INDIAN JOURNAL OF SCIENCE AND TECHNOLOGY

ORIGINAL ARTICLE

Multiple Language Document Indexing and Classification with Page Number

V. Nirmala[1]*, **B. Lavanya**[1]

1 Department of Computer Science, University of Madras, Guindy, Chennai-600025, Tamil Nadu, India

OPEN ACCESS

Received: 03/05/2025

Accepted: 13/06/2025

Published: 29/06/2025

Citation: Nirmala V, Lavanya B (2025) Multiple Language Document Indexing and Classification with Page Number. Indian Journal of Science and Technology 18(25): 1998-2008. https://doi.org/10.17485/IJST/v18i25.834

* **Corresponding author**.

vnirmala7488@gmail.com

Funding: None

Competing Interests: None

Copyright: © 2025 Nirmala & Lavanya. This is an open access article distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

Published By Indian Society for Education and Environment (iSee)

ISSN
Print: 0974-6846
Electronic: 0974-5645

Abstract

Objectives: To introduce a methodical and automated strategy for indexing and organizing PDF or Word documents. **Method**: This study proposes the Multiple Language Document Indexing System (MLDIS) Multinomial Naïve Bayes algorithm, an automatic indexing method that uses font-based characteristics, such as font size, type, and page numbers, to methodically create an index page while simultaneously classifying the document by language. By identifying the largest font as the title, progressively smaller fonts as headings, and consistent formatting patterns as subheadings, the system can effectively recognize and categorize textual elements. Hierarchical content organization is enabled through this structured approach, which also significantly reduces human error while ensuring increased precision and consistency in automated index development. The MLDIS Multinomial Naive Bayes algorithm is a scalable and dependable solution for both digital and print indexing, having demonstrated high efficiency in processing huge volumes of data. **Findings**: The method's ability to capture text structures, improve document navigation, and accelerate retrieval operations is confirmed by experimental results. The proposed MLDIS Multinomial Naïve Bayes algorithm achieved accuracy levels above 97% for PDF formats and 100% for Word formats, further advancing automated document processing. **Novelty/Applications**: This indexing technique can be applied across a wide range of areas, including academic journal publications, test question repositories, book indexing, and document collections in multiple languages such as English, Tamil, Telugu, and Hindi, as well as in banking management systems.

Keywords: Multiple Language Document Indexing System; Multinomial Naïve Bayes algorithm; Automatic Indexing method; Page number

1 Introduction

In existing methods, we identified four key gaps in current approaches to document indexing and organization. First, most systems lack support for subheadings, generating only a basic table of contents that captures main headings without any structural depth. Second, these solutions are typically limited to English and do not support multilingual

content. Third, many existing approaches rely heavily on manual indexing and organization, which is both time-consuming and prone to errors. Fourth, there is an absence of automatic classification based on the language used within the document. Our proposed method addresses all these gaps through a systematic and automated approach to indexing and organizing PDF or Word documents. It automatically extracts title of the page, both headings and subheadings with precise page references, supports multiple languages—including English, Tamil, Hindi, and Telugu—eliminates manual effort through full automation, and performs language-aware classification by segmenting content according to the language used in each section. This research introduces a methodical and automated strategy for indexing and organizing PDF or Word documents. The process entails extracting headings, subheadings, and associated content, along with accurate page references, while simultaneously classifying the material based on its language. We tested with the English, Tamil, Hindi, and Telugu languages. Conventional indexing techniques require careful manual input, rendering the process labor-intensive, susceptible to discrepancies and familiarity of each language. The indexing process commences with the extraction of text from each page, followed by a thorough analysis of its structure to identify headings and subheadings based on established patterns in font size, style, numbering, and textual hierarchy. The components are systematically indexed and aligned with their respective page numbers, facilitating a well-structured and easily navigable document. A primary heading, such as ”2” or ”3,” is arranged into subheadings like ”2.1,””2.2,””3.1,” and ”3.2,” following a coherent and logical hierarchical framework. This structured classification improves document clarity and enables swift access to particular sections. Traditionally, the process of creating an index for comprehensive documents such as books, research papers, and technical manuals necessitates that users manually compile the index during the content creation phase, as illustrated in LaTeX and various document processing systems. This approach poses considerable challenges and shows limited effectiveness, especially when dealing with lengthy documents. Our proposed solution removes the necessity for manual indexing by enabling users to dynamically generate a comprehensive index upon the completion of content. The MLDIS Multinomial Naïve Bayes algorithm accommodates multiple file formats like PDF and Word enabling users to automatically generate an index page that precisely aligns headings and subheadings with their corresponding page numbers. This guarantees a smooth and effective navigation experience across various document types. Our solution enhances document navigation by automatically linking headings and subheadings to their corresponding pages, thereby improving information accessibility and considerably minimizing user effort. This method efficiently conserves time and reduces the likelihood of human errors, all while maintaining consistency and precision in the indexing process. The resulting structured index improves document organization, serving as a vital resource for authors, researchers, publishers, and individuals who need well-organized, easily searchable, and effectively structured PDF documents.

