Journal of Big Data

METHODOLOGY **Open Access**

A hybrid Hadoop-based sentiment analysis classifier for tweets associated with COVID-19 utilizing two machine learning algorithms: CNN, and fuzzy C4.5

Fatima Es-sabery[1,2*†], Ibrahim Es-sabery[1†], Junaid Qadir[3†], Beatriz Sainz-de-Abajo[4†] and Begonya Garcia-Zapirain[5†]

†Fatima Es-sabery, Ibrahim Es-sabery, Junaid Qadir, Beatriz Sainz-de-Abajo and Begonya Garcia-Zapirain have contributed equally to this work.

*Correspondence: fatima.essabery@gmail.com

[1] Department of computer science, Sultan Moulay Slimane University, 23000 Beni Mellal, Morocco
[2] Department of Economics and Management Sciences, Hassan II University of Casablanca, 28830 Mohammedia, Morocco
[3] Department of Electrical, Electronic, and Telecommunications Engineering and Naval Architecture, University of Genoa, 16145 Genoa, Italy
[4] Department of Signal Theory, Communications, and Telematics Engineering, Universidad de Valladolid, 47011 Valladolid, Spain
[5] eVIDA Research Group, University of Deusto, 48007 Bilbao, Spain

Abstract
In recent years, research on opinion mining from X (formerly Twitter) has rapidly advanced, focusing on processing tweets to determine user sentiments about events. Many researchers prefer using machine and deep learning techniques for this analysis. This work proposes a novel approach integrating the C4.5 procedure, fuzzy rule patterns, and convolutional neural networks. The approach involves six steps: preprocessing to remove noisy data, vectorizing tweets with word embedding, extracting sentiment and contextual features using convolutional neural networks, fuzzifying outputs with a Gaussian fuzzifier to handle ambiguity, constructing a fuzzy tree and rule base using a fuzzy version of C4.5, and classifying tweets with fuzzy General Reasoning. This method combines the benefits of convolutional neural networks and C4.5 while addressing imprecise data with fuzzy logic. Implemented on a Hadoop framework-based cluster with five computing units, the approach was extensively tested. The results showed that the model performs exceptionally well on the COVID-19_Sentiments dataset, surpassing other classification algorithms with a precision rate of 94.56%, false-negative rate of 5.28%, classification rate of 95.15%, F1-score of 94.63%, kappa statistic of 95.12%, execution time of 11.81 s, false-positive rate of 4.26%, error rate of 4.26%, specificity of 95.74%, recall of 94.72%, stability with a mean deviation standard of 0.09%, convergence starting around the 75th round, and significantly reduced complexity in terms of time and space.

Keywords: Fuzzy version of C4.5 procedure, Convolutional neural network, Fuzzy rule pattern, Hadoop framework, X opinion mining, Sentiment analysis

Introduction

Communication is inherent to human nature, playing a pivotal role throughout history in fostering problem-solving, social responsibility, and societal participation. However, modern communication has evolved significantly with the rise of digital networks, becoming the primary mode of interaction across all societal segments [1]. Social networking platforms such as Instagram, TikTok, Facebook, WhatsApp, X, and YouTube dominate the digital landscape. Among these, the X platform stands out for enabling users to share short, immediate posts, fostering dynamic interactions. Individual users express their opinions, emotions, and ideas about products, events, and services, while organizations analyze these posts to inform decision-making-a process known as X opinion mining or X sentiment analysis [2].

The vast data generated on X holds substantial value for a range of domains, including economics, commerce, social issues, governance, and politics [3]. However, manually extracting insights from such data is impractical due to its sheer volume. As a result, automated X opinion mining has emerged as an effective solution, allowing for the analysis of user sentiment, opinions, and feedback on specific topics. This process, a core application of Natural Language Processing (NLP), involves identifying the sentiment expressed in tweets, whether at the sentence, feature, or document level. Our study focuses on sentence-level sentiment analysis [4]. X sentiment analysis has proven valuable in applications such as refining marketing strategies, improving educational tools, predicting stock market trends, and gauging political sentiment during elections [5].

Recent scholarly work in NLP has approached X sentiment analysis through key tasks: data collection, preprocessing, representation, feature extraction, and classification. While these technical aspects are vital, their detailed descriptions will be further discussed in the methodology and literature review sections. Our focus in the introduction remains on highlighting the importance of Twitter (X) as a platform for sentiment analysis and its broader applications across various fields.

Previous studies have explored various approaches to sentiment analysis on Twitter, employing machine learning and natural language processing (NLP) techniques. For instance, Marchenda et al. [6] introduced one of the earliest works utilizing machine learning algorithms such as Naive Bayes and SVM for sentiment classification of tweets, demonstrating that automated methods could effectively analyze large-scale social media data. Similarly, Aslan et al. [7] applied deep learning models to extract features from tweets and achieved promising results in sentiment prediction. These foundational works laid the groundwork for sentiment analysis using traditional machine learning approaches, but did not incorporate fuzzy logic, which offers potential advantages in handling uncertain or ambiguous data.

In more recent developments, hybrid approaches combining fuzzy logic and machine learning, such as the work by Wang et al. [8], have shown improved accuracy in sentiment classification by handling the inherent uncertainty of language in social media posts. However, their focus was primarily on long-form text, leaving room for further exploration in short, informal posts typical of Twitter. Our study builds on these findings by integrating fuzzy logic with convolutional neural networks (CNNs) to improve sentiment analysis performance on short-form Twitter data, a gap not extensively addressed in the existing literature.

Despite advancements in sentiment analysis, existing methods often struggle with handling large-scale datasets, capturing long-range dependencies in text, and managing ambiguous or fuzzy sentiment data. Traditional models face limitations in scalability and precision, while newer approaches lack effective integration of feature extraction and ambiguity management. To address these challenges, we propose a hybrid approach that leverages convolutional neural networks (CNNs), Fuzzy C4.5, and the Hadoop framework. The CNN component is employed for its advanced feature extraction capabilities, handling large-scale social media data effectively. Fuzzy C4.5 is integrated to manage ambiguous and uncertain sentiments through fuzzy logic, providing a nuanced understanding of sentiment expressions. The Hadoop framework is utilized to ensure scalability and efficient data processing, enabling our system to manage vast amounts of data seamlessly.

In this paper, we integrate the strengths of machine learning and deep learning to address their respective limitations. While machine learning methods excel in smaller datasets and are less prone to overfitting, deep learning models are superior in extracting more accurate features from larger datasets. To leverage these advantages, we employ convolutional neural networks (CNNs) for precise feature extraction from tweets. We use Fuzzy C4.5 as the classifier and incorporate a rule-based fuzzy system to handle imprecision and ambiguity in the data. To manage the growing volume of data efficiently, our approach utilizes a Hadoop cluster comprising one master node and four subordinate nodes, which facilitates implementation, development, and storage. Our proposed method aims to enhance opinion mining by overcoming previous shortcomings. We estimate the sentiment score for each tweet (negative, neutral, positive) and evaluate the performance of our hybrid framework against recent scientific approaches. We compare results across various metrics, including complexity, stability, convergence, precision, false-negative rate, classification rate, F1-score, kappa statistic, execution time, false-positive rate, error rate, specificity, and recall. The proposed approach demonstrates superior sentiment classification performance compared to other algorithms, affirming its effectiveness and value in addressing sentiment analysis challenges. *To guide our research, we pose the following key questions:*

1. Research Question 1: How can convolutional neural networks (CNN) be effectively utilized to extract and classify sentiments from tweets related to COVID-19?
2. Research Question 2: What role does the Fuzzy C4.5 algorithm play in enhancing the accuracy of sentiment classification when combined with CNN?
3. Research Question 3: How do the hybrid model's performance metrics compare to those of traditional machine learning models in sentiment analysis tasks?

The major contributions of this study are summarized as follows:

1. We develop a novel hybrid sentiment analysis model that integrates CNN for feature extraction and Fuzzy C4.5 for classification, specifically tailored for analyzing COVID-19-related tweets.
2. We demonstrate that the hybrid CNN-Fuzzy C4.5 model outperforms traditional machine learning models in terms of accuracy, precision, and recall, particularly in handling ambiguous and uncertain data.
3. We provide a detailed analysis of the impact of feature selection and parameter tuning on the performance of the proposed hybrid model, offering valuable insights for future research in the field.

The remainder of this investigation is organized as follows: Section 2 covers relevant literature, while Section 3 elucidates the fundamental principles behind convolutional neural networks, fuzzy C4.5, and fuzzy-rule pattern. In Section 4, we outline our research approach. Subsequently, Section 5 showcases the empirical findings and comparative outcomes. In conclusion, Section 6 encapsulates the conclusions drawn and outlines directions for future work.

Previous research

Within this section, we aim to deliver a succinct summary of groundbreaking research studies extracted from the literature. This encompasses a range of methods, including machine learning, deep learning, and fuzzy approaches, all applied to sentiment analysis.

Utilizing machine learning techniques to conduct opinion mining

In supervised learning approaches, tweets' sentiments are classified using the labeled tweets related to the given dataset. The overall process is to divine tweet' sentiments using the constructed classification model by applying diverse machine learning approaches, which are trained on the used tweet' dataset after carrying out the feature engineering extraction properly.

A comparative study conducted by Kanakaraddi et al. [9] examined several supervised algorithms for sentiment analysis, including support vector machine (SVM), random forest (RF), max entropy (ME) and naive Bayes (NB). Out of these approaches, the SVM demonstrated the highest classification rate, reaching an impressive 79.90%. In the research paper [10], the authors explored the application of five distinct machine learning approaches to analyze a movie review dataset. The supervised classifiers utilized in this study including multinomial NB, decision tree (DT), Bernoulli NB, SVM and ME. The results obtained from the experiments revealed that multinomial NB achieved commendable achievement when considering the precision (92.94%), F-score (87.87%), and accuracy (88.5%). On the other hand, the SVM demonstrated superior achievement when considering the recall (89.33%). In the work of ref. [11], the authors analyzed customer reviews of restaurants by employing various supervised algorithms, including SVM, RF, NB, k-nearest neighbor (KNN) and DT. Through their simulations, they discovered that the SVM classifier attained a superior accuracy of 94.56% for the given dataset compared to the other approaches.

Noor et al. [12] have implemented the support vector machine classifier on Urdu Roman reviews extracted from the e-commerce website of Pakistan-Daraz.pk. The collected review dataset contains 20 286 reviews labeled into positive, neutral, or negative class labels. They also used a bag-of-words approach to identify the relevant features from the dataset before passing it to the support vector machine classifier. The C4.5 procedure was employed by the author of the work [13] as a classification method to assign each sentence in the English dataset to one of the class labels: positive, negative, or neutral. The used dataset in their work consists of 140 000 English sentences. The classification performance reached 60.3% of the accuracy. According to ref. [14], the authors have used continuous naïve Bayes supervised learning algorithm for classifying the product reviews collected from a multi-domain and large-scale e-commerce platform. They have kept the high computational effectiveness of the conventional naïve Bayes. They have extended only the estimation parameter in the traditional algorithm to a continuous learning technique. As described by ref. [15], the study presents an innovative algorithm that combines the gradient boosting supervised algorithm ad harmony search-based technique. The major feature of applying the hybrid suggested approach is its fast rate of convergence. The experimental findings showcase a 93% accuracy achievement for the proposed hybrid approach.

The study detailed in ref. [16], proposes a novel approach for sentiment analysis of Arabic-language tweets that integrates machine learning with emoji-based features. They created an Emo-SL using a corpus of 58,000 tweets containing emojis and computed opinion scores for 222 frequently used emojis based on their distribution across negative and positive classes. Their findings demonstrated that their approach outperformed the VADER opinion analyzer by 26.7% in terms of accuracy. As reported in [17], an Arabic opinion mining system was developed to investigate user satisfaction with apps designed for COVID-19. A benchmark dataset containing 1,144,999 comments for 18 apps was provided. Six algorithms were implemented and evaluated: Support Vector Machine (SVM), K-Nearest Neighbor (KNN), Naive Bayes (NB), Logistic Regression (LR), Random Forest (RF), and Artificial Neural Network (ANN). The experimental results show that the ANN model achieved the best performance with 89% accuracy, and an F1 score of 89.71%. We have summarized in Table 1 all the characteristics of each cited machine learning technique including dataset type, sentiment analysis goal, performance metrics, strengths, and limitations.

Conventional machine learning techniques like SVM, k-NN, and decision trees face several research gaps in opinion mining, particularly in handling massive datasets, extracting long-range dependencies, and managing fuzzy sets. These models struggle with scalability and efficiency when processing large-scale social media data and lack the capability to capture contextual relationships between non-adjacent words, which are crucial for accurate sentiment analysis. Additionally, they are not well-equipped to handle ambiguous or uncertain sentiments due to their rigid classification boundaries, highlighting a need for enhanced algorithms or hybrid models that can incorporate fuzzy logic to better manage the nuances of human language.

Utilizing deep learning techniques to conduct opinion mining

In the area of opinion mining, most of the new deep neural network-based works have been focused on identifying term representation (word embedding) or employing diverse kinds of deep neural networks for clustering or classification tasks. Term embeddings approaches are trained to detect and compute term similarities and analyze their lexical relevance [18]. Among the various techniques for word embeddings, Word2Vec [19], FastText [20], and GloVe [21] have emerged as the most popular methods. These techniques compute the term embeddings vectors of each term based on its contexts. In other words, These methods assume that the terms in the same contexts have the same meanings. Therefore, they should vectorize with the same embedding vectors. The Word2Vec, FastText, and GloVe methods adopt an algorithmic structure that relies on the utilization of deep neural networks.

Within the realm of literature, there have been numerous investigations that have been conducted to perform sentiment classification tasks using deep neural networks. For

Table 1 Characteristics of each cited machine learning technique including dataset type, sentiment analysis goal, performance metrics, strengths, and limitations

Model	Dataset type	Sentiment analysis goal	Performance metrics	Strengths	Limitations
[9]	Movie review	Classification of review as positive, and negative	Accuracy of 80.17%, with a precision of 78.91%, recall of 82.33%, and an F-score of 80.58%	This approach is adaptable and efficient	Limited generalization to new data.
[10]	Restaurant Reviews	Classification of review as positive, negative, or neutral	Accuracy 94.56%	Meets key customer insight needs	Insufficient feature engineering details
[11]	Urdu Roman reviews dataset	Class label: Positive, negative, or neutral	Accuracy of 60.03%	Enhances e-commerce customer experience	SVM struggles with scalability
[12]	English training dataset	Class label: positive, and negative	Accuracy of 60.3%	Generates polarity association rules	C4.5 struggles with scalability
[13]	Amazon product and movie review datasets	Binary sentiment classifcation	Accuracy of 75.68%	Adapts to evolving datasets	Assumes feature independence inaccurately
[14]	Amazon product review	Binary classifcation	Accuracy 92.4%, Recall 92.31%, Precision 92.58% and F-measure 92.37%	Enhanced performance through optimization	Gradient boosting can overfit
[15]	Ohsumed, News-Group, Reuters and BBC datasets	Multiclass classification	F1-Micro score	Adaptable with lightweight models	May miss subtle nuances
[16]	58K Arabic tweets	Binary sentiment classifcation	Improves accuracy by 26.7% and achieves 89% F1 score	Emoji features boost accuracy	Preprocessing is complex and intensive
[17]	114,499 reviews from 18 Arabic COVID-19 Apps	Binary classifcation	89% accuracy and 89% F1.	Large dataset for analysis	Focuses on negative factors

example: The research conducted by ref. [22] discusses the use of a CNN to classify the X dataset, leveraging its capacity to effectively capture, identify, and extract overarching features by leveraging the linguistic and lexical relationships among these features. In the work of ref. [23], the authors in their pursuit of improvement, integrated a multi-Head attention technique with the CNN. The authors' proposed approach involves the integration of features to generate diverse feature channels, followed by the implementation of a multi-channel CNN for the detection of opinion words from multiple perspectives. Subsequently, the authors employed the multi-head attention approach to educe relevant characteristics from a wide range of dimensions. The practical results substantiate that the suggested method achieved the utmost classification accuracy, reaching 86.32%, when utilizing eight heads in the mechanism of multiple attention.

