Hindawi
Applied Computational Intelligence and Soft Computing
Volume 2024, Article ID 6671359, 17 pages
https://doi.org/10.1155/2024/6671359

Research Article

Parsing of Research Documents into XML Using Formal Grammars

Opeoluwa Iwashokun[1] **and Abejide Ade-Ibijola**[2]

[1]*Department of Applied Information Systems, College of Business and Economics, University of Johannesburg, Johannesburg, South Africa*
[2]*Research Group on Data, Artificial Intelligence, and Innovations for Digital Transformation, Johannesburg Business School, University of Johannesburg, Johannesburg, South Africa*

Correspondence should be addressed to Opeoluwa Iwashokun; opsygirl@gmail.com

Received 13 July 2023; Revised 30 November 2023; Accepted 6 January 2024; Published 31 January 2024

Academic Editor: Jyotismita Chaki

Automatic information extraction of content and style format in paged documents is challenging. It requires the conversion of the original document into a granular level of details for which every document section and content is identifiable. This functionality or tool does not exist for any academic research document yet. In this paper, we present an automated process of parsing research paper documents into XML files using a formal method approach of context-free grammars (CFGs) and regular expressions (REGEXs) definable of a standard template. We created a tool for the algorithms to parse these documents into tree-like structures organized as XML files named research_XML (RX) parser. The RX tool performed the extraction of syntactic structure and semantic information of the document's contents into XML files. These XML output files are lightweight, analyzable, query-able, and web interoperable. The RX tool has a success rate of 91% when evaluated on fifty varying research documents of 160 average pages and 8,004 total pages. The tool and test data are accessible on GitHub repo. The novelty of our process is specific to applying formal techniques for information extraction in structured multipaged documents and academic research documents thus advancing the research in automatic information extraction.

1. Introduction

Academic supervisors perform the duties of revising research documents with a writing guide (comparable to a set of rules) as part of their routine, a time-consuming review process that can be assisted with an intelligent tool. Automatic processing of documents requires discovery and recognition of the sequence and order of its content to the desired level of meaningful granularity [1–3]. This order referred to as the metastructure of a document can be described by a logical tree with nodes representing the sections of the documents [4–6]. The whole document converted into a tree-structured XML file makes an easily analyzable and queryable document for automatic revision. A semi-structured file format, such as XML, is both human comprehensible and machine readable containing uniquely identifiable tags that encode text in a tree-like structure providing rich content information on both logical and physical structures [5–7]. The insights of how headings are ordered on a page, pages ordered to form chapters, chapters ordered in a document, and other forms of possible subunits of the structural metadata of a document can be well captured with a tree-like structure such as an XML format.

Artificial Intelligence (AI) has been increasingly promoting research in applied areas of fourth industrial revolution (4IR) in education [8, 9]. This has led to improving teaching methods and innovative tools in education. Could AI tools be relevant in postgraduate supervision? Certainly, tools are important in postgraduate supervision for timely feedback in research studies and increased productivity rather than a traditional process [8–12]. The revision of written research documents is assisted by autocorrecting tools such as Grammarly; however, the whole research paper piece is not considered for structure-aware corrections. We propose an automated parsing of these research paper documents into XML files to facilitate structure-aware corrections of such documents since XML file format is structure defined. Standard documents can be generated automatically and reverse-engineered using custom parsers [13]. AI has been applied for efficient processing of various types of standard documents, such as business invoices [14], software requirements documents [15], financial reports [16], legal documents [17, 18], scientific documents [19, 20], and medical documents [21, 22]. We developed a tool named RX for automatic parsing of research proposals into XML as discussed in this paper.

Many attempts have been made to analyse different document syntax which defines different types of document structure. Previous research has approaches of structure information extraction from text documents using (1) machine and deep learning [23–26], (2) semantic network analysis [27], (3) syntactic pattern analysis [28], and (4) formal methods [6]. Conceptually, formal study techniques are still of interest for document understanding in modern NLP techniques [29]. There are existing formal-based information extraction techniques for encoding document's structure metadata as XML file format [6, 30, 31] or javascript object notation (JSON) file format [19]. In education pedagogy, we observed document analysis of scientific literature [6], scholarly articles [32], assignments [33], academic papers [34], and research proposals [35]. Document analysis and understanding of large documents is a difficult task [27, 32], and there is no agreeable best practice yet. The main difficulty lies in the understanding of the granular details and as a collective piece of the whole document. Previous research has analysed and extracted information from given sections of a research proposal document, that is the header and reference section [35]. Hence, the motivations to do this work are as follows:

(i) Motivation 1: we have not found a full-text information extraction of a research proposal document

(ii) Motivation 2: content and structure information contained in different sections of many students' research proposal submissions can be made available in a semistructured XML format, thus easily accessible for further processing

Underpinned on the formal language theory framework, which uses formal notation for representing language constructs, we have designed a generalized context-free grammar (CFG) that defines an entire research paper structure. We also designed spatial and feature characteristics rules that describes further the structure of the research proposal. The detailed structure of a research proposal document will be parsed using these rules into an abstract syntax tree (AST) represented as an XML file with tags representing the detailed structural formation. Figure 1 describes the end-to-end process of parsing the research proposal into XML. The approach is anchored on the knowledge that linguistic knowledge can be easily computed into an XML annotated document and vice-versa [31]. The conversion process of a research paper document (unstructured) to an XML document (semistructured) file format makes it easier to manipulate the file in an automatic revision tool that will be made available for postgraduate supervision. The CFG that defines the structure of varied research proposal documents and its usefulness for vetting many research proposals has been detailed in [36]. This paper adds to knowledge by the following points:

(1) Extracting syntactic and semantic information contained in different sections of many students' research proposals

(2) Develop algorithms for the automatic conversion of research documents into a tree-structured XML file at the desired level of granularity making it easier to manipulate the file for automatic extraction, summarization, or revision

(3) Advancing NLP research in information extraction techniques of documents using a formal language approach to extract structural metadata and construct abstract syntax tree of research documents

The rest of this paper is organized as follows. Section 2 contains background details of the formal language theory framework used in this paper. The details of related works of similar extraction techniques can be seen in Section 3. Details of our design and evaluation are in Sections 4 and 5, while Section 6 concludes the paper.

2. Information Extraction from Large Documents

Research documents contain important research explorations following a given order. An entire document might be divided into sections that must include *Introduction* and *Conclusion* for all academic disciplines and valid structure. Statistical-based NLP techniques or trained models are not fit for academic research documents since they contain domain-specific jargon, and writing style may vary widely [37]. In addition, the machine learning approach for working with text in documents requires a corpus of labeled data that is difficult and expensive for widespread forms of document structure or content [2, 37, 38]. The technique of information extraction from large documents is often preprocessed by building the document's representation from its constituent units, such as sentences or other symbolic tokens, which can be aggregated [38]. This approach is helpful in maintaining the overall context and storing the content in a readable and searchable format for ease of extracting, recognizing, or manipulating parts or whole documents. This process of encoding documents along with their rich structural dependencies benefits from formal techniques than machine learning [2, 37–39].

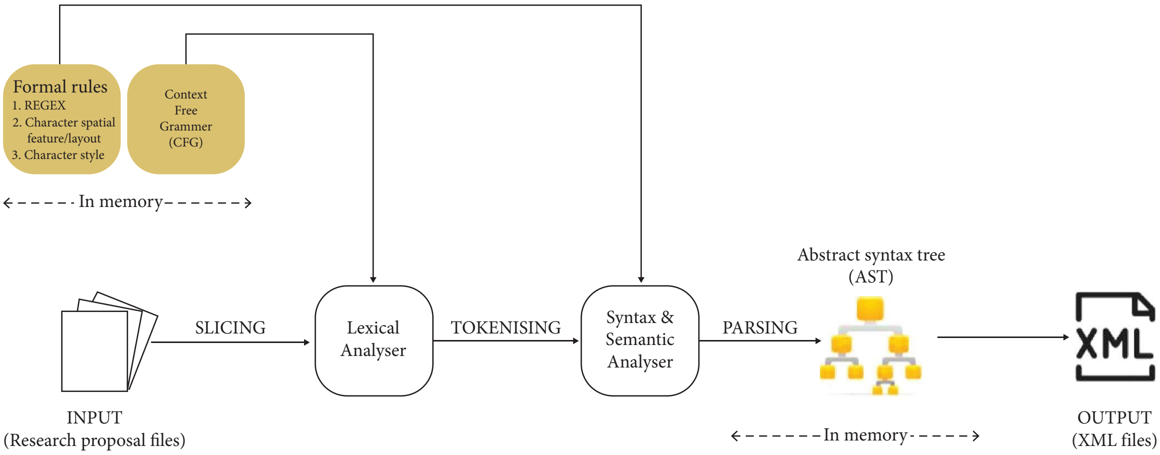

Formal rules
1. REGEX
2. Character spatial feature/layout
3. Character style
Context Free Grammer (CFG)
In memory
SLICING
Lexical Analyser
TOKENISING
Syntax & Semantic Analyser
PARSING
Abstract syntax tree (AST)
XML
INPUT (Research proposal files)
In memory
OUTPUT (XML files)

Figure 1: Workflow of parsing research documents into XML.

