IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940



International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

Research-Paper-to-Visual-Map Generator

Miss. Krutika Pravin Borse[1], Prof. Manoj V. Nikum[2], Prof. Mrs. S. N. Raul[3]

Research Scholar MCA, Shri Jaykumar Rawal Institute of Technology, Dondaicha, India[1]

HOD MCA & Assistant Professor, Shri Jaykumar Rawal Institute of Technology, Dondaicha, India[2]

Assistant Professor, Shri Jaykumar Rawal Institute of Technology, Dondaicha, India[3]

Abstract: Academic research papers are long and densely structured, and readers, particularly students and early-stage researchers, often need several careful readings before the way a paper's core concepts relate to one another becomes clear. This slows down literature review, especially when many papers have to be covered in limited time. This paper presents the design of a Research-Paper-to-Visual-Map Generator, a system that accepts an uploaded research paper and produces an interactive concept map of its key sections, main ideas and the relationships between them. Natural language processing is used to parse the document structure, identify section boundaries and extract key terms and phrases. A relationship-extraction step then links the extracted concepts using contextual proximity and section linkage, and the result is arranged as a hierarchical or network-style map in a web interface, where selecting a node shows the matching excerpt from the original text. The concept extraction and map construction stages were implemented as a prototype and run on the text of this paper to examine the output. The expected outcome is a working tool that shortens initial comprehension and supports faster assessment, note-taking and comparison of research papers.

Keywords: Natural Language Processing, Concept Extraction, Text Summarization, Knowledge Graph, Visual Mapping, Document Structure Analysis, Literature Review Automation, Information Visualization

I. INTRODUCTION

A research paper is written to be read in a fixed order, but it is rarely read that way. Most readers open the abstract, glance at the figures, skim the conclusion and only then decide whether the methodology deserves a closer look. For a student who is new to a field this habit does not work well. The terminology is unfamiliar, the sections depend on one another, and the main argument is spread over many pages. A paper often has to be read two or three times before the reader can say how the problem, the method and the findings fit together [1].

The difficulty grows with volume. A literature review for even a small project involves dozens of papers, and the time available for each one is short. Notes taken in a linear form, such as highlighted text or a list of bullet points, record what a paper says but not how its ideas are connected. A visual representation can fill this gap. Concept maps, which show ideas as nodes and the relationships between them as labelled links, have long been used in education to organise and share knowledge [2], [3]. Drawing one by hand for every paper, however, takes about as long as reading the paper again. This paper describes the design of a Research-Paper-to-Visual-Map Generator, a web-based tool that automates that step. The user uploads a paper. The system finds its sections, extracts the key terms in each of them, works out how those terms are related and draws the result as an interactive map. Selecting a node shows the passage of the original text that the node came from, so the map works as a way into the paper and not as a replacement for it.

The objectives of the work are: (i) to detect the section structure of an uploaded paper; (ii) to extract the central concepts and key phrases of each section; (iii) to find meaningful relationships between those concepts using contextual proximity and section linkage; (iv) to present the result as a hierarchical or network-style map in a web interface; and (v) to link every node back to its source text.

The rest of the paper is organised as follows. Section II reviews related work. Section III describes the proposed methodology. Section IV reports a prototype run of the concept extraction and map construction stages, and Section V concludes with the limitations of the present design and the work that remains.

II. LITERATURE REVIEW

The proposed system sits at the meeting point of four research areas: concept mapping, keyword extraction, document structure analysis and relation extraction. Each is summarised below, followed by the gap that this work tries to fill.

 This work is licensed under a Creative Commons Attribution 4.0 International License

A. *Concept Maps and Visual Knowledge Representation*

Novak and Cañas describe a concept map as a graph in which concepts are placed in nodes and the relationship between two concepts is written on the line that joins them, with the more general concepts placed higher up [2]. Eppler compared concept maps with mind maps, conceptual diagrams and visual metaphors and noted that the formats differ in how much structure they impose and in the kind of knowledge each one suits [3]. For a research paper, where the aim is to show how a problem, a method and a result depend on each other, the concept map is the closer fit. The practical value of an overview is also visible in Keshav's advice to read a paper in passes, beginning with a quick first pass that only establishes what the paper is about [1].