In ref. [24], it is detailed that the authors developed a new hybrid method by incorporating two models from the field of deep learning, specifically LSTM and CNN, to conduct opinion mining on reviews posted across various domains. It attained an accuracy rate of 83.13% when evaluated on a dataset consisting of movie reviews. To enhance the efficacy in dealing with short texts, The findings of ref. [25] show that the authors implemented a neural network algorithm based on bi-directional LSTM that enables fine-grained opinion mining of individual words. In the first step their proposed method divines the opinion data in terms of the smallest unit's arousal and valence rates. Later, it combines the extracted features into a bi-directional LSTM deep learning algorithm to analyze the entire text's sentiment.

In the paper authored by ref. [26], the research highlights that the authors devised a novel hybrid approach that used the GloVe technique as word vectorization method. Then, it applied the attention technique to the outcomes of bidirectional GRU and bidirectional LSTM neural network patterns to construct a hybrid deep learning model that gives greater or lesser consideration to diverse words and sentences. The analysis presented by ref. [27] reveals that the authors analyzed the sentiment classification performance using GRU, LSTM, and RNN. The empirical results provide evidence that the GRU attained the highest level of accuracy rate of 83.90%. In the research work [28], an innovative approach was suggested, which combines a convolutional neural network for extracting the features specific to the local context and a gated recurrent unit for learning the long-term dependency.

Based on the findings in ref. [29], the authors examine an proficient opinion mining technique that was developed for X data, with classification carried out using a modified deep learning neural network. The empirical findings were evaluated using evaluation metrics such as F-score, accuracy, precision, and recall. The results showed that the proposed approach outperformed ANN, K-Means, SVM, and DCNN, achieving a precision of 95.78%, recall of 95.84%, F-score of 95.87%, and accuracy of 91.65%.

In this research [30], thirty-six different deep learning models using Word2Vec and GloVe embeddings were developed based on various deep learning architectures, including Multilayer Perceptron (MLP), convolutional neural network (CNN), Long Short-Term Memory Network (LSTM), Bi-directional LSTM (Bi-LSTM), Gated Recurrent Unit (GRU), and Bi-directional GRU (Bi-GRU). The performance of all the developed models was compared using accuracy, F1-scores, and other evaluation metrics. The study produced several promising outcomes, with the Word2Vec embedded Bi-directional GRU model achieving the best performance, obtaining an average F1-score of 0.84 with a train-test split of 80:20. This comprehensive and detailed comparative analysis is unique and could be valuable for developing expert sentiment analysis tools.

According to ref. [31], the authors investigate an intelligent system for identifying Arabic COVID-19 text, named AraCovTexFinder. This approach aims to identify text related to COVID-19, a crucial step for detecting misinformation and harmful content. This is essential for effectively retrieving and processing vital information needed by policymakers and health authorities. AraCovTexFinder attains an impressive accuracy of 98.89 $\pm$ 0.001%, surpassing other baseline models, including those based on transformer architectures and deep learning techniques. A novel deep learning-based pattern called CovTiNet [32] has been introduced for extracting COVID-19 text in Bengali. This method combines an attention-based CNN for identifying COVID-19 text with attention-based positional embeddings for enhanced text-to-feature representation. Empirical results show that CovTiNet achieves a superior accuracy of 96.61 $\pm$ 0.001% on the BCovC dataset, outperforming other methods and baseline approaches.

We have summarized in Table 2 all the characteristics of each cited deep learning technique including dataset type, sentiment analysis goal, performance metrics, strengths, and limitations.

Deep learning techniques for opinion mining face several gaps: they require significant computational resources to handle massive datasets, struggle with capturing long-range dependencies due to issues like vanishing gradients, and lack effective methods for managing ambiguous or fuzzy sentiment data. There is a need for advanced models, such as those combining deep learning with fuzzy logic or attention mechanisms, to address these challenges and improve sentiment analysis.

Utilizing fuzzy approaches for the purpose of sentiment analysis

fuzzy logic theory aids us in handling the uncertainty and ambiguity sentiment data by giving us decision-making abilities in the existence of vagueness in the data under analysis. Within the existing body of literature, there are research works that conducted sentiment classification using principles derived from fuzzy logic theory, as an

Table 2 Characteristics of each cited deep learning technique including dataset type, sentiment analysis goal, performance metrics, strengths, and limitations

Model	Dataset type	Sentiment analysis goal	Performance metrics	Strengths	Limitations
[18]	30Music, Deezer, Twitter etc.	Recommendation Systems	Unconstrained Hyperparameter Optimization	Balanced performance for scalability	Impractical for large-scale systems
[19]	KBBI dataset	Dimensionality reduction	PCA and t-SNE	Valuable for understudied languages	Challenges in model interpretability
[20]	IMDB dataset	Binary sentiment classifcation	Accuracy of 83.90% and F1-score of 83.80%	Effective high-quality word embeddings	Limited by co-occurrence matrix
[21]	MR and Gold Dataset	Binary sentiment classifcation	Accuracy of 68%	Effective deep learning application	Risk of model overfitting
[22]	Chinese dataset	Binary sentiment classifcation	Accuracy of 86.32%	Rich feature combination integration	Increased model complexity challenges
[23]	Consumer dataset	Binary sentiment classifcation	Accuracy	Handles diverse domains effectively	Performance relies on data
[24]	Competition dataset	Binary sentiment classifcation	MAE and PCC	Predicts VA values for emotions	Bias may affect accuracy
[25]	Sentiment140 dataset	Multiclass classification	Recall of 90.88%, Precision of 95.70%, F1 of 93.22% and Accuracy 93.40%	Handles diverse data sources	High training costs and resources
[26]	SST-1 and SST-2 datasets	Multiclass classification	Accuracy of 83.90% and 81.10%	Effective for sequential data	Poor performance on the Amazon dataset.
[27]	Chinese product reviews	Multiclass classification	AUC, F1 and Accuracy	Strong domain generalization ability	Relies on data quality
[28]	IMDB and hotel reviews dataset	Binary sentiment classification	Accuracy of 65.45%	Fuzzy approach outperforms others	Depends on lexicon quality

illustration: The research conducted by ref. [33] discusses that the authors put forward a novel method by combining fuzzy logic word-level method, lexicon SentiWordNet-based approach, and lexicon AFINN-based technique for discovering and learning the sentiment polarity of online reviews. Maheswari et al. [34] came up with an innovative method that combines fuzzy logic and natural language processing techniques to conduct an analysis of customer's difficulties in purchasing quality products. As described by ref. [35], the study suggests to employ fuzzy rule-based systems as methods of computation to achieve accurate and interpret-able sentiment classification. The experimental findings demonstrate that the procedure based on fuzzy rules operates more effectively than the conventional machine learning methods while decreasing the time and space complexity and raising the interpretability.

The work referenced in ref. [36] introduced a novel hybrid method which combines the strengths of C4.5 for automated discovery of characteristic engineering from the provided data, and fuzzy logic theory for effectively handling lack of clarity and vagueness in the examined data. This integrated approach aims to provide the user with a more accurate sentiment forecast. Based on the findings in ref. [37], the authors presented an innovative fuzzy CNN approach that merges the principles of fuzzy logic theory and CNN. This integration leverages the advantages of both approaches, allowing for the extraction of meaningful data's local and global features characterized by imprecision and uncertainty. As described by ref. [38], the study reveals a hybrid approach for sentiment classification that merges fuzzy logic theory with a multilayer perceptron back-propagation network. In this method, the input online reviews are transformed into fuzzy sets using the Gaussian fuzzification technique, resulting in a fuzzification matrix. The generated matrix is subsequently inverted and fed into the implemented multilayer perceptron back-propagation network within the system.

As reported in ref. [39], the study introduces a novel classifier that combines the naive Bayes machine learning algorithm and fuzzy logic classifier for performing sentiment analysis. On analysis of their experimental results, they have proved that the proposed hybrid classifier provides better sentiment classification performance. The findings of ref. [40] show that the authors introduces a robust hybrid classifier that incorporates the support vector machine and fuzzy domain ontology to establish an automated procedure for extracting and identifying the semantic information and feature from the given reviews. Then apply the learning process to assign each review to class label of positive, neutral, or negative.

The proposed model in the study [41] offer improved representation for words with multiple meanings, leading to more precise sentiment analysis outcomes. Furthermore, the authors conducted a comparison between multi-sense word embedding and single embedding models, assessing the classification methods against classical machine learning technologies. In the final stage, a fuzzy system was employed to estimate concealed topics within a drug review dataset using the Latent Dirichlet Allocation (LDA) model. Additionally, a fuzzy rule-based system was utilized to elucidate the classification results of drug review polarity. Notably, both models demonstrated commendable performance in the classification task. Probabilistic Fast text achieved an accuracy of 82.1%, while multi-sense skip-gram achieved an accuracy of 79.8%.

The analysis presented by ref. [42] reveals that the authors suggested a hybrid fuzzy graph convolutional network model by combining fuzzy logic theory and graph convolution network to decrease the ambiguities of opinion in data. Also, in this model, they used the BERT+BiLSTM to convert each word into a numerical vector. Its obtained results show that the model is more complicated with millions of parameters, creating it challenging to interpret and implement in certain environments.

The research conducted by ref. [43] discusses an Opinion Mining (OM) model tailored for X to effectively address perspectivism and opinion ambiguity. To tackle perspectivism, they employ Social Network Analysis (SNA), where users undergo ranking and weighting using the UCINET tool and neural networks. An uncertainty classifier is implemented to amalgamate users' influence levels with the polarity scores of their texts, yielding a refined polarity score that better captures real-world reasoning of opinions. The necessary polarity scores for integration are generated using the TextBlob lexicon resource. The study evaluates three uncertainty classifiers in the proposed model: type1 fuzzy logic (T1-FL), type2 fuzzy logic (T2-FL), and neutrosophic logic (NL). A comparative analysis reveals the superior accuracy of NL in handling uncertainty within the data, highlighting the effectiveness of NL in enhancing the capability of OM in the context of social media.

We have summarized in Table 3 all the characteristics of each cited fuzzy logic technique including dataset type, sentiment analysis goal, performance metrics, strengths, and limitations.

Utilizing fuzzy approaches for sentiment analysis faces several gaps, including challenges with scalability due to computational intensity when processing large datasets, and difficulties in effectively capturing the nuanced nature of sentiment data with predefined rules and membership functions. Furthermore, fuzzy models can lack interpretability, making it harder to derive clear insights from sentiment analysis. Addressing these issues is essential for improving the effectiveness of fuzzy approaches in this field.

Given these advancements and limitations, our research is motivated to address the challenges identified in both conventional and fuzzy sentiment analysis methods. We propose a novel hybrid approach that integrates convolutional neural networks (CNNs) with fuzzy logic theory, leveraging the strengths of each method. The CNNs provide powerful feature extraction capabilities for handling large datasets, while the fuzzy logic component manages ambiguity in sentiment data. Additionally, we incorporate the Hadoop framework to ensure scalability and efficiency in processing massive datasets. By combining these techniques, our approach aims to overcome the limitations of existing methods and improve sentiment analysis outcomes.

In the following section, we will describe in detail the proposed hybrid model and its implementation, highlighting how it builds upon the findings from the literature review to advance the field of sentiment analysis.

Materials and methods

In sentiment analysis, traditional machine learning techniques such as Support Vector Machines (SVM), k-Nearest Neighbors (k-NN), and decision trees face notable challenges. These methods often struggle with scalability when processing massive datasets

Table 3 Characteristics of each cited fuzzy logic technique including dataset type, sentiment analysis goal, performance metrics, strengths, and limitations

Model	Dataset type	Sentiment analysis goal	Performance metrics	Strengths	Limitations
[33]	Mobile product reviews: Twitter, Flipkart	Aspect based Sentiment Analysis	Accuracy is 94.74% and f1-score is 97.14%	Real-time reviews for analysis	Noisy, biased social data.
[34]	Movie review dataset	Binary sentiment classification	Accuracy is 96.2%	Fuzzy systems enhance interpretability	Bag-of-words increases dimensionality
[35]	Video emotion dataset	Multimodal emotion understanding	Accuracy of 83.2%	Adaptive system improves classification	Partial transparency due to black-boxes
[36]	Record Linkage Comparison Patterns	Classification based on framework Hadoop	TPR, FNR, TNR, FPR, PR, KS, FS, and accuracy	Hadoop supports efficient scalability	Overhead impacts system efficiency
[37]	tweets and Facebook posts	Classification into positive, negative or neural	Accuracy, precision, and recall	Effective classification improves accuracy	Context sensitivity may challenge
[38]	RRSD dataset	Classification into strong negative, negative, neutral, positive and strong positive	Accuracy, precision, and recall	Improved precision with SVM-FDO	Dependency on Ontology Quality
[41]	Drug review dataset	Sentiment analysis of drug reviews	Accuracy of 82.1%	Enhanced accuracy with multi-Sense embeddings	Frequency and vague terms challenges
[42]	benchmark datasets	Sentence-Level Sentiment Analysis	Recall, Accuracy, Precision and F1-score	BERT+BiLSTM for contextualization	Potential overfitting
[43]	Twitter data	Handle perspectivism and ambiguity	Accuracy and error rate	Handling perspectivism and ambiguity	Dependence on lexicon resources

and frequently fail to capture long-range dependencies and contextual relationships between non-adjacent words, which are crucial for accurate sentiment analysis. Additionally, their rigid classification boundaries make it difficult to manage ambiguous or uncertain sentiments, highlighting a need for more sophisticated algorithms.

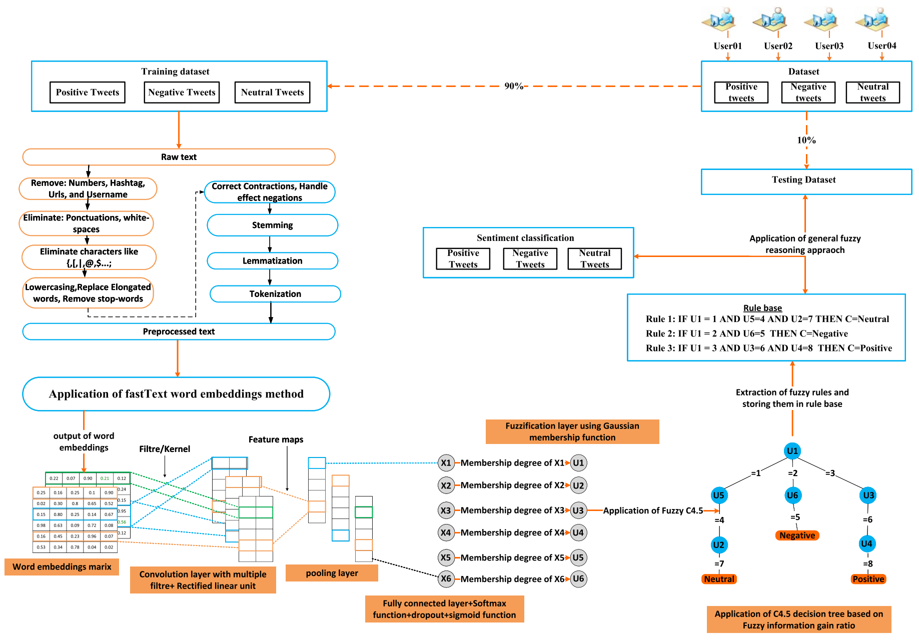
Deep learning methods, while offering advancements, also encounter significant challenges. They require substantial computational resources and can suffer from issues like vanishing gradients when attempting to capture long-range dependencies. Furthermore, these methods often lack effective strategies for managing ambiguous or fuzzy sentiment data, leading to potential inaccuracies in sentiment classification.