The contribution of the paper is

- To automatically classify indexing pages according to their respective multilingual language.
- To facilitate the automatic generation of an index that includes titles, headings, and multiple subheadings along with their corresponding page numbers.
- The MLDIS Multinomial Naïve Bayes algorithm has been implemented to support multiple languages, including English, Tamil, Hindi, and Telugu.
- MLDIS is compatible with various document formats, like PDF and Word.

The work in [1] evaluates the universal framework indexing methods like suffix arrays, FM-index, and r-index. The study [2] compares different text indexing approaches, in medical subject headings and highlighting the trade-offs between space efficiency and query speed. In [3], studied two key data management techniques: indexing and filtering and also explores parallel processing for efficient data filtering. In [4], uses an index-based approach to discovering cohesive subgraphs in hypergraphs and validates the efficiency of its indexing and querying methods. The article [5] introduces Multi-view Content-aware indexing (MC-indexing) for long document question answering (DocQA). Unlike fixed-length chunking, Multi-view Content-aware indexing (MC-indexing) segments documents into structured content chunks and represents each chunk in three views: raw text, keywords, and summary. MC-indexing is a plug-and-play solution that requires no training and integrates easily with any retriever to enhance performance. In [6], reviews database system Automatic Index Tuning (AIT) methods. It discusses AIT from problem definitions, frameworks, index types, automation levels, and more and also covers preprocessing, empirical and ML-based index benefit estimation, offline and online index selection. The study [7] introduces a novel hybrid approach for biomedical document indexing, combining the strengths of the Vector Space Model (VSM) and Description Logics (DL). VSM facilitates partial matching between document terms and external resources, while DL enhances knowledge representation for improved matching. In the article [8] addresses the limitations of traditional Information Retrieval (IR) systems, which fail to capture semantic meaning and struggle with vocabulary gaps when users and systems use different terms for the same concept. To overcome this, a novel hybrid semantic indexing approach is used, combining machine learning (skip-gram with negative sampling) and domain ontology to annotate and rank document concepts. The study [9] introduces an automatic PDF

indexing system to convert PDFs into structured data and extract key information for a keyword library. It uses text features and grammar rules to visualize and cluster text blocks, then applies tag-weight-based keyword extraction. The structured data enables intelligent e-books and knowledge graphs, helping publishers transition to intelligent knowledge services. The research work[10] introduces a query-based method for automated MongoDB indexing, requiring no data or usage stats and validated the algorithm on TPC-H and showed that all suggested indexes were used. The article[11] aims to help libraries adopt (semi-)automated subject indexing to enhance metadata and improve information retrieval. The article[12] discusses how ML can enhance index tuning by dynamically adapting to evolving workloads, opening new research directions for performance optimization in databases. In[13], describes how to automate important steps of subject indexing for specialized texts, such as word extraction, importance ranking, page reference recognition, and hierarchical term structuring based on semantic relationships and designed for Russian scientific documents, these methods combine formal linguistic rules and unsupervised machine learning. The study[14] presents improving an impact score model from the passage and tackles a vocabulary mismatch. The research paper[15] discusses improving document organization and retrieval in big data environments using Apache Lucene and Hadoop. It highlights the challenge of unstructured file storage in large organizations and gives a solution through parallel document clustering and indexing. The paper[16] address knowledge source limitations and introduce a phrase-based indexing model that parses documents into phrases using domain-specific terms and evaluates similarity based on concepts and common word stems. Incorporating word stems compensates for gaps in concept-based indexing, improving retrieval accuracy. The paper[17] discusses about the similarities, difference, advantages and disadvantages of automatic subject indexing of text categorization, document clustering and document classification. The study[18] focuses on open-domain QA models in large documents for each query that are too sluggish for real-time application. This research[19] discusses improving multimedia information retrieval (MMIR) by combining text and image data to reduce false positives when searching large multimodal collections like the web. In[20], the author suggests a way for constructing content-based information systems for digitized archives that makes use of historical indexing systems. During the time that archives were operational, these indexes were developed for the purpose of looking for documents that were similar to those that were already there. After presenting a conceptual model that can be used to define and make use of these technologies, the authors then proceed to present a framework that may be used to guide the digitization and indexing of historical data.