2.1. Formal Language Theory in NLP. Key areas of NLP have its fundamentals in the theory of formal languages which provides a systematic terminology and a specific set of rules describing well-formed structures of the language grammar [29]. The theory, postulated by Noam Chomsky in the 1950s and established in modern form in midnineties [40], describes a string as a finite sequence or length of tokens from finite alphabet Σ and an uncountable number of strings can be formed over Σ, i.e., Σ^* over Σ and a grammar as a computational system (automaton) for a language. Simply put, grammar is constrained by some specified rules that describe the structure of an infinite language and infinite string. This explanation by linguists and scientists to describe the formal relationship between the connecting words of a language using patterns and rule-based formal grammars has been applied in modern research to computer programs, music, and visual patterns [41]. Furthermore, Chomsky [40] described the complexities of grammar in a hierarchy of four classes of grammar, namely, unrestricted (type 0), context-sensitive (type 1), context-free (type 2), and regular (type 3) grammar. CFG is easier to deal with, computationally tractable, and able to handle complexities in languages better than regular [42].

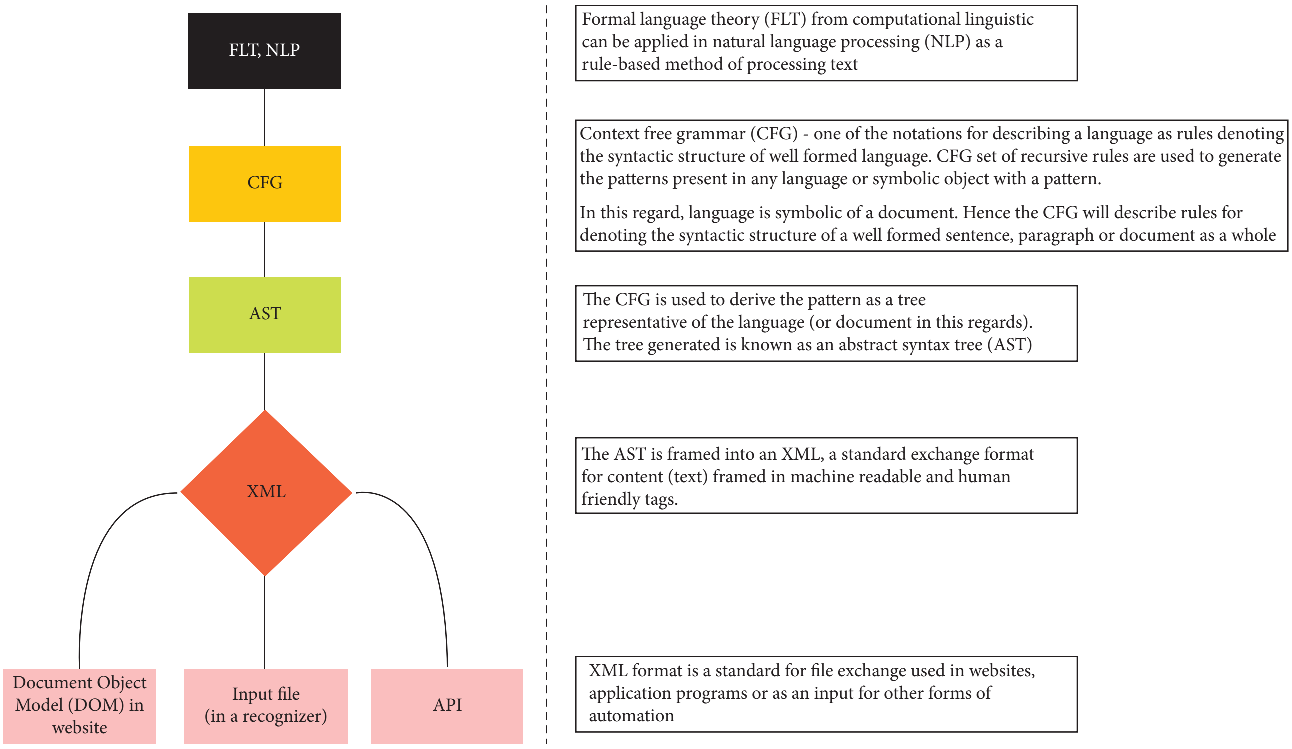
2.2. Context-Free Grammars and Parsers. A context-free grammar (CFG) is a formal system used to describe the syntax of a language represented as a 4-tuple [43, 44] given as $G = (N, \Sigma, P, S)$, where N and Σ are both set of nonterminals and terminals, respectively, which are disjoint finite sets. Elements of P are productions or predefined grammar rules while S is the start symbol. $S \in N$ and P is a finite set of formulas of the form $A \longrightarrow \alpha$, where $A \in N$ and $\alpha \in (N \cup \Sigma)^*$. The set of Σ contains terminal symbols, while elements of N are nonterminal symbols or variables. CFGs are efficient in defining languages recognizable by a parser. A parser checks whether a language (sentence) can be derived using the production rules (P) of a CFG to describe or derive its underlying strings [44]. We have applied the same theoretical understanding to parse the structure of research documents and create a recursive (tree) representation of the parsed documents into an XML file. See Figure 2 for the theoretical approach.

2.3. Parsing. The parsing process of analysing the constituent of a language string for syntactic relations has been used in a programming language to find the relationship between the tokens in a statement, routine, block, declarations, or procedures of a program [45]. At sentence level, parsing uses CFGs to derive the type and span of words in a sentence, and relations between the words are described as the syntactic structure of the sentence [4, 46]. For document analysis and information extraction, a parsing technique can be used to combine tokens according to the production rules defined in the procedure grammar to form a meaningful data structure of a document called an abstract syntax tree (AST), which can be formatted as a markup language, JSON text format, or any form of structured file format [4].

3. Related Work

Information extraction from pdf document sources has gained a lot of research interest with many varied research outputs and purposes [19]. We present related works of two broad categories of information extraction from pdf sources.

(i) Rule-based approach: it is based on predefined rules of known standards or document templates. This primitive way of analysing text has many applied research interests. The constituent details of documents in a rule-based approach are identified using text font style features and in combination with rules to extract the syntactic relationships between text and collection of texts. The approach has been used for computer programs, music, and visual patterns [41]. Previous research has applied rule-based in document types, such as invoices [14], legal documents [47], scientific documents [48], and CVs [49]. Abbott and Ade-Ibijola [21] used formal grammar to build complete syntactic and semantic analyses of sentences of clinical notes. ChemDataExtractor toolkit [50] extracts chemistry-related information from documents. Another existing literature created a tool named requirements analysis tool (RAT) which leverages


FLT, NLP
CFG
AST
XML
Document Object Model (DOM) in website
Input file (in a recognizer)
API

Formal language theory (FLT) from computational linguistic can be applied in natural language processing (NLP) as a rule-based method of processing text

Context free grammar (CFG) - one of the notations for describing a language as rules denoting the syntactic structure of well formed language. CFG set of recursive rules are used to generate the patterns present in any language or symbolic object with a pattern.

In this regard, language is symbolic of a document. Hence the CFG will describe rules for denoting the syntactic structure of a well formed sentence, paragraph or document as a whole

The CFG is used to derive the pattern as a tree representative of the language (or document in this regards). The tree generated is known as an abstract syntax tree (AST)

The AST is framed into an XML, a standard exchange format for content (text) framed in machine readable and human friendly tags.

XML format is a standard for file exchange used in websites, application programs or as an input for other forms of automation

Figure 2: Theoretical understanding of parsing research documents into XML.

the syntactic structure of requirement statements to extract the problems in a software requirements document [15].

(ii) Machine-based approach: recent techniques in big data and text analytics foster information extraction in this approach progressively. We refer to all other approaches different from formal or rule-based methods using any form of machine learning (ML), deep learning (DL), or large language model (LLM). It often requires the availability of all possible occurrences of word tokens as labeled data (i.e., annotated corpus) to produce its own training rules and build its own knowledge. Cermine [25] and GROBID [26] tools were designed for automatic information extraction based on support vector machines (SVMs) and conditional random fields (CRFs) machine learning techniques. Lee et al. [51] performed named entity recognition methods of information extraction of documents using the SVM machine learning model. Also, deep learning methods of convolutional neural network (CNN), long short-term memory (LSTM), and transformer deep learning models have been used for text processing and information extraction. Ji et al. [17] uses bidirectional LSTM to extract evidence information from court record documents (CRDs). CNN has often been used for information extraction from scanned or image-like documents [16, 52, 53]. Recently, there have been LLMs trained on vast text data, which use deep learning algorithms to understand the patterns and structures within that data. LLMs go beyond the scope of this article to generate new (short or long) relevant and grammatically correct text. LLMs, such as bidirectional encoder representations from transformers (BERTs) and generative pretrained transformers (GPTs), have been the toast to many since their release for the purpose of information generation or generative AI. These various approaches are more recently designed and have gained a lot of predominance with the leverage of big data. However, the rule-based approach often becomes more relevant in a case of not enough or hard-to-get labeled data or corpus.

We summarized an overview of information extraction from different document types and varied approaches that have been used in previous research work in Table 1.

3.1. Information Extraction as Text Generation.

AI-generated content (AIGC) engages models trained on vast text or document chunks providing content closely related to the prompts supplied by its user [58, 59]. Conversational LLMs, such as ChatGPT, extract outputs of related contents of patterns and structures similar to prompts given by a user in a process known as prompt engineering [60]. ChatGPT created by OpenAI has generated a high similarity index of text and documents as natural language responses to questions and statements asked by its users [59–61]. The use of ChatGPT to generate fake content is debatable but its applied use for educational materials cannot be overemphasized.