Fuzzy approaches offer solutions to some of these challenges by handling uncertainty and ambiguity in sentiment data. However, they face their own limitations, including scalability issues due to high computational demands and difficulties in effectively capturing nuanced sentiment data with predefined rules. Additionally, fuzzy models can lack interpretability, complicating the extraction of clear insights from sentiment analysis.

To address these limitations, we propose a novel hybrid approach that integrates fuzzy C4.5, fuzzy rule patterns (FRP), and convolutional neural networks (CNN). This approach is designed to specifically overcome the identified gaps in existing methods:

- *Robust Pre-processing and Feature Extraction:* Our approach begins with advanced pre-processing techniques to clean and prepare the data. This includes the removal of noise and irrelevant information from tweets. We utilize FastText vectorization for the numerical representation of text, which ensures effective and meaningful representation of the textual data.
- *Intelligent Feature Extraction with CNN:* Unlike traditional methods where CNNs are used primarily as classifiers, our approach employs CNN as a feature extractor. The CNN captures crucial sentiment and contextual features from tweets by identifying complex patterns and relationships in the text data, enhancing the understanding of sentiment nuances.
- *Managing Uncertainty with Fuzzy Logic:* The outputs from the CNN are fuzzified using the Gaussian Fuzzifier (GF), which manages uncertainty effectively by translating the continuous output into fuzzy sets. We then apply a fuzzy version of the C4.5 procedure to construct a fuzzy decision tree and a fuzzy rule base, enabling better handling of ambiguous or imprecise data.
- *Automated Fuzzy Rule Generation:* Our approach includes the automated generation of fuzzy rules using fuzzy CNN, eliminating the need for manual intervention in rule creation. This automation enhances efficiency and reduces the potential for human error.
- *Enhanced Classification with Fuzzy General Reasoning (FGR)*: For classifying new tweets, we use the Fuzzy General Reasoning (FGR) technique based on the established fuzzy rules. This technique improves classification accuracy by leveraging the fuzzy rule base to make more nuanced and accurate predictions.
- *Scalability and Efficiency:* The entire hybrid model is implemented on a Hadoop cluster, which ensures scalability and efficient processing of large datasets. Hadoop's distributed computing framework allows us to handle extensive data volumes effectively.

In summary, our approach represents a significant advancement by integrating advanced feature extraction with CNN, effective uncertainty management with fuzzy logic, and scalable processing with Hadoop. This methodology addresses the limitations of existing techniques and provides a more robust and accurate solution for sentiment analysis, particularly in the context of large-scale and complex datasets. Figure 1 depicts the composition of the developed hybrid model in this work, which encompasses eight stages in total including data capturing, data scrubbing, data representation, feature engineering using convolution neural network, data fuzzification, the building of classification training model using a fuzzy C4.5 decision tree, testing of the created classification model utilizing general fuzzy reasoning approach, parallelization of our hybrid model using the Hadoop framework.



User01
User02
User03
User04
Dataset
Positive tweets
Negative tweets
Neutral tweets
90%
Training dataset
Positive Tweets
Negative Tweets
Neutral Tweets
10%
Testing Dataset
Raw text
Remove: Numbers, Hashtag, Urls, and Username
Eliminate: Ponctuations, white-spaces
Eliminate characters like {,[,|,@,$...;
Lowercasing,Replace Elongated words, Remove stop-words
Correct Contractions, Handle effect negations
Stemming
Lemmatization
Tokenization
Preprocessed text
Application of fastText word embeddings method
Sentiment classification
Positive Tweets
Negative Tweets
Neutral Tweets
Application of general fuzzy reasoning appraoch
Rule base
Rule 1: IF U1 = 1 AND U5=4 AND U2=7 THEN C=Neutral
Rule 2: IF U1 = 2 AND U6=5 THEN C=Negative
Rule 3: IF U1 = 3 AND U3=6 AND U4=8 THEN C=Positive
Extraction of fuzzy rules and storing them in rule base
output of word embeddings
Filtre/Kernel
Feature maps
Fuzzification layer using Gaussian membership function
Membership degree of X1
Membership degree of X2
Membership degree of X3
Membership degree of X4
Membership degree of X5
Membership degree of X6
Application of Fuzzy C4.5
U1
=1
=2
=3
U5
U6
U3
=4
=5
=6
U2
Negative
U4
=7
=8
Neutral
Positive
Word embeddings marix
Convolution layer with multiple filtre+ Rectified linear unit
pooling layer
Fully connected layer+Softmax function+dropout+sigmoid function
Application of C4.5 decision tree based on Fuzzy information gain ratio

Fig. 1 The complete architecture of the hybrid approach developed in this work

Motivation

Sentiment analysis is an interesting research field that summarizes customer reviews of an event, service, or product and reveals whether the expressed sentiments about this event, service, or product are neutral, negative, or positive. Typically, sentiment analysis involves analyzing data gathered from social media platforms to help companies and organizations control their brands, services, events, and products by analyzing the expressed opinions, ideas, and attitudes in customer feedback and understanding buyer requirements. Generally, sentiment analysis aids big companies and enterprises in optimizing their marketing tactics, boosting the quality of their products, brands, and events, and improving buyer services. Motivated by the essential dual role of sentiment analysis in helping both customers and companies, for example, companies use sentiment analysis tools in decision-making to improve the quality of their brands, products, services, or events to better meet consumer needs and requirements. In this project, we propose an innovative hybrid model for conducting sentiment analysis on collected tweets. The suggested hybrid model integrates several advanced techniques: text-preprocessing methods to remove undesired and noisy data, the FastText word embedding method to represent sentences as numerical vectors, convolutional neural networks (CNN) to automatically extract and select relevant features, and the Gaussian membership function to fuzzify data to address challenges posed by uncertain and vague information. Additionally, the C4.5 technique with fuzzy information gain is utilized to create a fuzzy decision tree and generate a fuzzy rule base, while the general fuzzy reasoning technique is employed for categorizing new tweets. To enhance performance, the Hadoop framework is used to implement our approach in a parallel manner.

The novelty of the proposed system lies in this unique integration of CNN, Fuzzy C4.5, and the Hadoop framework, which collectively enhance sentiment analysis of tweets related to COVID-19. Unlike traditional sentiment analysis models that typically rely on either machine learning algorithms or rule-based approaches in isolation, our hybrid model leverages the strengths of both CNN and Fuzzy C4.5. This combination allows the system to capture complex patterns in textual data while simultaneously providing interpretable rules for decision-making, thereby improving both accuracy and explainability. Furthermore, the adoption of Hadoop's distributed computing capabilities addresses the scalability challenges faced by existing models, enabling efficient processing of large-scale, real-time social media data. This is particularly crucial in the context of a rapidly evolving global pandemic where the volume and velocity of data are exceptionally high. Additionally, our system is specifically tailored for COVID-19-related sentiment analysis, incorporating domain-specific preprocessing and feature extraction techniques that further enhance performance. These novel aspects set our system apart from existing sentiment analysis solutions and demonstrate its potential for more effective and scalable real-time sentiment analysis in dynamic scenarios.

Data capturing

Our suggested hybrid approach is implemented using the version 3.10.0a5 of the Python programming language. The execution of the model is carried out on two large datasets, as indicated below:

The first dataset, denoted as dataset1, is named *Sentiment140* [44]. Initially, this dataset comprises 1,600,000 instances that are annotated, with 800,000 tweets labeled as positive and the remaining tweets labeled as negative. Within this dataset, the decision attribute assumes two possible values: 4 or 0. A decision attribute value of 4 indicates a positive tweet, whereas a decision attribute value of 0 signifies a negative tweet. This dataset enables us to analyze the emotions and attitudes expressed towards a service, product, brand or topic on the X platform. It encompasses six attributes, each of which are Decision attribute, Ids, Date, Flag, User, and Text. In this proposal, we are interested in X sentiment classification. Therefore, only both attributes, which are "Target" and "Text," are useful for performing the sentiment analysis.

The second dataset, denoted as dataset2, is named *COVID-19_Sentiments* [45]. It comprises 259,458 instances with a neutral sentiment, along with 120,646 tweets conveying negativity and 257,874 tweets expressing positivity. Hence, the collected dataset consists of a total of 637,978 tweets. In this dataset, the decision attribute is labeled as either positive, negative, or neutral. The value of neutrality is represented by 0, while the value of negativity falls within the range of -1 to 0, and the value of positivity ranges from 0 to 1. There are five attributes included in this dataset, which are Target, Ids, Date, Location, and Text. The functional attributes in this dataset are the "Target" and "Text" attributes.

Data scrubbing

Typically, tweets are considered as semi-unstructured and unstructured data, often containing inconsistent, incomplete, unwanted, and noisy information. Consequently, in order to overcome these limitations and extract meaningful insights from such unstructured data, it becomes crucial to employ advanced methods of data pre-processing on tweets. This pre-processing step is necessary to prepare the tweets for the effective implementation of NLP and text mining approaches. Any sentiment classification method designed to treat and learn raw text collected from online social media platforms must effectively deal with unstructured and semi-structured raw text inconsistencies and eliminate semantically empty terms. As a result, this contribution incorporates the utilization of various data preprocessing tactics [46–48] with the objective of ensuring the quality of tweets as outlined below:

Abbreviations, Spelling mistakes and Slang: At present, the Twitter platform is the most widespread social network to forward a person or organization's ideas, impressions, and opinions about a particular subject within tiny messages limited to 280 characters, commonly known as tweets. Due to the limitation in tweet length, Twitter users employ multiple abbreviations, slang and make multiple spelling mistakes for expressing their entire opinions. And that influence negatively the Twitter opinion mining process. For avoiding these defies in this contribution, we have expanded every existing abbreviation in the tweets. We have replaced every slang with its complete vocabulary forms to avoid any vocabulary lookup restrictions. We have corrected spelling mistakes for meaningful processing using the WordNet dictionary. In this step, we applied three techniques, which are:

- Load and use a dictionary *abbreviation_dict* containing common abbreviations and their full forms (e.g., "btw" → "by the way").
- Load and use a dictionary *slang_dict* containing slang words and their standard vocabulary equivalents (e.g., "gonna" → "going to").
- Load and use a spell checker *WordNet* for correcting spelling mistakes.

Eliminate punctuation marks, URLs, special characters, hashtags, white spaces, numbers, and usernames: A widely used approach involves the exclusion of numbers, URLs, usernames, white spaces, hashtags, special characters, and punctuation marks from every processing tweet as they do not convey any negative or positive sentiments. After we have removed all these expressions, we saved only the lowercase and uppercase letters. The following algorithm 1 describes all the steps applied in this preprocessing technique:

Algorithm 1 Eliminate punctuation marks, URLs, special characters, hashtags, whitespaces, numbers, and usernames

```
function PREPROCESS_TEXT(text)
    Step 1: Convert to lowercase (optional)
    text = CONVERT_TO_LOWERCASE(text)
    Step 2: Remove URLs
    REMOVE_PATTERNS(text, ’http://’, ’https://’, ’www.’)
    Step 3: Remove usernames (mentions)
    REMOVE_PATTERNS(text, ”@username”)
    Step 4: Remove hashtags
    REMOVE_PATTERNS(text, ”#hashtag”)
    Step 5: Remove punctuation marks and special characters
    REMOVE_CHARACTERS(text, ’!”#$%&'()*+,-./:;¡=¿?@[\]ˆ_‘{—) ’}
    Step 6: Remove numbers
    REMOVE_PATTERNS(text, ’0-9’)
    Step 7: Remove extra white spaces
    REPLACE(text, multiple_whitespace,’ ’)
    Step 8: Return cleaned text
    return text
end function
```

Lower-case: After applying both earlier introduced data preprocessing strategies, we have saved only the uppercase and lowercase letters, and we have removed all other special characters. Consequently, as a component of this step, all capital letters within each tweet are changed to lowercase. Substitute elongated words with appropriate alternatives: This data preprocessing strategy aims to remove repeated letters within words, such as in the case of the elongated word “Glaaaaad”, where any letter that appears more than two times would be eliminated. Upon applying this data preprocessing strategy, the extended word “Glaaaaad” is transformed into “Glaad” and shortened to a length of two characters.

Exclude stop-words from the text: This data preprocessing strategy aims to remove words commonly encountered in tweets that do not have any significant relevance to a particular subject (“a”, “it”, “an”, “she”, “of”, “in”, “on”, “I”, “your”, “the”, “and” etc.). These commonly occurring words are typically ineffective and insignificant for the sentiment analysis process.

Tokenization: This data preprocessing strategy involves breaking down every input tweet into a collection of significant terms, and every term in this collection is called a token. For instance, This sentence “I am very happy” is divided into words “I”, “am”, “very”, “happy” and this process is done using NLTK tokenizer designed by Python.

Lemmatization: After the tokenization technique, lemmatization is carried out to turn out a diverse form of words to the root word, called a lemma, i.e., it converts the words that hold the same meaning to a single base word. For instance, the words “happy,” “Glad,” and “Good” are replaced with “Good”.

Stemming: Following the lemmatization strategy, stemming is performed to identify the stem of each word during the processing of tweets. The stemmed base of a word is often obtained by removing the prefix and/or suffix. Although, the achieved stem may or may not be the exact vocabulary form. The distinction between stemming and lemmatization strategies lies in the fact that lemmatization reduces the number of terms in the tweets using a lexicon. But, the stemming strategy decreases the length of a word to a root form. For instance, the stemming technique converts the words “running,” “runner,” and “runs” into their base form, “run,” while the words “better,” “best,” and “good” are reduced to their base form, “good”. These operations are performed by an algorithm called Porter stemmer. The following algorithm 2 describes all the steps applied in these preprocessing techniques:

Following the completion of the data scrubbing phase, the subsequent step involves data representation, where the instance is transformed into a numerical representation vector. For this purpose, FastText approach is employed due to its superior classification performance, as discussed in the paper [39]. The outputs of the data scrubbing phase serve as the input for the data representation phase within this study.