2 Methodology

The process begins by loading the document, followed by extracting text from all pages and performing preliminary preprocessing to remove punctuation. The next step involves applying the MLDIS Multinomial Naïve Bayes to refine the extracted text. Subsequently, the system separates the text from the page numbers and categorizes the content into two distinct sections:

1. Text formatted with bold, italics, or underlining.
2. Text without any of the aforementioned formatting styles.

The system then identifies the font size, font style, and font name associated with each text element, along with the corresponding page number. Additionally, it determines the language based on the font name and assigns appropriate labels accordingly. Following this, the system analyzes each text element based on font name to classify the language and compares the font size and style for hierarchical categorization. Text elements with significantly larger font sizes are classified as Titles. The next largest font size is designated as Headings, while consistent hierarchical subheadings of smaller font sizes are classified as Subheadings. The MLDIS Multinomial Naïve Bayes algorithm enables the system to automatically categorize text elements based on language and formatting. It then generates an index that includes Titles, Headings, and Subheadings, along with their corresponding page numbers, as illustrated in Figure 1.

2.1 Removal of stop words

Eliminating unwanted symbols in text is a crucial preprocessing step in natural language processing. Stop word removal, one of the most frequently used techniques, involves deleting commonly occurring words across all documents in a corpus to improve text analysis efficiency.

2.2 Label encoding

Label encoding converts categorical data into numerical values for machine learning models by assigning a unique integer to each category. After identifying the font type, its keywords are extracted and assigned a label to categorize the document.

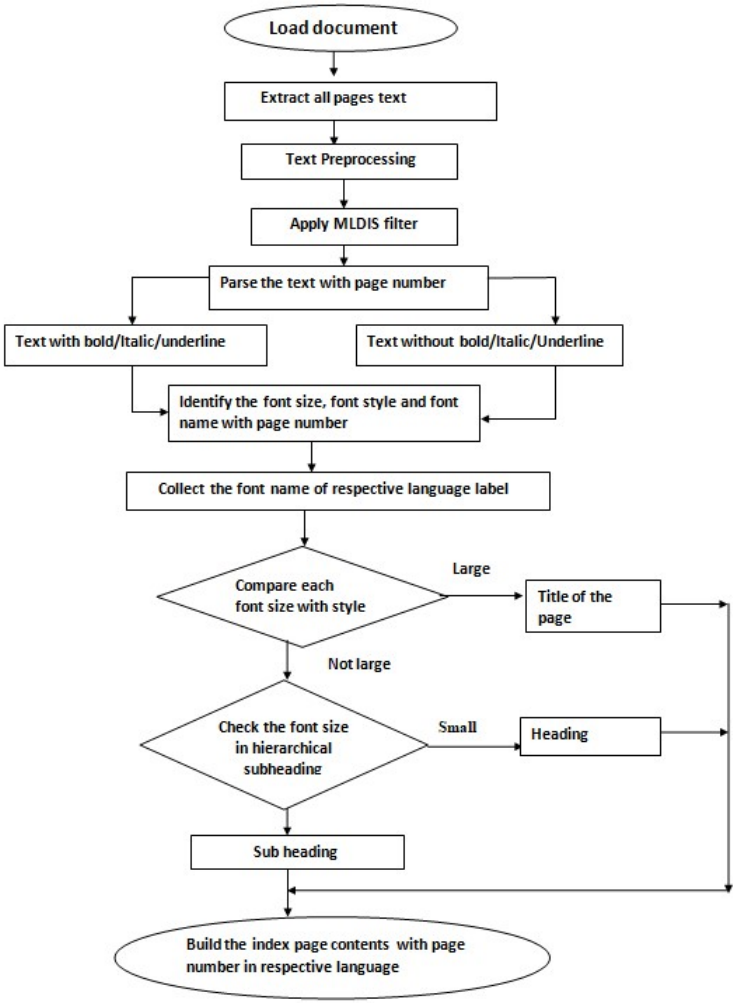

Fig 1. Proposed Architecture model

2.3 Naïve Bayes

Naive Bayes classifiers are supervised machine learning models used for classification, applying Bayes' Theorem to determine probabilities. They are called probabilistic classifiers because they assume that each feature is independent and affects the prediction individually.

