EPiC Series in Computing

Volume 98, 2024, Pages 103–115

Proceedings of 39th International Conference on Computers and Their Applications

Unveiling Text Mining Potential: A Comparative Analysis of Document Classification Algorithms

Sindhuja Penchala[1], Saydul Akbar Murad[1], Indranil Roy[2], Bidyut Gupta[3], and Nick Rahimi[1]

[1] School of Computing Sciences & Computer Engineering, University of Southern Mississippi, Hattiesburg, MS
Sindhuja.Penchala@usm.edu
saydulakbar.murad@usm.edu
nick.rahimi@usm.edu

[2] Department of Computer Science, Southeast Missouri State University, Cape Girardeau, MO
iroy@semo.edu

[3] School of Computing, Southern Illinois University; Carbondale, IL
bidyut@cs.siu.edu

Abstract

The importance of document classification has grown significantly in recent years, mostly due to the rise in digital data volumes. Since textual documents often contain more than 80% of all information, there is a perception that text mining has tremendous commercial potential. For future uses, knowledge extraction from these texts is essential. However, it is difficult to obtain this information due to the vast volume of files. As a result, since text classification was introduced, the practice of classifying documents by text analysis has grown in significance. We have primarily employed three different algorithms to compare the metrics between them in order to assess the performance of various models. For this, the dataset was created by extracting condensed information from a variety of textbook genres, including business, social science, and computer science textbooks. To classify textbooks within the same subject group, we used three supervised machine learning techniques in this study: decision trees, random forests, and neural networks. Among these three models, multilayer perceptron neural networks have performed and produced the best outcomes.

1 Introduction

Nowadays, large volumes of data are continuously being generated because of the growth of the Internet, social media, IoT, and mobile technologies. This volume is always increasing and contains insightful data when analyzed and linked to additional data sources [1]. Handling large numbers of documents on a regular basis presents a difficulty for both government agencies and private corporations. These documents come from a variety of sources, including external organizations like other governmental authorities or subcontractors, as well as internal processes. In the absence of an automated system, handling this large document load requires a substantial

number of resources. The amount of data on the earth is expected to surpass 180 zettabytes over the next five years. In other words, the demands for data are expanding too fast for humans to keep up with. Managing this enormous amount of data, assuring a trustworthy system for classifying it for effective reuse, and averting the annoying problem of information overload are just a few of the issues that need to be addressed [2]. By automatically analyzing and categorizing texts, Artificial Intelligence (AI) [3], Machine Learning (ML) [4], and Natural Language Processing (NLP) [5] allow us to save time and effort.

Document categorization became a problem in the domains of computer science, information science, and library science. Assigning a document to one or more classes or categories is its main goal. Automated methods for document classification are crucial in information retrieval systems like search engines because they enable users to locate specific information more quickly, easily, and conveniently while also saving a substantial amount of time and money. By putting in place an automatic document categorization system, you can expeditiously categorize each order according to parameters such as contents and delivery date. We only need to create a set of labels and include them into our document classification system to set this system up. After that, the system will handle the work of evaluating and categorizing every kind of shipment request, removing the need for human decision-making regarding special treatment and prioritization. The major objective of document classification is to create a robust classification model that can correctly classify documents into the appropriate categories.

Document classification can be done in two basic ways: text-based and visual-based. Text and graphics can both be found in documents.

Text Categorization: Text classification pertains to the interpretation of textual elements and is a subset of document classification. Anything from a single syllable to a page composed entirely of text might be considered a textual element. Compared to document classification, this procedure is more complex since contextual information is typically less available. While document classification can use the entire document for context, text classification solely considers the text that is present in a document. In text classification, text classifiers utilize Natural Language Processing algorithms to comprehend text-based data and classify words and phrases into categories depending on their context.

Image Classification: Image classification, on the other hand, is a subset of document classification that concentrates solely on individual images rather than entire texts. Photos and other visual documents are classified using visual technologies such as object identification, image recognition, and computer vision, which all rely on the visual characteristics and behaviors of the data. Then, using a range of criteria, the AI model can classify these images.

The growing amount of digital text data that permeates every aspect of our everyday lives highlights the importance of document classification. The capacity to categorize documents yields important insights automatically and reliably and speeds numerous procedures, consequently aiding businesses ranging from e-commerce to healthcare. Examples of these processes include classifying news articles, consumer evaluations, legal documents, and medical data. The diversified and unstructured form of the text makes this document classification process extremely difficult. Because natural language is context-dependent and flexible by nature, machines find it difficult to understand it. The creation of complex models and algorithms that can accurately capture the contextual information and semantic subtleties found in text is necessary to address these issues.