B. *Keyword and Key-Phrase Extraction*

Term weighting by TF-IDF, studied by Salton and Buckley, is still the usual starting point: a term is taken to be important when it is frequent in a document and rare in the rest of the collection [4]. Rose et al. introduced RAKE, which splits the text at stop words and punctuation to obtain candidate phrases and scores each phrase from the frequency and co-occurrence of its words; it works on a single document and needs no training data [5]. TextRank, by Mihalcea and Tarau, builds a graph of words linked by co-occurrence within a window and ranks them in the same way that PageRank ranks web pages [6]. Newer methods such as KeyBERT compare the embedding of each candidate phrase with the embedding of the whole document, so that phrases are chosen by meaning and not only by frequency [7], [8].

C. *Document Structure Analysis*

Scientific papers have a recognisable layout, but a PDF file does not store it. Text comes out of a PDF as a stream of lines, and headings, captions and column breaks have to be recovered. GROBID is an open-source tool that uses machine learning to recover the header, the body sections and the bibliography of scientific PDFs [9]. When only plain text is available, headings can be found from numbering or capitalisation. For splitting text by topic, Hearst's TextTiling compares the vocabulary of neighbouring blocks of sentences and places a boundary where the similarity drops [10].

D. *Relationship Extraction and Graph Representation*

The simplest way to relate two concepts is co-occurrence: if they appear close together, there is probably some connection between them. This is the idea behind TextRank [6] and it is cheap to compute, but it cannot say what kind of connection it is. Dependency parsers such as the one in Stanford CoreNLP make it possible to derive subject-verb-object relations that name the link [11], and language models trained on scientific text, such as SciBERT, represent technical vocabulary better than general-purpose models [12]. Cohan et al. also showed that representing the sections of a long scientific document explicitly helps when it is summarised [13]. Graphs built from any of these sources can be stored and analysed with libraries such as NetworkX [14].

E. *Interactive Visualization*

Shneiderman's information-seeking mantra, "overview first, zoom and filter, then details on demand", describes the behaviour wanted from a paper map very well [15]. In the browser, D3 binds data to SVG elements and supports force-directed graphs with zooming, dragging and click events [16].

F. *Research Gap*

Table I sets the approaches side by side. Keyword extractors return a flat list and ignore the structure of the paper. Structure parsers recover the sections but not the ideas inside them. Relation extractors work at sentence level. Tools for drawing concept maps exist, but they expect the user to supply the content. What is missing is one pipeline that takes a whole paper, keeps its section structure, links concepts across sections and lets the reader go from the map back to the source text. That is the gap addressed here.

TABLE I
SUMMARY OF RELATED APPROACHES AND THEIR LIMITATIONS

Approach	Representative work	Strength	Limitation
TF-IDF term weighting	Salton and Buckley [4]	Simple and fast; needs no training	Flat list of terms; no structure or relations
RAKE	Rose et al. [5]	Works on a single document	Depends on the stop-word list; no links between phrases
TextRank	Mihalcea and Tarau [6]	Ranks terms from a co-occurrence graph	Links are unlabelled; sections are ignored
KeyBERT	Grootendorst [7]	Chooses phrases by meaning	Needs an embedding model; structure is ignored

Approach	Representative work	Strength	Limitation
TextTiling	Hearst [10]	Finds topic boundaries in plain text	Gives boundaries, not concepts
GROBID	Lopez [9]	Recovers sections from PDF files	Stops at text structure; no concept layer
Manual concept mapping	Novak and Cañas [2]	Shows relationships clearly	Slow; has to be redone for every paper

III. METHODOLOGY

The system is organised as a pipeline of six stages, shown in Fig. 1. Every stage takes the output of the one before it, so a stage can be replaced by a better method later without changing the rest.