2.4 MLDIS Multinomial Naïve Bayes algorithm

The proposed Multiple Language Document Indexing System (MLDIS) Multinomial Naïve Bayes algorithm analyzes a PDF/Word document (d) by extracting text and categorizing it into Title (T), Headings (H), and Subheadings (SH), besides the document's language (e). The languages include English, Hindi, Tamil, and Telugu. Initially, the document is loaded, and its 'n' pages are extracted. Subsequently, the text is categorized into two groups: text with bold, italic, or underlined typefaces (tfb) and text devoid of these styles (tofb). Then, it recognizes essential characteristics including font size (fs), font name (fn), and font style (fe), simultaneously identifying the language of the document (e) and designating a suitable label (L). The system subsequently examines tfb and categorizes large font size text (Lfs) as the Title (T). If the text in tfb has a tiny font size (Sfs), it is classified as a Heading (H). To enhance the classification, it verifies whether H adheres to a pattern (P); if favorable, it is categorized as a Subheading (SH); if not, it retains the designation of Heading (H). The suggested MLDIS Multinomial Naïve Bayes algorithm autonomously organizes the index page by arranging the Title (T), Headings (H), and Subheadings

(SH) with their corresponding page numbers in the respective language (L). This methodical technique guarantees precise text classification, facilitating document navigation and analysis in algorithm 1.

Algorithm 1: Multiple Language Document Indexing System (MLDIS) Algorithm

Input: Load the document d
Define :
Lfs = Large font size text in a document
Dfs = Different least font size text in a document
Sfs = Small font size text in a document
tfb = Text document fonts with Bold/Italic/Underline
tofb = Text document font without Bold/Italic/Underline
Labels (L), Title (T), Subheadings (SH), Headings (H), Languages (e)
Font size(fs), Font name(fn), Font style(fe), Pattern in a text(P)
Output: Indexed structure with Title (T), Headings (H), Subheadings (SH) in Language(e)
Step1: Input d
Step2: Split the text document tfb, tofb with page number
Step3: Identify fs, fn, fe
Step4: Identify the language (e) and assign L
Step5: Compare each text font with bold tfb
if tfb = Lfs **then**
Set as Title (T)
else if tfb = sfs **then**
if sfs is in Pattern (P) **then**
Set as Subheading(SH)
else
Keep as Heading (H)
end if
end if
Step 6: Categorize the language (L), arrange and build the index page structure (T, H, SH)

2.5 Dataset

Table 1. List of Dataset-1

Sno	Name of the document	# PDF Files
1	Google scholar-journals	63
2	Indian bank -Internet/Mobile/Tele banking form	16
3	SBI - Account opening terms and condition	16
4	IOB - Account opening individual form	7
5	City union bank –FD&RD form	2
6	School prospectus	3
7	College prospectus	4
8.	Accountant (kaggle)	118
9.	Dataset of Pdf files (kaggle)	1078
10.	Tamil journals	30
11.	Hindi journals	20
12.	Telugu question papers	13
13.	Combined English, Tamil, Hindi question paper	10
14.	Combined English, Tamil, Hindi Admission form	15
Total		**1395**

Table 2. List of Dataset 2

Sno	Name of the document	# Word Files
1	Google scholar-journals	63
2	Tamil journals	30
3	Hindi journals	20
4	Combined English, Tamil, Hindi question paper	10
5	Combined English, Tamil, Hindi Admission form	15
6	Accountant (kaggle) converted in word	118
7	Dataset of Pdf files (kaggle) converted in word	1078
8.	Telugu question papers	13
Total		**1347**

3 Result and Discussion

In earlier approaches [16], the generation of the table of contents was limited to listing only the main headings without accounting for subheadings, page number or multilingual support. In contrast, our proposed system enhances the indexing process by creating a detailed and structured index page that includes both headings and subheadings, mapped precisely to page numbers. Furthermore, while previous methods operated solely in English, our implementation extends support to multiple languages, including English, Tamil, Telugu, and Hindi—thus improving accessibility and usability across diverse linguistic contexts.