Furthermore, the introduction of deep learning and the availability of large-scale datasets has led to notable advances in document categorization algorithms in recent years. This part will cover the essential details of document categorization, as well as its applications, constraints, strategies, and state-of-the-art techniques. We will also discuss the task's importance

in a variety of sectors and how it can lead to future advancements in NLP and information retrieval. Gaining a greater understanding of document classification will help researchers and practitioners select the most effective approaches and strategies to fulfill their specific needs for document organizing and categorization.

This primary objective is to perform an extensive overview of the body of knowledge and research on the classification and representation of documents. This entails summarizing important information about tokenization considerations, domain ontology, syntax, and semantics. Our main goal is to investigate different machine-learning methods for text classification using the body of existing literature. The following summarizes the fundamental reasons for which we investigated the text mining study areas:

1. Information Extraction (IE) approach is to identify specific details inside text, which is in line with the fundamental principles of text mining techniques for focused data retrieval.
2. Information retrieval (IR), which covers a range of information processing phases from data to knowledge retrieval, entails sorting through documents using statistical techniques to identify pertinent answers.

There are various sections in this study paper: Section 2 outlines the process of conducting a literature review in relation to document categorization; Section 3 provides the preprocessing stages and methodology for text extraction from various textbooks; and Section 4 explores the outcomes and supporting metric graphs. Finally, the Conclusion and the Opportunities for Further Improvements are covered in Section 5.

2 Literature Review

Due to its many useful applications, document categorization, a critical job in the fields of information retrieval and natural language processing (NLP), has become much more well-known in recent years. This work focuses on classifying documents into pre-established groups or categories, which makes it an essential part of many different fields, such as spam detection, content suggestion, and information organization. A great deal of research has been done to examine the effectiveness of different machine learning and deep learning techniques in the context of document classification [6]. Previous approaches mostly focused on traditional machine learning techniques, like Support Vector Machines (SVM) [7] and Naive Bayes [8], which performed rather well. But as deep learning has become more popular, methods like Multilayer Perceptron(MLP), Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN) have revolutionized the discipline by producing state-of-the-art results [9, 10].

The process of choosing and creating features that best capture the content of documents is a key topic of research in document categorization. Conventional methods depended on handcrafted features such as term frequency-inverse document frequency (TF-IDF) representations and bag-of-words. However, more recently, distributed representations such as Word Embeddings (like Word2Vec and GloVe) have become more popular as a means of more effectively capturing semantic information. Furthermore, pre-trained language models with contextual embeddings that significantly improve accuracy and generalization, such as BERT, have expanded the limits of document categorization [11].

Another important area of study in document categorization is the creation of customized models for managing various document formats, such as text, photos, audio, and multi-modal data. Audio-based models, such as RNNs, are used to identify spoken documents, whereas image classification networks, such as CNNs, have been adapted for tasks involving the classification of

document images. The amalgamation of multimodal data has also acquired impetus, permitting the categorization of documents with a variety of content types. Moreover, the accessibility of extensive and diverse document classification methods has been made easier by the availability of large and varied document datasets [12, 13].

In summary, document categorization is an active study field in the fields of machine learning and natural language processing. Developments in feature engineering, machine learning techniques, and the accessibility of a variety of datasets are the main forces behind this improvement. The discipline is constantly pushing the bounds of precision and application through the incorporation of deep learning approaches, the study of novel characteristics, and model adaptation to suit a variety of document types. Future work is expected to focus on a few learnings, fine-grained document classification, and problems related to organizing large, unstructured document collections.

3 Methodology

In this section, we have presented a model for classifying textbooks. On a smaller dataset, we have found the best accuracy utilizing the supervised machine learning method, the neural network – MLP model. We have trained the model and generated predictions for textbooks in the same category.

3.1 Dataset Creation

We created a unique dataset by acquiring diverse coursebooks through downloading. First, we downloaded study book PDFs and extracted text using Python. The textbooks covered a wide range of topics, including computer science, social sciences, and business, and the content was summarized using the Sumy library and was then treated as a feature. Several records were produced by repeating this method, each labeled with the appropriate subject name. Two columns make up the final dataset: one for the summary feature and one for the subject label. The dataset consists of 243 records and 2 columns. Here is the reference link of the dataset, which is uploaded on the Kaggle account (https://www.kaggle.com/datasets/sindhujapenchala/document-classification-dataset/data).