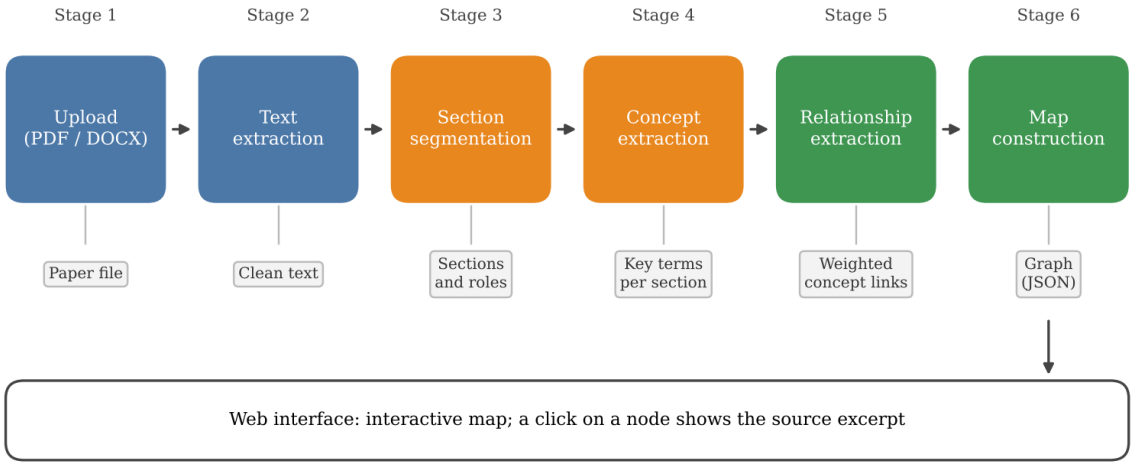


Fig. 1 Architecture of the proposed Research-Paper-to-Visual-Map Generator.

A. *Document Upload and Text Extraction*

The user uploads a paper in PDF or DOCX format through the web interface. Text is read page by page. For PDF files a layout-aware extractor is used so that two-column pages are read in the correct order. Headers, footers and page numbers that repeat on every page are removed and hyphenated line breaks are joined. Line breaks are otherwise kept, because they help in recognising headings in the next stage.

B. *Section Segmentation*

A line is treated as a heading when it follows one of the usual patterns in papers: a Roman or Arabic numeral followed by a title, a title written in capital letters, or a standard name such as Abstract, Introduction, Methodology, Results or Conclusion standing alone on a line. The text between two headings forms one section. If a paper has no usable headings, the text is divided by the change in vocabulary, in the manner of TextTiling [10], and each block is named after its highest-scoring terms. A section is also given a role (background, method, result or conclusion) from its heading, and the role decides where the section sits in the final map.

C. *Concept Extraction*

Inside each section the text is converted to lower case, split into sentences and words, stripped of stop words and reduced to word stems. The candidates are single words and two-word phrases made of adjacent content words. The score of a candidate is its frequency in the section multiplied by $\ln(1 + S/df)$, where S is the number of sections and df is the number of sections in which the candidate appears. Terms that occur in every section, such as the word paper itself, are pushed down by this factor, while terms that characterise one section rise. A single word that is only part of a selected phrase is dropped. The k best candidates of each section ($k = 4$ by default) become concept nodes. An embedding-based re-ranking step in the style of KeyBERT [7] can be placed on top of this baseline.

D. *Relationship Extraction*

Two kinds of link are created. A contextual link joins two concepts that occur in the same sentence, and its weight is the number of sentences in which they occur together. A section link joins a concept to each other section in which it also appears, which shows how an idea introduced in the background comes back in the method or in the findings. Contextual

 This work is licensed under a Creative Commons Attribution 4.0 International License

links that fall below a threshold (by default, fewer than two shared sentences) are dropped so that the map stays readable. Where the parser finds a verb between two concepts in a sentence, the verb can be stored as a label on the link.

E. *Map Construction*

The concepts and links form a graph G = (V, E) with three kinds of node: a root node for the paper, one node for each section and the concept nodes. Section nodes hang from the root in the order in which they occur in the paper, and each concept node hangs from the section in which its score is highest. In hierarchical mode this tree is drawn as it is, with the contextual and section links added as thin cross-connections. In network mode a force-directed layout is used, with node size set by the number of links and node colour set by section role.

F. *Web Interface*

The front end is a single-page web application. After the upload, the server returns the graph as JSON and the map is drawn in the browser. The user can zoom, drag nodes, collapse a section and search for a term. A click on a node opens a side panel that shows the section name and the sentences from which the concept was taken, with the concept highlighted. This is the "details on demand" step of Shneiderman's mantra [15]. The map can be exported as an image or as JSON, so that it can be placed in notes or compared with the map of another paper.

G. *Technology Stack*

The tools proposed for each layer are listed in Table II. All of them are free and open source, which keeps the system within the reach of a student project.

TABLE II
PROPOSED TECHNOLOGY STACK

Layer	Tools	Purpose
Text extraction	PyMuPDF, python-docx	Reading PDF and DOCX files
Language processing	NLTK or spaCy, scikit-learn	Sentence splitting, stop words, term scoring
Graph handling	NetworkX [14]	Building and analysing the concept graph
Back end	Python, Flask	Upload handling and a JSON interface
Front end	HTML, CSS, JavaScript, D3.js [16]	Interactive map and source-text panel

H. *Evaluation Plan*

The tool will be evaluated at two levels. For technical quality, section detection will be checked against headings marked by hand, and the extracted concepts will be compared with the author keywords printed on each paper, which are available without extra labelling effort. For usefulness, a small group of students will answer questions about a paper after reading it normally and after using its map, and the time taken and the number of correct answers will be recorded for both conditions.