The Table 3 shows the technique applied in five classification methods with proposed MLDIS method and showing good performance in MLIDS. In Table 4 shows the document applied in both PDF and Word file format. Figure 2 shows the multiple language document indexing system accuracy comparison on document in PDF and word.

Table 3. Accuracy compared 6 methods with proposed MLDIS Multinomial Naïve Bayes

S no	Document	#Word Files	Methods	Accuracy (%)
1	Google scholar-journals	63	**Proposed-MLDIS**	**100**
			XGBoost	92
			Random Forest	92
			Adaboost	63
			Logistic Regression	75
			Decision Tree	67
			Support vector machine	78
2	Indian bank--Internet/Mobile/Tele	16	**Proposed-MLDIS**	**100**
			XGBoost	82
			Random Forest	86
			Adaboost	87
			Logistic Regression	85
			Decision Tree	69
			Support vector machine	86
3.	SBI - Account opening terms and condition	16	**Proposed-MLDIS**	**100**
			XGBoost	87
			Random Forest	86
			Adaboost	75
			Logistic Regression	67
			Decision Tree	46
			Support vector machine	56
4	IOB - Account opening individual form	7	**Proposed-MLDIS**	**100**
			XGBoost	65
			Random Forest	86
			Adaboost	85
			Logistic Regression	67
			Decision Tree	46
			Support vector machine	86

Continued on next page

Table 3 continued

			Proposed-MLDIS	**100**
			XGBoost	86
			Random Forest	75
5	City union bank –FD&RD form	2	Adaboost	79
			Logistic Regression	89
			Decision Tree	82
			Support vector machine	69
			Proposed-MLDIS	**100**
			XGBoost	87
			Random Forest	75
6	School prospectus	3	Adaboost	88
			Logistic Regression	67
			Decision Tree	68
			Support vector machine	78
			Proposed-MLDIS	**100**
			XGBoost	89
			Random Forest	92
7	College prospectus	4	Adaboost	71
			Logistic Regression	86
			Decision Tree	78
			Support vector machine	86
			Proposed-MLDIS	**98**
			XGBoost	87
			Random Forest	87
8	Accountant(kaggle)	118	Adaboost	86
			Logistic Regression	84
			Decision Tree	75
			Support vector machine	68
			Proposed-MLDIS	**99**
			XGBoost	86
			Random Forest	89
9.	Dataset of Pdf files(kaggle)	1078	Adaboost	90
			Logistic Regression	82
			Decision Tree	83
			Support vector machine	86
			Proposed-MLDIS	**99**
			XGBoost	89
			Random Forest	77
10.	Tamil journals	30	Adaboost	75
			Logistic Regression	68
			Decision Tree	78
			Support vector machine	89
			Proposed-MLDIS	**98**
			XGBoost	65
			Random Forest	**68**
11.	Hindi journals	20	Adaboost	71
			Logistic Regression	73
			Decision Tree	89
			Support vector machine	78
			Proposed-MLDIS	**98**
			XGBoost	89
			Random Forest	85
12.	Telugu question papers	13	Adaboost	77
			Logistic Regression	79
			Decision Tree	75
			Support vector machine	88

Continued on next page

Table 3 continued

			Proposed-MLDIS	**100**
			XGBoost	86
	Combined English,		Random Forest	82
13.	Tamil,Hindi question paper	10	Adaboost	87
			Logistic Regression	87
			Decision Tree	70
			Support vector machine	78
			Proposed-MLDIS	**100**
	Combined English,		XGBoost	87
14.	Tamil,Hindi Admission form	5	Random Forest	87
			Adaboost	82
			Logistic Regression	91
			Decision Tree	94
			Support vector machine	78

