Text Analyzing Tool for Simplifying the Syllabus Creation Process

Ritu Sodhi* and Jitendra Choudhary

***Abstract*—In curriculum development, syllabus creation is an important activity. The need for a syllabus repository and data extraction is essential for creating a new syllabus. Faculties do this job manually by adding topics from their intelligence as well as analyzing syllabuses of the same course available on the search engine. Because it is a time-consuming job, this research aims to propose and implement a text-analyzing tool. The methodology is used, first to create a syllabus repository of computer science courses, do data extraction & normalization, analyzes the contents of different syllabuses, and suggests the contents to the faculties. This tool is experimented on the sample for creating the syllabus of C Programming. The result of the tool is that it suggests the topics that can be included in the syllabus to the faculties. The time and effort required to create a syllabus are reduced by using this tool. In the future weightage to the syllabuses can be given for classification that will give more accurate results while analyzing syllabuses. This tool can be improved by using machine learning algorithms for creating syllabus repositories and data extraction. Semantic comparison can give more accurate results by creating a specific model for computer terminologies.**

***Index Terms*—Curriculum, syllabus, faculty, data extraction, analysis, semantic comparison.**

I. Introduction

The syllabus plays a vital role in education. A syllabus is an important resource for faculties, students, and learners [1]. The syllabus consists mainly of course objectives, course contents, course outcomes, learning methods, online resources, references, etc. It is defined by members of the academic team [2, 3]. The most important part of the syllabus is its contents. Syllabus creators in universities create new syllabuses as and when it is required. They also take references from search engines.

This research proposed and implemented a methodology for creating and analyzing syllabuses of computer science courses that will

1) Create an automatic syllabus repository of pdf from search engines.
2) Extracting the contents using either of two ways.
   a) Create individual text files of syllabuses manually from pdfs, by keeping only the syllabus contents.
   b) Extract the contents automatically from PDFs and normalize them.
3) This paper implements two different algorithms after data extraction to analyze different syllabus contents
   a) The first algorithm finds the frequency of occurrences of every topic among different syllabuses.

   Sort the topics in the decreasing order of their frequency and display them to the syllabus creator.
   b) The second algorithm finds the semantic similarity value of every topic of the syllabus among different syllabuses. Sort the topics in decreasing order of their semantic similarity value and show it to the syllabus creator.

This tool helps the syllabus creator to view and analyze the material from other computer science syllabuses (licensed appropriately). It gives suggestions to the syllabus creator about the contents that can be added by faculties in their syllabus. Tool suggest as per the contents in the repository. This tool can suggest core as well as specific topics. The faculty can use this suggestion, can combine one or more suggestions, can add new topics, or filter it while creating his/her syllabus.

This tool is important because it gives suggestions to the syllabus creator about the topics that can include which will reduce the manual work. Its impact is to save time and effort, and reduce repetitive activities in the syllabus creation process.

The structure of the paper is: Section II contains background and related work. Section III states the problem. Section IV states the research hypothesis. Section V contains materials and methods that explain the methodology implemented by the text-analyzing tool. Section VI contains results and discussion that describe insight into the tool. Section VII discusses the conclusion and future work.

II. Background and Related Work

A. The Creation of a Repository of Syllabuses and Extracting Contents

Research has been done on creating intelligent queries to send to search engines, creating repositories, text retrieval, converting unstructured into structured content, and creating their content & publish it to the web.

Researchers developed a database of the syllabus. They used an efficient crawling mechanism for these purposes [4].

Searching on the web is an error-prone and time-consuming process because of irrelevant links on the web. Yu *et al.* [5] take the syllabuses from search engines and convert the unstructured syllabi into structured syllabi using various techniques. They created a structured syllabus repository that is accessible for searching Text Books, Syllabus contents, and other syllabus-related attributes.

Tungare *et al.* [6] created a syllabus repository for Courses at US universities. They also created a tool that allows professors to create syllabuses and publish them to the repository.

Manuscript received September 19, 2022; revised November 2, 2022, accepted November 15, 2022.

Ritu Sodhi and Jitendra Choudhary are with the Department of Computer Science, Medi-Caps University, India.

*Correspondence: ritu.sodhi@medicaps.ac.in

doi: 10.18178/ijiet.2023.13.2.1820

Joorabchi and Mahdi [7] worked on creating a national syllabus repository in Ireland. They developed and tested a prototype for the syllabus repository. It is a semi-automated framework for the national repository system.

B. The Semantic Analysis

There are studies related to the need for semantic analysis, methods used to do semantic analysis, improving the existing methods of semantic analysis, and combining the different methods on different data sets.

There is a need to improve techniques for teaching and learning that lead to how curricula are modeled and managed semantically [8–12].

Saquicela *et al.* [13] addressed a proposal to find the similarities between academic content by using semantic technologies and text mining.