IV. RESULT

The complete system is still under development. To check that the concept extraction and map construction stages behave sensibly, they were written as a Python script, using the stop-word list of scikit-learn and the NetworkX library, and run on the text of Sections I to III of this paper. The section headings were supplied as boundaries. The input was 1,907 words in 95 sentences. Only one document and no reader study are involved, so the results below describe what the prototype produced and are not a measurement of accuracy.

A. *Extracted Concepts and Map*

Table III lists the four highest-scoring concepts of each section, and Fig. 2 shows the resulting map. It has 12 concept nodes under three section nodes, 16 contextual links and 10 section links. The strongest contextual links were concept and section (11 sentences) and concept and map (8 sentences).

 This work is licensed under a Creative Commons Attribution 4.0 International License

TABLE III
CONCEPTS SELECTED BY THE PROTOTYPE FOR EACH SECTION (SCORE IN BRACKETS)

Section	Words	Sentences	Top concepts
I. Introduction	407	24	extract (2.77), methodology (1.83), reader (1.83), work (1.83)
II. Literature Review	685	32	concept (7.62), co-occurrence (6.24), map (5.55), scientific (5.55)
III. Methodology	815	39	section (14.56), node (7.62), word (6.41), link (5.55)

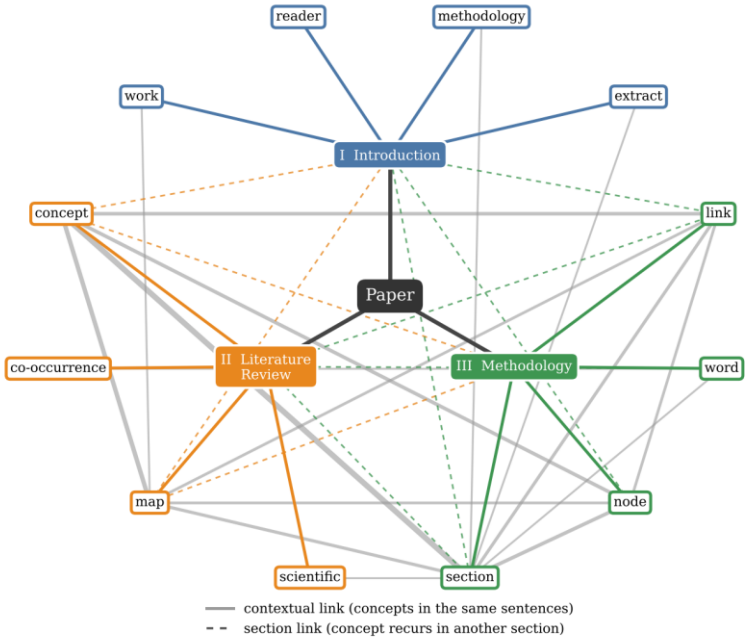


Fig. 2 Concept map produced by the prototype from Sections I to III of this paper. Solid coloured lines show the section hierarchy, grey lines show contextual links and dashed lines show section links.

B. *Discussion*

For Sections II and III the concepts describe the content fairly well. The Literature Review is represented by concept, map, co-occurrence and scientific, and the Methodology by section, node, link and word, which is the vocabulary of the graph construction described there. The word concept appears eleven times in each of Sections II and III and four times in Section I, and the section links in Fig. 2 show this recurrence across the paper. That is the kind of cross-section connection the map is meant to expose.

The Introduction gave the weakest result. Its concepts (extract, methodology, reader and work) are words that happen to be repeated and do not describe the section. The section is short and written in general language, so few terms occur twice and the pool of candidates is small. A frequency baseline is therefore least reliable on short sections, which is where the embedding-based re-ranking proposed in Section III-C should help. A second weakness is that only one two-word phrase was selected. The phrase concept map occurs five times in Section II but loses to the single word concept, which occurs eleven times.

The prototype covers only two of the six stages. PDF extraction, automatic heading detection, link labelling and the web interface were not part of this run. If the full system behaves like the prototype, a reader would see the main vocabulary of each section and the links between sections in one view before reading, which is the quick first pass that Keshav recommends [1]. Whether this actually reduces reading time can only be settled by the evaluation described in Section III-H.

V. CONCLUSION

Reading a research paper well means knowing how its parts fit together, and that is exactly what continuous text hides. This paper proposed a Research-Paper-to-Visual-Map Generator that detects the sections of an uploaded paper, extracts key concepts from each section, links them through contextual proximity and section linkage, and presents the result as an interactive map that leads back to the original text. A prototype of the concept extraction and map construction stages produced a readable map of 12 concepts from about 1,900 words of text.

