2017 Conference on Information and Communication Technology (CICT'17)

SemAcSearch: A Semantically Modeled Academic Search Engine

Rishab Ketan Doshi
Department of Information Technology
NITK Surathkal
Mangalore, INDIA
rishabketandoshi@gmail.com

Shravan Karthik
Department of Information Technology
NITK Surathkal
Mangalore, INDIA
shravan1994@gmail.com

Sowmya Kamath S
Department of Information Technology
NITK Surathkal
Mangalore, INDIA
sowmyakamath@nitk.edu.in

***Abstract*—Scholarly article search is a new vertical search paradigm that has gained popularity fast, due in part to the large volumes of research output from universities across the globe. The ranking given to scholarly articles on a search engine's result page is a significant factor in determining its citation rate and audience. A higher Search Engine(SE) rank can help in garnering more reads and possibly more citations for an article, while a lower rank can actually hinder the perceived value of an article from the users' perspective. Hence, searching academic journals and scholarly articles may need special consideration to other factors, beyond the keyword search and context-based querying strategies adopted by most conventional search engines. Academic search engine optimization (ASEO) is a crucial requirement for search engines dealing with scholarly articles. In this paper, we present a specialized, vertical search engine focusing on journal and scholarly article search, that considers context and semantics of the query and articles in computing the overall ranking of publications. Using this, the effectiveness of various ranking algorithms in determining the rank of individual articles was explored and their performance compared.**

***Index Terms*—Semantic Search, Learning to Rank, Search Engine Optimization**

I. Introduction

With the popularity of the Web and widespread access to the Internet, the volume and variety of information available has increased beyond one's wildest expectations. Given the emergence of both core and interdisciplinary areas of research, the number of scholarly articles indexed on the Web have also grown rapidly. As per the results of a scholarly research output survey conducted by Jinha [1], there are currently more than 50 million publications in print, starting from the year 1665, when articles first appeared in print. Mabe and Amin [2] estimated that the number of journals themselves was approximately 15,000 in 2001. Björk et al [3] estimated that 23,750 journal articles were published in 2006, while Jinha [1] found the number had grown to 26,406 in 2009. Given this phenomenal growth trend, specialized search engines for scholarly articles are now more critical than ever before. From users' perspective, such specialized systems can facilitate quick access to most relevant documents as per the information need. From scholarly article author's perspective, these systems can aid in boosting citation counts and peer recognition, which is critical to success in the highly competitive academic field.

Academic search, thus, has grown into a critical area in the field of Information Retrieval (IR) on the Web over the past decade. While the advent of large-scale search engines with vast indexing capabilities and fast matching algorithms has improved access to a variety of Web content, the process of searching for academic articles is considered to be a vertical domain within traditional IR [4]. The concepts of Search Engine Optimization (SEO) is central in enhancing IR and search performance on the Web. SEO techniques are used to restructure content on websites for achieving higher rank in search results displayed by conventional search engines like Google and Bing. The primary aim is to increase the audience size for particular content and also increase click-through rate.

Just as website builders have an interest in ensuring that their websites are discoverable by the larger public, academicians and researchers also aspire for better dissemination of their research findings into the research community and beyond. It is also important to note that a lot of research work undertaken builds on the work done by peers in the respective research fields. Thus, it is imperative that pertinent articles appear higher in the results list returned in response to a users query. Academic Search Engine Optimization (ASEO) [5] is a field that deals primarily with improving ranking and exposure of academic articles, for better user experience. ASEO is the fundamental paradigm over which popular academia-oriented vertical search like Google Scholar and Microsoft Academic are built on. Although the fundamental concepts upon which such academic search engines are built are the same as conventional web search engines, there are certain finer granular level differences between the two. One of the main differences is - academic and scholarly literature mainly cater to a specialized audience. Other crucial differences are discussed in the subsequent sections.