3.1.1 Summary Extraction

Extracting relevant material from textbooks in a variety of areas is an important part of learning. Our approach makes full analysis and condensing of content possible by using sophisticated text processing tools. We methodically extract important insights using natural language processing methods, then present them in succinct summaries that encapsulate the main ideas and important information. This method simplifies understanding, enabling consumers to comprehend a variety of topics with accessibility and attention. We preserve accuracy and coherence in the extracted summaries by utilizing state-of-the-art technology, providing readers with insightful information about the subject matter without compromising depth or relevance.

SUMY: Unlike abstractive methods, which conceptualize and rephrase a summary, extractive methods for text summarizing choose sections of the text and create a summary from them. Abstractive summarization includes a range of methods, each applied differently according to researchers' methodologies. These techniques include WordNet, lexical chains, graph theory, clustering, and others. Aiming to robustly combine two or more methodologies, some approaches are statistical, some have language roots, and still others are profound. Tokenizing

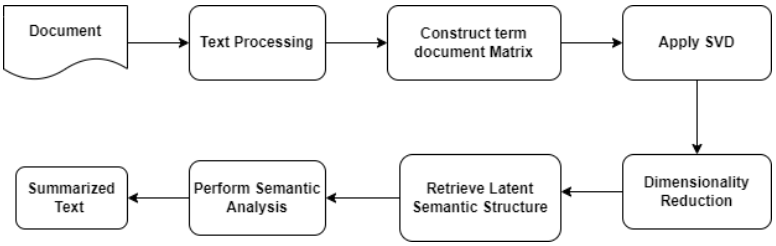

Figure 1: Text Summarization using LSA algorithm.

the content comes after we load the text file, parse it, and import the libraries we need for text preparation.

A range of text-summarizing techniques are available on Sumy, including:

1. Latent Semantic Analysis (LSA) with LSA Summarizer.
2. Kull back-Leibler (KL) with KL Summarizer.
3. Lex Rank with Lex Rank Summarizer.
4. Luhn with Luhn Summarizer.
5. Edmundson with Edmundson Summarizer.

In this study, we used the LSA algorithm for extracting the summary from textbooks.

Latent Semantic Analysis: Latent Semantic Analysis (LSA) is a technique used in information retrieval and natural language processing that aims to reveal the underlying structure in a collection of texts. Reducing dimensionality and doing semantic analysis on large-scale textual corpora are its main uses. The basic idea behind LSA is to capture the implicit correlations between keywords and their environments by representing words and documents in a condensed-dimensional space.

A term-document matrix that shows the relationships between terms (words) and documents in a particular corpus which is created as part of the LSA process. This matrix keeps track of a term's frequency in each individual document in each cell. The matrix then goes through a mathematical process called Single Value Decomposition (SVD), which divides it into three matrices: a term-concept matrix, a singular value matrix, and a concept-document matrix.

The concept-document matrix shows the connections between these latent concepts and the documents, whereas the term-concept matrix outlines the linkages between terms and latent concepts. LSA overcomes issues with high-dimensional data and noise by reducing the dimensionality of the original term-document matrix and locating the most important latent concepts in the dataset. Applications for LSA can be found in a variety of natural language processing tasks, such as text summarization, document clustering, and information retrieval. Based on their semantic significance to the document's overall content, LSA can identify the most important sentences or phrases in a document during text summarization. LSA helps provide more contextually meaningful summaries by capturing the innate semantic structure. Figure 1 illustrates how the LSA algorithm is used to summarize text. Prior to creating the term-document matrix, the text in the document is preprocessed.

Figure 1 shows the steps involved in Latent Semantic Analysis (LSA). First, a corpus of text documents is gathered, and then text preprocessing, which includes tokenization, stop word removal, and punctuation, is carried out. The next step is to create a term-document

matrix, which contains word frequencies or other weightings. This matrix is divided into three parts by applying Singular Value Decomposition (SVD): U (Term-concept matrix), $\sum$ (Singular value matrix), and V^T (Concept-document matrix). This reduces the dimensionality of the representation by keeping the top k singular values. Latent semantic structures are created by utilizing the reduced matrices, which enables semantic analysis to identify links between terms and texts in the lower-dimensional space. Lastly, this representation is used for tasks like text summarization, document grouping, and information retrieval.