The prototype also showed where the present design is weak. Frequency-based scoring is unreliable on short sections, favours single words over phrases, and says nothing about the type of a relationship. PDF extraction, automatic heading detection and the web interface have been designed but not yet tested, and no reader study has been carried out, so the claim that the tool saves reading time remains to be shown.

The next steps follow from these limits: re-ranking candidates with sentence embeddings, labelling links with the verb found between two concepts, building the web front end, and running the evaluation described in Section III. Placing the maps of several papers on one canvas, so that shared concepts can be compared directly, is a natural extension for literature review.

ACKNOWLEDGMENT

The authors thank the Department of MCA, Shri Jaykumar Rawal Institute of Technology, Dondaicha, for the facilities and the guidance given during this work.

REFERENCES

[1]. S. Keshav, "How to read a paper," *ACM SIGCOMM Computer Communication Review*, vol. 37, no. 3, pp. 83-84, 2007.

[2]. J. D. Novak and A. J. Cañas, "The theory underlying concept maps and how to construct and use them," Florida Institute for Human and Machine Cognition, Pensacola, FL, USA, Tech. Rep. IHMC CmapTools 2006-01 Rev 01-2008, 2008.

[3]. M. J. Eppler, "A comparison between concept maps, mind maps, conceptual diagrams, and visual metaphors as complementary tools for knowledge construction and sharing," *Information Visualization*, vol. 5, no. 3, pp. 202-210, 2006.

[4]. G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Information Processing & Management*, vol. 24, no. 5, pp. 513-523, 1988.

[5]. S. Rose, D. Engel, N. Cramer and W. Cowley, "Automatic keyword extraction from individual documents," in *Text Mining: Applications and Theory*, M. W. Berry and J. Kogan, Eds. Chichester, U.K.: Wiley, 2010, pp. 1-20.

[6]. R. Mihalcea and P. Tarau, "TextRank: Bringing order into text," in *Proc. Conf. Empirical Methods in Natural Language Processing (EMNLP)*, Barcelona, Spain, 2004, pp. 404-411.

[7]. M. Grootendorst, "KeyBERT: Minimal keyword extraction with BERT," Zenodo, 2020, doi: 10.5281/zenodo.4461265.

[8]. J. Devlin, M.-W. Chang, K. Lee and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. NAACL-HLT*, Minneapolis, MN, USA, 2019, pp. 4171-4186.

[9]. P. Lopez, "GROBID: Combining automatic bibliographic data recognition and term extraction for scholarship publications," in *Research and Advanced Technology for Digital Libraries (ECDL 2009)*, LNCS vol. 5714, Berlin, Germany: Springer, 2009, pp. 473-474.

[10]. M. A. Hearst, "TextTiling: Segmenting text into multi-paragraph subtopic passages," *Computational Linguistics*, vol. 23, no. 1, pp. 33-64, 1997.

[11]. C. D. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. J. Bethard and D. McClosky, "The Stanford CoreNLP natural language processing toolkit," in *Proc. 52nd Annual Meeting of the ACL: System Demonstrations*, Baltimore, MD, USA, 2014, pp. 55-60.

[12]. I. Beltagy, K. Lo and A. Cohan, "SciBERT: A pretrained language model for scientific text," in *Proc. EMNLP-IJCNLP*, Hong Kong, China, 2019, pp. 3615-3620.

[13]. A. Cohan, F. Dernoncourt, D. S. Kim, T. Bui, S. Kim, W. Chang and N. Goharian, "A discourse-aware attention model for abstractive summarization of long documents," in *Proc. NAACL-HLT (Short Papers)*, New Orleans, LA, USA, 2018, pp. 615-621.

[14]. A. A. Hagberg, D. A. Schult and P. J. Swart, "Exploring network structure, dynamics, and function using NetworkX," in *Proc. 7th Python in Science Conf. (SciPy)*, Pasadena, CA, USA, 2008, pp. 11-15.

 This work is licensed under a Creative Commons Attribution 4.0 International License

[15]. B. Shneiderman, "The eyes have it: A task by data type taxonomy for information visualizations," in *Proc. IEEE Symp. Visual Languages*, Boulder, CO, USA, 1996, pp. 336-343.

[16]. M. Bostock, V. Ogievetsky and J. Heer, "D3: Data-driven documents," *IEEE Trans. Visualization and Computer Graphics*, vol. 17, no. 12, pp. 2301-2309, 2011.