aruvithura, Erattupetta, India

Article Info
Volume 8, Issue 5
Page Number: 86-91

Publication Issue :
September-October-2022

Article History
Accepted : 10 Sep 2022
Published : 20 Sep 2022

ABSTRACT

Subject classification is an indispensable part of all academic search engines to facilitate faster search and retrieval of scholarly articles based on search queries. The widely used approach uses the metadata of journal papers like title, abstract, paper keywords, etc., to classify articles. This paper compares full text-based subject classification with metadata-based subject classification using a graph-based indexing approach. Comparing both methods is an extension of my previous work, GASE, a Graph-based Academic Search Engine based on the subject classification of research articles using an efficient full-text indexing approach. The results show that full-text indexing-based subject classification yields high accuracy than metadata-based classification. Also compared the space complexity and time complexity of both indexing methods. Full-text indexing will have higher space complexity, as it requires storing the entire contents. But subject labeling takes up a generalized time complexity of ϴ (n2 log(n) 2) for both full-text and metadata indexing by considering only the higher-order term and ignoring other constant values.

Keywords : Subject Classification, GASE, Academic Search Engine, Full-Text Indexing, Metadata Indexing

I. INTRODUCTION

Classification of research papers into relevant subject areas enables efficient information retrieval and usually, classification methods follow 2 approaches: (i) using subject experts and (ii) using metadata like title, abstract, etc. [1]. Subject classification using subject experts is a highly accurate method, but the time, effort, and cost of finding efficient experts are not affordable. GASE is a graph-based academic search engine that uses a full-text indexing approach using a graph-based sequence word model [2][3] The main idea behind GASE is the subject classification of each indexed article using a pre-indexed Graph-based Subject Classifier, GSC. This paper presents a comparison study of the full text-based classification of papers with metadata-based subject classification using the GSC.

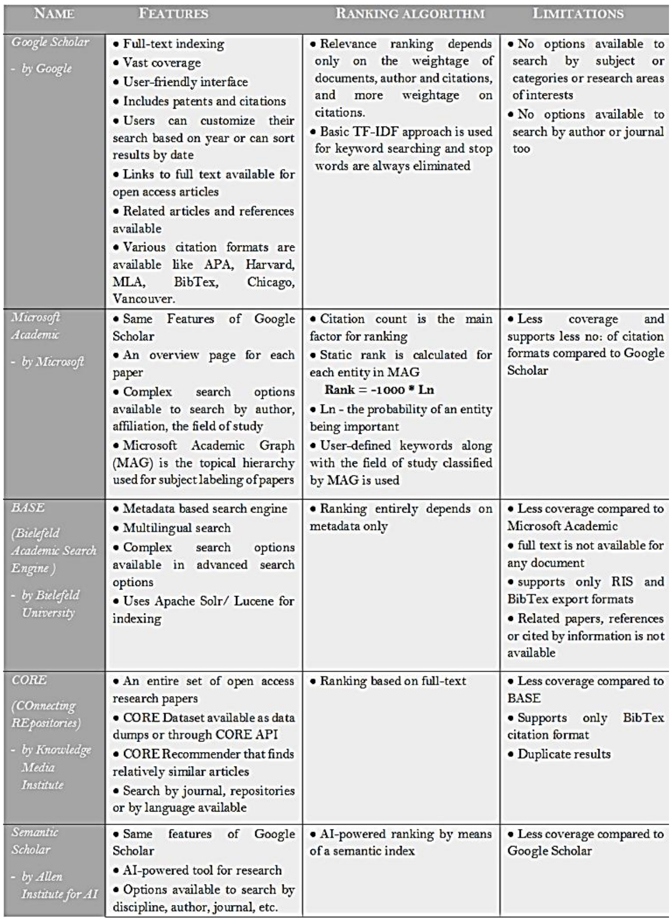
II. REVIEW

Most popular academic search engines include Google Scholar, BASE, CORE, Microsoft Academic, etc., and Google Scholar is the leading player in the market. Academic search engines normally index only metadata of the papers, which include title, abstract, keywords, etc., and completely ignore the full text of the article. The unavailability of the full text of the article due to journal copyright policies, pricing, etc. is the main issue behind this. The BASE is a metadata-based search engine. There are also full-text search engines that enable search from the full text of scientific documents, e.g., JURN. Some others use a mixed approach of indexing the full text of all open access documents and metadata of others, e.g., Google Scholar. Table 1 lists out some of the latest and most popular academic search engines, their important features, and their limitations [4-8].