3.2 Text Preprocessing

The preparation of text data is a crucial step in any natural language processing project as it helps to prepare and clean the textual data for further analysis or processing. A text preprocessing pipeline is a methodical set of steps that are used on unprocessed text input to prepare it for use in different natural language processing applications.

While the exact steps in a text preprocessing pipeline can differ, they usually include tokenization, stop word removal, stemming, and lemmatization. These procedures are essential for decreasing the amount of data and improving the accuracy of natural language processing (NLP) tasks, including information extraction and text classification. Textual data presents difficulties since it is unstructured and often contains noise, such as typos, grammatical errors, and erratic formatting. Eliminating this noise is the main objective of a text preparation pipeline, which makes text data analysis and understanding simpler. Here, figure 2 illustrates the text classification procedure.

3.2.1 NLTK

The Natural Language Toolkit, or NLTK, is a robust Python application development package for handling natural language data. It provides accessible interfaces for more than 50 lexical resources and corpora, such as WordNet. A variety of text processing libraries covering functions like tokenization, parsing, tagging, categorization, and stemming are also included. Additionally, NLTK offers wrappers for powerful NLP libraries, promoting a lively community.

Tokenization: Tokenization, an essential step in many NLP applications, is the process of dissecting a text into individual words or sentences. Tokenization techniques developed by NLTK are flexible enough to handle a wide range of languages and text formats, making it a valuable tool for processing a wide range of textual data.

Stop words: Stop words are frequently used terms in natural language processing that are filtered out during the text preprocessing stage since they usually don't add anything to the meaning of a phrase. A list of stop words for several languages is provided by NLTK, which can be used to remove these frequently occurring words from text data. A commonly used word (such as "the," "a," "an," or "in") is referred to as a stop word. It is important that these terms be removed from our database to save space and avoid them.

Stemming: By treating many versions of a word as interchangeable, stemming serves to simplify text analysis by reducing words to their root or base form. Many stemming techniques, including the Porter stemming algorithms, which is frequently employed in text mining and information retrieval applications, are incorporated into NLTK.

Lemmatization: Lemmatization is a linguistic process that involves reducing words to their elementary or root form, often known as the lemma. Lemmatization is the process of classifying inflected variants of a word as a single, cohesive base form. This helps standardize word usage and improve accuracy in activities like text mining, information retrieval, and

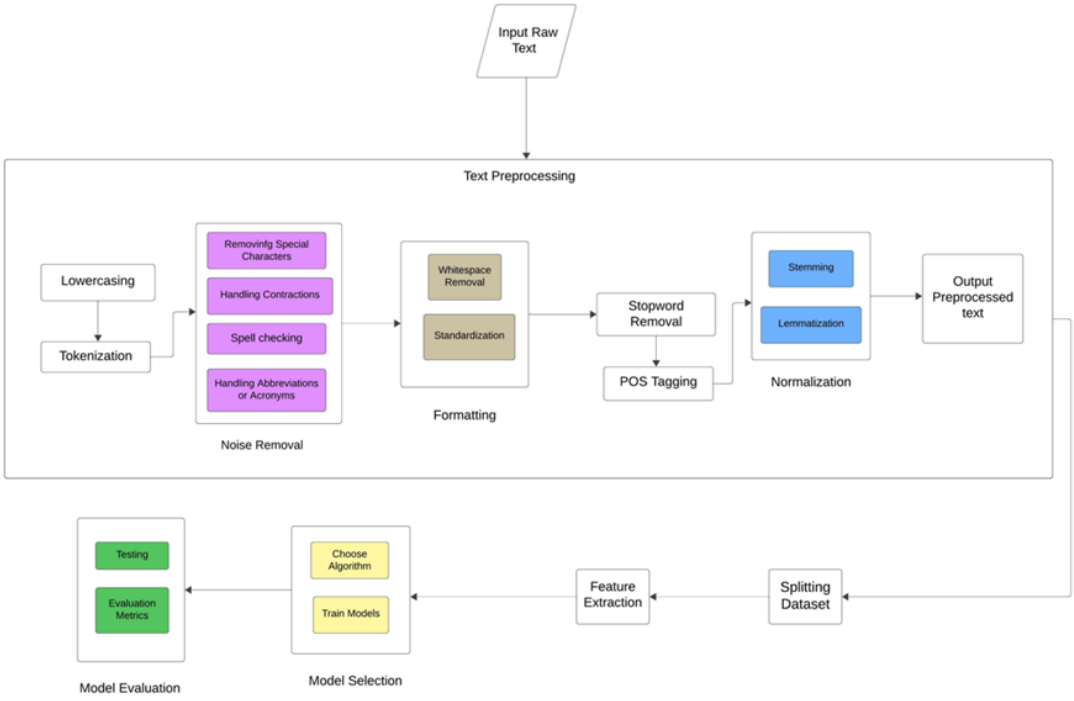

Figure 2: Process of Text Classification.

sentiment analysis. It is especially helpful in natural language processing (NLP) and text analysis.