Natural language is one of the important and difficult techniques in the field of artificial intelligence; the results of text retrieval are needed in many applications [14, 15].

According to Jelodar *et al.* [16], searching in the field for the meaning of words is needed.

The Word2vec technique is used to perform semantic analysis [17].

Maulud *et al.* [18] compared recent NLP techniques and conclude that advanced semantic methods have good accuracy.

Chen *et al.* [19] examined the biomedical domain for finding the performance of sentence similarity. In this research, a study of PubMed documents was conducted to determine the effectiveness of sentence similarity measures for sentence ranking. For sentence ranking, neither lexical nor semantic measures provide the desired results.

The attention weight method, syntactic information, and semantic information are all combined by Quan et al. to create a novel tree kernel for sentence similarity called the ACVT kernel [20].

Peng *et al.* [21] propose an improved Enhanced-RCNN model for sentence similarity. The architecture of Enhanced-RCNN is not as complicated as the BERT model. Based on the experimental results, Enhanced-RCNN exceeds baseline levels and achieves a competitive return on two real-world data sets.

C. The Automatic Generation of Questions and Evaluation

Research has been done on the automatic creation of question papers and evaluation of answer sheets.

Das *et al.* [22] suggested the automatic generation of questions and evaluating their answers. They extracted key phrases from the course curriculum (syllabus), and from that, they generate questions. A set of model answers was taken from different textbooks and subject experts to evaluate the answers of students' responses.

Ragasudha and Saravanan [23] did automatic question paper generation by creating a database of questions with blooms taxonomy and also evaluating the answer which is computerized in a subjective format using a keyword matching algorithm.

The previous studies do not compare the various syllabuses of the same courses and also do not suggest the topics that can be included in the syllabus to the syllabus creator.

III. Problem Statement

It is always a cumbersome job for faculties to manually create syllabi by analyzing various syllabuses. Research has been done regarding automatic syllabus repository creation and semantic comparisons. But no research has been done on analyzing various syllabuses of the same course. In this paper, this research proposes and implements a text-analyzing tool for creating a repository of computer science courses, analyzing various syllabuses of the same course, and giving suggestions to the faculties about the topic that can be included in syllabuses.

IV. Research Hypothesis

The tool for creating syllabus repositories and analyzing the syllabuses will reduce the amount of time and effort of faculties during syllabus creation. This research will let the education field and the researchers will start working on the automation of creating syllabuses.

V. Materials and Methods

This research developed the text analyzing tool using python programming language, streamlit framework, and MySQL database management system.

The methodology first created the repository of syllabuses of a particular course from a search engine and extracts the contents from the syllabuses given in the pdf. After that compare the different syllabuses and find similar topics. And then output the topics to the syllabus creator in the descending order of their occurrences or semantic similarity value.

The steps followed by methodology are shown in Fig. 1.

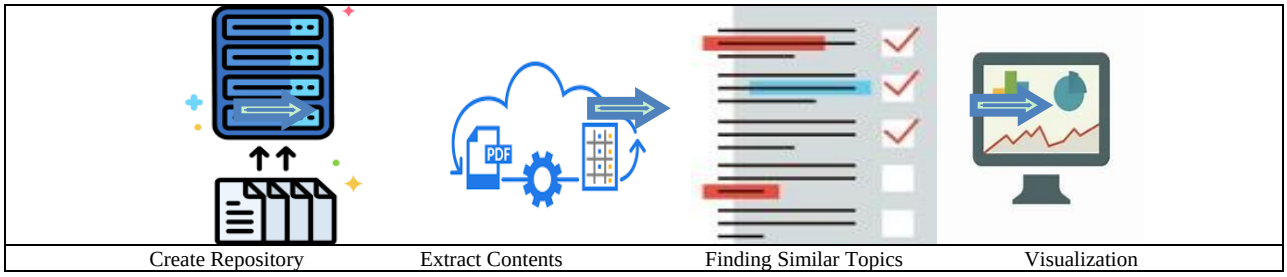

Fig. 1. Methodology for creating and analyzing syllabus repository.

The methodology described in Fig. 1 consists of four methodologies having some common modules. A description of each of the methodologies is given in subsections.

A. Methodology 1

The steps followed by Methodology 1 are

- Firstly create repository
- And extract contents manually
- And compare syllabuses by finding the frequency of each topic
- And finally, output the results

A detailed description of the steps are

1) Create a repository

Firstly the module creates a syllabus repository from a search engine. The detailed process of creating the repository module is shown in Fig. 2.