Table 4. Accuracy comparison on documents applied in both PDF and word format

Sno	Name of the document	# Word Files	Methods	Accuracy (%)	
				In PDF	In Word
			Proposed-MLDIS	**100**	**100**
			XGBoost	92	93
1	Google scholar-journals	63	Random Forest	92	93
			Adaboost	63	86
			Logistic Regression	75	84
			Decision Tree	67	88
			Support vector machine	78	81
			Proposed-MLDIS	**99**	**100**
			XGBoost	89	88
			Random Forest	77	87
2.	Tamil journals	30	Adaboost	75	84
			Logistic Regression	68	87
			Decision Tree	78	79
			Support vector machine	89	78
			Proposed-MLDIS	**98**	**100**
			XGBoost	65	72
			Random Forest	68	64
3.	Hindi journals	20	Adaboost	71	82
			Logistic Regression	73	82
			Decision Tree	89	87
			Support vector machine	78	89
			Proposed-MLDIS	**98**	**100**
			XGBoost	89	89
			Random Forest	85	87
4.	Telugu question papers	13	Adaboost	77	81
			Logistic Regression	79	81
			Decision Tree	75	89
			Support vector machine	88	89
			Proposed-MLDIS	**100**	**100**
			XGBoost	86	88
	Combined English,		Random Forest	82	87
5.	Tamil,Hindi question paper	10	Adaboost	87	91
			Logistic Regression	87	91
			Decision Tree	70	89
			Support vector machine	78	88

Continued on next page

Table 4 continued

			Proposed-MLDIS	**100**	**100**
			XGBoost	87	89
	Combined English,		Random Forest	87	88
6.	Tamil, Hindi	5	Adaboost	82	89
	Admission form		Logistic Regression	91	91
			Decision Tree	94	91
			Support vector machine	78	89
			Proposed-MLDIS	**98**	**100**
			XGBoost	87	89
			Random Forest	87	89
7	Accountant (kaggle)	118	Adaboost	86	88
			Logistic Regression	84	89
			Decision Tree	75	89
			Support vector machine	68	87
			Proposed-MLDIS	**99**	**100**
			XGBoost	86	89
	Dataset of Pdf files		Random Forest	89	89
8.	(kaggle) converted	1078	Adaboost	90	91
	in word		Logistic Regression	82	86
			Decision Tree	83	87
			Support vector machine	86	88

The proposed Multiple Language Document Indexing System (MLDIS) Multinomial Naïve Bayes algorithm demonstrates superior performance in multilingual document classification, consistently outperforming traditional classifiers. It achieves perfect accuracy in Word documents and exhibits remarkable accuracy in PDF files, consistently reaching or exceeding 96%. Our approach utilizes PyMuPDF in Python, an effective toolkit for PDF text extraction and analysis, to systematically parse, classify, and organize document content into a unified index, enabling accurate and efficient document classification and indexing for downstream tasks. The results in Table 3 and Table 4 highlights the model's robustness, efficiency, and adaptability across various document types and formats, making it a highly reliable solution for automated document classification.

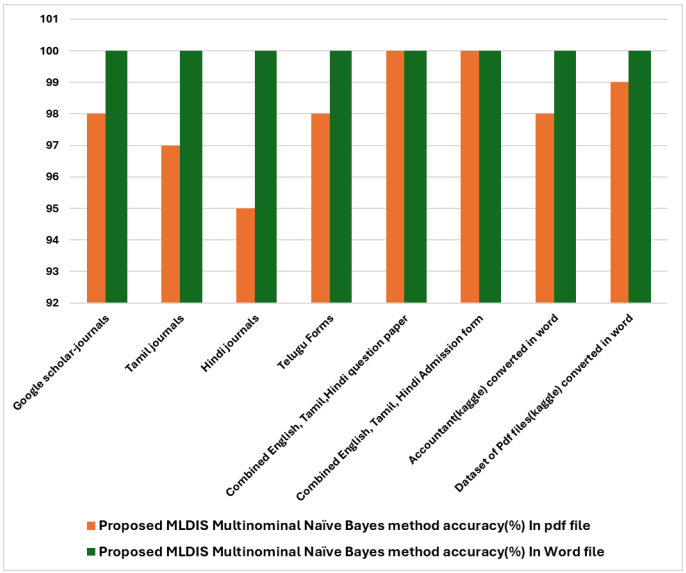

Fig 2. Multiple Language Document Indexing System Accuracy Comparison on document in PDF and word

3.1 F-statistics and verification

To evaluate the effectiveness of the Proposed Multiple Language Document Indexing System (MLDIS) Multinomial Naïve Bayes algorithm framework, an F-test for ANOVA is conducted to compare classification accuracy across different methods. The F-statistic measures whether the variance in classification accuracy between the MLDIS Multinomial Naïve Bayes algorithm and other methods are statistically significant. A higher F-value with a low p-value suggests that the proposed method provides a significant improvement in accuracy.