Algorithm 2 Exclude stop-words from the text, Tokenization, Lemmatization and Stemming techniques

```
1: **Input:** Tweet text $T$
2: **Output:** Preprocessed text $T'$
3: **Step 1: Exclude Stop-words**
4:         Define stop-words list $S = \{$”a”, ”it”, ”an”, ”she”, ”of”, ”in”, ”on”, ”I”, ”your”, ”the”, ”and”, $\ldots\}$
5: for each word $w$ in $T$ do
6:    if $w \in S$ then
7:       Remove $w$ from $T$
8:    end if
9: end for
10: **Step 2: Tokenization**
11:         Tokenize $T$ into tokens $t_1, t_2, \ldots, t_n$ using NLTK tokenizer
12:         $T_{tokens} = [t_1, t_2, \ldots, t_n]$
13: **Step 3: Lemmatization**
14: for each token $t$ in $T_{tokens}$ do
15:    Convert $t$ to its lemma $t_{\text{lemma}}$ using a lemmatizer
16:    Replace $t$ with $t_{\text{lemma}}$
17: end for
18: **Step 4: Stemming**
19: for each token $t$ in $T_{tokens}$ do
20:    Apply Porter Stemmer to convert $t$ to its stem $t_{\text{stem}}$
21:    Replace $t$ with $t_{\text{stem}}$
22: end for
23: **Return:** Preprocessed text $T' = T_{tokens}$
```

Data representation

The objective of this work is to ascertain the sentiment orientations of tweets gathered from X. Consequently, the preprocessed tweets must be translated into assembly language to facilitate subsequent analysis, processing, and classification. During this phase, we have performed unsupervised training utilizing the FastText [20] data vectorization method, which converts words with an n-gram value of 2 into vectors of reduced

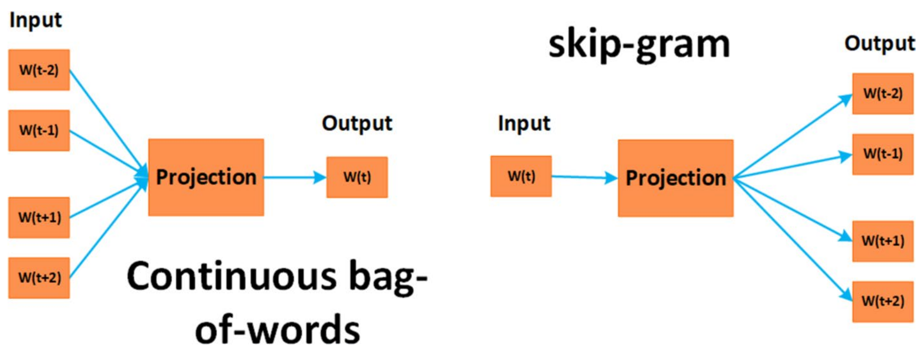


Fig. 2 Continuous bag-of-words model and Skip-gram approach

dimensionality. FastText is based on either the skip-gram procedure or the Continuous BOW model, producing low-dimensional vectors for each sentence, as illustrated in Fig. 2. The results of a comparative study [49] between the skip-gram procedure and the continuous BOW approach revealed that the skip-gram model offers superior accuracy in word representation. However, it comes at the expense of longer training time. As a solution, we have chosen to utilize the skip-gram model in our work, effectively addressing the training time issue through the utilization of the Hadoop framework.

As mentioned earlier, various word vectorization methods have been developed, including Word2Vec, bag-of-words, GloVe, IF-IDF, FastText, and n-gram. According to a comparative study conducted in the paper [39], these methods achieved accuracy levels of 77.43%, 64.24%, 72.23%, 71.05%, 87.13%, and 51.76%, respectively. Therefore, we can explain the excellent performance achieved by FastText compared to other word embedding methods by the ability of FastText in capturing the sense of more abbreviated words and help the term vectorization operation to detect the prefixes and suffixes of the learned bag of n-grams.

Following the implementation of FastText technique as the data representation stage in our contribution, the subsequent step involves utilizing a convolutional neural network to perform feature extraction and choose features, as detailed in subsequent subsection.

Feature engineering

The process of extracting and selecting the most relevant and significant features play a pivotal role in diverse applications of NLP. Numerous studies have been conducted to generate resilient, comprehensive, and suitable features. In our study, we have extensively reviewed various research works from the literature [50–52] Based on our analysis of recent works, we have deduced a noticeable trend towards prioritizing feature engineering in deep learning models, as opposed to relying on handcrafted features in conventional machine learning techniques. In essence, deep neural networks possess the capability to autonomously extract and choose the most pertinent features throughout the learning process, eliminating the requirement for human intervention. This stands in contrast to traditional machine learning algorithms that necessitate predefined features

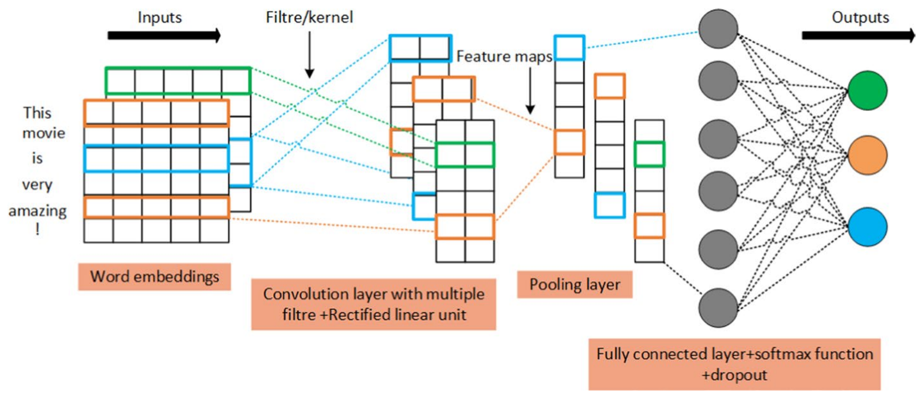

Inputs
Filtre/kernel
Feature maps
Outputs
This
movie
is
very
amazing
!
Word embeddings
Convolution layer with multiple filtre +Rectified linear unit
Pooling layer
Fully connected layer+softmax function +dropout

Fig. 3 Feature engineering using convolutional neural network

prior to initiating the training process. In the literature, numerous deep learning models exist, with the CNN standing out as one of the most widely recognized and popular types. CNNs have been specifically designed to provide an effective representation of the input data. Given its overall structure, the CNN stands out as an advantageous option to extract and choose the most suitable attributes in this specific study. Typically, the fundamental architecture of the CNN consists of a single convolutional layer, a sole pooling layer, and a singular fully connected layer, as illustrated in Fig. 3.

As we stated earlier and as elucidated in Fig. 3 the architecture of the simple convolutional neural network includes three essential layers which are: The convolution layer is utilized to derive a collection of features from the matrix acquired during the data representation stage. The pooling layer serves the purpose of choosing the most significant and suitable features from the acquired set during the convolution layer step. Hence, the outputs of the pooling layer are converted into straight values through a fully connected layer, utilizing the softmax activation function.

Following the utilization of a convolutional neural network to retrieve and choose the most suitable characteristics in this study, the subsequent phase involves employing a Gaussian membership function to fuzzify the generated features from the convolutional neural network stage. In this study, we implemented the fuzzification function to equip our suggested hybrid approach with the capability to handle vague and uncertain data, thereby enhancing the accuracy of the hybrid approach we introduced for sentiment classification.

Data fuzzification phase

Upon utilizing the convolutional neural network to retrieve and choose the most consistent and appropriate characteristics, the subsequent stage involves the data fuzzification step, aiming to fuzzify the produced collection of features from the previous phase. The main goal of this stage is to fuzzify the characteristics, enabling the application of the fuzzy C4.5 decision tree, thereby equipping our model to to effectively handle vague and uncertain data [53]. Within the scope of this study, we employ the fuzzification process to convert the neural activation's in the fully connected layer into a collection of vague values. This process involves determining the membership degree of every neural activation's by implementing a Gaussian fuzzifier. The reason for selecting the Gaussian membership function over trapezoidal or triangular membership functions is supported by experimental results outlined in the paper [54]. The findings demonstrate that the Gaussian fuzzifier attains a high classification rate of 94.87%, outperforming the triangular fuzzifier with a classification rate of 90.14% and the trapezoidal fuzzifier with a classification rate of 91.21%. The Gaussian fuzzifier is characterized by a pair of parameters which are: $sd > 0$ denotes the typical deviation and cv represents the core value as outlined in Fig. 4. The computation of the membership degree of y involves the use of the subsequent eq. (1):

$$\mu_A(y) = e^{-}\frac{(y-cv)^2}{2.sd^2} \tag{1}$$

Once the neural activation's in the fully connected layer have been fuzzified employing the Gaussian fuzzifier, we obtain the vague value for each neuron. Consequently, the subsequent step involves implementing the fuzzy version of C4.5 procedure, which relies on fuzzy information gain ratio.

Fuzzy C4.5 procedure

After completing the data fuzzification stage [55], wherein the crisp neuron values of the fully connected layer were fuzzified, the subsequent stage focuses on constructing the

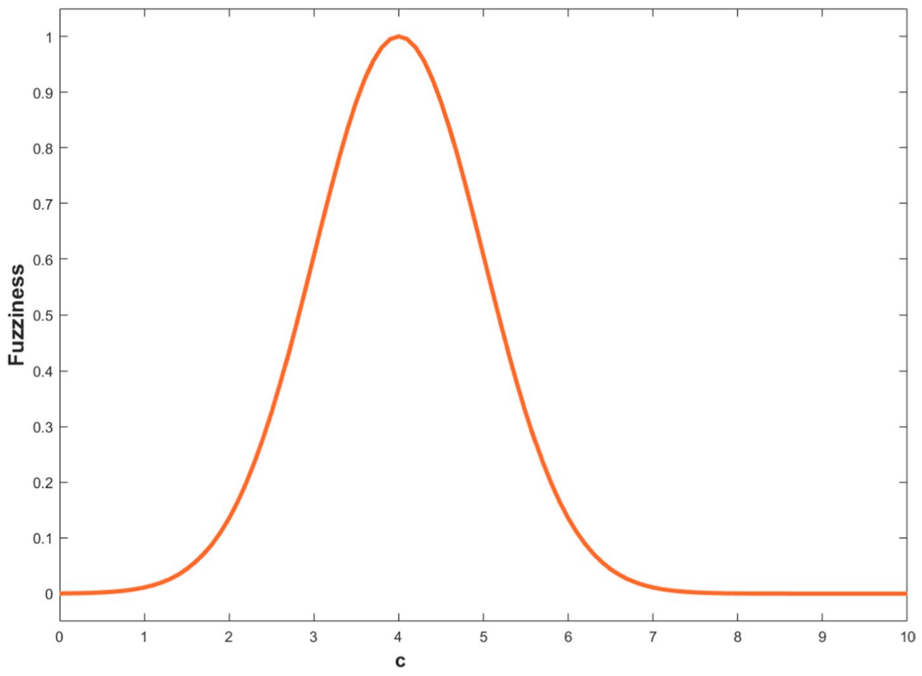


Fig. 4 Representation graphic of Gaussian fuzzifier

fuzzy tree. This is achieved by utilizing the fuzzy C4.5 algorithm and subsequently deriving imprecise rules from the resultant tree of vagueness, which are then stored in the rule repository. In this work, our classifier merges the principles of the fuzzy set theory with the C4.5 algorithm.

C4.5 machine learning algorithm is the followed-up decision tree procedure for constructing a decision tree from the given training dataset. The obtained decision tree after applying C4.5 is an oriented tree involving three kinds of nodes: a base node, internal nodes, decision nodes [56]. The prescribed procedure for constructing a tree with the C4.5 algorithm is outlined as follows: Initially, the provided dataset undergoes division into separate testing and training datasets through the implementation of the cross-validation approach employing 10-fold segmentation. Secondly, The training dataset is utilized to implement the C4.5 algorithm and construct the trained model which is a decision tree. In this second phase, the C4.5 algorithm computes at each iteration the ratio of information gain of all existed features in the used training dataset. Afterward, it picks the feature having the greatest information gain rate as the optimal attribute during the splitting process. Once the better feature is set, the training dataset is partitioned into multiple partitions equal to the better feature's number of values. The C4.5 algorithm stops if all examples in every partition have the same class label. Otherwise, the C4.5 algorithm is recursively performed until all partition instances have the same class label or no feature remainders in the dataset features list. The decision tree is constructed after the algorithm stopped, and the testing dataset is employed to assess the performance of constructed decision tree.

In this contribution, we have utilized the fuzzy version of the C4.5 algorithm that combines fuzzy theory and decision trees to tackle the tweets' ambiguity and vagueness. The fuzzy C4.5 algorithm employs vague linguistic words to choose superior attribute splits. It allows the dataset examples to concurrently proceed downward multiple edges with diverse membership degree values ranging in the interval [0, 1]. The used procedure for creating the fuzzy tree using fuzzy C4.5 is similar to conventional C4.5. But there is one difference: the conventional C4.5 algorithm computes the ratio of the information gain by measuring the likelihood of all dataset instances. Unlike, fuzzy version of C4.5 procedure computes the fuzzy ratio of the information gain by calculating the likelihood of the belonging degree of all dataset instances. Therefore, in the data fuzzification phase, we have computed the membership degree of the dense layer's neurons for the purpose of utilizing the fuzzy C4.5 over the dense layer's neurons and then construct a fuzzy decision tree.

The algorithm 3 and the flowchart Fig. 12 describe our developed fuzzy parallel C4.5 used in this contribution for creating the fuzzy tree by implementing it on training tweet dataset.

Algorithm 3 Our proposed Fuzzy Parallel C4.5 Algorithm

```
Require: Fuzzified dataset $E$, Feature set $F$
Ensure: Fuzzy Decision Tree
1: Initialize Fuzzy Decision Tree with a root node
2: Configure Hadoop with Mapper and Reducer jobs
3: Split dataset $E$ into subsets $E_j$ for parallel processing
4: for each Mapper job on subset $E_j$ do
5:     for each feature $F(n)$ do
6:         Compute fuzzy values for $F(n)$
7:         Calculate fuzzy information gain for $F(n)$
8:     end for
9: end for
10: for each Reducer job do
11:     Combine results from Mapper jobs
12:     Select the feature $F(n)^*$ with the highest fuzzy information gain
13: end for
14: Add $F(n)^*$ as a node in the Fuzzy Decision Tree
15: for each value of $F(n)^*$ do
16:     if the subset is pure then
17:         Label as a leaf node with corresponding class
18:     else
19:         Recursively split the subset based on $F(n)^*$
20:     end if
21: end for
22: Repeat until all subsets are processed
23: Return Fuzzy Decision Tree
```

Upon constructing the fuzzy tree using the C4.5 technique with the application of fuzzy ratio of information gain. Once we derived all potential vague rules using the created fuzzy tree, we proceed to save them in the rule base. Consequently, the subsequent phase of our contribution involves applying the GFR procedure to the acquired rule repository for the purpose of categorization the new instances.

General fuzzy reasoning technique

Following the completion of both phases, which involve constructing the vague tree and generating the vague rules using the C4.5 algorithm with IGRF. Hence, the pretrained pattern has been created, and the subsequent step involves assessing our developed learning pattern. That is, we have made use of the GFR [57] method to classify the fresh input instance and determine the feature label to which it belongs based on the produced rule base. General Fuzzy Reasoning (GFR) is well-suited for handling uncertainty and complex, non-linear relationships by using fuzzy logic principles and Gaussian fuzzy set functions, making it more adaptable and nuanced compared to traditional methods. Fuzzy Logic Controllers (FLCs) are effective for control tasks with predefined rules but lack flexibility for complex or dynamic datasets, requiring extensive tuning. Fuzzy Inference Systems (FIS) are useful for applying fuzzy logic but can struggle with scalability and adaptability for large or intricate datasets, where GFR provides better performance. Adaptive Neuro-Fuzzy Inference Systems (ANFIS) offer advanced modeling by combining neural networks and fuzzy logic but are computationally intensive and require significant training data. GFR was chosen for its balance of flexibility, scalability, and computational efficiency, which suits the needs of our hybrid model.