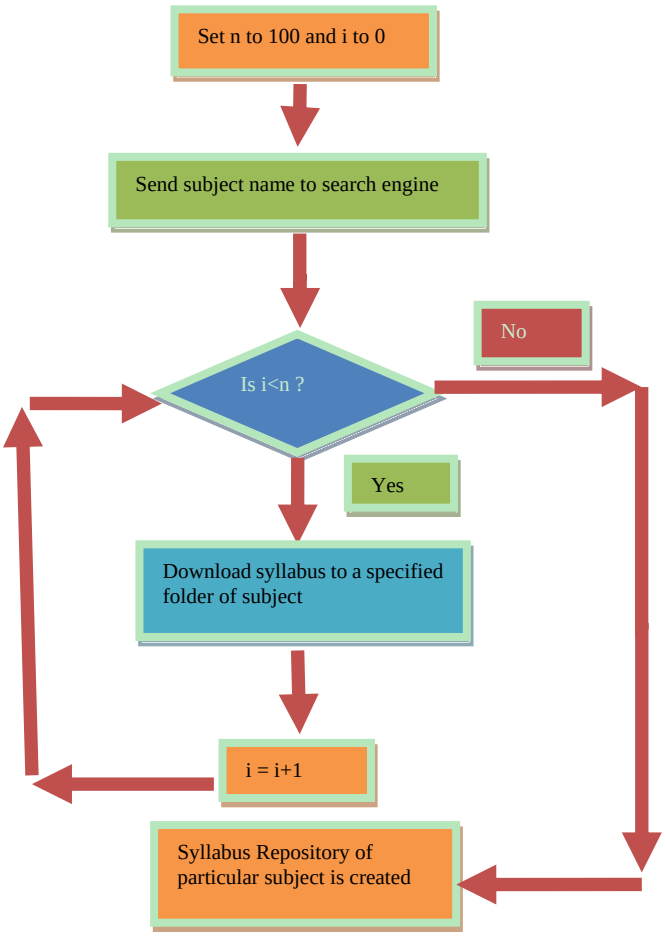

Fig. 2. Flow chart explaining the process used by the tool for creating the repository.

The syllabus available on search engines is in various formats such as PDF, HTML, WORD, etc ([13]). But this research is limited to only PDFs. This module creates a repository of pdf syllabus files ([7]).To create a syllabus repository first set n to 100 and i to 0 (n is the number of syllabuses that need to be downloaded). Send the name of the syllabus to the search engine and download the syllabus one by one from the search engine to the structured folder for later retrieval [6]. The documents found in the repository are of three type's full syllabus, syllabus of one subject, and noise. Manually removed the full syllabus and noise from the repository and kept only the syllabus of one subject in the repository.

2) Extract contents manually

This module extracted the contents manually. Because extracting contents from pdf is itself a big research topic. This module created separate text files for each of the syllabi in pdf manually. After extraction, it becomes easier for the tool to compare various contents.

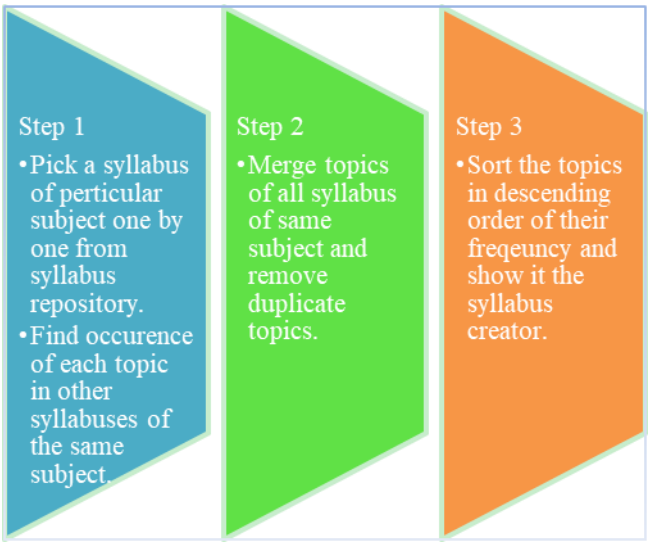

Fig. 3. Steps implemented by the tool for finding the most frequent topics in various syllabuses.

3) Compare syllabuses by finding the frequency of each topic

Sometimes faculty want to see the relevancy and importance of a particular topic by finding how many other syllabuses, the topic is included. This methodology has done this by finding the frequency of different topics.

This module finds occurrences of each topic in the syllabus repository. The detailed process is shown in Fig.3.

In past research, work has been done to analyze the contents of academics ([13]). This technique shown in Fig. 3 analyzes course contents of different syllabuses and found the most frequent topics.

This comparison mechanism can be used as a similarity score. This tool recommends to its users what topics can be included in their syllabuses by finding the most frequent topics in various syllabuses. This will enable future tools to recommend textbooks, course objectives, course outcomes, etc. obtained from the set of various syllabi [6].