Table 1: Literature of information extraction from various types of documents.

S/N	Document type	Technique	Approach	Authors
1	Invoices	(i) Bidirectional LSTM deep neural network and trained data extracted end-to-end from invoice	Machine-based	[2]
		(ii) Named entity recognition using BERT (bidirectional encoder representations from transformers)	Machine-based	[54]
		(iii) Optical character recognition and graph convolution network from invoice images	Machine-based	[53]
2	Financial reports	(i) Detection of key performance indicators (KPI) from a report using the density of alpha-numeric characters in a rule-based fashion	Rule-based	[16]
3	Medical clinical notes	Parse meaningful critical values from clinical notes and perform a semantic lookup	Rule-based	[21, 55]
4	Legal documents: (i) Court record docs (CRDs)	(i)Bidirectional LSTM for training and extracting information	Machine-based	[17]
	(ii) Compliance documents	(ii) Context-free grammar for complex rule interpretation	Rule-based	[56]
5	Software requirements documents	Syntactic and semantic analysis approach to align with standard writing best practices	Rule-based	[15]
6	CVs	Rule-based text extraction from CV	Rule-based	[49, 57]
7	Academia: literature research	Optical character recognition and graph convolution network from invoice images	Machine-based	[19, 20]

3.2. Information Extraction as Text Comprehension. An automated understanding of textual knowledge is a comprehension task extracting the syntactic and semantic knowledge in a document [15, 46]. It is also a pipeline for AIGC as discussed earlier. Text comprehension is an easy task for humans but time-consuming to rely solely on human manual extraction when there are lots of documents. Arguably, AIGC generalizes information extracted from lots and lots of content using deep learning models. However, formal language and hybrid techniques have proven to have higher precision for extracting detailed insights of text than deep models and are of important use when data are hard to find or not enough.

3.3. The Gap. Information extraction from scientific documents has attracted increasing attention from the NLP community. Oftentimes, only certain parts of the document have been the target for information extraction, e.g., extraction of citations from literature, abstracts from publications, or scientific jargon from related publications. Our work identifies the need for automatic manipulation of all the constituent parts of a research paper (large) document which has not been done in any existing literature.

4. Design

The architecture of the RX tool for parsing research proposals into XML files is described in Figure 1. It is made up of the following three key processes: (1) slicing: which takes the research proposal input document and CFG definition of a valid proposal structure, (2) tokenisation: which takes the disjoint slices and formal rule definitions of elements of a research proposal, and (3) parsing: creates the abstract parse tree in memory of the research proposal document. The system receives as input the research proposal document and set of rules (production rules of the CFG and other formal rules) and then outputs the XML document file. In this section, we describe the CFG for defining a valid proposal structure in Section 4.1. The CFG is used during the slicing process. Section 4.2 explains the tokenising process. The description of the formal rules that are necessary for parsing the document tokens is contained in Section 4.3 while the description of the parsing process is given in Section 4.4. The general algorithm is given in Section 4.5.

4.1. Context-Free Grammar (CFG). A CFG is a four-tuple type of formal grammar. It is used in this design to generate all conceivable formations of the structure of a research proposal and is represented as G. We define it as follows:

$$G = (N, \Sigma, P, S), \tag{1}$$

where

Definition 1. Nonterminal symbol (N): it is the set of all nonterminal symbols. In our design, we define a nonterminal as a unique section in a proposal document. These sections of the proposal document are as follows: preliminary_parts (P_p), chapter_parts (P_c), appendix_parts (P_a), and references_parts (P_r). Equation (2) is as follows:

$$N = \{P_p, P_c, P_a, P_r\}. \tag{2}$$

Definition 2. Set of terminals (Σ): it is a set of all terminal symbols. In our design of CFG, we represent the pages as terminals of the document with each terminal resenting unique page types in a proposal organized often as preliminary pages or chapters in a proposal document. The set is given in the following equation:

$$\Sigma = \{t_p, d_p, a_p, c_p, l_f, l_t, a_{\mathrm{bp}}, c_1, c_2, \cdots, c_q, p_a, p_r\}, \tag{3}$$

where t_p, d_p, a_p, c_p, l_f, l_t, and a_{bp} are terminal symbols for the title, declaration, table_of_contents, list_of_figures, list_of_tables, and abstract_pages, respectively, and found in the preliminary_parts of a proposal document. Also, c_1, c_2, $\cdots$, and c_q are terminal symbols for chapters of the proposal which may represent the introduction_chapter, literature_review_chapter, methodology_chapter,. . ., and conclusion_chapter pages, respectively. We may not have more than ten chapters in a research proposal document. Lastly, p_a and p_r are the pages of the appendix_part and pages of the reference_part of a proposal document, respectively.

Definition 3. Productions (P): these are derivatives or predefined rules (often recursive) that produce the layout structure metadata of any input research proposal. In our design, we define these rules as derivatives of the terminal symbols string that corresponds to any input proposal structure in the algorithm. For instance, a valid string structure of this CFG given as $(t_p \cdot d_p \cdot c_p \cdot a_{bp} \cdot c_1 \cdot c_2 \cdots c_5 \cdot p_r)$ is a valid proposal structure containing a title_page, declaration, table of contents, abstract, introduction_chapter (c_1), literature_review_chapter (c_2), methodology_chapter (c_3), work_plan_chapter (c_4), conclusion_chapter (c_5), and reference_pages (p_r). The rules are expressed in productions 4–8.

$$S \longrightarrow P_p P_c P_a P_r, \tag{4}$$

$$P_p \longrightarrow t_p (d_p | \lambda)(a_p | \lambda) c_p (l_f | \lambda)(l_t | \lambda) a_{\mathrm{bp}}, \tag{5}$$

$$P_c \longrightarrow c_1 \cdot (c_2 | \lambda) \cdot (c_3 | \lambda) \cdots (c_{n-1} | \lambda) \cdot c_n, \tag{6}$$

$$P_a \longrightarrow p_a | p_a P_a | \lambda, \tag{7}$$

$$P_r \longrightarrow p_r. \tag{8}$$

Definition 4. Start symbol (S): it is a nonterminal symbol representing the entire language. In our design, it is the input research proposal document to be parsed.

4.2. Tokenising. We have described a four-tuple CFG of an input proposal document S. The document is abstracted into symbolic representations of its constituent parts and structure. This knowledge abstraction of the research paper's structural metadata is obtained through a tokenising process. The input proposal document is first sliced into "lines of texts" which are programmatically tokenised as meaningful text strings described as document tokens.

Definition 5. Set of document token: it is a closed set of meaningful text chunks identifiable as symbolic elements in a proposal, represented in our design as t, given in the following equation:

$$t = \left\{t_t, t_a, t_s, t_{\mathrm{tp}}, t_d, t_{\mathrm{st}}, t_p, t_n, t_f, t_{\mathrm{ta}}, t_a, t_r\right\}, \tag{9}$$

where t_t, t_a, t_s, t_{tp}, t_d, t_{st}, t_p, t_n, t_f, t_{ta}, t_a, and t_r are text strings symbols for proposal_title, author, supervisor, title_paragraph, proposal_date, section_title, paragraphs, page_number, figure_label, table_label, appendix_label, and reference, respectively. These meaningful text chunks are a group of characters and words that may also include numbers and or special characters.

The novelty of our work is the granularised tokenization of any research proposal (a scholarly large document) into meaningful parts of all the document tokens contained in all the pages of the proposal document. These are the inputs for the parser to produce the document's parse tree, see Figure 3. An example of the description of document tokens that may be contained in the title page (t_p), declaration page (d_p), acknowledgment page (a_p), content page (c_p), list of figures page (l_f), list of tables page (l_t), abstract page (a_{bp}), chapter page (c_q), appendix page (p_a), and reference page (p_r), respectively, is contained in equations (10)–(19).

$$t_p = t_t t_a t_s t_{\mathrm{tp}}^{(1,2)} t_d, \tag{10}$$

$$d_p = t_{\mathrm{st}} t_p^{+} t_n, \tag{11}$$

$$a_p = t_{\mathrm{st}} t_p^{+} t_n, \tag{12}$$

$$c_p = t_{\mathrm{st}} t_{ta} t_n, \tag{13}$$

$$l_f = t_{\mathrm{st}} \left(t_i | t_{ta}\right) t_n, \tag{14}$$

$$l_t = t_{\mathrm{st}} \left(t_i | t_{ta}\right) t_n, \tag{15}$$

$$a_{\mathrm{bp}} = t_{\mathrm{st}} t_p^{+} t_n, \tag{16}$$

$$\begin{aligned} c_q = {} & t_{\mathrm{st}}\left(\left(t_p | t_{\mathrm{sst}} | t_f | t_i | t_{\mathrm{ta}}\right)^{+} t_n\right)^{+} \\ & + \text{where } 1 \le q \le 10, \end{aligned} \tag{17}$$

$$p_a = t_{\mathrm{st}} t_f^{+} t_n, \tag{18}$$

$$p_r = t_{\mathrm{st}} t_r^{+} t_n. \tag{19}$$

According to equations (10)–(19), in the instance of a research proposal document, the title page (t_p) will contain a title, author, supervisor, two short paragraphs, and a date. Hence, the equivalent of a title page is given by the equation $t_p = t_t t_a t_s t_{\mathrm{tp}}^{(1,2)} t_d$. The document tokens (t) are equivalent to their respective constituent text and not a derivative. Our algorithm design extracts these document tokens of meaningful chunks of words or texts, belonging to a closed set t from lines of text for any input research proposal document S.