POS Tagging: Part-of-speech (POS) tagging is a crucial component of natural language processing (NLP) that offers instruments for categorizing words in a text according to their grammatical meaning. Users can easily use part-of-speech tagging techniques to their text data without having to train a model from scratch.

Parsing: NLTK also provides parsing tools that let users examine the grammatical construction of sentences. The library supports several parsing strategies, such as chart parsing and recursive descent parsing. Applications like information extraction and syntactic analysis benefit greatly from these features.

NLTK's working algorithm is designed to be both modular and extensible. Users can tailor their approach to NLP tasks by selecting modules according to their needs. Moreover, NLTK's smooth integration with other Python libraries and tools makes it a top option for NLP practitioners, educators, and academics. In conclusion, NLTK is essential to natural language processing since it offers an extensive collection of text analysis tools and methods. It is a go-to library for Python users dealing with textual data because of its modular design, support for different languages, and range of NLP applications. An excellent tool for raising the potential of Python applications is NLTK.

3.3 Splitting Dataset

This crucial stage guarantees an impartial assessment of the model's functionality. Following a methodical process, eighty percent of the dataset was put aside for training, and the remaining twenty percent was set aside for testing only. The famous 'scikit-learn' machine learning library's 'train_test_split' function was used to split the dataset into separate training and testing sections. By maintaining the natural distribution patterns in the training and testing sets, the stratified splitting technique guarantees a proportionate representation of the various classes within the dataset. By training the model on a significant fraction of the data and rigorously testing its performance on unseen samples, this division technique helps to assess the model's generalizability and promotes resilience and dependability in subsequent analyses and predicting tasks. In this case, we used the random state parameter, which has a fixed value of 123 to ensure consistency between model evaluations.

3.4 Encoding

This study used the 'LabelEncoder' module from the well-known scikit-learn toolkit in Python to do feature encoding. The preprocessing processes applied for category target variables are demonstrated in the code sample. For many machine learning methods that require numerical inputs, this procedure efficiently converts the category labels into numerical representations.

3.5 Feature Extraction

We employed word vectorization using Term Frequency-Inverse Document Frequency (TF-IDF), a fundamental technique in natural language processing (NLP) that converts textual data into numerical vectors while accounting for word significance within a corpus. TF-IDF assigns weights to words according to how frequently they occur in each text and how infrequently they occur across the corpus. Terms that appear frequently are assigned larger weights, signifying their importance, whereas unusual terms receive lower weights due to their rarity in other publications. By encoding the text into a vector space, where each dimension represents a unique word with a value indicating its relevance in each document, this method helps capture the essence of the text. Many natural language processing (NLP) activities, such as text categorization, information retrieval, and sentiment analysis, make extensive use of TF-IDF. By giving textual data in a numerical format that machine learning models can easily interpret and utilize, it enables machines to process and grasp textual data.

3.6 Model Selection and Training

During this stage, the model was chosen and trained using three different classification algorithms: Decision Tree, Random Forest, and Neural Network-MLP [14, 15]. The optimal results are seen for a very small dataset using the Multi-Layer Perceptron Neural Network model. A neural network technique that learns the relationships between linear and non-linear data is called a multilayer perceptron [16].

Figure 3 represents the multilayer perceptron model, which is a feed-forward neural network algorithm. It is divided into three layers: the input layer, the hidden layer, and the output layer.

1. Input Layer: The input layer receives input data and forwards it to the hidden layer. Each input layer node represents a feature or property of the incoming data.

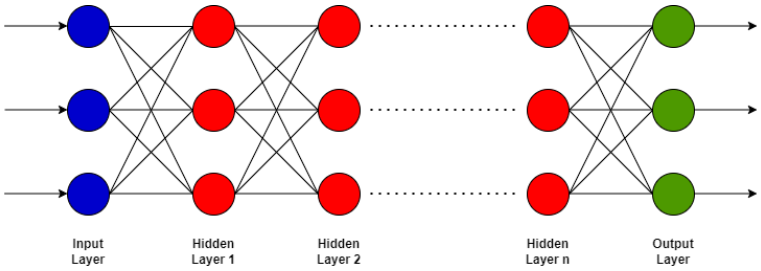

Figure 3: Diagram representing Multilayer Perceptron.