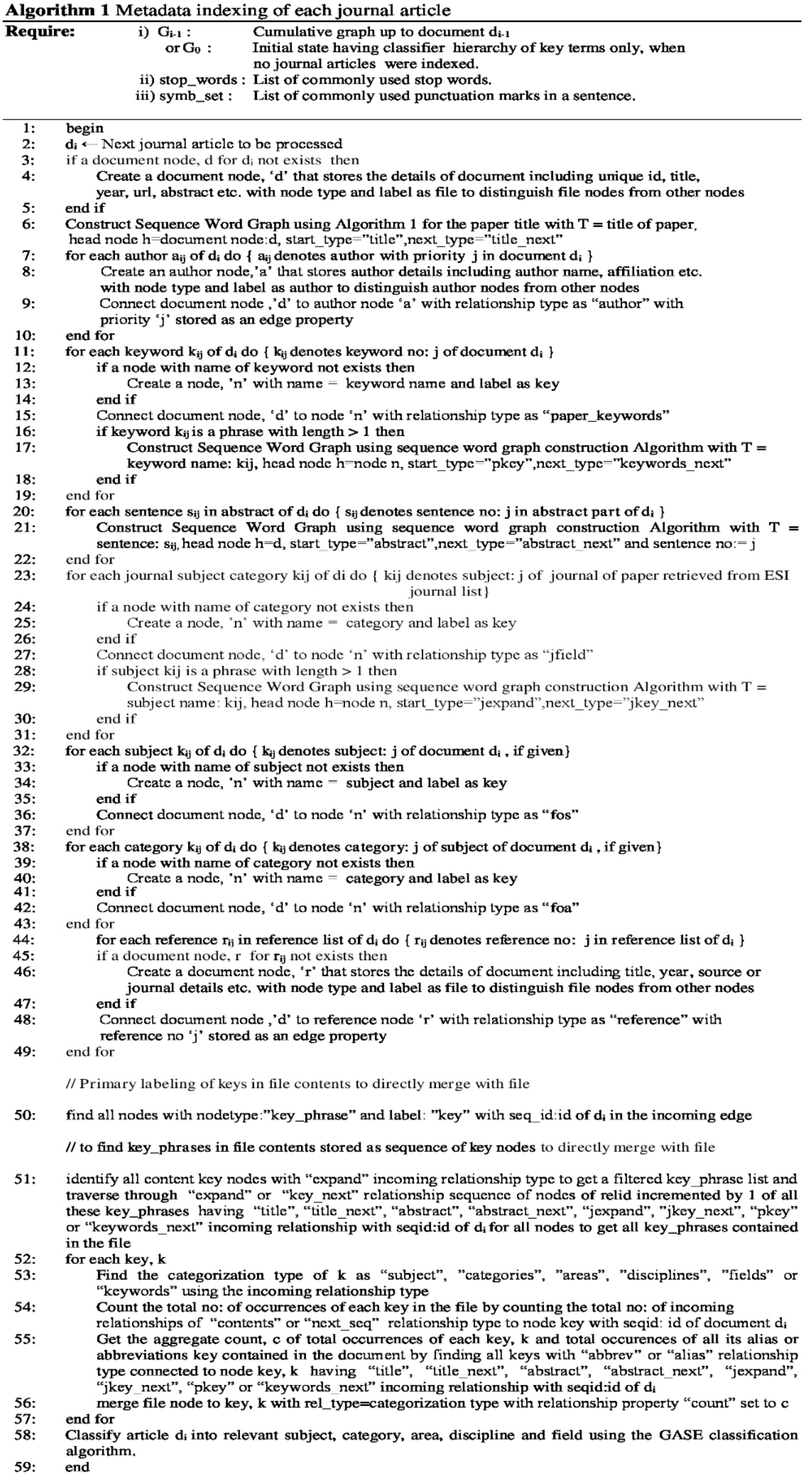
METADATA-BASED SUBJECT CLASSIFICATION USING GASE CLASSIFIER

Metadata-based subject classification is the most commonly used approach. Metadata includes the title of papers, abstract, user-defined keywords, author details, etc., [9]. The important phases of metadata-based classification are similar to GASE which include: (i) construction of classifier hierarchy of important keywords, (ii) Graph-based index creation of each article using WSG, Word sequence graph model, and primary labeling of content keys, and (iii) Subject classification of articles using a classification algorithm. The only difference between the two approaches is that journal contents will be omitted from indexing except the abstract. Phrases among paper keywords defined by the user will be also indexed using the WSG model to capture appropriate matches between user-given keywords and those in the classifier. Also, the subject classification of each paper's journal is also indexed by using the ESI list of journal subject categories [10]. The algorithm to construct metadata graph-based indexing is shown in Fig. 1 using the sequence word graph construction algorithm [3].

Table. 1 : Features and limitations of some popular search engines



NAME	FEATURES	RANKING ALGORITHM	LIMITATIONS
Google Scholar *- by Google*	• Full-text indexing • Vast coverage • User-friendly interface • Includes patents and citations • Users can customize their search based on year or can sort results by date • Links to full text available for open access articles • Related articles and references available • Various citation formats are available like APA, Harvard, MLA, BibTex, Chicago, Vancouver.	• Relevance ranking depends only on the weightage of documents, author and citations, and more weightage on citations. • Basic TF-IDF approach is used for keyword searching and stop words are always eliminated	• No options available to search by subject or categories or research areas of interests • No options available to search by author or journal too