4.3. Formal Rules of REGEX, Style, and Spatial Features.

These sets of rules are implemented as recognisers. A REGEX recognizer will be used as a language acceptor algorithm that validates the element token based on the matching regular expression. For instance, the author's name (t_a) text string recognition on the title page of an input proposal can be matched by a regular expression given as $(By{:}\ |By)?(\setminus \mathrm{s} \setminus \mathrm{w})^{+}$. A valid text string example for an author's name recognition in a proposal document is "By: Thakiso Ogero" extracted from the proposal in Figure 4. The text position or spatial feature in the document, as well as text font characteristics, is also implemented as recognisers for matching text string symbols. A set of styles or text line features of font boldness, font size, top line spacing, bottom line spacing, left alignment space, and right alignment space is given as a set $\{y_1, y_2, \cdots y_n\}$.

4.4. Parsing.

The parser defines the syntax (i.e., arrangements) and the semantics (i.e., meaningful relationship) between all the document tokens in the research proposal. Figure 5 describes in a simple diagram how this formation that must be recognisable as a meaningful or valid abstract research proposal is formed. The content of the research proposal document is preserved all through the process and becomes the text content in between the terminal nodes or tags of the abstract syntax tree (AST). The annotations in parse tree nodes make it possible for its conversion into XML having tags corresponding to nodes.

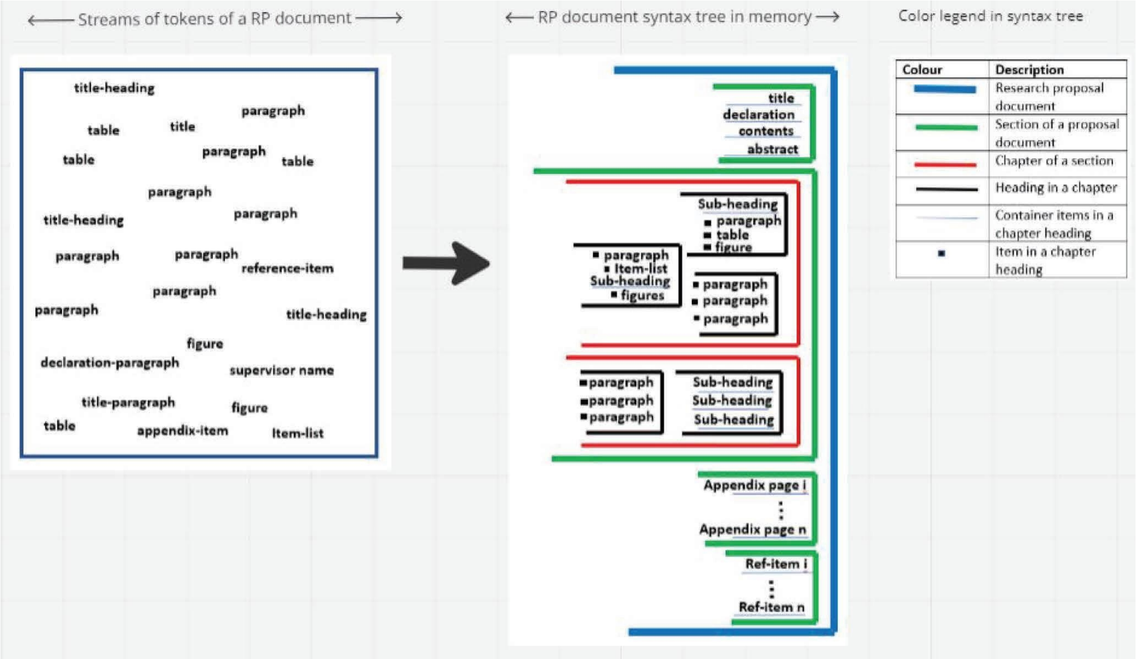
4.5. Algorithm.

The algorithm (see Algorithm 1) takes the research proposal document input and outputs an XML document. The algorithm steps are given as follows.

(i) Step 1: (Slicing) a function defined by ITextSharp software library is used for slicing the input document into "lines of text." The function takes as parameter the input document and a set of document predefined styles, e.g., font_bold, font_size, top_space, bottom_space, left_align, and right_align. See Line 8 in Algorithm 1.

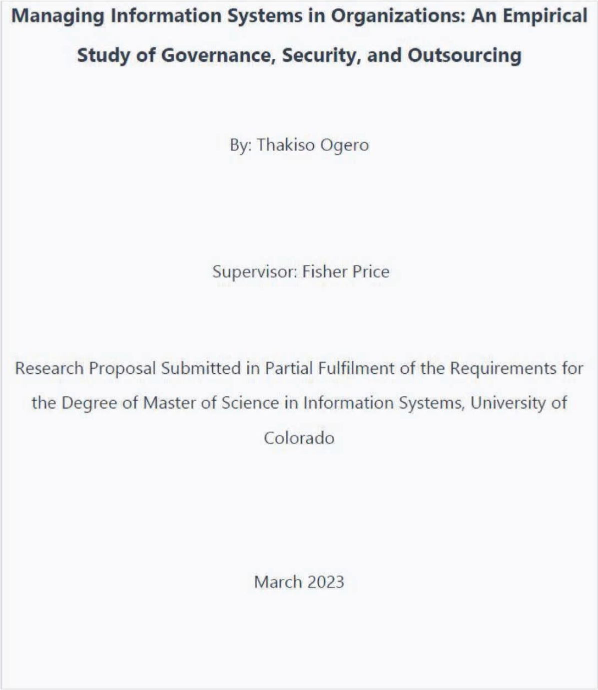
(ii) Step 2: (Tokenising) "lines of text" are matched with the corresponding REGEX of text string symbol. The "lines of text" are identified as any of text string symbols of title (t_t), author (t_a), supervisor name (t_s), chapter paragraphs ((t_p), and so on. See Line 11 in Algorithm 1.

(iii) Step 3: (Parsing) valid text string symbolic tokens are organized into a nested array using production rules defined in P. See Lines 13 and 14 in Algorithm 1.



Colour	Description
🟦	Research proposal document
🟩	Section of a proposal document
🟥	Chapter of a section
━	Heading in a chapter
─	Container items in a chapter heading
▪	Item in a chapter heading

Figure 3: Research proposal abstract syntax tree (AST) generation.


Managing Information Systems in Organizations: An Empirical Study of Governance, Security, and Outsourcing

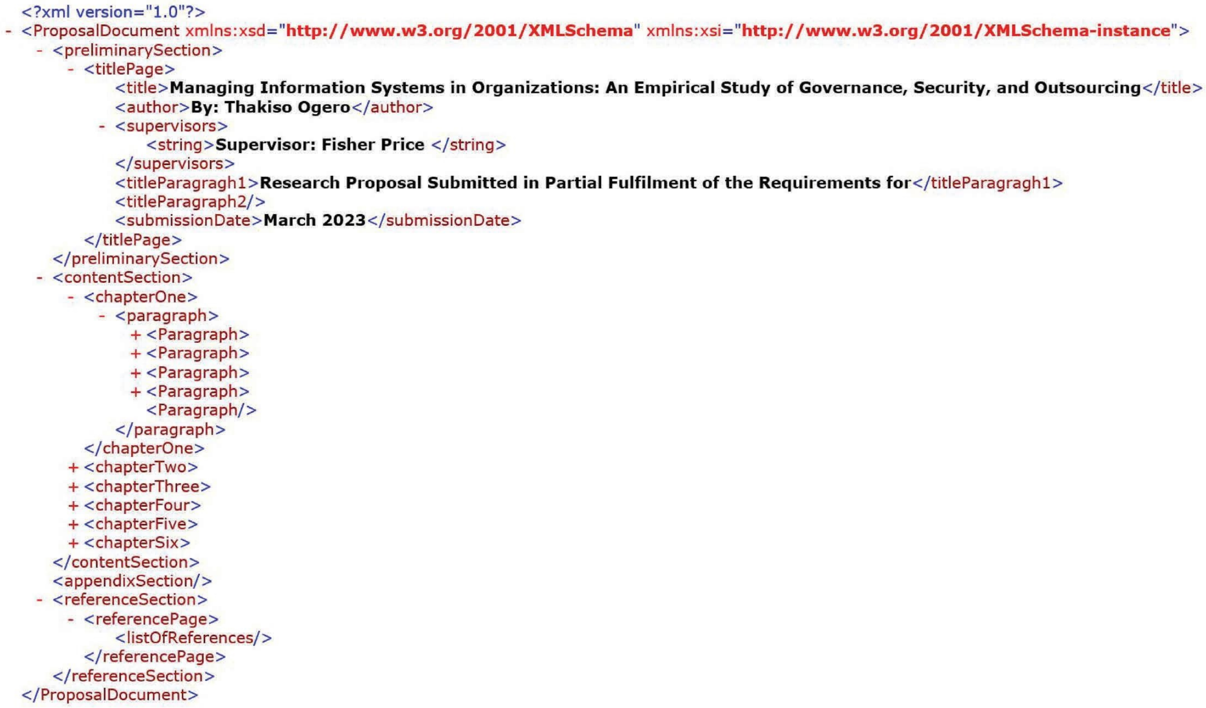
By: Thakiso Ogero

Supervisor: Fisher Price

Research Proposal Submitted in Partial Fulfilment of the Requirements for the Degree of Master of Science in Information Systems, University of Colorado

March 2023

Figure 4: First page of a sample ChatGPT-generated research proposal.