The general fuzzy reasoning algorithm proceeds with the subsequent steps introduced subsequently for the categorization of a novel tweet w_n:

Let $w_n = \{c_{n1}, c_{n2}, ..., c_{nl}\}$ a new example to be classified and determined its class label and $\{LR_1, LR_2, ..., LR_K\}$ a set of K extracted linguistic rules from the generated fuzzy tree in the previous phase of our work. Let $MV_j(c_{nj}), j = 1, ..., l$ be the computed membership value of the feature value c_{nj} by employing the Gaussian fuzzy set function.

- Determine the compatibility rate (CR) between the new tweet w_n and every linguistic rule LR_x for $x = 1, ..., K$ in the rule base by applying the t-norm **t** as described in the following eq. (2):

$$CR(w_n, LR_x) = \mathbf{t}[MV_1(c_{n1}), MV_2(c_{n2}), .., MV_l(c_{nl})] \tag{2}$$

- For every value of the decision feature, compute the classification degree MD_z. MD_z is determined as the gathering of the CR of all fuzzy rules, which are measured in the first step with the value z_a of the decision feature and expresses the CR of the new example to be classified with all the fuzzy rules whose divined class label is z_a by the following eq. (3):

$$MD_{z_a} = \mathbf{f}\{CR(w_n, LR_x)|z_a\} \tag{3}$$

Where z_a is the class label of LR_x, and **f** is a gathering operator.

To classify the new tweet $w_n = \{c_1, c_2, c_3, ?\}$, which possesses four attributes, including an unknown decision attribute "?" by employing the general fuzzy reasoning technique. Considering the values of c_1, c_2, and c_3 attributes are characterized as fuzzy classes. Furthermore, we posit that the $MV(c_1) = 0.56$, $MV(c_2) = 0.24$, and $MV(c_3) = 0.75$ are the measured membership values utilizing the Gaussian fuzzy set function of c_1, c_2, and c_3 respectively. Moreover, we have four linguistic rules outlined as follows:

LR_1: IF Q is q_1 AND R is r_1 AND S is s_1 THEN O is o_1. With MV(q_1) = 0.60 MV(r_1) = 0.21 and MV(s_1) = 0.93

LR_2: IF Q is q_2 AND R is r_2 AND S is s_2 THEN O is o_2. With MV(q_2) = 0.14, MV(r_2) = 0.87 and MV(s_2) = 0.68

LR_3: IF Q is q_3 AND R is r_3 AND S is s_3 THEN O is O_3. With MV(q_3) = 0.24, MV(r_3) = 0.95 and MV(s_3) = 0.50

LR_4: IF Q is q_4 AND R is r_4 AND S is s_4 THEN O is o_4. With MV(q_4) = 0.79, MV(r_4) = 0.51 and MV(s_4) = 0.32.

Stage 1: determine the degree of compatibility DC for the input tweet (c_1, c_2, c_3) matches every fuzz rule term (q_1,q_2,q_3,q_4,r_1,r_2, r_3,r_4, s_1,s_2,s_3,s_4). Subsequently, these measured rates will be employed to compute the CR for each linguistic rule.

1. DC1(c_1,q_1) = min(MV(c_1),MV(q_1)) = min(0.56,0.60) = 0.56
2. DC2(c_2,r_1) = min(MV(c_2),MV(r_1)) = min(0.24,0.21) = 0.21
3. DC3(c_3,s_1) = min(MV(c_3),MV(s_1)) = min(0.75,0.93) = 0.75
4. DC4(c_1,q_2) = min(MV(c_1),MV(q_2)) = min(0.56,0.14) = 0.14
5. DC5(c_2,r_2) = min(MV(c_2),MV(r_2)) = min(0.24,0.87) = 0.24

6. DC6(c_3,s_2) = min(MV(c_3),MV(s_2)) = min(0.75,0.68) = 0.68
7. DC7(c_1,q_3) = min(MV(c_1),MV(q_3)) = min(0.56,0.24) = 0.24
8. DC8(c_2,r_3) = min(MV(c_2),MV(r_3)) = min(0.24,0.95) = 0.24
9. DC9(c_3,s_3) = min(MV(c_3),MV(s_3)) = min(0.75,0.50) = 0.50
10. DC10(c_1,q_4) = min(MV(c_1),MV(q_4)) = min(0.56,0.79) = 0.56
11. DC11(c_2,r_4) = min(MV(c_2),MV(r_4)) = min(0.24,0.51) = 0.24
12. DC12(c_3,s_4) = min(MV(c_3),MV(s_4)) = min(0.75,0.32) = 0.32

As a result:

- min(DC1,DC2,DC3) = min(0.56,0.21,0.75) = 0.21
- min(DC4,DC5,DC6)= min(0.14,0.24,0.68) = 0.14
- min(DC7,DC8,DC9) = min(0.24,0.24,0.50) = 0.24
- min(DC10,DC11,DC12) = min(0.56,0.24,0.32) = 0.24

Consequently, the CR between the tweet w_n and every linguistic rule LR_1,LR_2,LR_3, and LR_4, equals: CR(w_n, LR_1) = 0.21, CR(w_n, LR_2) = 0.14, CR(w_n, LR_3) = 0.24, and CR(w_n, LR_4) = 0.24.

Stage 2: The next step is to measure the classification degree MD_z for each decision attribute value. In this example, we have two decision attribute value z_1, and z_2, as follows:

- $MD_{z_1} = f\{CR(w_n, LR_x)|z_1 = \text{CR}(w_n, LR_1) + \text{CR}(w_n, LR_3) = 0.21 + 0.24 = 0.45$
- $MD_{z_2} = f\{CR(w_n, LR_x)|z_2 = \text{CR}(w_n, LR_2) + \text{CR}(w_n, LR_4) = 0.14 + 0.24 = 0.38$

Stage 3: In stage 2, we deduce that the decision attribute value z_1 has the highest sum MD_{z_1} = 0.45. So, Assign the decision attribute value z_1 to the input tweet $w_n = \{c_1, c_2, c_3, z_1\}$.

As mentioned earlier, for the classification of new input tweets, we have opted to employ the general fuzzy reasoning technique rather than the classic fuzzy reasoning technique. This decision is based on the experimental results presented in the work [58], which demonstrated that the GFR technique achieves higher classification rate (86.56%) compared to the CFR technique, attaining a classification rate of 63.96%. Having elucidated the application of the GFR technique in this contribution, the subsequent step involves explaining the utilization of the Hadoop software in this proposal.

Parallelization using Hadoop

At present, the volume of data is growing exponentially due to the digital world's progress. The utilization of cloud data techniques and the extensive adoption of intelligent sensors and smart machines have increased Big Data techniques. Nowadays, Big Data is present in each side of our quotidian life and can be applied to solve modernized issues. Big Data fetches two crucial difficulties: how to stock and to operate with a large-scale dataset, and more importantly, how to learn data and convert it into an aggressive benefit. Hadoop framework overcomes these two difficulties by efficiently storing and affording considerable computational abilities over the large-scale datasets [59]. It is a distributed model, and it provides a training approach to parallelize, implement, and schedule the tasks to be learned on a group of computers. Hadoop provides appropriate solutions to overcome modernized issues that are fault-tolerant, scalable, resilient, and cost-effective. Hadoop possesses two primary sub-projects, which are the MapReduce programming pattern and the Hadoop Distributed File System (HDFS). Hadoop framework employs the MapReduce programming model to treat massive data by dividing data over a cluster of computing nodes and aggregating the obtained computational results. MapReduce is the most used approach to learn the large-scale data stored on large groups of machines. In this approach, tasks are trained by partitioning into small chunks and distributing across multiple computing nodes. In general, The Hadoop cluster consists of a sole master node and multiple slave nodes. The MapReduce framework consists of two components: multiple task trackers and a job tracker. The daemon responsible for job tracking is executed on the master node and every task tracker daemon performed on slave nodes. JobTracker is employed to prepare the location of a data file in diverse DataNodes and monitor the TaskTracker jobs. It is an essential component in the Hadoop framework as if it fails, all the executing tasks will get stopped. TaskTracker performs many diverse tasks, such as a mapper function, reducer function, and shuffle operation. Therefore it is deemed as an important component because it is carrying out the principal MapReduce operations. The Hadoop framework also uses distributed file system HDFS as a fault-tolerant, reliable, and scalable data storage on computing nodes' clusters. It operates jointly with MapReduce by sharing storage and computational tasks over a large cluster of computing nodes. HDFS stores data in any format like videos, text, images, etc., and it consists of two important components such as NameNode and multiple DataNodes. NameNode daemon runs on master node and it controls and manages tasks carried out by DataNodes which are ruining on slave node and they stocks all data files and trains them [56]. To overcome the two crucial difficulties confronted by big data, we have implemented the Hadoop framework as shown in Fig. 5 that describes our suggested hybrid model's implementation over a Hadoop cluster of 5 machines: single master computer and four slave computers.

The block diagram in Fig. 5 illustrates the process of sentiment classification using the Hadoop framework with a Fuzzy Parallel C4.5 Algorithm, implemented through the MapReduce programming model. The process begins with the collection of tweets or text data, which is partitioned into a training dataset and a testing dataset. These datasets are distributed across a Hadoop cluster consisting of five machines (one master and four slaves), utilizing the Hadoop Distributed File System (HDFS). HDFS manages storage by assigning data blocks to its DataNodes (located on slave machines) and maintaining metadata through its NameNode (on the master node)

In the *Preprocessing Phase*, each chunk of the data is passed through preprocessing mappers that clean the data by removing noise, tokenizing the text, and preparing it for further processing. After preprocessing, the data flows through a series of specialized mappers, each responsible for different steps in the feature extraction and fuzzification process. Specifically: *FastText Mapper:* Converts the processed text into numerical vectors using the FastText method, providing a dense vector representation of each word. *CNN Mapper:* Applies a convolutional neural network (CNN) to automatically extract the most relevant features from the data. The CNN is designed to capture significant characteristics from short texts, such as tweets. *Gaussian Mapper:*

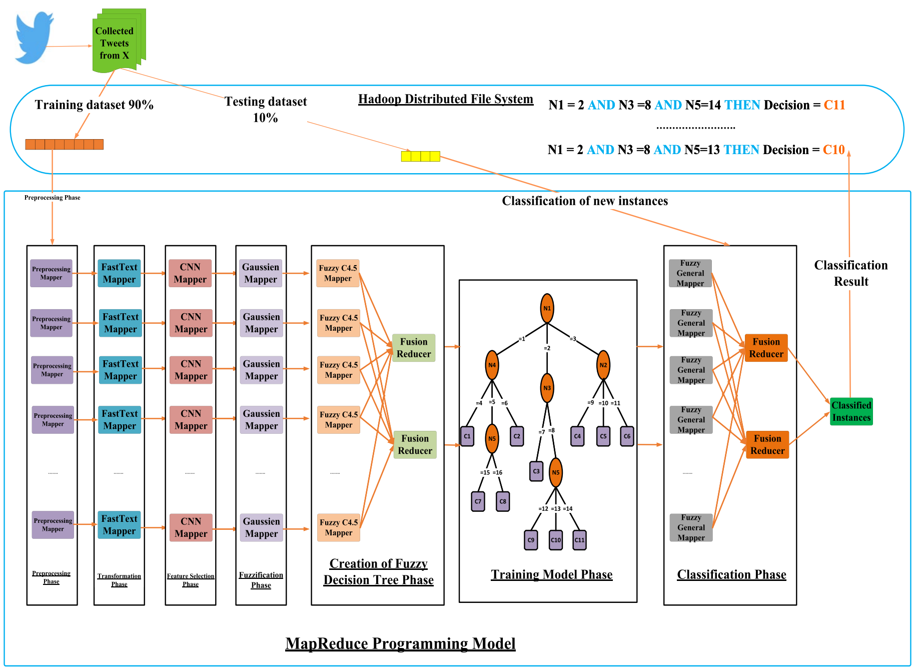


Fig. 5 Parallelization of our developed hybrid model using Hadoop framework

Introduces fuzziness by applying Gaussian fuzzy set functions, allowing the model to handle uncertainty and vagueness in the data.

After feature extraction, the *Fuzzy C4.5 Mapper* is used to create small fuzzy decision trees based on the partitioned training data. The C4.5 algorithm is modified to incorporate fuzzy logic, allowing for more nuanced decision boundaries. Multiple fuzzy decision trees are created in parallel across the mappers. These partial fuzzy trees are then combined by the *Fusion Reducer*, resulting in a complete, aggregated fuzzy decision tree.

In the *Classification Phase*, the final fuzzy tree generated from the training data is applied to the testing dataset. Here, the *Fuzzy General Mapper* performs General Fuzzy Reasoning (GFR) on each chunk of the testing data, classifying the instances based on the rules derived from the fuzzy decision tree. The outputs from these individual mappers are then aggregated by another *Fusion Reducer*, which combines the classification results from all the mappers to generate the final set of classified instances.

In this model, MapReduce is essential for ensuring scalability and parallel processing. Each phase, from data preprocessing to feature extraction, fuzzification, and classification, is carried out in parallel, significantly improving the system's efficiency and capability to handle large-scale datasets. The final output provides a comprehensive sentiment classification based on fuzzy logic, addressing the inherent uncertainties in natural language data.

Algorithm 4 Our proposed parallel hybrid model

```
Require: Training tweet dataset E.
Ensure: FuzzyTree.
 1: Define the job of Hadoop
 2: Designate the Mapper class as ChooseMapperJob
 3: Designate the Reducer class as ChooseReducerJob
 4: Altering the the size of each block of HDFS is necessary to enable the division of
    dataset E into S subsets $E_j = \{j = 1, 2, ..., S\}$
 5: for $j = 1$ to $S$ chunk do
 6:   In the j-th ChooseMapperJob
 7:   if (a word in $j$-th data $chunk_j$ not in English) then Translate(word into English)
 8:   T = data-pretreatment($chunk_j$)
 9:   F = FastText(T)
10:   C = ConvolutionNeuralNetwork(F)
11:   D = GaussianMembershipFunction(C)
12:   $Tree_j$ = Algorithm 3
13: end for
14: for $j = 1$ to $n$ Reducer do
15:   In the j-th ChooseReducerJob
16:   FuzzyTree = $\sum_{j=1}^{S} Tree_j$
17: end for
18:   Input: Testing tweet dataset E, and FuzzyTree
19:   Output: Classification Result (CR)
20: for $j = 1$ to $C$ chunk do
21:   In the j-th ChooseMapperJob
22:   $CR_j$ = GeneralFuzzyReasoning($Chunk_j$,FuzzyTree)
23: end for
24: for $j = 1$ to $n$ Reducer do
25:   In the j-th ChooseReducerJob
26:   CR = $\sum_{j=1}^{C} CR_j$
27: end for
28: return CR to be stored in HDFS as classification result
```

Where:

- *Training tweet dataset (E)*: The set of tweets used for training the model.
- *FuzzyTree*: The fuzzy decision tree generated during the training phase.
- *Testing tweet dataset (E)*: The set of tweets used for testing the model.
- *Classification Result (CR)*: The output classification result to be stored in HDFS.
- *S*: Number of subsets into which the training dataset *E* is divided for parallel processing.
- *j*: Index variable representing the current subset or chunk during iterations over the subsets.
- *T*: Result of data pretreatment for each chunk.
- *F*: Result of applying FastText for vectorization.
- *C*: Result of applying convolutional neural network (CNN) processing.
- *D*: Result of applying Gaussian Membership Function to fuzzify the data.
- $Tree_j$: Fuzzy decision tree generated for each subset E_j.
- CR_j: Classification result for each chunk during the testing phase.