Microsoft Academic *- by Microsoft*	• Same Features of Google Scholar • An overview page for each paper • Complex search options available to search by author, affiliation, the field of study • Microsoft Academic Graph (MAG) is the topical hierarchy used for subject labeling of papers	• Citation count is the main factor for ranking • Static rank is calculated for each entity in MAG **Rank = -1000 * Ln** • Ln - the probability of an entity being important • User-defined keywords along with the field of study classified by MAG is used	• Less coverage and supports less no: of citation formats compared to Google Scholar
BASE *(Bielefeld Academic Search Engine)* *- by Bielefeld University*	• Metadata based search engine • Multilingual search • Complex search options available in advanced search options • Uses Apache Solr/ Lucene for indexing	• Ranking entirely depends on metadata only	• Less coverage compared to Microsoft Academic • full text is not available for any document • supports only RIS and BibTex export formats • Related papers, references or cited by information is not available
CORE *(COnnecting REpositories)* *- by Knowledge Media Institute*	• An entire set of open access research papers • CORE Dataset available as data dumps or through CORE API • CORE Recommender that finds relatively similar articles • Search by journal, repositories or by language available	• Ranking based on full-text	• Less coverage compared to BASE • Supports only BibTex citation format • Duplicate results
Semantic Scholar *- by Allen Institute for AI*	• Same features of Google Scholar • AI-powered tool for research • Options available to search by discipline, author, journal, etc.	• AI-powered ranking by means of a semantic index	• Less coverage compared to Google Scholar