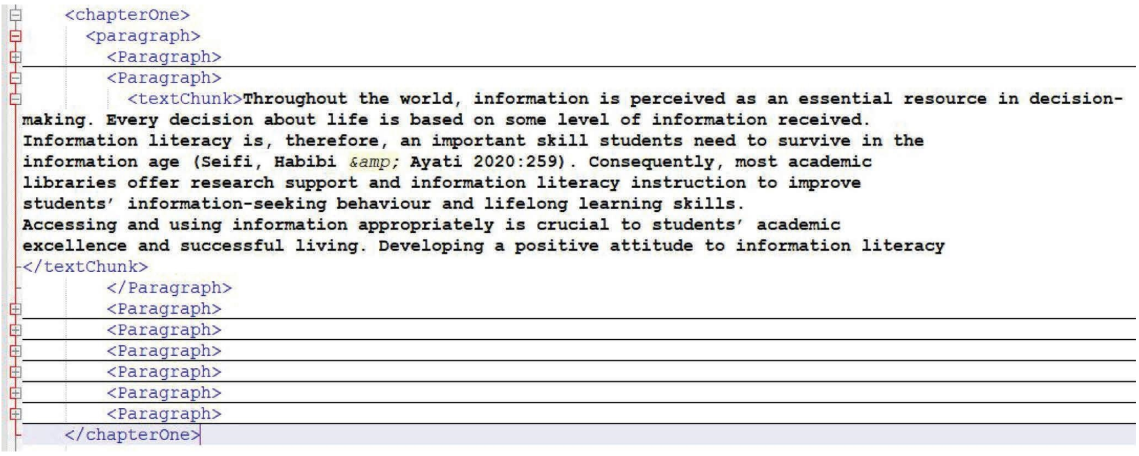


```
  <?xml version="1.0"?>
- <ProposalDocument xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  - <preliminarySection>
    - <titlePage>
        <title>Managing Information Systems in Organizations: An Empirical Study of Governance, Security, and Outsourcing</title>
        <author>By: Thakiso Ogero</author>
      - <supervisors>
          <string>Supervisor: Fisher Price </string>
        </supervisors>
        <titleParagragh1>Research Proposal Submitted in Partial Fulfilment of the Requirements for</titleParagragh1>
        <titleParagraph2/>
        <submissionDate>March 2023</submissionDate>
      </titlePage>
    </preliminarySection>
  - <contentSection>
    - <chapterOne>
      - <paragraph>
        + <Paragraph>
        + <Paragraph>
        + <Paragraph>
        + <Paragraph>
          <Paragraph/>
        </paragraph>
      </chapterOne>
    + <chapterTwo>
    + <chapterThree>
    + <chapterFour>
    + <chapterFive>
    + <chapterSix>
    </contentSection>
    <appendixSection/>
  - <referenceSection>
    - <referencePage>
        <listOfReferences/>
      </referencePage>
    </referenceSection>
  </ProposalDocument>
```

Figure 5: Generated XML object notation of sample proposal from the RX tool.

```
(1) **Input:** PDF proposal document, $S$
        **Output:** XML document, $X$
        $Y$ = set of document styles, $\{y_1, y_2, y_3, \dots y_n\}$
        $T$ = set of predefined text strings, $\{t_t, t_a, t_s, t_{tp}, t_{st}, t_{sst}, t_p, t_n, t_d, t_r, t_f, t_i, t_{ta}\}$
        $Q$ = empty nested array/set as empty Parse tree object, [[]]
        $P \rightarrow \{P_p, P_c, P_a, P_r\}$, / * set of Productions * /
(2) $L$ = PDF Library.Extract_textLines_frm_Pdf Doc (SY)
    / * Extract text string tokens from $L$ into nested array object * /
    **for** pair $(l, t) \in$ pair $(L, T)$ **do**
(3)     **if** $l$ = regex_match $(t)$ **then**
(4)         / * Add extracted text string tokens to nested array object * /
            **if** $l \in$ RHS $(P)$ **then**
(5)             Add $l$ to $Q$
(6)         **end**
(7)     **end**
(8) **end**
(9) $X \rightarrow$ Serialize_Nested Array Object_to_Xml File $(Q)$
        / * Serialize nested array (a parse tree) object into an XML file * /
        **return** XML_Document $(X)$
```

Algorithm 1: Parsing document to XML algorithm.



```
<chapterOne>
  <paragraph>
    <Paragraph>
    <Paragraph>
      <textChunk>Throughout the world, information is perceived as an essential resource in decision-
making. Every decision about life is based on some level of information received.
Information literacy is, therefore, an important skill students need to survive in the
information age (Seifi, Habibi &amp; Ayati 2020:259). Consequently, most academic
libraries offer research support and information literacy instruction to improve
students' information-seeking behaviour and lifelong learning skills.
Accessing and using information appropriately is crucial to students' academic
excellence and successful living. Developing a positive attitude to information literacy
</textChunk>
    </Paragraph>
    <Paragraph>
    <Paragraph>
    <Paragraph>
    <Paragraph>
    <Paragraph>
    <Paragraph>
</chapterOne>
```

Figure 6: Display content of a paragraph tag.

(iv) Step 4: (Serialization) the function Serialize_Nested Array Object_to_XmlFile takes as input the nested array object (Q, representing parse tree of the document in memory) and output as XML file using C# inbuilt serialize function. See Lines 19 in Algorithm 1.

(v) Step 5: return output (XML file). See Line 20 in Algorithm 1.

5. Application and Evaluation

The algorithms for this work were implemented using C# and iTextSharp API into a tool named RX. The source codes and dataset are available in the GitHub repo. The test dataset is a corpus of 50 research proposals with a mean page average of sixty pages. The test data corpus is described in Section 5.1, and a sample demo of a "fake" research proposal result output is described in Section 5.2. The output results of 50 research proposals are discussed in Section 5.3. We conclude the section with the performance evaluation of the tool and compare the performance with an online available alternative in Section 5.4.

5.1. Input Test Dataset. Fifty large research documents of proposal theses in Information Systems and related disciplines available online on institutional repositories were used for testing. These were selected using purposive sampling from Universities across South Africa, Ghana, Nigeria, and India. These documents are 160 pages on average and a total of 8004 pages for all 50 documents. The dataset is publicly accessible, and a copy is available on a GitHub repo. These proposals vary in total number of pages, structure, and content since universities may use slightly differing writing guideline. The varying dataset for testing the tool successfully provides the generalisation capability of our approach.

5.2. XML Result Output Description. For reasons of personal information protection, we generated a sample fake proposal document using ChatGPT to display the sample contents of an output XML file in this article. See Figures 4 and 5 for the first page of the sample proposal document and XML output generated. The XML file in Figure 5 embodies in tags the abstract syntax tree of a research proposal shown as a picture in Figure 3. The XML output gives insights into the structural metadata and content of the original proposal document in a lightweight document format structure.

5.3. Discussion of Results. The resulting XML file output in Section 5.2 embodies the structure and semantics of the input document. The text content of the input research proposal is within the leaf node tags in the XML document. For instance, a "paragraph" XML tag contains text strings of words or sentences, see Figure 6. We have described the results obtained from 50 research documents proposal tests using the RX tool in Table 2. In the table, "1" indicates true when the given tokens in the document represented by the text string symbol is correctly identified in the XML output, "0" is when it is incorrectly identified, and "NG" when the text is missing in the XML output file owing to the text string symbol not present or incorrect format in the original file. According to the results obtained, the tool had a low accuracy recognition of figures/images (t_f) and tables (t_{ta}) contained in the chapter pages (c_n), performed better with recognising tokens on the title page (t_p) and best with recognising tokens on the preliminary pages. The method of evaluation is explained in Section 5.4.1.

5.4. Performance Evaluation. We measured the performance of the tool using "confusion matrix" performance metrics. We also compared results with some alternative tools including "FormX.ai" available freely online here for converting PDF files to XML assessed on 15th November 2023.

5.4.1. Tool Performance. The confusion matrix is a four-quadrant comprising classifications of true positives (TPs), true negatives (TNs), false positives (FPs), and false negatives (FNs). See Figure 7 for the confusion matrix for evaluating this research work. The description of its quadrants for result evaluation of the research proposals vs the XML output of the RX tool is given:

Table 2: Proposal sections and element tokens evaluation.