- *C*: Number of chunks into which the testing dataset is divided for parallel processing.

As data volume continues to grow exponentially due to advancements in digital technology, the use of Big Data techniques has become increasingly crucial. Hadoop's framework provides a robust solution to handle these challenges by efficiently managing and processing large-scale datasets through its distributed model. However, implementing Hadoop's parallelization introduced several challenges. One significant challenge was the effective distribution and balancing of data and computational tasks across the cluster to avoid bottlenecks and ensure optimal performance. To address this, we partitioned the dataset into training and testing sets, distributing these across five computers using the Hadoop Distributed File System (HDFS). HDFS allowed for scalable and fault-tolerant storage, while the MapReduce model facilitated parallel processing by dividing data into smaller chunks and assigning them to multiple mapper nodes for concurrent processing.

Another challenge was ensuring fault tolerance and system reliability, particularly with critical components such as the JobTracker. Any failure in the JobTracker could disrupt the entire processing pipeline. To mitigate this, we implemented redundancy and fail over mechanisms within the Hadoop cluster, allowing the system to recover from node failures without halting the processing. Additionally, integrating various processing steps-such as data preprocessing, FastText for numerical vector conversion, convolutional neural networks (CNN) for feature extraction, and fuzzy logic for classification-required meticulous synchronization to prevent performance bottlenecks. These challenges were addressed by optimizing configurations and carefully managing the interaction between different processing stages.

Experimental validation

Experimental validation refers to the rigorous process of testing and evaluating our proposed algorithm and other models in a controlled setting. It involves executing the models on specific datasets and analyzing their performance metrics. The datasets used in our experiments are as follows:

The Sentiment140 dataset: consists of tweets (X messages) labeled with sentiment polarity: positive, negative, or neutral. Each tweet in the dataset is labeled based on the sentiment expressed by the user who posted it.

COVID-19 Sentiments dataset: It contains text data, such as social media posts, news articles, or public comments, related to the COVID-19 pandemic. This dataset includes sentiments expressed by individuals in India regarding various aspects of the pandemic, such as government responses, public health measures, vaccines, and societal impacts.

IMDB Dataset: The IMDB dataset is commonly used in the context of movie-related research and recommendation systems. It contains information about movies, including titles, genres, release dates, user ratings, and reviews. The dataset is valuable for tasks such as sentiment analysis and movie recommendation algorithms.

Academic Yelp Dataset: The Academic Yelp dataset focuses on reviews and ratings related to academic institutions, courses, or educational resources. It includes information such as institution names, course titles, reviewer ratings, and written reviews.

Researchers may use this type of dataset for educational quality assessment and sentiment analysis.

World Cup2014 Dataset: The World Cup2014 dataset contains information related to the FIFA World Cup held in Brazil in 2014, encompassing data about football (soccer) matches, teams, players, goals, and various statistics. Researchers and sports analysts often utilize such datasets to study team performance, player dynamics, and predict match outcomes.

The datasets were carefully selected based on their relevance to the specific domains under consideration. Prior research and literature validate the credibility of these datasets within their respective fields. Moreover, we employed rigorous preprocessing techniques to ensure data quality.

Experiment and results

As we introduced previously, our suggested hybrid model comprises six steps: In the first stage, we have selected two big datasets, namely Sentiment140 and COVID-19_Sentiments containing the tweets collected from the X platform. In the second phase, we have applied the necessary text pre-processing techniques to remove all the noisy and undesired data in every tweet and prepare the tweet for proper analysis. In the third step, we have implemented the FastText vectorization method for converting every text based-tweet into a numerical vector. Multiple vectorization methods can be found in the literature. However, in this study, we opted for FastText due to the promising experimental results presented in the paper [39], which has been shown that FastText exhibits superior performance compared to other vectorization approach in the domain of sentiment analysis. During the fourth stage, we employed a CNN to extract the most relevant features through the convolution layer and subsequently reduce their dimensionality using the pooling layer.

During the fifth phase, we applied fuzzification to the outputs of dense layer using the Gaussian fuzzifier. We opted for the Gaussian fuzzy set function over the Triangular and Trapezoidal alternatives based on empirical results from the paper [54]. These results clearly demonstrate that the Gaussian fuzzy set function outperformed the others in terms of performance. Once the data undergo fuzzification, we proceeded to employ C4.5 with fuzzy information to create the fuzzy tree. Subsequently, the general fuzzy reasoning technique was utilized to classify new tweets, utilizing the fuzzy rule base extracted from the newly formed fuzzy tree. Based on the experimental findings presented in the paper [58], we chose to utilize the GFR technique over the CFR technique. The results from the study demonstrate that the general fuzzy reasoning technique offers superior performance, solidifying our decision to adopt it. In the sixth phase, Our suggested hybrid approach has been put into practice on a Hadoop cluster consisting of five machines, with one acting as the master computing node, and the remaining four functioning as slaves or computing nodes. This section presents multiple experiments to confirm our suggested hybrid model's accuracy and performance by comparing it with other models chosen from the literature. And to judge its effectiveness and performance, we have picked nine assessment criteria which are: *Time Consumption* (TC) represent the total time used for running our proposed model, *True Positive Rate* (TPR), *False*

Positive Rate (FPR),*True Negative Rate* (TNR), *Kappa Statistic* (KS), *Precision* (PR), *False Negative Rate* (FNR), *Error Rate* (ER), *Accuracy* (AC), and *F1-score* (FS) [54].

In this empirical analysis, we used a standard train-validation-test (t/v/t) split technique to guarantee the fairness and reliability of our experimental findings. The dataset was divided into 70% for training, 15% for validation, and 15% for testing. This split was uniformly applied across all tested approaches. Also, our experiments were conducted using Google Colab Pro+, which provides longer runtimes, more frequent access to premium GPUs/TPUs (such as the T4 or V100), and faster hardware. In our case, we utilized NVIDIA T4 GPUs and a high-RAM environment, which provides more memory (up to around 25GB). Each algorithm was executed under the same hardware configuration, ensuring equal access to computational resources, including GPU cores, memory, and storage. To guarantee fairness, we imposed similar runtime constraints per core across all algorithms, preventing any advantage from computational resources or time, particularly for more computationally demanding algorithms like CNN.

Comparison of our approach with baseline models

In this section, we will present the experimental outcomes of our hybrid framework. These results were obtained by applying our approach as well as other methods, including CNN, C4.5, fuzzy C4.5, fuzzy rule-based systems, BART-large-mnli (zero-shot), RoBERTa (few-shot), and DistilBERT (fine-tuned), on two specifically selected datasets with a time per core limit of 7200 s. To assess the efficiency and superiority among these approaches, we calculated accuracy, F1-score, Recall, precision, and average time as evaluation metrics, as illustrated in the following Table 4.

From Table 4, it is evident that our hybrid framework surpasses all other algorithms in terms of accuracy, recall, precision, and F1 score across both datasets. In other words, the hybrid method demonstrates superior efficiency in classifying new instances

Table 4 The results derived from the ablation studies rely on metrics such as Accuracy, Recall, Precision, F1-score, and Average time

Dataset	Techniques	Accuracy	Recall	Precision	F1-score	A. time (s)
dataset2	Our model	95.15	94.72	94.56	94.63	7.65
	CNN	82.46	79.35	80.02	77.96	2.54
	C4.5	65.34	64.87	66.55	63.98	4.48
	Fuzzy C4.5	79.65	78.41	79.21	77.86	5.98
	FRS	86.50	85.64	84.78	85.36	4.36
	BART-large-mnli (zero-shot)	93.76	92.46	93.18	92.56	420
	RoBERTa (few-shot)	94.64	95.06	94.02	93.98	196
	DistilBERT (fine-tuned)	90.40	93.03	87.98	90.43	1500
dataset1	Our model	93.73	91.83	92.31	92.06	9.42
	CNN	81.30	76.46	77.72	75.39	4.91
	C4.5	63.84	62.27	64.54	61.32	6.47
	Fuzzy C4.5	78.10	76.11	76.96	75.36	7.32
	FRS	84.89	83.24	82.68	82.96	6.58
	BART-large-mnli (zero-shot)	92.17	91.02	92.22	90.87	609.25
	RoBERTa (few-shot)	92.65	92.19	92.18	91.77	315
	DistilBERT (fine-tuned)	89.54	91.32	86.99	89.75	2280

compared to each individual method. Additionally, our experiments show that our approach outperforms the implemented transformers (RoBERTa, BART, and DistilBERT) across different evaluation methods (zero-shot, few-shot, and fine-tuned scenarios). This indicates that our proposed hybrid approach effectively balances efficiency and long-range dependency modeling in natural language processing (NLP) by optimizing the trade-offs between computational resources and the ability to capture complex relationships within text.

The average time results show a significant variation across the different techniques. For dataset 2, CNN is the most time-efficient, with an execution time of just 2.54 s, followed by C4.5 at 4.48 s and FRS at 4.36 s. In contrast, our model takes 7.65 s, which is relatively higher but still far more efficient than models like RoBERTa (few-shot) and DistilBERT (fine-tuned), which take 196 s and 1500 s, respectively. Similarly, for dataset 1, the fastest models remain CNN (4.91 s) and C4.5 (6.47 s), with our model showing a time of 9.42 s. DistilBERT has the slowest execution time at 2280 s for this dataset. While our model has a higher execution time than some classical approaches like CNN and FRS, it is much faster compared to large, transformer-based models like RoBERTa and DistilBERT.

Sensitivity analysis of different algorithms

The Table 5 shows the sensitivity analysis of different algorithms that are part of our hybrid model for sentiment analysis. It assesses the effect of preprocessing, FastText, CNN, and fuzzy logic on the hybrid model's performance employing accuracy, recall, precision, and F1-score. The empirical findings proves that the preprocessing is important as the model attaining 94.02% accuracy when preprocessing techniques are applied, in comparison to only 64.41% without them. Another thing that greatly affects the performance is using FastText which after creating high quality word embeddings capturing semantic meaning results in an accuracy of 91.34%. In contrast, performance decreases to 78.61% if FastText is removed from the hybrid model. CNN is another critical component, as it improves the accuracy to 82.46%, due to its ability to extract local features and dependencies in the data. Without CNN, the model's performance drops notably to 65.94%. Fuzzy logic is effective in handling uncertainties and making decisions in ambiguous cases, contributing to an

Table 5 The sensitivity analysis of all techniques used in our approach is based on empirical findings and evaluated using metrics such as accuracy, recall, precision, and F1-score

Techniques	Accuracy	Recall	Precision	F1-score
With preprocessing	94.02	94.65	95.51	93.98
Without preprocessing	64.41	63.18	65.35	63.54
With FastText	91.34	92.34	91.98	92.70
Without FastText	78.61	79.45	78.23	77.01
With CNN	82.46	79.35	80.02	77.96
Without CNN	65.94	64.22	63.87	64.00
With fuzzy logic	86.50	85.64	84.78	85.36
Without fuzzy logic	71.12	72.04	71.97	70.64
Our model	95.15	94.72	94.56	94.63

accuracy of 86.50%. Without fuzzy logic, the accuracy decreases to 71.12%. Overall, the hybrid model that incorporates all techniques-preprocessing, FastText, CNN, and fuzzy logic-delivers the highest performance, with an accuracy of 95.15%. This demonstrates the strength of combining these methods, as each contributes to improving the overall classification results in sentiment analysis.

Comparison of our approach with others

Within this section, we provide a third comprehensive overview of the performance of our proposed hybrid approach, which integrates multiple preprocessing tasks, the FastText method, CNN, Gaussian fuzzy set function, C4.5 algorithm, general fuzzy reasoning technique, and the Hadoop framework. We evaluate the effectiveness of the introduced approach by assessing the nine mentioned evaluation criteria, in comparison to other hybrid approaches chosen from the existing literature. Furthermore, we assess the effectiveness of our approach by calculating its complexity, stability, convergence, and consumption time. In this study, we have successfully implemented and applied four approaches from the existing literature to both the *dataset1* and *dataset2*. The objective was to compare their effectiveness with our contribution. The description of the implemented approaches is provided below:

- *Paper* [14]: Xu et al. described a Sentiment classification model using harmony random forest and harmony gradient boosting machine over product review datasets.
- *Paper* [22]: Liao et al. introduced a new model CNN for situations understanding based on sentiment analysis of X data.
- *Paper* [34]: Maheswari et al. introduced a new procedure that combines fuzzy theory and NLP techniques for computing the word polarity scores.
- *Paper* [39]: Es-sabery et al. proposed a new classifier named "A MapReduce Opinion Mining for COVID-19-Related Tweets classification Using Enhanced ID3 Decision Tree Classifier".

During the initial experiment, we evaluated the error rate (ER %) and accuracy (AC %) of our proposed methodology in comparison to four alternative methods selected from existing literature.

Figure 6 displays the practical findings achieved in terms of classification rate through the implementation our suggested methodology. We compare the efficacy of our proposed methodology with that of the aforementioned classifiers over Dataset1 and Dataset2. This comparison is undertaken to demonstrate the effectiveness of our introduced methodology.

In Fig. 6, the study by ref. [14] shows lower classification rates (33.09% for dataset1 and 51.44% for dataset2) compared to other models. This under-performance is due to insufficient text preprocessing and the use of TF-IDF, which is less effective than GloVe, Word2Vec, and FastText. The direct application of the naive Bayes algorithm without feature extraction further harmed their results.

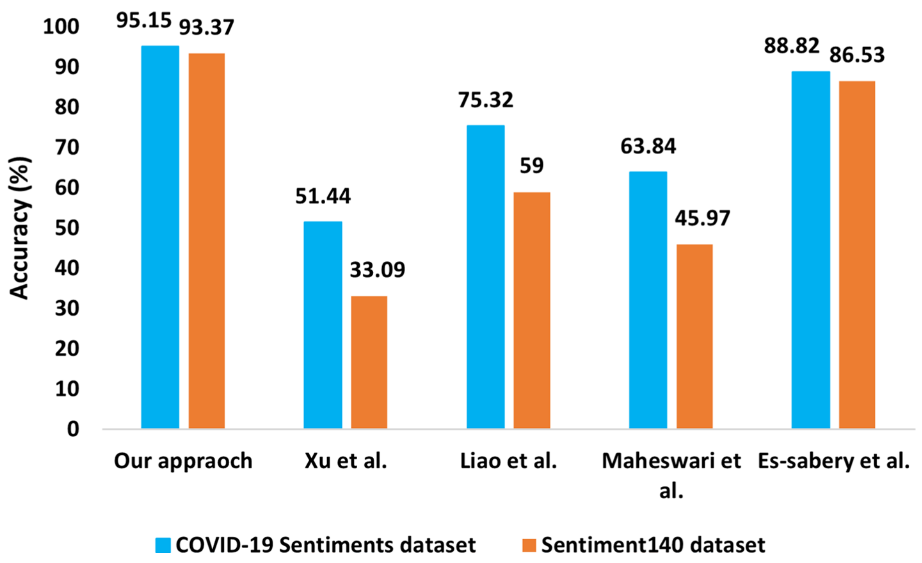


Fig. 6 Accuracy achieved through the implementation of our suggested hybrid approach and alternative methods

Maheswari et al. [34] achieved better results (45.97% and 63.84%) due to effective preprocessing and the use of the Mamdani fuzzy approach. However, their performance is limited by the AFINN dictionary and some missing preprocessing steps.