Algorithm 1 Metadata indexing of each journal article

```
Require:  i) $G_{i-1}$ :      Cumulative graph up to document $d_{i-1}$
             or $G_0$ :       Initial state having classifier  hierarchy of key terms only, when
                              no journal articles  were indexed.
          ii) stop_words :    List of commonly used stop words.
          iii) symb_set :     List of commonly used punctuation marks in a sentence.

begin
$d_i \leftarrow$ Next journal article to be processed
if a document node, d for $d_i$ not exists  then
    Create a document node, 'd' that stores the details of document including unique id, title,
    year, url, abstract etc. with node type and label as file to distinguish file nodes from other nodes
end if
Construct Sequence Word Graph using Algorithm 1 for the paper title with T = title of paper,
 head node h=document node:d, start_type="title",next_type="title_next"
for each author $a_{ij}$ of $d_i$ do { $a_{ij}$ denotes author with priority  j in document $d_i$ }
     Create an author node,'a' that stores author details including author name, affiliation etc.
     with node type and label as author to distinguish author nodes from other nodes
     Connect document node ,'d' to author node 'a' with relationship type as "author" with
     priority 'j' stored as an edge property
end for
for each keyword $k_{ij}$ of $d_i$ do { $k_{ij}$ denotes keyword no: j of document $d_i$ }
    if a node with name of keyword not exists then
        Create a node, 'n' with name =  keyword name and label as key
    end if
    Connect document node, 'd' to node 'n' with relationship type as "paper_keywords"
    if keyword $k_{ij}$ is a phrase with length > 1 then
        Construct Sequence Word Graph using sequence word graph construction Algorithm with T =
        keyword name: kij, head node h=node n, start_type="pkey",next_type="keywords_next"
    end if
end for
for each sentence $s_{ij}$ in abstract of $d_i$ do { $s_{ij}$ denotes sentence no: j in abstract part of $d_i$ }
    Construct Sequence Word Graph using sequence word graph construction Algorithm with T =
    sentence: $s_{ij}$, head node h=d, start_type="abstract",next_type="abstract_next" and sentence no:= j
end for
for each journal subject category kij of di do { kij denotes subject: j of  journal of paper retrieved from ESI
                                                 journal list}
    if a node with name of category not exists then
        Create a node, 'n' with name =  category and label as key
    end if
    Connect document node, 'd' to node 'n' with relationship type as "jfield"
    if subject kij is a phrase with length > 1 then
        Construct Sequence Word Graph using sequence word graph construction Algorithm with T =
        subject name: kij, head node h=node n, start_type="jexpand",next_type="jkey_next"
    end if
end for
for each subject $k_{ij}$ of $d_i$ do { $k_{ij}$ denotes subject: j of document $d_i$ , if given}
    if a node with name of subject not exists then
        Create a node, 'n' with name =  subject and label as key
    end if
    Connect document node, 'd' to node 'n' with relationship type as "fos"
end for
for each category $k_{ij}$ of $d_i$ do { $k_{ij}$ denotes category: j of subject of document $d_i$ , if given}
    if a node with name of category not exists then
        Create a node, 'n' with name =  category and label as key
    end if
    Connect document node, 'd' to node 'n' with relationship type as "foa"
end for
    for each reference $r_{ij}$ in reference list of $d_i$ do { $r_{ij}$ denotes reference no:  j in reference list of $d_i$ }
    if a document node, r  for $r_{ij}$ not exists then
        Create a document node, 'r' that stores the details of document including title, year, source or
        journal details etc. with node type and label as file to distinguish file nodes from other nodes
    end if
     Connect document node ,'d' to reference node 'r' with relationship type as "reference" with
     reference no 'j' stored as an edge property
end for

// Primary labeling of keys in file contents to directly merge with file

find all nodes with nodetype:"key_phrase" and label: "key" with seq_id:id of $d_i$ in the incoming edge

// to find key_phrases in file contents stored as sequence of key nodes to directly merge with file

identify all content key nodes with "expand" incoming relationship type to get a filtered key_phrase list and
traverse through  "expand" or  "key_next" relationship sequence of nodes of relid incremented by 1 of all
these key_phrases  having  "title", "title_next", "abstract", "abstract_next", "jexpand", "jkey_next", "pkey"
or "keywords_next" incoming relationship with seqid:id of $d_i$ for all nodes to get all key_phrases contained
in the file
for each key, k
    Find the categorization type of k as "subject", "categories", "areas", "disciplines", "fields" or
    "keywords" using the incoming relationship type
    Count the total no: of occurrences of each key in the file by counting the total no: of incoming
    relationships of  "contents" or "next_seq"  relationship type to node key with seqid: id of document $d_i$
    Get the aggregate count, c of total occurrences of each key, k and total occurences of all its alias or
    abbreviations key contained in the document by finding all keys with "abbrev" or "alias" relationship
    type connected to node key, k  having  "title",  "title_next",  "abstract",  "abstract_next",  "jexpand",
    "jkey_next", "pkey" or "keywords_next" incoming relationship with seqid:id of $d_i$
    merge file node to key, k with rel_type=categorization type with relationship property "count" set to c
end for
Classify article $d_i$ into relevant subject, category, area, discipline and field using the GASE classification
algorithm.
end
```