For example, there are three syllabuses the first syllabus has a topics array, function, and pointer. The second syllabus has a topics array, structure, and pointer. The third Syllabus has the topics union, pointer, and file handling.

By finding frequency pointer occurred 3 times, the array occurs 2 times, and function, structure, union, and file handing each occurs one time. So the output will be in descending order of their frequency.

Fig. 4 is a screenshot of implemented tool. In this Fig., the user enters the name of the syllabus of which they want to see the most frequent topics.

Fig. 5 is another screenshot of implemented tool. It displays the most frequent topics in the syllabus of C Programming.

4) Visualization of results

Visualization of information is important to show the results [24]. This research used a streamlit framework for displaying the results.

B. Methodology 2

The steps followed by Methodology 2 are

- Firstly create repository
- And extract contents manually
- And compare syllabuses semantically and find semantic similarity value
- And output the results

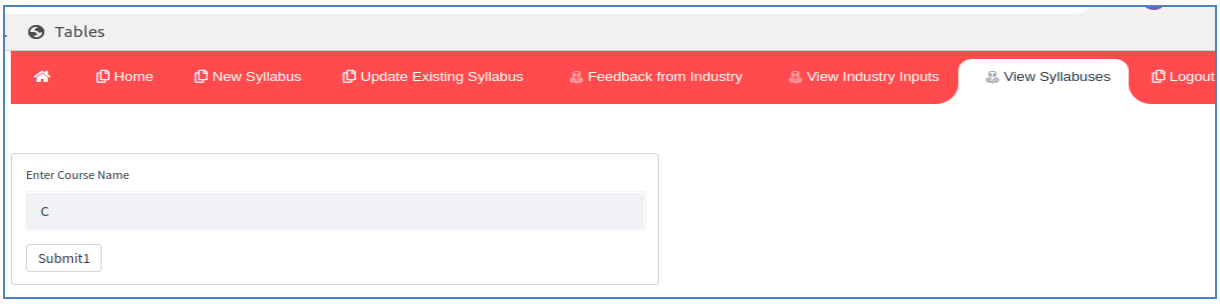

Fig. 4. Screenshot of the tool for course name input.