- **NULL Hypothesis H0:** The Proposed MLDIS Multinomial Naïve Bayes algorithm does not improve classification accuracy significantly when compared with other methods.
- **Alternate Hypothesis H1:** The Proposed MLDIS Multinomial Naïve Bayes algorithm improves Classification accuracy significantly when compared with other methods.

Table 5. F-statistic and P-value of MLDIS Multinomial Naïve Bayes algorithm

S.no	Name of the document	F-stat	P-value
1	Google scholar-journals	5.44	0.04
2	Indian bank -Internet/Mobile/Tele banking form	1.77	0.00
3	SBI - Account opening terms and condition	0.72	0.00
4	IOB - Account opening individual form	1.51	0.00
5	City union bank –FD&RD form	1.52	0.04
6	School prospectus	1.25	0.00
7	College prospectus	1.25	0.00
8.	Accountant (kaggle)	3.52	0.02
9.	Dataset of Pdf files (kaggle)	4.25	0.03
10.	Tamil journals	4.52	0.03
11.	Hindi journals	4.00	0.03
12.	Telugu question papers	4.25	0.03
13.	Combined English, Tamil, Hindi question paper	3.25	0.02
14.	Combined English, Tamil, Hindi Admission form	3.25	0.02

In Table 5, the MLDIS Multinomial Naïve Bayes algorithm shows statistically significant improvement across all datasets, with p-values below 0.05. In Table 5 (PDF format), datasets such as Google Scholar journals (F-stat: 5.44, p-value: 0.04), Indian bank-Internet/Mobile/Tele banking form (1.77,0.00), SBI- Account opening terms and condition (0.72,0.22), IOB - Account opening individual form (1.51,0.00) ,City union bank –FD&RD form(1.52,0.04)Tamil Journals (4.52, 0.03), Hindi Journals (4.00, 0.03), Telugu Question Papers (4.25, 0.03), and Combined English, Tamil, Hindi datasets in question paper and admission form(3.25, 0.02) show strong results. Additionally, Accountant (Kaggle) (3.52, 0.02) and Dataset of PDF Files (Kaggle) (4.25, 0.03) also perform well and achieve higher accuracy due to their structured text format. Since all p-values are below 0.05, the alternative hypothesis (H_1) is accepted, confirming that the MLDIS Multinomial Naïve Bayes significantly improves classification accuracy when compared to other methods thus accurately indexes the documents.

4 Conclusion

The proposed MLDIS Multinomial Naïve Bayes algorithm shows strong performance results in automatic document indexing across both Word and PDF formats. It effectively analyzes font types, sizes, and layout structures to identify titles, headings, subheadings, and page numbers, even in multilingual documents. Here the proposed works build the indexing with page number at different languages in a single document at a time. The languages are used in the proposed techniques are English, Tamil, Telugu and Hindi. Word files yield higher accuracy due to consistent font rendering, while PDFs may suffer from font embedding issues that affect text recognition. Despite minor PDF limitations, the algorithm achieved up to 100% accuracy in Word and 98% in PDF. Originally developed in 2020 for automated Table of Contents generation, the approach has been significantly enhanced using the Proposed MLDIS Multinomial Naïve Bayes algorithm, achieving improved classification accuracy for both PDF and Word documents.

Strength:

It automatically classifies the indexed pages, including titles, headings, and subheadings, with page numbers according to their respective languages. It is able to quickly build the index pages separately for each language in a single document. The supported languages are English, Tamil, Hindi, and Telugu. In a Word file, it achieves 100% accuracy.

Weakness:

The index is also classified in PDF files however, there is a minor distinction when compared to Word files due to the predefined nature of PDF fonts. Consequently, it attains an accuracy rate of 98%.