Fig. 1: Algorithm to create metadata indexing of journal articles

III. EXPERIMENTAL EVALUATIONS & RESULTS

In order to compare the metadata subject classification with GASE full text-based approach, the same data set of GASE subject classified papers of 1307 papers is used. GASE classification accuracy yields around 91% for full text-based subject classification [4]. The subject classification accuracy value, (AV) for metadata classification is calculated using the same formula of GASE by dividing the total no: of papers correctly classified (CC) by the total no: of papers in the data set (TP). The accuracy value obtained for each of the category types from level 2 to level 6 of the classifier hierarchy is shown in Table 2 to Table 6.

Table. 2: Accuracy measure for category type, 'fields'

Fields	TP	CC	AV (%)
algebraic topology	10	7	70
group theory	12	6	50
algebraic geometry	49	35	71.4
computational geometry	12	6	50

Table. 3: Accuracy measure for category type, 'disciplines'

Disciplines	TP	CC	AV (%)
atomic physics	1	0	0
functional analysis	20	11	55
differential geometry	36	21	58.3
fluid dynamics	3	2	66.6

Table. 4: Accuracy measure for category type, 'areas'

Areas	TP	CC	AV (%)
combinatorics	38	15	39.4
dynamical systems	18	4	22.2

Table. 5: Accuracy measure for category type, 'categories'

Categories	TP	CC	AV (%)
software engineering	3	1	33.3
machine learning	1	0	0
mathematical physics	55	26	47.3
number theory	25	9	36

Table. 6: Accuracy measure for category type, 'subject'

Subject	TP	CC	AV (%)
Mathematics and statistics	426	303	71.1
computer science / informatics	363	291	80.2
physics	428	315	73.6

Comparison of accuracy value results of metadata with full text-based subject classification for each category type from level 2 to level 6 is shown by their graphical representation in Fig. 2 to Fig. 6.

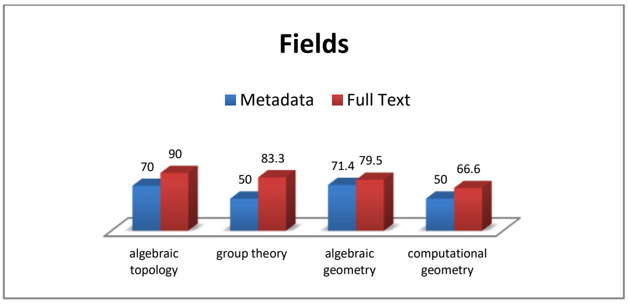


Fig. 2: Metadata Vs. Full Text-based comparison of accuracy value in % for category type, 'fields'

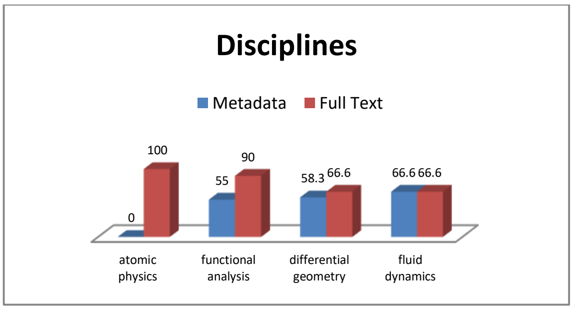


Fig. 3: Metadata Vs. Full Text-based comparison of accuracy value in % for category type, 'disciplines'

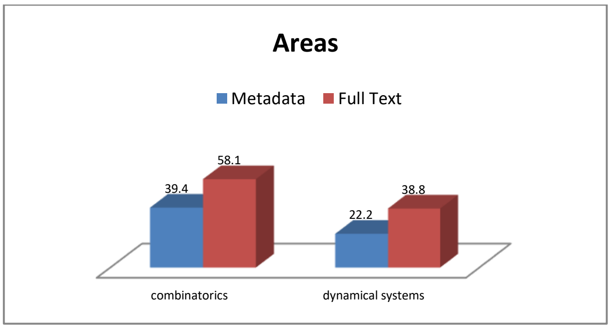


Fig. 4: Metadata Vs. Full Text-based comparison of accuracy value in % for category type, 'areas'