Liao et al. [22] outperformed both previous methods with accuracies of 75.32% and 59% by employing extensive preprocessing and CNN. Despite this, their use of one-hot-vector embedding and lack of fuzzy logic lead to lower accuracy compared to our proposed method.

Es-sabery et al. [39] achieved the highest accuracy (86.53% and 88.82%) due to their comprehensive approach, including effective preprocessing, FastText embedding, and Information Gain for feature selection. However, their accuracy is still lower than our hybrid model.

Our proposed hybrid approach excels with accuracies of 95.15% and 93.37% by integrating comprehensive preprocessing, FastText embedding, CNN, Gaussian membership functions, and the fuzzy C4.5 method. Additionally, we leverage Hadoop for enhanced computational efficiency.

In Fig. 7, The experimental results regarding the error rate are presented for different classifiers, including our proposed hybrid approach and the other selected methods. These results were obtained from evaluations conducted on Dataset1 and Dataset2.

Our hybrid model achieves error rates of 6.63% and 4.85% for dataset1 and dataset2, respectively, significantly outperforming other methods. In comparison, the Es-sabery et al. model, which previously had the best performance, shows higher error rates of 13.47% and 11.18% on the same datasets. This demonstrates a clear improvement of approximately 6.84% on dataset1 and 6.33% on dataset2. The significant reduction in error rates with our hybrid model demonstrates a marked improvement in accuracy, indicating superior data classification. This enhancement

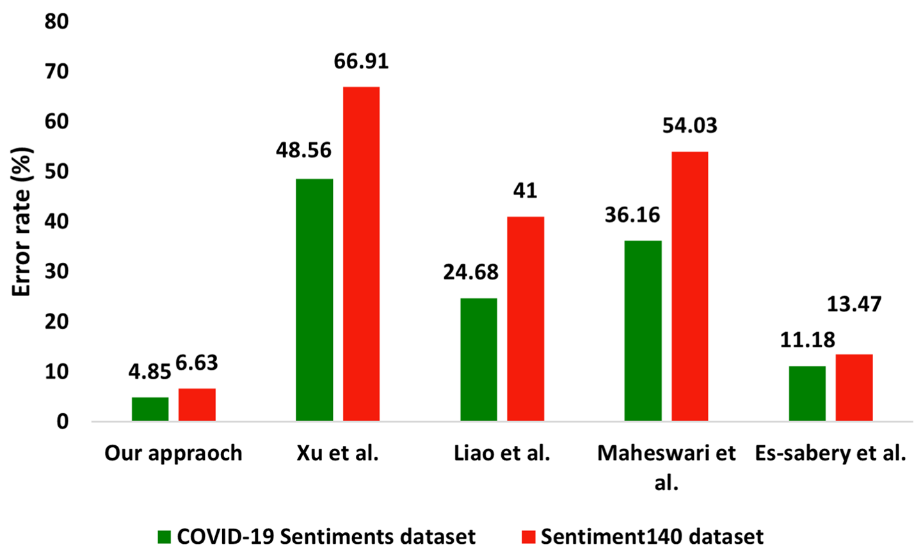


Fig. 7 Error rate resulting from the implementation of our proposed hybrid method and other models

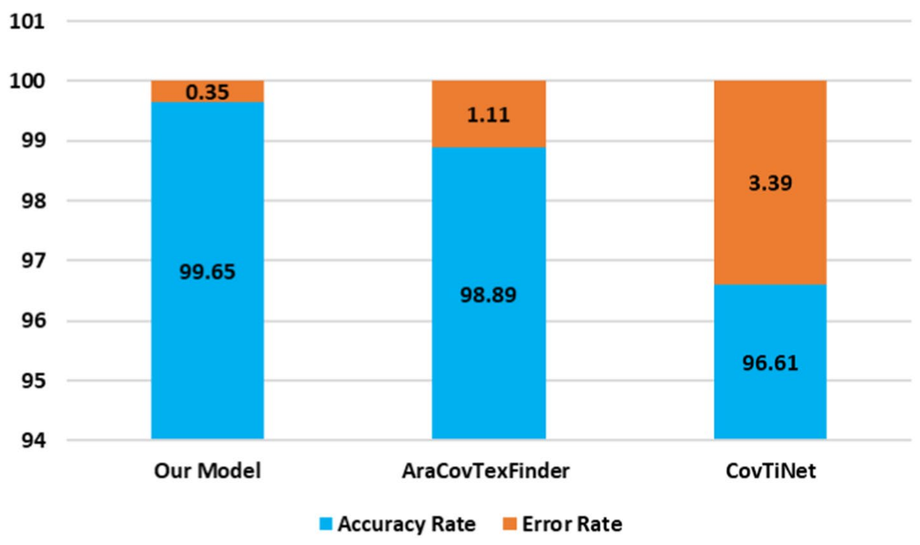


Fig. 8 Error and Accuracy rates resulting from our model and other transformer-based approaches

reflects the successful integration of advanced techniques, resulting in better performance compared to existing methods.

Comparison of our approach with other transformer-based approaches

In this section, we compare our proposed approach with AraCovTexFinder [31] and CovTiNet [32] for the identification of COVID-19-related text. Both AraCovTexFinder and CovTiNet utilize advanced machine learning techniques tailored for Arabic-language datasets; however, our approach introduces a hybrid model that integrates convolutional neural networks (CNN) with fuzzy C4.5 decision trees to enhance classification accuracy. By leveraging this combination, we aim to improve the identification and classification of COVID-19 textual data, especially in handling short and informal text common in social media platforms. In Fig. 8, the experimental results regarding the error and accuracy rates of our model, along with the other transformer-based approaches.

Our Model outperforms both AraCovTexFinder and CovTiNet, demonstrating a significantly higher accuracy rate and a lower error rate. The performance difference indicates that the hybrid approach used in Our Model is more effective in handling the complexities of COVID-19 text identification. While AraCovTexFinder performs relatively well, with less than a 1% drop in accuracy compared to Our Model, CovTiNet shows a noticeable performance gap, with the lowest accuracy and the highest error rate among the three models.

Our approach outperforms CovTiNet and AraCovTexFinder by combining CNN, fuzzy logic, and the fuzzy C4.5 decision tree, allowing it to handle ambiguous and unstructured COVID-19 text more effectively. Unlike CovTiNet's focus on CNN and AraCovTexFinder's use of transformers, our model addresses uncertainty and noisy data with fuzzy logic, ensuring higher accuracy. Additionally, its Hadoop-based architecture improves scalability, making it more efficient at processing large datasets, leading to superior performance in COVID-19 text identification.

Execution time

In this section, we analyze the training execution time of our proposed approach and the comparison models from the literature. This assessment is conducted on both the dataset2 and dataset1. In Fig. 9, the experimental results regarding the execution time of our suggested hybrid classifier, along with the other analyzed methods, are presented for both the dataset1 and dataset2.

As illustrated in Fig. 9, our proposed hybrid model requires 108.27 s and 58.56 s to execute on dataset1 and dataset2, respectively. In comparison, the classifier developed by Es-sabery et al. achieves the shortest execution times of 45.21 s and 15.95 s for dataset1

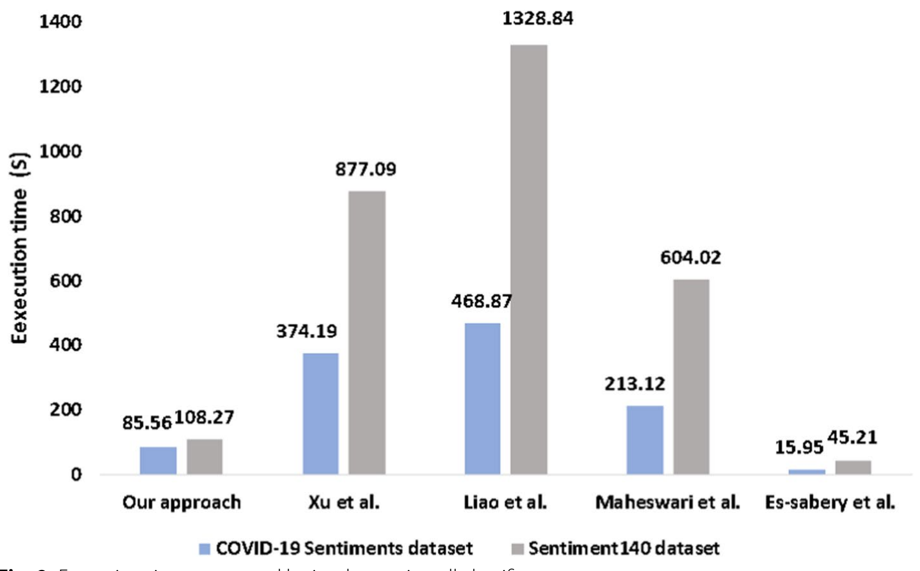


Fig. 9 Execution time consumed by implementing all classifiers

and dataset2. The differences in execution times are due to the complexity of our hybrid model, which uses multiple algorithms including CNNs and fuzzy techniques, leading to longer processing times. In contrast, the Es-sabery et al. classifier, with its simpler approach, executes faster. This makes Es-sabery *et al.'s* model preferable for applications needing quick responses, while our model, though slower, offers better accuracy and functionality. In order to diminish the execution time demanded by our proposition, we systematically augmented the quantity of computing nodes within the Hadoop cluster, as illustrated in the Fig. 10

Figure 10 illustrates the results of our proposition implementation on multiple Hadoop computing nodes, reflecting the experimental outcomes. Thus, as shown in Fig. 10, the execution time of our suggested hybrid model decreased significantly from 108.27 s and 85.56 s for the dataset1 and dataset2, respectively, employing five computing nodes to 20.23 s and 11.81 s for the same datasets, respectively, employing twelve computing nodes. Following the completion of this third experiment, it is evident that our proposed hybrid approach exhibits a shorter execution time compared to all the other examined methods.

Space and time complexity

In this experiment, we evaluated the efficacy of our proposition alongside the other selected approaches, which were chosen from the literature. We evaluated the training process for both the dataset2 and dataset1 by quantifying their space and time complexity. The experimental results concerning the complexity in terms of space are displayed in the Table 6. This table includes measurements of the storage capacity occupied by parameter allocation and the execution of instructions for both our proposal and all other classifiers.

As shown in Table 6, our developed hybrid approach executed a considerable number of instructions, employing a memory space of 39.80 M and 26.01 M for the dataset1 and dataset2, respectively. Additionally, the storage capacity designated by the parameters of our proposal amounts to 25.06 M and 18.75 M for the dataset1 and dataset2, respectively. In accordance with the results obtained from the experiments, our proposal exhibits lower memory space compared to DLMNN, SDLBSA,

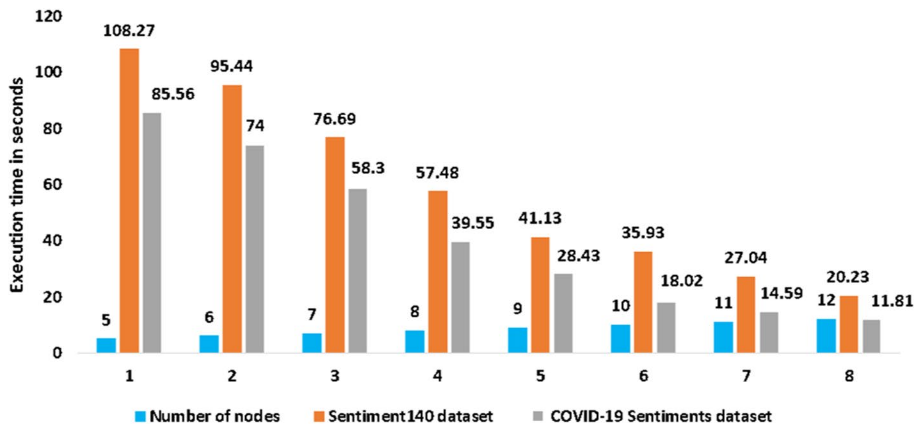


Fig. 10 Execution time consumed through the implementation our suggested hybrid approach on different number of computing nodes in the Hadoop cluster

Table 6 Complexity in terms of space of our proposal and other classifiers

Name of dataset	Techniques	No. instructions (M)	No. parameters (M)
dataset2	Our approach	26.01	18.75
	Paper [14]	9.79	5.34
	Paper [22]	17.62	15.98
	Paper [34]	14.47	10.63
	Paper [39]	12.60	9.57
	DLMNN [29]	31.31	25.02
	SDLBSA [30]	29.56	22.45
	FGNN [42]	27.16	20.81
dataset1	Our approach	39.80	25.06
	Paper [14]	19.87	10.94
	Paper [22]	25.76	23.41
	Paper [34]	21.37	16.57
	Paper [39]	29.10	20.19
	DLMNN [29]	50.42	40.98
	SDLBSA [30]	49.02	34.25
	FGNN [42]	39.97	26.59

and FGNN approaches but it exhibits significantly higher memory space usage when compared to the rest of approaches. Because our approach employs multiple techniques to conduct sentiment analysis, including accurate application of all necessary preprocessing tasks, implementation of FastText, CNN, Gaussian Fuzzifier, and fuzzy C4.5, as well as the GFR method. In contrast to other approaches that utilized a maximum of three techniques for opinion mining, these methods achieved inferior performance in terms of AC, ER, TPR, TNR, KS, FNR, FS, PR, and FPR when comparing them to our suggested hybrid model.

The proposed model's higher memory usage is a trade-off for its substantial performance benefits, including superior accuracy and efficient data processing. While the increased memory requirements may necessitate more robust hardware or cloud resources, the enhanced classification accuracy justifies these costs for precision-critical applications. Optimizing memory usage is important, and we will explore strategies such as refining algorithms, employing memory-efficient data structures, and applying model compression techniques in future work to reduce memory overhead while maintaining performance.

The experimental results regarding time-related intricacy, including the testing and training time expended by our proposal and all other classifiers, are presented in Table 7.

As indicated in Table 7, our developed hybrid approach required a training time of 16.21 s and 9.62 s for the dataset1 and dataset2, respectively. Furthermore, our developed hybrid approach required a testing time of 4.023 s and 2.181 s for the dataset1 and dataset2, respectively. The practical results reveal that our proposed hybrid approach demonstrates considerably lower time-complexity in comparison with all other approaches. The excellent time-complexity performance achieved by our proposal can be attributed to its utilization of a Hadoop cluster comprising twelve computing nodes: single master

Table 7 Complexity in terms of time of our proposal, and other selected classifiers

Name of datasets	Techniques	Training time (s)	Testing time (s)
dataset2	Our approach	9.62	2.181
	Paper [14]	336.77	37.42
	Paper [22]	421.98	46.88
	Paper [34]	191.80	21.31
	Paper [39]	11.96	3.98
	DLMNN [29]	300.24	70.65
	SDLBSA [30]	457.20	89.32
	FGNN [42]	234.15	37.98
dataset1	Our approach	16.21	4.023
	Paper [14]	789.38	87.71
	Paper [22]	1189.32	139.53
	Paper [34]	542.98	61.04
	Paper [39]	33.90	11.30
	DLMNN [29]	801.63	110.45
	SDLBSA [30]	650.42	94.10
	FGNN [42]	462.31	87.77

node and eleven slave nodes, as depicted in Fig. 5. As follows, the description of the time complexity for each technique used in our hybrid model.