References

1) Belazzougui D, Cunial F, Karkkainen J, Makinen V. Linear-time String Indexing and Analysis in Small Space. *ACM Transactions on Algorithms*. 2020;16(2):1–54. https://doi.org/10.1145/3381417.
2) Amar-Zifkin A, Ekmekjian T, Paquet V, Landry T. Algorithmic indexing in Medline frequently overlooks important concepts and may compromise literature search results. *Journal of the Medical Library Association*. 2025;113(1):39–48. Available from: https://jmla.pitt.edu/ojs/jmla/article/view/1936#:~:text=Three%20main%20issues%20with%20algorithmically,significant%20concept%20not%20represented%20in.
3) Kesavan T, Kumar S. Graph based Indexing techniques for big data Analytics: a Systematic Survey. *International Journal of Recent Technology and Engineering*. 2019;7(6S5):641–647. Available from: https://www.ijrte.org/wp-content/uploads/papers/v7i6s5/F11120476S519.pdf.
4) Luo Q, Zhang W, Yang Z, Yu D, Lin X, Wang L. Cohesive Subgraph Discovery in Hypergraphs: A Locality-Driven Indexing Framework. *The VLDB Journal*. 2025;34(34):1–20. https://doi.org/10.1007/s00778-025-00915-x.
5) Dong K, Goh DXD, Lee YQ, Zhang H, Li X, Zhang C, et al. Multi-view Content-aware Indexing for Long Document Retrieval. *Computational and Language*. 2024;15(103):1–14. Available from: https://arxiv.org/abs/2404.15103.
6) Wu Y, Zhou X, Zhang Y, Li G. Automatic Index Tuning: A Survey. *IEEE Transactions on Knowledge and Data Engineering*. 2024;36(12):7657–7667.
7) Boukhari K, Omri MN. DL-VSM based document indexing approach for information retrieval. *Journal of Ambient Intelligence and Humanized Computing*. 2023;14(5):5383–5394. Available from: https://link.springer.com/article/10.1007/s12652-020-01684-x#:~:text=5%20VSM-DL%20based%20document,by%20exploiting%20VSM%20and%20DL.
8) Sharma A, Kumar S. Machine learning and ontology-based novel semantic document indexing for information retrieval. *Computers & Industrial Engineering*. 2023;176(2):108940. Available from: https://www.sciencedirect.com/science/article/abs/pii/S0360835222009287.
9) Chen K, Huang J, Zhang O, Cui Y. Research and Implementation of Automatic Indexing Method of PDF for Digital Publishing. *ACM Transactions on Asian and Low-Resource Language Information Processing*. 2023;22(3):1–21. https://dl.acm.org/doi/10.1145/3501400.
10) Espona L, Vichalkovski, Steingartner A, Pustulka W, Elzbieta. Automatic indexing for MongoDB. 2023;p. 535–543. Available from: https://irf.fhnw.ch/entities/publication/ec51fb02-f3f1-454a-8e7e-b972f5ada2b5.
11) Golub K. Automated subject indexing: An overview. *Cataloging & Classification Quarterly*. 2021;59(8):702–719. Available from: https://www.tandfonline.com/doi/full/10.1080/01639374202120123 11.
12) Valavala M, Alhamdani W. Automatic database index tuning using machine learning. 2021;p. 1–8. Available from: https://ieeexplore.ieee.org/abstract/document/9358646.
13) Bolshakova E, Ivanov KM. Automating Hierarchical Subject Index Construction for Scientific Documents. 2020;p. 201–214.
14) Mallia A, et al. Learning passage impacts for inverted indexes. 2021. Available from: https://dl.acm.org/doi/pdf/10.1145/3404835.3463030.
15) Aliyu MB, Iqbal R, James A, Nkantah D. SED: An Algorithm for Automatic Identification of Section and Subsection Headings in Text Documents. *International Journal of Computer Science Issues*. 2020;17(6):40–47. Available from: https://www.ijcsi.org/papers/IJCSI-17-6-40-47.pdf.
16) Lydia L, Moses J, Varadarajan V, Nonyelu F, Shankar. Clustering and indexing of multiple documents using feature extraction through apache hadoop on big data. *Malaysian Journal of Computer Science*. 2020;1(1):108–123. https://doi.org/10.22452/mjcs.sp2020no1.8.
17) Golub K. Automatic subject indexing of text. *Knowledge Organization*. 2019;46(2):104–121. Available from: https://www.nomos-elibrary.de/de/10.5771/0943-7444-2019-2-104.pdf.
18) Seo M, et al. Real-time open-domain question answering with dense-sparse phrase index. *arXiv preprint arXiv:190605807*. 2019. Available from: https://arxiv.org/pdf/1906.05807.
19) Seo M, et al. Real-time open-domain question answering with dense-sparse phrase index. *arXiv preprint arXiv:190605807*. 2019. Available from: https://arxiv.org/pdf/1906.05807.
20) Colavizza G, Ehrmann M, Bortoluzzi F. Index-driven digitization and indexation of historical archives. *Frontiers in Digital Humanities*. 2019;6:4. https://doi.org/10.3389/fdigh.2019.00004.