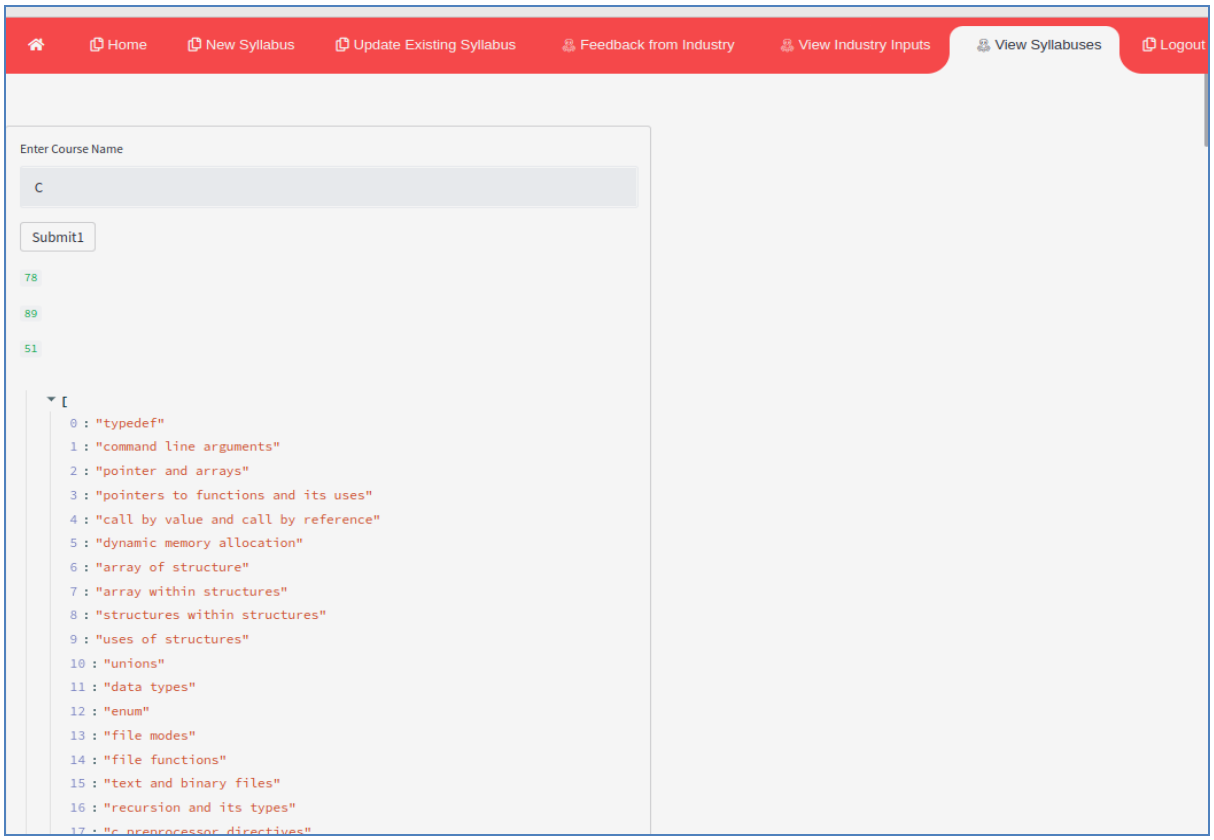

Fig. 5. Screenshot of the tool that displays the most frequent topics in different syllabuses.

The detailed description of the steps followed by Methodology 2 is

1) Create a repository

In this step, the tool creates a repository automatically from the web. This step is the same as described in step 1 of Methodology 1.

2) Extract Contents Manually

In this step, the tool extracts the contents from each of the syllabi for further comparison. This step is the same as described in step 2 of Methodology 1.

3) Compare syllabuses semantically and find the semantic similarity value

This module calculates a semantic similarity value for each topic of the syllabus of the repository. If the repository has 3 syllabuses. Syllabus 1 has topics a1, a2, and a3. Syllabus 2 has topics b1, b2, and b3. Syllabus 3 has topics c1, c2, and c3. Topic a1 is semantically compared to b1,b2,b3,c1,c2 and c3. The value comes after each comparison is between 0 and 1, Sum up all the values. Similarly, each topic of one syllabus is compared with the topics of other syllabuses, and the semantic similarity count for that topic is calculated. Topics will be sorted in descending order of their semantic similarity count.

The semantic approach can improve the results and accuracy. For example, terms given in Table I are semantically similar in the syllabus so they must be counted as are same terms.

TABLE I: SEMANTICALLY SIMILAR TERMS APPEAR IN C PROGRAMMING SYLLABUS

Topic 1	**Topic 2**
Formal Parameters	Formal Arguments
Method	Function
Loop	Iteration
Function Declaration	Function Prototype

In Table I topic 1 is semantically similar to topic 2 of the same row. There are also many different sentences such as “Define an array”, “what is an array”, and “Introduction of an array” which are semantically similar. These semantic similar sentences can be identified during semantic comparison. So it will improve the accuracy as they are the same terms that can appear in different syllabuses. Taking another example of topics of Software Testing syllabuses is given in Table II.

TABLE II: SEMANTICALLY SIMILAR TERMS APPEAR IN SOFTWARE TESTING SYLLABUS

Topic 1	**Topic 2**
Error	Mistake
White Box Testing	Glass Box Testing
Volume Testing	Flood Testing
Testing Goals	Testing Objectives

In Table II topic 1 is similar to topic 2 of the same row. Another example is syllabus creator may write the topics in different ways such as “Attributes of testing” or “Testing attributes”, “Characteristics of test case” or “Test case characteristics” etc.

From the above examples, it is clear that there is a need to findsimilar topics as many topics are presented in different ways or have different names for the same topic. So there is a need to do a semantic comparison.

This module compares each topic of each syllabus with other topics of other syllabuses and finds the total semantic similarity count for each topic. The process is shown in Fig. 6. Using a spacy large English model for semantic comparison [25]. It can be further improved by training a model for computer terminologies.

4) Visualization of results

This step will display the topics in descending order of their semantic value.

C. Methodology 3

Methodology 3 will do

- Firstly create repository
- And then extract contents automatically
- And then normalizing the text
- And compare Syllabuses by finding the frequency of each topic
- And output the results

A detailed description of methodology 3 is given below

1) Create a repository

In this step, the tool creates a repository of syllabuses automatically from the web. This step is the same as described in step 1 of Methodology 1

2) Extract contents automatically

This module extract contents from syllabuses automatically,observed 100 syllabuses for C Programming, and found the most frequent terms for starting and end points of syllabus contents. Table III contains some common terms for starting and end points of syllabus contents. The contents of the syllabus are in between starting and end points ([7]). In the future, this process can be improved by building an automatic classifier that will allow us to find the start and end points of syllabi. This can be improved by the process of finding topical terms by using machine learning algorithms such as the decision tree algorithm [26], support vector machines, or Naıve Bayes methods [26] for this purpose.

TABLE III: MOSTLY USED TERMS FOR STARTING AND END POINTS OF SYLLABUS CONTENTS

Start Point	**End Point**
UNIT-I	Text Books
Module-I	Suggested Readings
Session- I	Reference Books
Chapter-I	RECOMMENDED BOOKS
Course Contents	BOOKS
Syllabus	Suggested Activities

3) Text normalization

This module uses normalization techniques that are removing stop words, handling whitespaces, converting to lowercase, and handling numbers, and removing header & footer from pdfs ([13]).