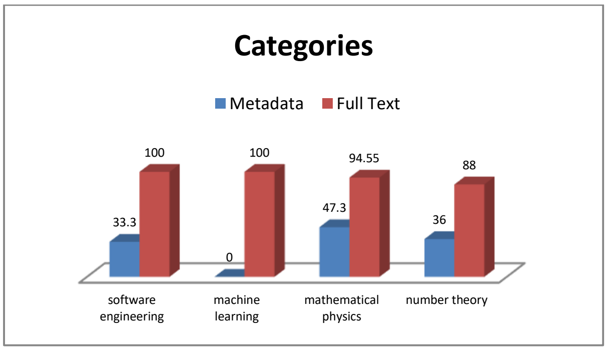


Fig. 5: Metadata Vs. Full Text-based comparison of accuracy value in % for category type, 'categories'

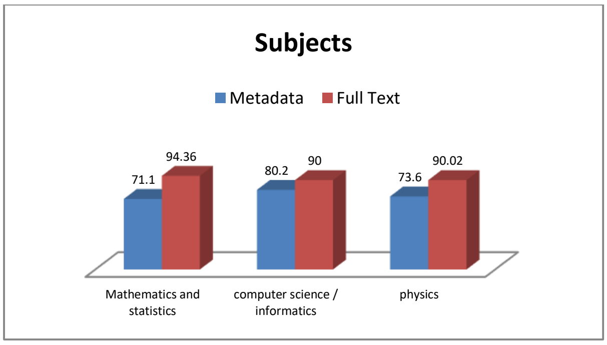


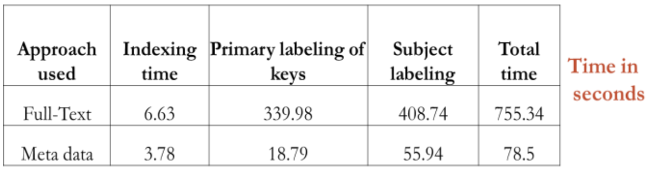
Fig. 6: Metadata Vs. Full Text-based comparison of accuracy value in % for category type, 'subjects'

Results show that full text-based subject classification yields far better accuracy than metadata-based subject classification.

IV. SPACE COMPLEXITY & TIME COMPLEXITY - FULL-TEXT INDEXING VS. METADATA-BASED INDEXING

Initially the average indexing time, including the primary labeling of keys and subject labeling of full-text and metadata is calculated. Metadata includes only title, author details, abstract, keywords, journal category, and list of references to the index. Whereas full-text indexing requires indexing the full text of each article in sequence order by catching the word order of each sentence and also need to concatenate stop words in between too. The no: of sentences are high and, therefore, the no: of keys present in the contents when full contents of the article is used. This increases the time for indexing, primary labeling of keys, and the final subject labeling. So total indexing time will be higher for full-text indexing when compared to metadata. The average indexing time in seconds for full-text and metadata-based indexing is shown in Table. 7.

Table. 7: Full-text indexing time Vs. metadata-based indexing



Approach used	Indexing time	Primary labeling of keys	Subject labeling	Total time
Full-Text	6.63	339.98	408.74	755.34
Meta data	3.78	18.79	55.94	78.5

Time in seconds

The space complexity of an algorithm is the total space taken based on the input size. The Neo4j graph database is used to store the entire data. In neo4j, each node takes up 15B, each relationship requires 34B, and each of the properties consumes 41B to store data. The index requires ˜ 33% of the total space needed for the entire nodes, relationships, and properties [11]. Based on the number of indexed files used for comparison, the total space complexity required for both full-text indexing and metadata indexing is calculated. Auxiliary space for the algorithm includes space utilized by other data structures for running the algorithm like array lists, hash maps, etc. Since only the higher-order term were considered, the space complexity for full-text indexing can be generalized as ϴ (n2.54905), and for metadata indexing, space complexity can be generalized as ϴ (n2.5305) based on the number of files considered. The complexity depends entirely on the length of the documents being considered and varies according to the nature of input documents. But full-text indexing will have higher space complexity, as it requires storing entire contents.