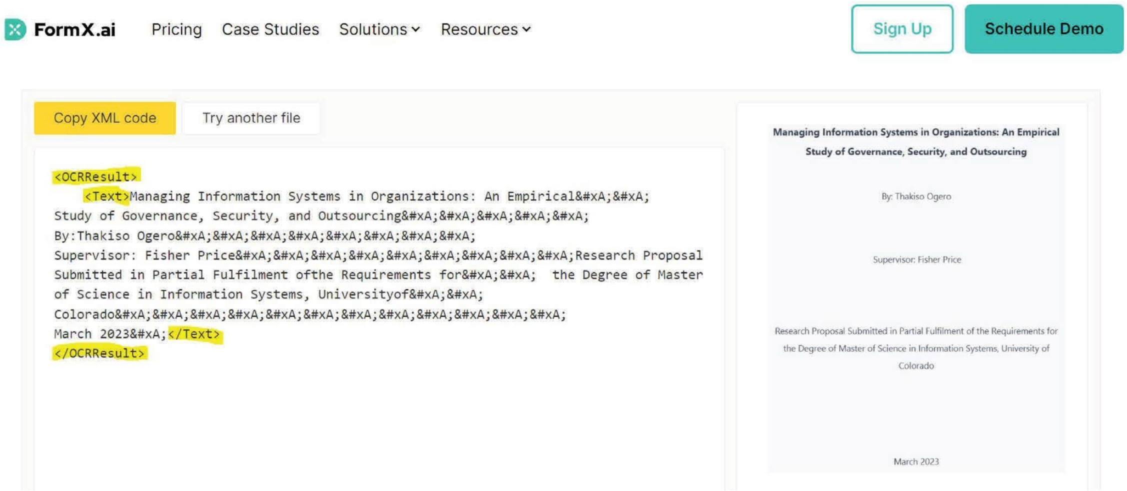
S	t_p	d_p	a_p	c_p	lf	l_t	a_{bp}	$[c_1, c_2 \cdots c_n]$	p_r
X	$[t_t\ t_a\ t_{\text{tp}}\ t_d]$	$[t_{\text{st}}\ t_p\ t_n]$	$[t_{\text{st}}\ t_p\ t_n]$	$[t_{\text{st}}\ t_a\ t_n]$	$[t_{\text{st}}\ t_i\ t_n]$	$[t_{\text{st}}\ t_i\ t_n]$	$[t_{\text{st}}\ t_p\ t_n]$	$[t_{\text{st}}\ t_p\ t_{\text{sst}}\ t_f\ t_i\ t_{\text{ta}}\ t_n]$	$[t_{\text{st}}\ t_f\ t_n]$
P1	[1 1 1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1 0 0 1 1]	[1 1 1]
P2	[0 1 0 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 0 0 1 0 1]	[1 1 1]
P3	[1 1 ng 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[0 1 1]	[ng ng ng ng ng ng ng]	[ng ng ng]
P4	[1 1 ng 1 0]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[ng 1 1]	[ng ng ng ng ng ng ng]	[ng ng ng]
P5	[1 1 1 1 0]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1 0 1 0 1]	[1 1 1]
P6	[1 1 ng 1 0]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[ng ng ng ng ng ng ng]	[ng ng ng]
P7	[1 1 ng 1 0]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1 0 1 0 1]	[1 1 1]
P8	[1 1 ng 0 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1 0 1 0 1]	[1 1 1]
P9	[1 1 ng 1 0]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[ng ng ng ng ng ng ng]	[ng ng ng]
P10	[1 1 ng 1 ng]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1 0 1 0 1]	[1 1 1]
P11	[1 1 0 1 ng]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1 0 1 0 1]	[1 1 1]
P12	[0 1 0 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1 0 1 0 1]	[1 1 1]
P13	[1 1 1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1 0 1 0 1]	[1 1 1]
P14	[1 1 0 0 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1 0 1 0 1]	[1 1 1]
P15	[1 1 ng 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1 0 1 0 1]	[1 1 1]
P16	[1 1 1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1 0 1 0 1]	[1 1 1]
P17	[1 1 1 1 0]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1 0 1 0 1]	[1 1 1]
P18	[1 1 ng 0 ng]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1 0 1 0 1]	[1 1 1]
P19	[1 1 1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1 0 1 0 1]	[1 1 1]
P20	[1 1 0 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1 0 1 0 1]	[1 1 1]
P21	[1 1 1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1 0 1 0 1]	[1 1 1]
P22	[1 1 1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1 0 1 0 1]	[1 1 1]
P23	[1 1 1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1 0 1 0 1]	[1 1 1]
P24	[0 1 1 0 0]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1 0 1 0 1]	[1 1 1]
P25	[1 1 1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1 0 1 0 1]	[1 1 1]
P26	[1 1 0 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1 0 1 0 1]	[1 1 1]
P27	[1 1 0 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1 0 1 0 1]	[1 1 1]
P28	[1 0 1 1 0]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1 0 1 0 1]	[1 1 1]
P29	[1 1 ng 1 ng]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1 0 1 0 1]	[1 1 1]
P30	[1 1 0 0 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1 0 1 0 1]	[1 1 1]
P31	[1 1 0 0 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1 0 1 0 1]	[1 1 1]
P32	[1 0 0 1 0]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[ng ng ng ng ng ng ng]	[ng ng ng]
P33	[1 1 0 1 0]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[ng ng ng ng ng ng ng]	[ng ng ng]
P34	[1 0 0 1 0]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[ng ng ng ng ng ng ng]	[ng ng ng]
P35	[1 1 1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1 0 1 0 1]	[1 1 1]
P36	[1 1 1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1 0 1 0 1]	[1 1 1]
P37	[1 1 0 1 0]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1 0 1 0 1]	[1 1 1]
P38	[1 1 1 1 0]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1 0 1 0 1]	[1 1 1]
P39	[0 1 0 1 0]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1 0 1 0 1]	[1 1 1]
P40	[1 1 0 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1 0 1 0 1]	[1 1 1]
P41	[1 1 0 1 0]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1 0 1 0 1]	[1 1 1]
P42	[1 1 1 1 0]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1]	[1 1 1 0 1 0 1]	[1 1 1]

Table 2: Continued.

S	t_p	d_p	a_p	c_p	lf	l_t	a_{bp}	$[c_1, c_2 \cdots c_n]$	p_r
P43	$[1\ 1\ 1\ 1\ 0]$	$[1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1\ 0\ 1\ 0\ 1]$	$[1\ 1\ 1]$
P44	$[1\ 1\ 1\ 1\ 0]$	$[1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1\ 0\ 1\ 0\ 1]$	$[1\ 1\ 1]$
P45	$[1\ 1\ 1\ 1\ 0]$	$[1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1\ 0\ 1\ 0\ 1]$	$[1\ 1\ 1]$
P46	$[1\ 0\ 1\ 1\ 0]$	$[1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1\ 0\ 1\ 0\ 1]$	$[1\ 1\ 1]$
P47	$[1\ 1\ 1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1\ 0\ 1\ 0\ 1]$	$[1\ 1\ 1]$
P48	$[0\ 0\ 1\ 1\ 0]$	$[1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1\ 0\ 1\ 0\ 1]$	$[1\ 1\ 1]$
P49	$[1\ 1\ 1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1\ 0\ 1\ 0\ 1]$	$[1\ 1\ 1]$
P50	$[1\ 1\ 0\ 1\ 0]$	$[1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1\ 0\ 1\ 0\ 1]$	$[1\ 1\ 1]$
Ave	$[0.90\ 0.90\ 0.67\ 0.88\ 0.56]$	$[1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 1]$	$[1\ 1\ 0.98\ 0.14\ 0.98\ 0.16\ 1]$	$[1\ 1\ 1]$

Research Proposal \ XML Output	XML tag represented? YES	XML tag represented? NO
Document token found? YES	**TP**	**FP**
Document token found? NO	**FN**	**TN**

Figure 7: Confusion matrix.



```
<OCRResult>
    <Text>Managing Information Systems in Organizations: An Empirical&#xA;&#xA;
Study of Governance, Security, and Outsourcing&#xA;&#xA;&#xA;&#xA;&#xA;
By:Thakiso Ogero&#xA;&#xA;&#xA;&#xA;&#xA;&#xA;&#xA;&#xA;
Supervisor: Fisher Price&#xA;&#xA;&#xA;&#xA;&#xA;&#xA;&#xA;&#xA;&#xA;Research Proposal
Submitted in Partial Fulfilment ofthe Requirements for&#xA;&#xA;  the Degree of Master
of Science in Information Systems, Universityof&#xA;&#xA;
Colorado&#xA;&#xA;&#xA;&#xA;&#xA;&#xA;&#xA;&#xA;&#xA;&#xA;&#xA;&#xA;
March 2023&#xA;</Text>
</OCRResult>
```

Managing Information Systems in Organizations: An Empirical Study of Governance, Security, and Outsourcing

By: Thakiso Ogero

Supervisor: Fisher Price

Research Proposal Submitted in Partial Fulfilment of the Requirements for the Degree of Master of Science in Information Systems, University of Colorado

March 2023

Figure 8: Alternative tool (FormX) XML file.

(i) True positives (TPs): the document token is found in the input research proposal, detected by the RX tool and represented in the XML tag. TP has a value of 1 in the evaluation for a given document token.

(ii) False positive (FP): the document token is found in the input research proposal but not detected by the RX tool. FP has a value of 0 in the evaluation for a given document token.

(iii) False negative (FN): the document token is not found in the input research proposal but detected by the RX tool. A situation like this cannot exist; hence, this is not applicable.

(iv) True negative (TN): the document token is not found in the input research proposal and not detected by the RX tool. TN has a value of "not given" (NG) in the evaluation for a given document token.