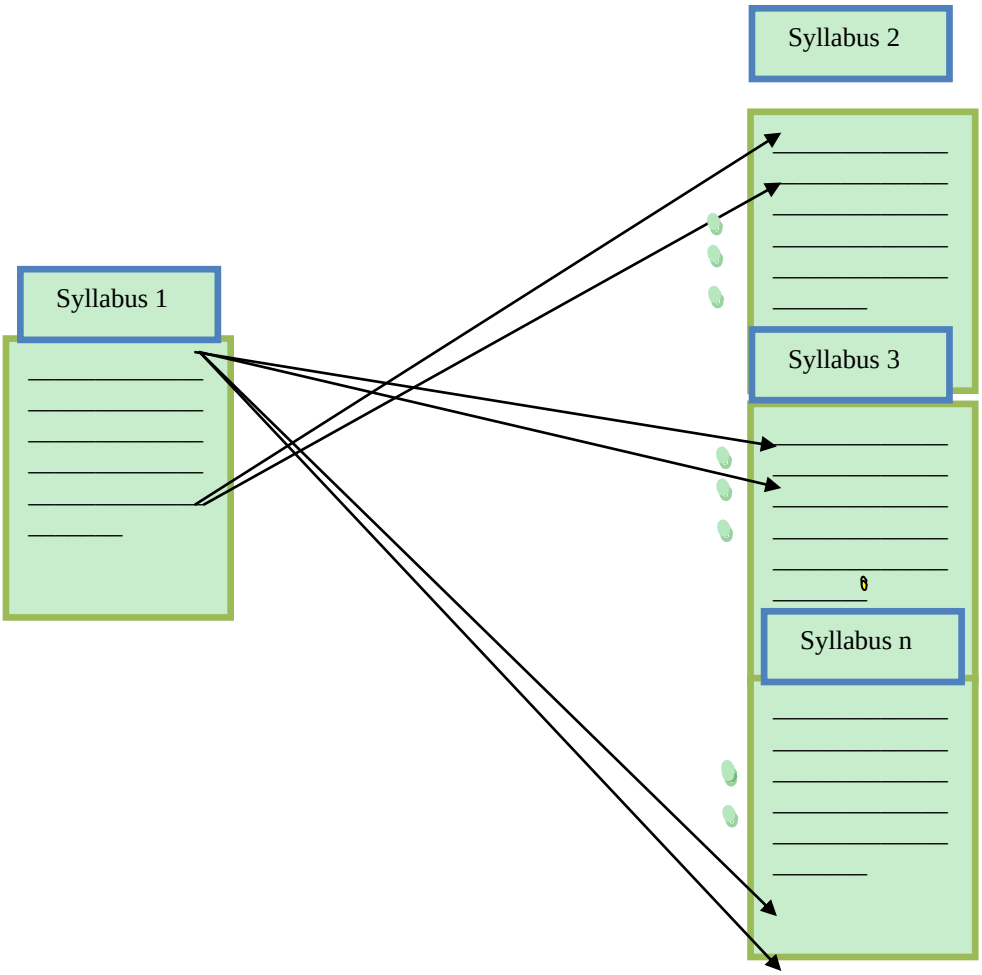

Fig. 6. Comparison of each topic of the syllabus with other syllabuses semantically.

4) Compare syllabuses by finding the frequency of each topic

After normalization frequency of each topic in the repository is calculated. This step is the same as described in step 3 of Methodology 1.

5) Visualization of results

This step will display the topics in descending order of their frequency.

D. Methodology 4

Methodology 4 will do

- Firstly create repository
- And then extract contents automatically
- And then normalizing the text
- Compare syllabuses semantically and find semantic value
- And output the results

A detailed description of methodology 4 is given below

1) Create a repository

In this step repository of syllabuses is created as in step 1 of Methodology 1.

2) Extract contents automatically

As there are many things in the syllabus other than the contents. Code is developed to find the core contents and extract the contents from pdfs. This step is the same as step 2 of Methodology 3

3) Text normalization

In this step, the tool normalizes the contents before comparison. This step is the same as described in step 3 of Methodology 3

4) Compare syllabuses semantically and find semantic value

After normalizing the text, this tool compares topics of one syllabus with another and calculates each topic's semantic count. This step is the same as described in step 3 of Methodology 2

5) Visualization of results

This step will give the output in descending order by their semantic value.

VI. Results and Discussion

The results of this tool are shown in Fig. 4 and Fig. 5.

The summary of the results of this tool are:

1) Firstly repository of the 100 syllabuses of C Programming is created from the web by executing the code.
2) Extracted the contents from the Syllabuses.
3) Compared the topics of the syllabuses of the C Programming
4) Give Suggestions about similar topics, the result is shown in Fig. 5.