Our work focuses on search engines specific to academic literature and scholarly articles. In this paper, we present a semantically modeled academic search engine we call SemAcSearch and compare the results obtained after applying various page ranking algorithms to scholarly documents. The scholarly literature used as a dataset includes journal publications, conference proceedings and other research publications. Just as search engines crawl various websites and pages, SemAcSearch scans through a database of scholarly articles and ranks

978-1-5386-1866-0/17/$31.00

articles based on relevance to the search query.

The rest of this paper is organized as follows: In section II, we discuss relevant related work in the area of ASEO, highlighting the differences between academic search engines and web search engines, and also illustrating the ranking parameters used by the existing academic search engines. Section III describes the proposed methodology and implementation specifics of Crawling,Indexing and Ranking. Section IV presents a discussion on the experimental results and an analysis of the validity and performance of our system, lastly in section V we present the concluding remarks and possible future improvements.

II. Related Work

Search engines consist of several components, chief among which are the crawler, indexer and a page ranking component. Although academic search engines have the same components, they differ from regular search engines in many ways. This can be evidenced by the fact despite having well-established traditional search engines, Google and Microsoft have developed separate search engines catering to scholarly article search.

There has been extensive research in the area of academic search engines and scholarly search. Harzing et al [6] performed a comprehensive longitudinal and cross-disciplinary comparison of three major bibliometric databases, namely, Google Scholar, Scopus and the Web of Science, which showed a reasonably stable quarterly growth for both publications and citations across the three databases. One significant difference that academic search engines have over regular search engines is the challenge of having to crawl and index scholarly articles which are mostly articulated in the form of unstructured data such as PDF's or Word documents, with varied formatting used by different publication houses and conferences/journals. Hence, extracting crucial information like author details, affiliation, keywords, abstract etc, from a paper can be challenging. This fact has been expounded by Minh-Tang et al [10] who used Optical Character Recognition(OCR) techniques in addition to simple text parsing to perform Logical Structure Recovery of Scholarly documents.

Conventional search engines rely on vital information present in the webpage like meta-tags, links to dependent websites etc. to rank a page. However, in academic search engines the factors that contribute to the ranking of a publication are completely different, this can be illustrated by considering the example of Semantic Scholar [13] which is an academic search engine developed at the Allen Institute of Artificial Intelligence. Semantic Scholar uses a variety of factors like Author Influence Score, Highly Influential Citations etc., to rank a scholarly article. It also lays strong emphasis on the citation of a publication by defining and employing unique terminologies like Citation Velocity and Citation Acceleration. The former is an indicator of how popular and lasting a publication is, which is essentially the weighted average of the publications citations for the last 3 years, the latter measures the change in citation velocity over time indicating whether the number of citations for a publication is increasing or decreasing.

Martín et al [11] explored the importance of citations of a publication for its ranking on Google Scholar. They collected data by firing a null query, only filtered by year of publication (from 1950-2013) and built visualizations using various data points like rank, citation number, paper versions and publication year to compare the importance of these data points. They found that citations are indeed an important factor in relevance ranking of a publication on Google Scholar. Zhu et al [12] investigated the importance of a particular citation, by presenting an idea to automatically identify the subset of references in a bibliography that have a central academic influence on the citing paper. The problem is modeled as a binary classification problem, wherein each research paper is identified as either influential or non-influential. They concluded that number of times a paper is cited was one of the best predictors of how influential it was.

Beel et al [5] suggested a method to rank academic papers by distinguishing the paper into various sections and assigning a weight to each section. A function that accounts for intentional repetition of words is proposed to avoid false ranking of the paper. Lei et al [7] proposed a semantic model to analyze the search queries. The proposed semantic model breaks the query into varied components of the sentence and assigns a weight to each component. Each component of the sentence are then queried on the index table and the resulting individual sums are aggregated, normalized to indicate a rank. This semantic model also factors in other entities such as synonyms to the token of a query, root words of the query obtained by stemming the query.