2. Hidden Layer: Hidden layers are intermediate layers that exist between the input and output levels. Each hidden layer is made up of many nodes that use weights and activation functions to compute the input data. Deep neural networks are distinguished by the presence of numerous hidden layers, which distinguishes them as "deep" networks.

3. Output Layer: The final layer is responsible for the network's output. The number of nodes in this layer is determined by the type of problem that the MLP is attempting to solve.

3.7 Model Evaluation

The trained model was used for the testing subset to predict labels based on the learned patterns for evaluation. To ensure consistency and model compatibility, the textual features of the testing subset were converted using the same preprocessing techniques as the training data. The predictive performance and generalizability of the model were evaluated by generating predictions using the prediction function and comparing them to the real labels. This evaluation step aims to assess the model's accuracy in classifying text data as well as its potential for real-world applications. Among the three algorithms used, the neural network MLP achieved a notable high accuracy of 91.83%, while the decision tree achieved a lesser accuracy of 71.42%. Random Forest, on the other hand, achieved an accuracy of 85.71%. The ensuing results for accuracy, precision, and recall are as follows:

Accuracy: The frequency with which a classification machine learning model is correct in its overall predictions are indicated by the accuracy.

$$Accuracy = \frac{TP + TN}{TP + TN.FP + FN} \tag{1}$$

Precision: Precision is the frequency with which an ML model is correct when predicting the intended target class.

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

Recall: The ability of an ML model to recognize all instances belonging to the target class is measured by the recall.

$$Recall = \frac{TP}{TP + TN} \tag{3}$$

Table 1: Performance Matrices for Three Different Models.

	Accuracy	Precision	Recall
Decision Tree	71.42	0.72	0.71
Random Forest	85.71	0.86	0.86
Neural Network	91.83	0.92	0.92

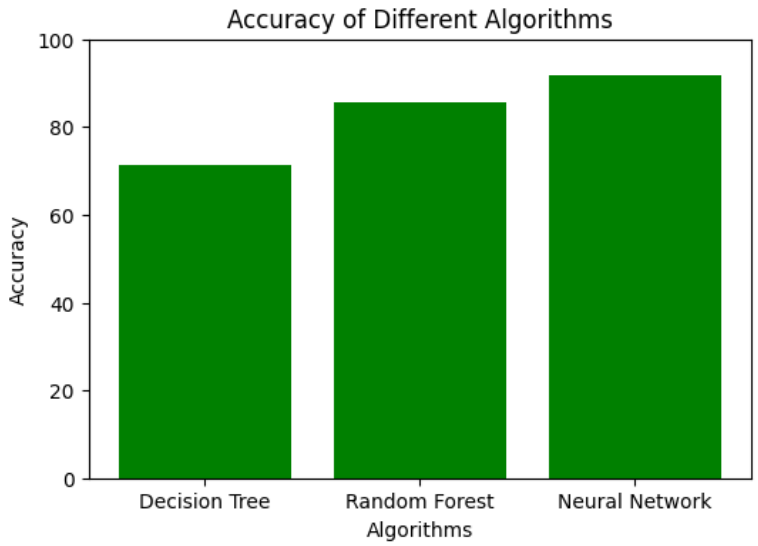

Figure 4: Bar plot displaying the performance of Accuracy for Decision Tree, Random Forest, and Neural Network algorithms.

4 Result

In this research paper, we computed various metrics, which include accuracy, precision, and recall, and they are represented in graphs.

Table 1 displays performance characteristics for three different machine learning algorithms: decision trees, random forests, and neural networks [17]. These metrics indicate each algorithm's performance in its respective classification challenge. The Decision Tree algorithm obtained 71.42% accuracy, 72% precision, and 71% recall. The Random Forest algorithm, on the other hand, performed better, with an accuracy of 85.71% and precision and recall of 86%. The Neural Network-MLP method outperformed both, with the greatest accuracy at 91.83% and precision and recall scores of 92% [18]. These measurements provide insights into these algorithms' predictive powers and robustness in classification tasks, emphasizing the neural network's superior performance based on these evaluation criteria [19].

Graph of Accuracy: Figure 4 depicts the accuracy performance of various algorithms, including the Decision Tree, Random Forest, and Neural Network. The Decision Tree algorithm has a lesser accuracy of 71.42% than the other two algorithms. Random Forest scores 85.71%, while Neural Network tops with an accuracy of 91.83%.