From previous studies, repositories of syllabuses from the web are created by sending intelligent queries to the web and converting unstructured data into structured data ([5]). Tungare et al., (2007) compared the contents of syllabuses to find common knowledge units in similar courses. Saquicela et al., (2018) analyze the academic content ([13]). But they did not compare syllabuses of the same course for the purpose to suggest the topics to the faculties.

Some problems occur while fully automatically extracting contents. Problems while extracting contents

1) Most of the syllabuses available on search engines have syllabi of whole courses not of a single subject.
2) Tutorials, sessions, notes, etc documents are also coming while searching the syllabus of a particular course.
3) PDFs are having headers and footers, and this tool is unable to remove them automatically.
4) Syllabus contents on search engines are not of the same format.

These problems can be resolved by training a machine language classifier to extract the content of a particular syllabus from the whole syllabus and identify the syllabuses from other academic documents. Code can be developed to remove the headers and footers from pdf automatically and convert the heterogeneous format documents extracted from the web into a single format.

After extracting the contents this tool compares topics of various syllabuses. The result is promising and gives more frequent topics to appear in various syllabuses. This tool is also executed for semantic comparison among topics. Semantic comparison can be improved by creating a dedicated model for computer terminology and synonyms.

VII. Conclusion and Future Work

This paper proposed and implemented a tool that creates a syllabus repository of computer science courses from search engines and analyzes it. This tool first creates a syllabus repository in pdf format and extracts the contents. After that, this tool performs a comparison among topics using two different algorithms. The first algorithm finds out the most frequent topics in various syllabuses and the second finds the most frequent semantically similar topics.

In the future, the algorithm can be improved at many levels. It can be trained to create a syllabus repository from search engines. This research can also be improved by using machine learning techniques to find out the start and end points of syllabuses. This tool can also be improved by creating a model for computer terminology like the English model that will give more accurate results for semantic comparison of topics of computer science courses.

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

Ritu Sodhi and Jitendra Choudhary conducted the research work and wrote the paper. Jitendra Choudhary analyzed the paper. All authors had approved the final version.

References

[1] Y. Xiaoyan, T. Manas, F. Weiguo, P. Q. Manuel, A. F. Edward, C. William, T. Guo-Fang, and C. Lillian, "Automatic syllabus classification," in *Proc. the ACM IEEE Joint Conference on Digital Libraries, Vancouver, BC, Canada*, 2007, Doi: http://doi.acm.org/10.1145/1255175.1255265

[2] D. Habanek, "An examination of the integrity of the syllabus," *College Teaching*, vol. 53, no. 2, pp. 62–64, 2005.

[3] A. M. Tokatl and Y. Keli, "Syllabus: How much does it contribute to the effective communication with the students?" *Social and Behavioral Sciences*, vol. 1, no. 1, pp. 1491–1494, 2009.

[4] Y. Matsunaga, S. Yamada, E. Ito, and S. Hirokawa, "A web syllabus crawler and its efficiency evaluation," in *Proc. the International Symposium on Information Science and Electrical Engineering 2003 (ISEE 2003)*, Fukuoka, Japan, 2003.

[5] X. Yu, M. Tungare, W. Fan, M. P ́erez-Qui˜nones, E. A. Fox, W. Cameron, G. Teng, and L. Cassel, "Using automatic metadata extraction to build a structured syllabus repository," *International Conference on Asian Digital Libraries*, pp. 337-346, 2007.

[6] M. Tungare, X. Yu, W. Cameron, G. Teng, M. A. P ́erez-Qui˜nones, L. Cassel, W. Fan, and E. A. Fox, "Towards a syllabus repository for computer science courses," in *Proc. SIGCSE 2007*, Covington, Kentucky, USA, vol. 39, pp. 55-59, 2007.

[7] A. Joorabchi and A. E. Mahdi, "An automated syllabus digital library system for higher education in Ireland," *The Electronic Library*, vol. 27, no. 4, pp. 640-658, 2009.

[8] H. Chung and J. Kim, "Semantic model of syllabus and learning ontology for intelligent learning system," in *Proc. Computational Collective Intelligence. Technologies and Applications: 6th International Conference*, pp. 175-183, 2014.

[9] H. Chung and J. Kim, "An ontological approach for semantic modeling of curriculum and syllabus in higher education," *International Journal of Information and Education Technology*, pp. 365-369, 2016, DOI: 10.7763/IJIET.2016.V6.715,2016

[10] B. Deliyska and P. Manoilov, "Ontologies in intelligent learning systems," *Intelligent Learning Systems and Advancements in Computer-Aided Instruction: Emerging Studies*, pp. 31–48, 2012.