Based on this extensive review of relevant research, we found that there is no definite standard for academic search engine optimization. While there have been many methods that have been proposed and many websites implement their own algorithms, there is no clearly established leader that monopolizes the academic search domain. Most search queries are complex and a sense of the query has to be developed before displaying results. Incorporating semantics along with other ASEO techniques could possibly enhance the ranking of most relevant articles, an avenue which we explore in the work presented in this paper. We propose SemAcSearch, an academic search engine, that factors in a semantic approach to improve the relevance of search queries and a ranking algorithm that discriminates various sections of the paper, for influence estimation.

III. Proposed Approach

The methodology adopted to build SemAcSearch, the semantically modeled academic search engine is depicted in Fig. 1. below. The primary modules are the academic article crawler and the indexer which creates an inverted index of extracted domain-specific terms, over which the user query is matched after applying various ranking algorithms. The process is described in detail next.

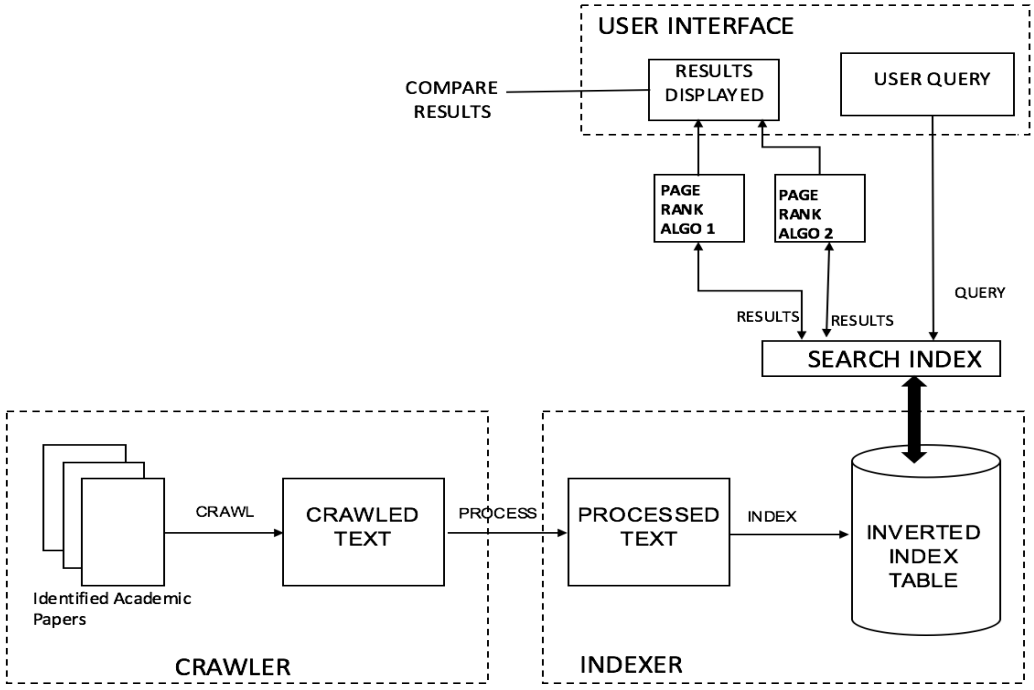

Fig. 1. Workflow and processes of SemAcSearch

A. The SemAcSearch Crawler

Firstly, we start by identifying the academic papers that the search engine will crawl, ensuring that the results returned by it will be one of these identified papers. The papers chosen are such that they belong to similar set of conferences to ensure similarity in the structure of the paper and also broadly belong to a fixed set of topics. We identified 140+ papers that fit these parameters for the purposes of our study. The papers predominantly belong to the publication houses Elsevier, IEEE and ACM.

Scholarly articles often comprise of important sections, some common sections being *Abstract, Title, Authors, Keywords, References* etc. All these sections can be indexed in the crawling phase, also references which are an integral part of every paper can be used to link papers. Using this concept, we propose a unique ranking algorithm based on a graph model, that estimates the influence of other peer/referred articles on the paper. SemAcSearch limits extracting information to Title, Authors, Abstract and keywords. Sophisticated learning algorithms can be used to extract information from images and graphs from these articles, which we intend to explore as future work.