Graph of Precision: Figure 5 illustrates a bar plot of the precision values of various algorithms (Decision Tree, Random Forest, and Neural Network). The precision score for the neural network was notable at 0.92, whereas the precision of the decision tree remained relatively

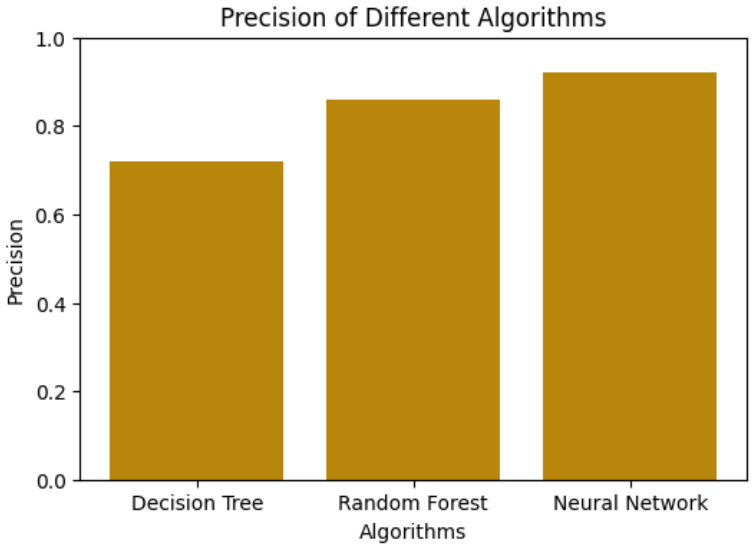

Figure 5: Bar plot displaying the performance of Precision for Decision Tree, Random Forest, and Neural Network algorithms.

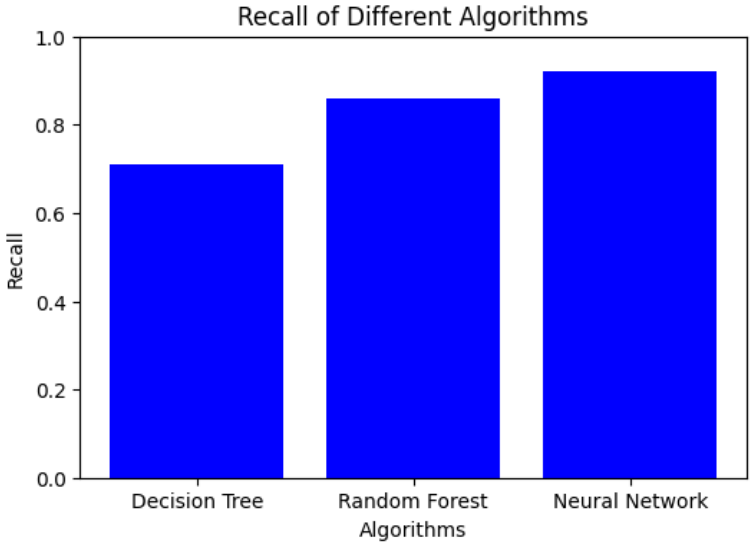

Figure 6: Bar plot displaying the performance of Recall for Decision Tree, Random Forest, and Neural Network algorithms.

low at 0.72. The precision value achieved by Random Forest was 0.86.

Graph of Recall: Figure 6 depicts the recall metrics of distinct machine learning methods, including decision trees, random forests, and neural networks. The recall score for the neural network was 0.92; the recall score for the decision tree was considerably lower at 0.71; and the recall value for the random forest was 0.86.

5 Conclusion

Text mining, machine learning, and natural language processing techniques and approaches must be used to efficiently organize and extract patterns and knowledge from texts as the use of textual data grows. This review focused on current research, diving into approaches for document representation and feature selection analysis. Data cleansing and summarizing were accomplished using Python programs. To some extent, the Neural Network approach outperforms other text classification algorithms, with 91.83 percent accuracy. More research is needed to pursue concept-based or semantic document representations. Consider semantic representations to achieve improved categorization results. Given the exponential expansion of digital documents over time, text classification problems find application in a variety of domains. To produce more accurate and effective classification results. Categorization algorithms are critical for properly managing this massive volume of documents.

In the future, our goal is to expand the dataset and improve document classification approaches by employing complex NLP algorithms and deep learning models to produce more exact and effective classification results.