Let i be the number of elements in the set $\{t_t, t_a, t_s, t_{\text{tp}}, t_d, \cdots\}$ given in Table 2 which denotes the set of all possible text string symbols of a given document. We measured the evaluation for each text string symbol element (t_i) as given in equations (20)–(22). We measured the accuracy of each token recognition as follows:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{Total}}. \tag{20}$$

For example, t_s in Table 2 is calculated as follows:

$$\frac{23 + 10}{50} \approx 0.67. \tag{21}$$

Therefore, the overall measure of accuracy of the tool is given by the percentage average of all the possible token strings evaluation and given as follows:

Table 3: Comparison of RX tool with other alternatives.

Software tool	Successful XML conversion?	Meaningful XML?	Rich XML tree?	XML tag content?	Comments	Link
FormAI	Y	N	N	Y	XML structure has all the contents of the pdf file in a single tag	https://www.formx.ai
i2pdf	Y	N	Y	Y	XML structure was too complex with many verbose tags of the input file metadata	https://www.2pdf.com
Nanonets pdf_to_xml	N	NA	NA	NA	NA	https://www.nanonets.com
Vertopal	Y	N	Y	Y	XML structure was too complex with many verbose tags of the input file metadata	https://www.vertopal.com
Aconvert	Y	N	Y	Y	XML structure was too complex with many verbose tags of the input file metadata	https://www.aconvert.com
RX	Y	Y	Y	Y	The tags are meaningful and has meaningful content, often a text chunk of complete information	github repo

$$\begin{aligned}\text{Average Accuracy} &= \sum_{i=1}^{n} t_i \\ &= \frac{0.90 + 0.90 + 0.67 + \cdots + 0.16 + 1 + 1 + 1 + 1}{33} \approx 0.91.\end{aligned} \tag{22}$$

5.4.2. Comparison with Other Alternatives. The XML structure generated of RX is rich and meaningful compared to other alternatives. See Figure 8 for the XML structure generated by an alternative tool "FormX" of the same input research proposal document. Also, the XML file output given RX can be reverse-engineered easily into the original input document thus allowing for further analysis of many research documents. Other alternative online tools provided an XML file with an unmeaningful structure for large documents while some others returned no result. See Table 3 for a comparison of the results from the tool and an alternative.

6. Conclusion

This paper expresses a formal method approach to information extraction from unstructured documents, specifically a research proposal document. The paper examined various approaches used in the previous research and expresses the most appropriate approach for knowledge-based and format-specific kind of challenges in this work: extraction from large documents research proposals. We have implemented a tool named "RX" for parsing research documents into an abstract syntax tree structure organized into an XML file format using text features, regular expression, and CFGs to determine the hierarchical relationship (i.e., parse) between text string symbols. The XML file format embodies the structure and content of the document having tags representing the nodes of the AST and contents of the research document as text between tags.

The RX tool performs the processes of (1) document slicing, (2) tokenisation, and (3) lexical analysis. RX tool converts research proposals successfully and meaningfully. Its evaluation of fifty research documents of 160 average number of pages and 8004 total pages from different institutional repositories has an overall success rate of 91%. The generated XML files were meaningful, rich, and recognised the different structures in a research proposal with descriptive tag names. Comparatively, the RX tool performed much better than existing tools for converting pdf to XML. This demonstrates the relevance and effectiveness of formal methods approach for information extraction from unstructured texts.

6.1. Limitations. The tool performed well in identifying various sections and constituents of the research paper document but slightly lower in the performance of the title page section. This was due to the varying title page organization of various research documents. The algorithm currently has some limitations in parsing tables and figures as the interpretations of its structure and appearance in the document are not based only on text fonts.

6.2. Future Work. Tables in a document present a unique tabular structure in terms of rows, columns, cells, and other details. For future work, we will explore the parsing of tables and figures in structure and content. The formal technique presented in this paper will be extended to embed these nontextual contents into the final XML file output. We will also improve the recognition and parsing accuracy of each document section, especially the title page and the overall accuracy.

Data Availability

The input files data and C# code used to support the findings of this study have been deposited in the GitHub repository: https://github.com/opsygirl/Document_to_XML_Parser. The repository is publicly available, includes a read me file, and code may be reproduced in a Microsoft Visual Studio environment. The input files and XML output files are in a zip file named Test Proposals and XML files in the repository. The findings of our evaluations are included in the article in Table 2.

Disclosure

The authors, Opeoluwa Iwashokun and Abejide Ade-Ibijola, hereby declare that this article titled "Parsing of Research Documents into XML using Formal Grammars" is a joint effort. We have both agreed to submit this work to the Applied Computational Intelligence and Soft Computing Journal

Conflicts of Interest

The authors declare that they have no conflicts of interest regarding the publication of this paper.

Acknowledgments

This research was partially funded by the National Research Fund (NRF) with Grant no. 119041 and supported by the Global Excellence Stature (GES) awards.

References

[1] S. Maji, A. Appe, R. Bali, A. G. Chowdhury, V. C. Raghavendra, and V. M. Bhandaru, "An interpretable deep learning system for automatically scoring request for proposals," in *Proceedings of the 2021 IEEE 33rd International Conference on Tools with Artificial Intelligence (ICTAI)*, pp. 851–855, IEEE, Washington, DC, USA, November 2021.

[2] R. B. Palm, F. Laws, and O. Winther, "Attend, copy, parse end-to-end information extraction from documents," in *Proceedings of the 2019 International Conference on Document*

Analysis and Recognition (ICDAR), pp. 329–336, Sydney, NSW, Australia, September 2019.

[3] F. Graliński, T. Stanisławek, A. Wróblewska et al., "Kleister: a novel task for information extraction involving long documents with complex layout," 2020, https://arxiv.org/abs/2003.02356.

[4] S. Whitmore, "Procedure parsing: a method for parsing handwritten documents into computer-based procedures," *Human Factors in Software and Systems Engineering*, vol. 61, p. 21, 2022.

[5] G. Zaman, H. Mahdin, K. Hussain, and A.-U. Rahman, "Information extraction from semi and unstructured data sources: a systematic literature review," *ICIC Express Letters*, vol. 14, no. 6, pp. 593–603, 2020.

[6] A. Constantin, S. Pettifer, and A. Voronkov, "Pdfx fully-automated pdf-to-xml conversion of scientific literature," *DOCENG*, vol. 13, pp. 177–180, 2013.

[7] L. Schmidt, F. Shokraneh, K. Steinhausen, and C. E. Adams, "Introducing raptor: revman parsing tool for reviewers," *Systematic Reviews*, vol. 8, no. 1, pp. 151–154, 2019.

[8] C. Kayembe and D. Nel, "Challenges and opportunities for education in the fourth industrial revolution," *African Journal of Public Affairs*, vol. 11, no. 3, pp. 79–94, 2019.

[9] C. W. Okonkwo and A. Ade-Ibijola, "Chatbots applications in education: a systematic review," *Computers & Education: Artificial Intelligence*, vol. 2, Article ID 100033, 2021.

[10] G. Suganya, "A study on challenges before higher education in the emerging fourth industrial revolution," *International Journal of Engineering Technology Science and Research*, vol. 4, 2017.

[11] V. Sanchez-Anguix, R. Chalumuri, J. M. Alberola, and R. Aydogan, "Artificial intelligence tools for academic management: assigning students to academic supervisors," in *Proceedings of the 4th International Technology, Education and Development Conference*, pp. 4638–4644, Valencia, Spain, March 2020.

[12] Y. Xue and Y. Wang, "Artificial intelligence for education and teaching," *Wireless Communications and Mobile Computing*, vol. 2022, Article ID 4750018, 10 pages, 2022.

[13] Y. Sheng, Y. Lu, J. Xie et al., "Template-based structured document classification and extraction," *uS Patent*, vol. 10, p. 158, 2020.

[14] B. P. Rasmus, *End to End Information Extraction from Business Documents*, 2019.

[15] K. Verma, A. Kass, and R. Vasquez, "Using syntactic and semantic analyses to improve the quality of requirements documentation," *Semantic Web*, vol. 5, pp. 405–419, 2014.

[16] E. Brito, R. Sifa, C. Bauckhage, R. Loitz, U. Lohmeier, and C. Pünt, "A hybrid ai tool to extract key performance indicators from financial reports for benchmarking," in *Proceedings of the ACM Symposium on Document Engineering 2019*, New York, NY, USA, September 2019.

[17] D. Ji, P. Tao, H. Fei, and Y. Ren, "An end-to-end joint model for evidence information extraction from court record document," *Information Processing & Management*, vol. 57, no. 6, Article ID 102305, 2020.

[18] S. Kubeka and A. Ade-Ibijola, "Automatic comprehension and summarisation of legal contracts," *Contract*, vol. 9, p. 10, 2021.

[19] M. Zhu and J. M. Cole, "Pdfdataextractor: a tool for reading scientific text and interpreting metadata from the typeset literature in the portable document format," *Journal of Chemical Information and Modeling*, vol. 62, no. 7, pp. 1633–1643, 2022.

[20] Z. Bodó and L. Csató, "A hybrid approach for scholarly information extraction, Studia Univ. Babes-Bolyai," *Inform*, vol. 62, no. 2, pp. 5–16, 2017.