The design of the crawler is critical to the architecture of an academic search engine like SemAcSearch. Here, each article's PDF was uploaded using a GUI created for facilitating the addition of PDF files. The PDF file is parsed using an open-source PDF parser, Composer[1].Before the PDF is parsed using the composer tool, it is first verified to see if the composer is capable of parsing the particular PDF. If this verification succeeds, the PDF is scanned to extract the paper's title. The composer tool also extracts certain meta-tags of the PDF such as author. The abstract and keywords of each scholarly paper lies in the body of the parsed PDF. Thus, a script was written to extract these two attributes from the PDF. The process of extraction of relevant attributes was performed heuristically, after considering the structure of the papers chosen for this study.

During this heuristic process, several observations were made, one of the simplest being - the title of the publication appears at the beginning, followed by the list of authors in 99% of the articles. Another observation made is that the abstract and list of keywords all lie within the first page of the document and there is usually a paragraph spacing between the two entities. These parameters were found to fit well for most papers even beyond those considered in this study. Upon parsing the uploaded paper using these assumptions, the identified values are displayed to the author who can then make changes to these values in case of errors produced by the crawler. Once the user approves the associated values, he/she can submit it to the database, wherein the indexer performs further processing and stores the index.

B. The SemAcSearch Indexer

An inherent problem with rudimentary text processing is that a computer is unable to distinguish between words with the same root, for example: *'computing' 'computation', 'computations'* when all three are essentially indicating the same thing. We avoid this problem by stemming the tokens obtained

[1]Available online at http://pdfparser.org/

from the uploaded article. Stemming of the tokens ensures that the root word is extracted and all verbal forms of the root word fall under the same category. This step also results in space optimization ensuring that the root word is stored and all its verbal forms are indexed under the same root word rather than as separate entities.

This indexing step is performed by feeding the extracted tokens from the crawled PDF (obtained in the previous step) to a python script, which performs the text processing. The processing includes converting all the tokens into lower case, removing stop words and using the NLTK python library[2] to stem every entity. These processed entities are then pushed into the database. Data extracted from all the papers that was processed in the previous step is further pushed into a MySql database. Each word is indexed in a table and further has attributes associated with it as shown below:

- *PID (Paper ID)* - ID of the paper in which the word/token appears, serves as an index for pagerank.
- *Abstract Count* - Frequency of the word in the abstract of the paper.
- *Title Count* - Frequency of the word in the title of the paper.
- *Keywords count* - Frequency of the word in the keywords of the paper.
- *Author Count* - Frequency of the word in the Author's list for the paper.

At the end of these steps there are roughly 8000 tables each representing a token. Another separate table stores the information about all the papers that have been crawled. This table contains the information to be displayed for a search query associated with every result. Attributes associated with each record in the paper table are:

- *Paper ID* - Unique ID of the paper
- *Title* - Title of the paper
- *Author* - Author Name for the paper
- *Citations* - Citation count of the paper

As an example, we demonstrate the indexing process for a single paper with details as below:

- Title- *First Aid in Football: The Changing Face of Aid and Paramedics*
- Abstract- *Football is a dangerous sport and aid is of prime importance. Hence we've asked stewards to learn basic aid...*
- Paper ID- 1234
- Author- *Sam Bale*
- Citation count- 25
- Keywords - *Football, Medical Aid*

The table for the token '*aid*' will have the entries as shown in Table I. The paper table will have entries as shown in Table II. This process is repeated for all the papers in the chosen dataset.

[2]Available online at http://www.nltk.org/

TABLE I
TOKEN TABLE FOR 'AID'

PID	Abstract Count	Title Count	Keywords Count	Author Count
1234	2	2	1	0

TABLE II
PAPER TABLE

PID	Title	Author	Citations
1234	First Aid in Football: The Changing Face of Aid and Paramedics	Sam Bale	25

C. Querying the Search Engine

The search engine can be queried through the search engine bar. Once the query is submitted, a couple of steps are performed before querying the database for results.