References

[1] J. Voerman, I. S. Mahamoud, M. Coustaty, A. Joseph, V. P. d'Andecy, J.-M. Ogier, Automatic classification of company's document stream: comparison of two solutions, Pattern Recognition Letters (2023).

[2] J. Xiong, L. Yu, X. Niu, Y. Leng, Xrr: Extreme multi-label text classification with candidate retrieving and deep ranking, Information Sciences 622 (2023) 115–132.

[3] S. A. Murad, A. Adhikary, A. J. M. Muzahid, M. M. H. Sarker, M. A. R. Khan, M. B. Hossain, A. K. Bairagi, M. Masud, M. Kowsher, Ai powered asthma prediction towards treatment formulation: An android app approach, Intelligent Automation & Soft Computing 34 (1) (2022) 87–103.

[4] S. A. Murad, Z. R. M. Azmi, Z. H. Hakami, N. J. Prottasha, M. Kowsher, Computer-aided system for extending the performance of diabetes analysis and prediction, in: 2021 International Conference on Software Engineering & Computer Systems and 4th International Conference on Computational Science and Information Management (ICSECS-ICOCSIM), IEEE, 2021, pp. 465–470.

[5] H. Broome, Y. Shrestha, N. Harrison, N. Rahimi, Sms malware detection: A machine learning approach, in: 2022 International Conference on Computational Science and Computational Intelligence (CSCI), IEEE, 2022, pp. 936–941.

[6] K. Sharifani, M. Amini, Machine learning and deep learning: A review of methods and applications, World Information Technology and Engineering Journal 10 (07) (2023) 3897–3904.

[7] A. Adhikary, S. A. Murad, M. S. Munir, C. S. Hong, Edge assisted crime prediction and evaluation framework for machine learning algorithms, in: 2022 International Conference on Information Networking (ICOIN), IEEE, 2022, pp. 417–422.

[8] A. Gautam, N. Rahimi, Viability of machine learning in android scareware detection, Proceedings of 38th International Confer 91 (2023) 19–26.

[9] L. Waikhom, R. Patgiri, A survey of graph neural networks in various learning paradigms: methods, applications, and challenges, Artificial Intelligence Review 56 (7) (2023) 6295–6364.

[10] S. Albawi, T. A. Mohammed, S. Al-Zawi, Understanding of a convolutional neural network, in: 2017 international conference on engineering and technology (ICET), Ieee, 2017, pp. 1–6.

[11] X. Jiang, M. Ringwald, J. Blake, H. Shatkay, Effective biomedical document classification for identifying publications relevant to the mouse gene expression database (gxd), Database 2020 (2020) baaa043.

[12] D. Jatnika, M. A. Bijaksana, A. A. Suryani, Word2vec model analysis for semantic similarities in english words, Procedia Computer Science 157 (2019) 160–167.

[13] A. Laleh, R. Shahram, Analyzing facebook activities for personality recognition, in: 2017 16th IEEE international conference on machine learning and applications (ICMLA), IEEE, 2017, pp. 960–964.

[14] O. I. Abiodun, A. Jantan, A. E. Omolara, K. V. Dada, N. A. Mohamed, H. Arshad, State-of-the-art in artificial neural network applications: A survey, Heliyon 4 (11) (2018).

[15] Y. Jang, K. Won, H.-d. Choi, S. Y. Shin, Classification of research papers on radio frequency electromagnetic field (rf-emf) using graph neural networks (gnn), Applied Sciences 13 (7) (2023) 4614.

[16] Z. Wang, Y. Cai, D. Liu, F. Qiu, F. Sun, Y. Zhou, Intelligent classification of coal structure using multinomial logistic regression, random forest and fully connected neural network with multisource geophysical logging data, International Journal of Coal Geology 268 (2023) 104208.

[17] A. M. Rinaldi, C. Russo, C. Tommasino, A semantic approach for document classification using deep neural networks and multimedia knowledge graph, Expert Systems with Applications 169 (2021) 114320.

[18] Q. Li, H. Peng, J. Li, C. Xia, R. Yang, L. Sun, P. S. Yu, L. He, A survey on text classification: From traditional to deep learning, ACM Transactions on Intelligent Systems and Technology (TIST) 13 (2) (2022) 1–41.

[19] R. Haque, N. Islam, M. Tasneem, A. K. Das, Multi-class sentiment classification on bengali social media comments using machine learning, International Journal of Cognitive Computing in Engineering 4 (2023) 21–35.