[11] G. Demartini, I. Enchev, J. Gapany, and P. Cudr-Mauroux, "The bowlogna ontology: Fostering open curricula and agile knowledge bases for europes higher education landscape," *Semantic Web*, vol. 4, no. 1, pp. 53-63, 2013.

[12] P. Miltenoff, J. Keengwe, and G. Schnellert, "Technological strategic planning and globalization in higher education," *Learning Tools and Teaching Approaches through ICT Advancements*, pp. 348–358, 2013.

[13] V. Saquicela, F. Baculima, G. Orellana, N. Piedra, M. Orellana, and M. Espinoza, "Similarity detection among academic contents through semantic technologies and text mining," in *Proc. IWSW 2018 - International Workshop on Semantic WebAt: Habana 2096*, 2018.

[14] X. Liaoand and Z. Zhu, "Classification of natural language semantic relations under deep learning," in *Proc. IEEE International Conference on Advances in Electrical Engineering and Computer Applications (AEECA)*, pp. 1025-1027, 2020.

[15] A. A. Maksutov, V. I. Zamyatovskiy, V. N. Vyunnikov, and A. V. Kutuzov, "Knowledge base collecting using natural language processing algorithms," in *Proc. IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus)*, pp. 405-407, 2020.

[16] H. Jelodar, Y. Wang, M. Rabbani, G. Xiao, and R. Zhao, "A collaborative framework based for semantic patients-behavior analysis and highlight topics discovery of alcoholic beverages in online healthcare forums," *Journal of Medical Systems*, vol. 44, no. 5, 2020.

[17] X. Wang, X. Dong, and S. Chen, "Text duplicated-checking algorithm implementation based on natural language semantic analysis," in *Proc. IEEE 5th Information Technology and Mechatronics Engineering Conference (ITOEC)*, pp. 732-735, 2020.

[18] D. H. Maulud, S. R. M. Zeebaree, K. Jacksi, M. A. M. Sadeeq, and K. Hussein, "State of art for semantic analysis of natural language processing," *Qubahan Academic Journal*, vol. 1, no. 2, pp. 21-28, 2021.

[19] Q. Chen, S. Kim, W. J. Wilbur, and Z. Lu, "Sentence similarity measures revisited: Ranking sentences in pubmed documents," in *Proc. 2018 ACM International Conference*, Washington, DC, USA, 2018.

[20] Z. Quan, Z. Wang, Y. Le, B. Yao, K. Li, and J. Yin, "An efficient framework for sentence similarity modeling," in *Proc. IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 27, no. 4, pp. 853-865, 2019.

[21] S. Peng, H. Cui, N. Xie, S. Li, J. Zhang, and X. Li, "Enhanced-RCNN: An efficient method for learning sentence similarity," in *Proc. the Web Conference*, pp. 2500-2506, 2020.
[22] B. Das, M. Majumder, A. A. Sekh, and S. Phadikar, "Automatic question generation and answer assessment for subjective examination," *Cognitive Systems Research*, vol. 72, pp. 14–22, 2022.
[23] R. Ragasudha and M. Saravanan, "Secure automatic question paper generation with the subjective answer evaluation system," in *Proc. International Conference on Smart Technologies and Systems for Next Generation Computing (ICSTSN)*, vol. 1-5, doi: 10.1109/ICSTSN53084.2022.9761323, 2022.
[24] E. Segel and J. Heer, "Narrative visualization: Telling stories with data," in *Proc. IEEE Transactions on Visualization and Computer Graphics*, vol. 16, no. 6, pp. 1139-1148, 2010.
[25] N. Sanprasit, T. Titijaroonroj, and K. Kesor, "A semantic approach to automated design and construction of star schemas," *Engineering and Applied Science Research*, vol. 48, no. 5, pp. 518-528, 2021.
[26] T. Mitchell, *Machine Learning*, McGraw-Hill, 1997.

Copyright © 2023 by the authors. This is an open access article distributed under the Creative Commons Attribution License which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited (CC BY 4.0).

Ritu Sodhi was born in 1984 in Indore, India. She received the MCA degree from IIT – Roorkee in 2008.

She is presently working on a Ph.D. degree in computer science at Medi-Caps University, Indore. Her areas are natural language processing, algorithms, and education. She is an assistant professor at Medi-Caps University, Indore, M.P., India.

She has 14 years of teaching work experience at the UG and PG levels.

Jitendra Choudhary was born in 1984 at Dewas, India. He received his Ph.D. degree from D.A.V.V. Indore in 2014.

His areas are software engineering, JAVA Programming & Technologies, and software testing. His research area includes Extreme Programming and Software Maintenance. He is an associate professor and HOD CS at Medi-Caps University, Indore, M.P., India.

He has 17 years of teaching work experience at the UG and PG levels.