1) **Synonyms extraction** The synonyms associated with each query term are obtained using Python's NLTK package. These synonyms are vital to dispel the issues related to keyword search and ensure that a context-specific perspective of the query is obtained. Without the presence of synonyms, the entities without a direct match with regards to root words are ignored. A classic example would be, while querying papers on '*data mining*', papers titled '*information extraction*' are not returned without the use of the synonyms model. The synonyms model ensures that a query having the word '*data*' also include results containing the word '*information*', as both the words are synonyms. Similarly for the word '*mining*' results containing the word '*excavation*' are included. These synonyms help broaden the scope of tokens associated with a query thereby providing more results. These tokens are further fed to the unique page rank algorithm.
2) **Stemming of tokens and synonyms:** The stemming of the query along with the synonyms (obtained in the previous step) is performed to ensure a match with the database entries. Once the entries are matched to that of the database, the page rank algorithm ranks results based on relevance.

The users query is lemmatized to ensure a match between the user query and database index, further a list of synonyms are generated on the fly which are then also inserted into the database as synonyms to the particular token creating a two way index between the synonym and token. This ensures that synonyms for this particular token are cached and needn't be generated for future queries.

D. Ranking of Results

Rank for every paper based on the search query takes into account the important factors such as - title of the article, author of the article, citation count of the paper, keywords

associated with the article, abstract of the document and the synonyms of the query. All these parameters are assigned different weights in order to calculate the page rank for the matched result. Also, to ensure that the authors of the paper don't spam the paper with particular keywords in order to gain a higher page rank, we have employed a slow growth function. The ranking algorithm also sets a precedence for title and author match as content which has matches in these important parts of the paper will naturally be relevant to the query. This is followed by a match with the list of associated keywords and finally there is a match to the abstract associated with the paper. Following this there is a match with the list of synonyms associated with each query. Naturally, the synonyms are assigned a lesser weight than the weight assigned for the exact query token. However, as demonstrated in the paper this decreased weight needn't necessarily hamper search results.

Heuristically, the model that gives most relevant results is the algorithm which assigns different weights to each entity, and accounts for unnecessary repetition of certain words to boost the ranking of a page. This is achieved by inducing a slow growth function $f(x)$ defined as follows:

$$f(x) = \begin{cases} k\sqrt{x} & 0 \leq x \leq 1000 \\ k(\sqrt{1000} + log(x - 1000)) & x > 1000 \end{cases}$$

where, k is the weight assigned for each factor.

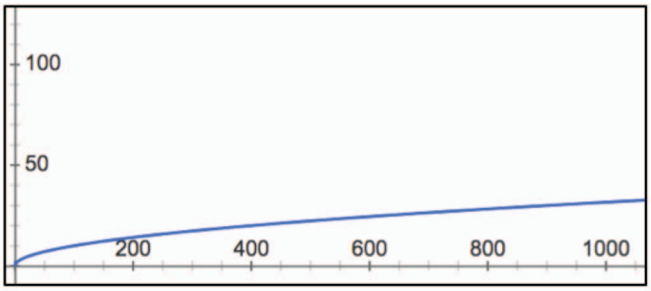

Fig. 2. $k.\sqrt{x}$ for k=1

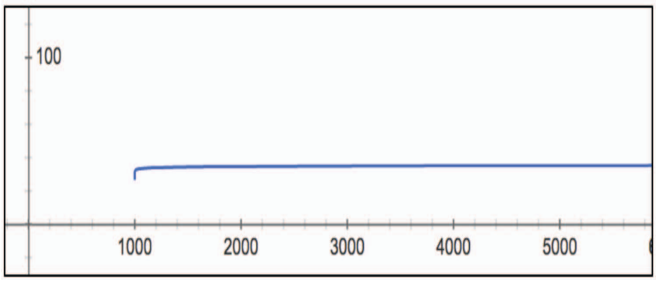

Fig. 3. $k.(\sqrt{1000} + \log(x - 1000))$ for k=1

As seen in Fig. 2, the square root of x is a slow growing function, which grows almost linearly for small values of x and very slowly for larger values of x. Further as seen in Fig. 3, the graph for $(k * (sqrt(1000) + log(x - 1000)))$, is almost a constant and grows only negligibly, hence offsetting the problem of word repetition.