The time complexity of an algorithm is the total time required to run the algorithm. The core part of the algorithm is the subject labeling of articles. Subject labeling entirely depends on the number of keys, 'n' present in each article. Based on the algorithm, subject labeling takes up a generalized time complexity of ϴ (n2 log(n)2) for both full-text and metadata indexing by considering only the higher-order term and ignoring other constant values.

V. CONCLUSIONS AND FUTURE SCOPE

Subject classification of research papers is an indispensable procedure of any academic search engine to provide a better search experience for users. This paper presents a comparison study of full text-based subject classification with metadata-based classification using the GASE classifier model. Subject classified arXiv data set of around 1307 papers was used for each type of classification. Evaluation results show that the full text-based approach provides far better accuracy when compared to the metadata-based subject classification.

VI. ACKNOWLEDGEMENTS

I sincerely thank Dr. M. Sudheep Elayidom, Professor, Division of Computer Engineering, Cochin University of Science and Technology and T. Santhanakrishnan, Scientist, Govt. of India, Ministry of Defense, NPOL, Kochi-21 for their valuable suggestions and guidelines.

VII.REFERENCES

[1] Kang, M., Shin, J. D., & Kim, B. (2015). Automatic subject classification of korean journals based on kscd. Indian Journal of Science and Technology, 8(S1), 452-456.

[2] Soumya George, M. Sudheep Elayidom, T. Santhanakrishnan(2017) ,"A Novel Sequence Graph Representation for Searching and Retrieving Sequences of Long Text in the Domain of Information Retrieval",International Journal of Scientific Research in Computer Science, Engineering and Information Technology (IJSRCSEIT),pp 108-113, Volume 2 ,Issue 5 , September-October-2017

[3] Soumya George, M. Sudheep Elayidom, T. Santhanakrishnan," Knowledge Graph Based Subject Classification of Scholarly Articles", Journal of Advanced Research in Dynamical & Control Systems, (JARDCS), Volume. 11, 02-Special Issue, 2019

[4] Gusenbauer, M. (2019). Google Scholar to overshadow them all? Comparing the sizes of 12 academic search engines and bibliographic databases. Scientometrics, 118(1), 177-214.

[5] Arum, N. S. (2016). A look at semantic scholar and Google scholar.

[6] Khalid, S., Khusro, S., Ullah, I., & Dawson-Amoah, G. (2019). On the curren state of scholarly retrieval systems. Engineering, Technology & Applied Science Research, 9(1), 3863-3870.

[7] Paszcza, B. (2016). Comparison of Microsoft academic (graph) with web of science, scopus and google scholar (Doctoral dissertation, University of Southampton).

[8] The top list of academic search engines, https://paperpile.com/g/academic-search-engines/

[9] de Waard, A., & Kircz, J. (2003, November). Metadata in science publishing. In Proceedings Conferentie Informatiewetenschap (pp. 03-11).Zhang, L., Liu, X., Janssens, F., Liang, L., & Glänzel, W. (2010). Subject clustering analysis based on ISI category classification. Journal of Informetrics, 4(2), 185-193.

[10] ESI_Journal_Category_Map_2012.xlsx, Retrieved from: https://alldocs. net/esi-journal- category-map-2012 –xlsx

[11] Jose Rocha. Understanding Neo4j's data on disk. Retrieved from : https://neo4j.com/developer/kb/understanding-data-on-disk

Cite this article as :

Soumya George, "Comparison of Full-Text Indexing with Metadata Indexing Based Subject Classification Using Graph-Based Index", International Journal of Scientific Research in Computer Science, Engineering and Information Technology (IJSRCSEIT), ISSN : 2456-3307, Volume 8 Issue 5, pp. 86-91, September-October 2022.
Available at doi : https://doi.org/10.32628/CSEIT228516
Journal URL : https://ijsrcseit.com/CSEIT228516