[21] S. Abbott and A. Ade-Ibijola, "Algorithms and a tool for automatic decryption of clinical notes," in *Proceedings of the 2019 6th International Conference on Soft Computing & Machine Intelligence (ISCMI)*, pp. 137–143, Johannesburg, South Africa, November 2019.

[22] M. Y. Landolsi, L. Hlaoua, and L. Ben Romdhane, "Information extraction from electronic medical documents: state of the art and future research directions," *Knowledge and Information Systems*, vol. 65, no. 2, pp. 463–516, 2023.

[23] P. Sondhi, M. Gupta, C. Zhai, and J. Hockenmaier, "Shallow information extraction from medical forum data," in *Proceedings of the International Conference on Computational Linguistics*, Beijing, China, August 2010.

[24] P. Li and K. Mao, "Knowledge-oriented convolutional neural network for causal relation extraction from natural language texts," *Expert Systems with Applications*, vol. 115, pp. 512–523, 2019.

[25] D. Tkaczyk, P. Szostek, M. Fedoryszak, P. J. Dendek, and Ł. Bolikowski, "Cermine: automatic extraction of structured metadata from scientific literature," *International Journal on Document Analysis and Recognition*, vol. 18, no. 4, pp. 317–335, 2015.

[26] P. Lopez, "Grobid: combining automatic bibliographic data recognition and term extraction for scholarship publications," in *Proceedings of the Research and Advanced Technology for Digital Libraries: 13th European Conference, ECDL 2009*, pp. 473-474, Corfu, Greece, September 2009.

[27] G. Zhou and M. Zhang, "Extracting relation information from text documents by exploring various types of knowledge," *Information Processing & Management*, vol. 43, no. 4, pp. 969–982, 2007.

[28] T. Bayer and H. Walischewski, "Experiments on extracting structural information from paper documents using syntactic pattern analysis," *Proceedings of 3rd International Conference on Document Analysis and Recognition*, vol. 1, pp. 476–479, 1995.

[29] W. Merrill, "Formal language theory meets modern nlp," 2021, https://arxiv.org/abs/2102.10094.

[30] O. Altamura, F. Esposito, and D. Malerba, "Transforming paper documents into xml format with wisdom++," *International Journal on Document Analysis and Recognition*, vol. 4, no. 1, pp. 2–17, 2001.

[31] C. Grover, E. Klein, A. Lascarides, and M. Lapata, "Xml-based nlp tools for analysing and annotating medical language," in *Proceedings of the 2nd workshop on NLP and XML*, Stroudsburg, PA, USA, September 2002.

[32] M. M. Rahman and T. Finin, "Deep understanding of a document's structure," 2017, https://ebiquity.umbc.edu/_file_directory_/papers/857.pdf.

[33] S. A. Ajetunmobi and O. Daramola, "Ontology-based information extraction for subject-focussed automatic essay evaluation," in *Proceedings of the 2017 International Conference on Computing Networking and Informatics (ICCNI)*, pp. 1–6, Lagos, Nigeria, October 2017.

[34] F. Peng and A. Mccallum, "Accurate information extraction from research papers using conditional random fields," pp. 329–336, 2004, https://aclanthology.org/N04-1042.pdf.

[35] B. M. Knisely and H. H. Pavliscsak, "Research proposal content extraction using natural language processing and semi-supervised clustering," *A demonstration and comparative analysis*, vol. 128, p. 2023.

[36] O. Iwashokun and A. Ade-Ibijola, “Structural vetting of academic proposals,” *International Journal of Advanced Computer Science and Applications*, vol. 13, no. 7, 2022.

[37] B. Banerjee, W. A. Ingram, J. Wu, and E. A. Fox, “Applications of data analysis on scholarly long documents,” in *Proceedings of the 2022 IEEE International Conference on Big Data (Big Data)*, pp. 2473–2481, IEEE Computer Society, Los Alamitos, CA, USA, December 2022.

[38] Y. Liu and M. Lapata, “Learning structured text representations,” *Transactions of the Association for Computational Linguistics*, vol. 6, pp. 63–75, 2018.

[39] E. Smith, D. Papadopoulos, M. Braschler, and K. Stockinger, “Lillie: information extraction and database integration using linguistics and learning-based algorithms,” *Information Systems*, vol. 105, Article ID 101938, 2022.

[40] N. Chomsky, *The Essential Chomsky*, New Press/ORIM, The Netherlands, 2011.

[41] W. Fitch and A. Friederici, “Artificial grammar learning meets formal language theory: an overview,” *Philosophical Transactions of the Royal Society B: Biological Sciences*, vol. 367, no. 1598, pp. 1933–1955, 2012.

[42] H. Nagarajan, P. Vancha, and M. Supriya, “Recognising the English language using context free grammar with pyformlang,” in *Proceedings of the 2022 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT)*, pp. 1–6, IEEE, Bangalore, India, July 2022.

[43] A. Ade-Ibijola, “Finchan a grammar-based tool for automatic comprehension of financial instant messages,” *Proceedings of the Annual Conference of the South African Institute of Computer Scientist and Information Technologists*, vol. 1, p. 0, 2016.

[44] V. Thorsteinsson, H. Oladottir, and H. Loftsson, “A wide-coverage context-free grammar for Icelandic and an accompanying parsing system,” in *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pp. 1397–1404, Varna, Bulgaria, September 2019.

[45] J. Levine, D. Brown, and T. Mason, *Lex & Yacc*, Oreilly library, Dublin, Ireland, 1998.

[46] C. Moukrim, T. Abderrahim, E. H. Benlahmer, and A. Tarik, “An innovative approach to autocorrecting grammatical errors in Arabic texts,” *Journal of King Saud University-Computer and Information Sciences*, vol. 33, no. 4, pp. 476–488, 2021.

[47] G. Boella, L. Di Caro, and V. Leone, “Semi-automatic knowledge population in a legal document management system,” *Artificial Intelligence and Law*, vol. 27, no. 2, pp. 227–251, 2019.

[48] R. Ahmad, M. T. Afzal, and M. A. Qadir, “Information extraction from pdf sources based on rule-based system using integrated formats,” in *Proceedings of the Semantic Web Challenges: Third SemWebEval Challenge at ESWC 2016*, pp. 293–308, Heraklion, Crete, Greece, May 2016.

[49] M. Cronje and A. Ade-Ibijola, “Automatic slicing and comprehension of cvs,” in *Proceedings of the 2018 5th International Conference on Soft Computing & Machine Intelligence (ISCMI)*, pp. 99–103, IEEE, Nairobi, Kenya, November 2018.

[50] M. Swain and J. Cole, *Chemdataextractor: A Toolkit for Automated Extraction of Chemical Information from the Scientific Literature*, ACS Publications, Washington, DC, USA, 2016.

[51] K.-J. Lee, Y.-S. Hwang, S. Kim, and H.-C. Rim, “Biomedical named entity recognition using two-phase model based on svms,” *Journal of Biomedical Informatics*, vol. 37, no. 6, pp. 436–447, 2004.

[52] C. Wick and F. Puppe, “Fully convolutional neural networks for page segmentation of historical document images,” in *Proceedings of the 2018 13th IAPR International Workshop on Document Analysis Systems (DAS)*, pp. 287–292, IEEE Computer Society, Los Alamitos, CA, USA, April 2018.

[53] A. C. Tran, L. T. Ho, and H. T. Nguyen, “Information extraction from invoices by using a graph convolutional neural network: a case study of Vietnamese stores,” *IEIE Transactions on Smart Processing & Computing*, vol. 11, no. 5, pp. 316–323, 2022.

[54] A. Hamdi, E. Carel, A. Joseph, M. Coustaty, and A. Doucet, “Information extraction from invoices,” in *Proceedings of the Document Analysis and Recognition–ICDAR 2021: 16th International Conference*, pp. 699–714, Springer, Lausanne, Switzerland, September 2021.

[55] E. Yehia, H. Boshnak, S. AbdelGaber, A. Abdo, and D. S. Elzanfaly, “Ontology-based clinical information extraction from physician’s free-text notes,” *Journal of Biomedical Informatics*, vol. 98, Article ID 103276, 2019.

[56] Y.-C. Zhou, Z. Zheng, J.-R. Lin, and X.-Z. Lu, “Integrating nlp and context-free grammar for complex rule interpretation towards automated compliance checking,” *Computers in Industry*, vol. 142, Article ID 103746, 2022.

[57] A. Barducci, S. Iannaccone, V. La Gatta, V. Moscato, G. Sperlì, and S. Zavota, “An end-to-end framework for information extraction from Italian resumes,” *Expert Systems with Applications*, vol. 210, Article ID 118487, 2022.

[58] B. Townsend, E. Ito-Fisher, L. Zhang, and M. May, “Doc2dict: information extraction as text generation,” 2021, https://arxiv.org/abs/2105.07510.

[59] R. J. M. Ventayen, “Openai chatgpt generated results: similarity index of artificial intelligence-based contents,” 2023, https://papers.ssrn.com/sol3/papers.cfm?abstract_id=4332664.

[60] D. Kirtania and S. Patra, “Openai chatgpt generated content and similarity index: a study of selected terms from the library & information science (lis),” *Annals of Library and Information Studies*, vol. 70, 2023.

[61] Ö. Aydın and E. Karaarslan, “Openai chatgpt generated literature review: digital twin in healthcare,” 2022, https://papers.ssrn.com/sol3/papers.cfm?abstract_id=4308687.