In SemAcSearch, different weights were assigned to each entity. Title and authors were given the highest weight as words/ tokens appearing in the title will have the most relevance to the search query. The text matches from the abstract were assigned the least weight among all the other associated text in the document. Synonyms were assigned a far lesser weight however, the function applied was the same.

IV. Results and Discussion

As SemAcSearch is an academic search engine, we will discuss the results obtained by illustrating results on various search queries. The time taken to return the results for a search query was roughly 5 seconds. The slight delay lies in getting the python script that processes the query to run on a separate process. While this seems large, the advantage of having a separate process to run this feature ensures there is batch processing that can occur with the two modules working independently.

The performance of the search engine was evaluated using the three standard IR metrics - precision, recall and f1-score. In our case, these metrics were measured in terms of True Positive (TP), False Positive (FP), True Negative (TN) and False Negative (FN). TP measures the number of relevant results correctly identified as relevant, while TN gives the number of irrelevant results correctly identified as irrelevant. FP measures the number of irrelevant results that were incorrectly identified as relevant. Finally, FN measures the number of relevant results that were incorrectly identified as irrelevant.

Precision measures the relevancy of the results of the search engine, i.e the precision value is the ratio of relevant search results, to the total results retrieved by the engine for a particular query. The precision value is a good estimate of the false-positives in the search result.

$$Precision\ (P) = \frac{TP}{TP + FP} \tag{1}$$

Recall measures the fraction of relevant results that are identified, i.e recall value is the ratio of correct results returned, to the total relevant search results for the query. Recall value is a good estimate of the false-negatives in the search result.

$$Recall\ (R) = \frac{TP}{TP + FN} \tag{2}$$

F1-score is the harmonic mean of the precision and recall values. The f1-score value ranges from 0 (indicative of a weak score attributed to minimal relevance of the results returned) and 1 (indicative of the strong score attributed to the high relevance of the results returned).

$$F1 - score = 2 \times \frac{P \times R}{P + R} \tag{3}$$

To illustrate the difference in the performance of the two retrieval approaches, i.e., without semantic modeling (naive approach) and with semantic modeling, we present a simple example. We compare the results obtained by the naive search

engine algorithm as a plain inverted index table and the semantically modeled search engine that returns a separate inverted index table with ranked results generated as per its algorithm. Considering a sample search query '*Neural Networks*', the performance of the two approaches is compared. When the naive algorithm that doesn't factor in the semantic modeling performed on each query is used, the confusion matrix is as indicated in Table III. The same query on applying the semantic model achieved results as indicated by the confusion matrix shown in Table IV.

TABLE III
CONFUSION MATRIX FOR QUERY 'NEURAL NETWORKS' FOR THE NAIVE APPROACH (WITHOUT SEMANTIC MODELING)

	True	**False**
True	14	6
False	3	0

TABLE IV
CONFUSION MATRIX FOR QUERY 'NEURAL NETWORK' WITH SEMANTIC MODELING

	True	**False**
True	20	0
False	3	0

The performance of the naive search algorithm in comparison to the semantically modeled algorithm in terms of precision, recall and f1-score values is presented in Table V below. These values are derived from the data in Tables III and IV. The dataset chosen had 20 papers related to Neural Networks. As seen from Tables III and IV, the naive retrieval algorithm achieved significantly lower precision. The semantic model due to its nature of determining the context of the query is able to capture the users query and model it to yield suitable results. However, both suffer from the flaw that papers having the words '*Neural*' or '*Networks*' are matched even if they are not the main focus of the paper, this happens due to a simple string match where matched article gets pushed onto the search result. Thus, the recall value for the engine is almost never 1. However, the recall value is far greater for the semantic model than for the naive algorithm.

TABLE V
PERFORMANCE OF THE NAIVE AND SEMANTICALLY MODELED SEARCH ALGORITHMS

Metric	**Naive Algorithm**	**Semantically Modeled**
Precision	0.7	1
Recall	0.824	0.870
F1-score	0.756	0.930

A consequence of the higher precision and recall value achieved by the semantically modeled search engine is an improved f1-score. When compared to the naive string-matching algorithm, we notice a 23% increase in the f1-score of the semantically modeled search engine. Hence, it can be concluded that, delving into the context of the query through the semantic modeling has a profound impact on the relevance of the search results.

V. CONCLUSION & FUTURE WORK

In this paper, SemAcSearch, a vertical search engine focusing on journal and scholarly article search, that considers context and semantics of the query in computing the overall ranking of publications is presented. Semantic modeling and parametric weight based pageranking helped achieve a clear improvement in the relevance of the search results when compared to a naive keyword-based matching approach. To avoid the adverse effects of keyword stuffing employed by authors for achieving a higher page rank, a slow growth function was also used in the pagerank algorithm employed in the search engine. Currently, SemAcSearch only focuses on making sense of the context associated with each word. A more foolproof and relevant method would be to also reference other sections of each paper to develop an impact factor for a paper with respect to other peer authors. This leads to a graph model based on which a more effective ranking can be computed, which can ensure greater relevance of results. Further analysis of the query like, assigning a weight based on POS tagging and score based on the component will also be explored next.

REFERENCES

[1] Jinha, Arif E. "Article 50 million: an estimate of the number of scholarly articles in existence." Learned Publishing 23.3 (2010): 258-263.

[2] Mabe, Michael, and Mayur Amin. "Growth dynamics of scholarly and scientific journals." Scientometrics 51.1 (2001): 147-162.

[3] Björk, Bo-Christer, Annikki Roos, and Mari Lauri. "Global annual volume of peer reviewed scholarly articles and the share available via different Open Access options." ELPUB 2008.

[4] Chen, Min, Shiwen Mao, and Yunhao Liu. "Big data: A survey." Mobile networks and applications 19.2 (2014): 171.

[5] Joeran Beel, Bela Gipp, Erik Wilde, M. (2010): "Academic Search Engine Optimization (ASEO): Optimizing Scholarly Literature for Google Scholar & Co.", Journal of scholarly publishing 41.2 (2009): 176-190.

[6] Harzing, Anne-Wil. "Publish or perish.", Melbourne, Australia: Tarma Software Research (2007)

[7] Yuangui Lei, Victoria Uren, and Enrico Motta., "SemSearch: A Search Engine for the Semantic Web", In: Proceedings of the 15th International Conference on Knowledge Engineering and Knowledge Management (EKAW). (2006) pp23845, Springer

[8] Page, Lawrence, Sergey Brin, Rajeev Motwani, and Terry Winograd. "The PageRank citation ranking: Bringing order to the Web.", Stanford InfoLab, 1999.

[9] Brin, Sergey, and Lawrence Page. "The anatomy of a large-scale hypertextual web search engine." Computer networks 56.18 (2012): 3825-3833.

[10] Luong, Minh-Thang, Thuy Dung Nguyen, and Min-Yen Kan. "Logical structure recovery in scholarly articles with rich document features." Multimedia Storage and Retrieval Innovations for Digital Library Systems 270 (2012): 2.

[11] Martin-Martin, Alberto, Enrique Orduna-Malea, Anne-Wil Harzing, and Emilio Delgado Lpez-Czar. "Can we use Google Scholar to identify highly-cited documents?." Journal of Informetrics 11, no. 1 (2017): 152-163.

[12] Zhu, Xiaodan, Peter Turney, Daniel Lemire, and Andr Vellino. "Measuring academic influence: Not all citations are equal." Journal of the Association for Information Science and Technology 66, no. 2 (2015): 408-427.

[13] Semantic Scholar: www.semanticscholar.org, developed at the Allen Institute of Artificial Intelligence.