Hadoop MapReduce framework: The time complexity is $O(n/m)$, where n is the dataset size and m is the number of nodes. This reduces the processing time by parallelizing the workload across multiple nodes, resulting in faster training and testing times.

HDFS (Hadoop Distributed File System): HDFS has a time complexity of $O(r \cdot \log(m))$, where r is the replication factor and m is the number of nodes. It ensures reliable data access and replication, slightly increasing training time but enhancing fault tolerance and reliability.

FastText: The time complexity is $O(n \cdot d)$, where n is the dataset size and d is the embedding dimension. FastText efficiently handles large datasets and generates word vectors quickly, reducing training time.

CNN (convolutional neural network): The time complexity of a convolutional layer is $O(k^2 \cdot c \cdot h \cdot w)$, where k is the filter size, c is the number of channels, and $h \times w$ are the height and width of the input feature map. Although CNNs are computationally intensive, they extract high-level features, improving classification accuracy.

Fuzzy C4.5 Algorithm: The time complexity is $O(n \cdot \log(n))$, where n is the number of data points. This algorithm improves prediction accuracy with some added computational cost due to recursive partitioning.

Overall Time Complexity: The hybrid model's overall time complexity is $O((n/m) + r \cdot \log(m) + n \cdot d + k^2 \cdot c \cdot h \cdot w + n \cdot \log(n))$. This combines the complexities of each component, balancing performance and accuracy. The following Table 8 includes detailed calculations along with the measured training and testing times.

Convergence

Within the context of the this experiment, we evaluated the effectiveness of our proposal alongside the other selected approaches, all chosen from the existing literature.

Table 8 Time complexity and training/testing time analysis

Technique	Time complexity	Training time (ms)	Testing time (ms)
MapReduce	$O(n/m)$	1500	700
HDFS I/O	$O(r \cdot \log(m))$	750	300
FastText	$O(n \cdot d)$	1250	200
CNN	$O(k^2 \cdot c \cdot h \cdot w)$	3000	600
Fuzzy C4.5	$O(n \cdot \log(n))$	2500	800
Total Hybrid Model	$O((n/m) + r \cdot \log(m) + n \cdot d + k^2 \cdot c \cdot h \cdot w + n \cdot \log(n))$	9600	2600

The assessment involved training over both dataset2 and dataset1, and we demonstrated the convergence of each assessed approach employing eq. (4). This allowed us to identify the count of iterations at which the evaluated approach satisfied the condition outlined in the eq. (4).

$$\mathrm{Error}_{\mathrm{ratep}} - \mathrm{Error}_{\mathrm{ratec}} \geq \mathrm{T}_{\mathrm{value}} \tag{4}$$

Where $\mathrm{Error}_{\mathrm{ratep}}$ denotes the mean error rate attained by the analyzed method during the preceding iteration of the learning procedure. $\mathrm{Error}_{\mathrm{ratec}}$ quantifies the mean rate of inaccuracies of the assessed approach at the present round of the learning procedure. And $\mathrm{T}_{\mathrm{value}}$ indicates the boundary rate that instigated the convergence process. Following the execution of several analyzed experiments, we established the threshold rate at 0.0001. Consequently, we estimated the mean error rate for each evaluated approach employing the succeeding eq. (5):

$$\mathrm{E} = \frac{1}{2} \times \frac{\sum_{j=1}^{I} \sum_{i=1}^{D} (z - z_{\mathrm{label}})^2}{I} \tag{5}$$

Where I indicates the overall number of saved examples within the pretrained dataset, D denotes the cumulative number of class attributes labels within the utilized dataset, z denotes the anticipated and obligatory class attribute label resulting from the classification procedure's output, and z_{label} signifies the acquired label of the class feature upon reaching the classification procedure's result. If the condition stated in eq. (5) is satisfied, we declare the trained approach as converged, and the algorithm continues execution until the mean error rate of the trained approach fulfills the specified condition. Conversely, if the condition is not met, we classify the trained method as not converged.

In Fig. 11, we illustrate the rate at which convergence occurs of our developed hybrid methodology during its execution over the dataset1 and dataset2. As depicted in Fig. 11, we observed that our proposal achieved convergence towards the boundary ratio value of 0.0001 after 254 and 75 rounds when applied to the dataset1 and dataset2, respectively, using our suggested hybrid model algorithm.

Table 9 presents the convergence iteration comparison of our developed hybrid approach with other selected approaches from literature. Based on the result presented in Table 9, it is evident that our developed hybrid approach demonstrates significantly faster convergence than the other assessed methods. This deduction is corroborated by the reality that our approach exhibits a lower misclassification value when compared to the analyzed alternatives.

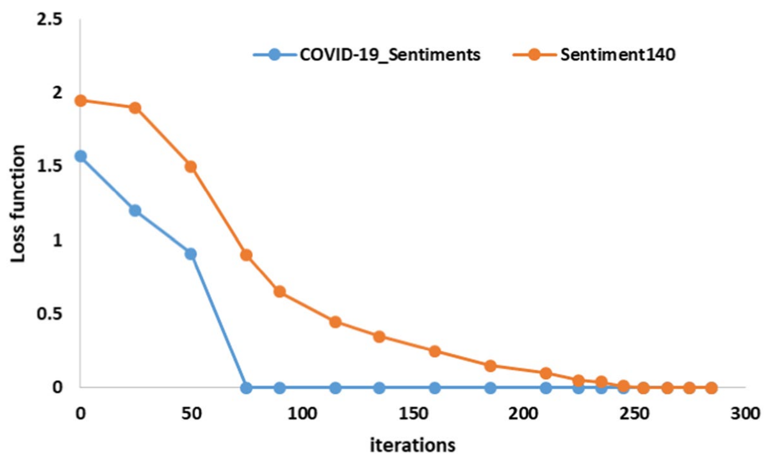


Fig. 11 Convergence rate of our proposal when it was performed over dataset2 and dataset1

Table 9 Convergence rounds of our proposal and other analyzed classifiers

Datasets	Algorithms	Iterations
dataset2	Our approach	75
	Paper [14]	675
	Paper [22]	351
	Paper [34]	514
	Paper [39]	90
dataset1	Our approach	254
	Paper [14]	2198
	Paper [22]	1347
	Paper [34]	1775
	Paper [39]	270

Our proposed model achieves faster convergence due to several key factors. Firstly, the hybrid architecture combines FastText for efficient word representation, CNNs for high-level feature extraction, and fuzzy C4.5 for accurate classification, all of which contribute to a more streamlined learning process. Additionally, the use of adaptive learning rates and momentum-based optimization techniques enhances the model's ability to converge quickly by adjusting the learning trajectory dynamically. The efficiency of data processing, facilitated by Hadoop MapReduce and HDFS, further supports rapid convergence by ensuring that data handling does not become a bottleneck. Empirically, as shown in Fig. 11 and Table 9, our model converges significantly faster than other methods-achieving convergence in 75 iterations on dataset2 and 254 iterations on dataset1, compared to much higher iterations for the other models listed. This faster convergence not only indicates a reduction in training time but also correlates with improved performance, as evidenced by lower misclassification rates. These findings suggest that our model's efficient design and optimization strategies lead to quicker learning and better overall effectiveness compared to traditional approaches.

Stability

Based on this experimental phase, we conducted a performance evaluation of our developed hybrid approach alongside the other selected approaches. The assessment involved computing the average standard deviation (MSD) for each method over the provided dataset employing five instances of cross-validation. The primary aim of this experiment is to identify the most stable methods compared to all the assessed methods.

Table 10 denotes the average accuracy (AVA) and the average deviation standard of our proposal in comparison to the selected classifiers. These values were acquired through five rounds of cross-validation applied to both datasets utilized in this study.

As indicated in Table 10, we conclude that our developed hybrid approach exhibits greater stability when compared to the other selected approaches across five different cross-validations. The validity of this conclusion is substantiated by the reality that our developed hybrid approach achieved higher average accuracy rates of 95.15% and 93.37% while maintaining lower average standard deviations of 0.09% and 0.18% when applied to the dataset2 and dataset1, respectively.

The stability analysis presented in Table 10 demonstrates that our proposed model achieves high accuracy with a low standard deviation across different datasets. This indicates that the model performs consistently well and exhibits minimal variability in its predictions. Specifically, the high accuracy suggests that the model is effective at capturing the underlying patterns in the data, while the low standard deviation reflects its robustness and reliability.

Our model shows stable performance across diverse datasets, demonstrating robustness to noisy and varied data. This reliability ensures consistent results in real-world applications, making it suitable for scenarios where high accuracy and low variance are crucial, such as financial forecasting, medical diagnostics, and sentiment analysis.

Scalability

The scalability assessment in Table 11 of the hybrid model, based on increasing dataset sizes, reveals a clear trend: accuracy decreases and processing time increases as the dataset size grows. The model achieves its highest accuracy (98.97%) with the smallest dataset (IMDB movie with 25,000 reviews) and its lowest (93.73%) with the largest dataset (Sentiment140 with 1,600,000 reviews). Processing times also increase correspondingly,

Table 10 Stability of our proposal and other classifiers over five rounds of cross-validation

Datasets	Algorithms	AVA (%)	MSD (%)
dataset2	Our approach	95.15	0.09
	Paper [14]	51.44	1.67
	Paper [22]	75.32	0.54
	Paper [34]	63.84	1.12
	Paper [39]	88.82	0.12
dataset1	Our approach	93.37	0.18
	Paper [14]	33.09	2.34
	Paper [22]	59.00	0.98
	Paper [34]	45.97	2.03
	Paper [39]	86.53	0.26

Table 11 Scalability assessment of our hybrid model based on increasing dataset sizes

Dataset name	Data size	Accuracy (%)	Processing time (s)
IMDB movie	25,000 reviews	98.97	5.72
Movie Short V2	200,000 reviews	98.16	6.61
COVID-19_Sentiments	259,458 reviews	95.15	7.65
Academic Yelp	1,000,000 reviews	95.97	8.87
World Cup 2014	1,415,958 reviews	94.86	9.16
Sentiment140	1,600,000 reviews	93.73	9.42

from 5.72 s for the IMDB movie dataset to 9.42 s for Sentiment140. This indicates that while the model performs exceptionally well with smaller datasets, it faces challenges with both accuracy and efficiency as data volume increases, underscoring the need for optimization to better handle large-scale data in future work.

Cross-domain application

According to this experiment, we implemented our proposed algorithm along with various machine learning models including C4.5 decision tree, Naive Bayes (NB), Support Vector Machine (SVM), Random Forest (RF), convolutional neural network (CNN), Feedforward Neural Network (FFNN), and Recurrent Neural Network (RNN). We applied these models to three distinct domains: Movie Recommendation Systems using the IMDB dataset, Educational Quality Assessment using Academic Yelp, and Mobile Apps and Games related to the World Cup 2014.

As shown in Table 12, our proposed algorithm outperforms all other algorithms in terms of classification rate (IMDB: 98.97%, Movie Short V2: 98.16%, Academic Yelp: 95.97%, World Cup2014: 94.86%) and running time (IMDB: 5.72s, Movie Short V2: 6.61s, Academic Yelp: 8.87s, World Cup2014: 9.16s) across all used datasets.

In the context of Movie Recommendation Systems using the IMDB dataset, our model demonstrated exceptional performance with a classification rate of 98.97% and a running time of just 5.72 s. This high accuracy reflects the model's ability to effectively capture complex patterns in movie ratings, while the efficient running time underscores its computational efficiency compared to other models. For Educational Quality Assessment using the Academic Yelp dataset, our model achieved a classification rate of 95.97% and completed the task in 8.87 s. This performance indicates the model's robustness in managing diverse and nuanced feedback data, surpassing other methods in accuracy and efficiency. In the domain of Mobile Apps and Games related to the World Cup 2014, our algorithm reached a classification rate of 94.86% with a running time of 9.16 s. This result highlights the model's adaptability and superior performance in a domain characterized by potentially diverse user interactions.

Table 12 Classification rates and running times of various algorithms across different datasets

Datasets	Algorithms	Classification rate (%)	Running time (s)
IMDB movie	C4.5	60.07	90.30
	NB	78.65	100.46
	SVM	82.32	97.65
	RF	58.24	101.54
	CNN	81.08	98.72
	FFNN	74.63	79.95
	RNN	83.15	86.69
	Our model	98.97	5.72
Movie Short V2	C4.5	59.12	316.59
	NB	73.15	265.48
	SVM	79.40	218.96
	RF	57.94	318.65
	CNN	79.87	298.54
	FFNN	71.35	243.21
	RNN	78.60	304.32
	Our model	98.16	6.61
Academic Yelp	C4.5	51.36	512.24
	NB	65.21	458.13
	SVM	70.93	435.63
	RF	49.54	536.48
	CNN	76.31	408.64
	FFNN	68.55	399.75
	RNN	76.42	521.77
	Our model	95.97	8.87
World Cup 2014	C4.5	54.65	704.32
	NB	63.12	624.67
	SVM	64.11	618.30
	RF	35.95	599.78
	CNN	71.71	542.81
	FFNN	66.53	600.07
	RNN	65.47	582.15
	Our model	94.86	9.16

For limitations and challenges, the unique characteristics of each dataset-such as their size, data type, and feature distribution-can influence the model's performance. Although our model excels across all domains, it is important to recognize that domain-specific features and data quality can impact results. Additionally, the competitive running times reflect the computational resources required. While our approach achieves a balance between high accuracy and efficient computation, scaling the model to larger datasets or more complex domains may necessitate further optimization.

Statistical significance analysis

In the final experiment, we conducted a statistical significance analysis [60] of the performance differences among all models evaluated across multiple datasets, we used Friedman test which is an appropriate non-parametric statistical procedure for repeated measures. This test is notably well-suited for comparing multiple related groups or conditions, as we saw while evaluating different models over the same datasets. Friedman Test is a non-parametric test that requires no assumptions of normality or homogeneity of variances, so it may be robust for performance metrics which do not follow the traditional statistical requirements. We assessed significant differences in performance of the models by conducting a Friedman Test, considering paired data nature.

For the Friedman test, we first arranged the data as shown in the Table 13, where each row represents a different dataset and each column represents the accuracy metric of a different model.

The results of the Friedman Test show a test statistic of 23.3333 and a p-value of 0.0001, which is far below the significance threshold of 0.05. This indicates that there are statistically significant differences in performance among the models, allowing us to confidently reject the null hypothesis that the models perform similarly. The higher test statistic suggests a substantial disparity in their performance.

Discussion

The proposed hybrid sentiment analysis model, integrating CNN, Fuzzy C4.5, and Hadoop, is designed to handle large-scale data efficiently and accurately classify sentiments in real-time. This system is particularly relevant for analyzing social media data, such as COVID-19-related tweets, and helps organizations better understand public sentiment, improve decision-making, and adapt their strategies accordingly. The main key contributions of this research are summarized below:

- We propose a hybrid approach that combines convolutional neural networks (CNN) for feature extraction with Fuzzy C4.5 for uncertainty management, effectively capturing complex text patterns and generating interpretable rules for sentiment classification.
- We demonstrate that the integration of Hadoops distributed computing significantly enhances the model's scalability, enabling real-time processing of large datasets and reducing computational time compared to traditional models, making it suitable for high-volume data environments like social media monitoring.
- We validate the performance of the hybrid model on a domain-specific dataset of COVID-19-related tweets, revealing superior accuracy in sentiment trend detection, and show that it not only outperforms conventional machine learning methods but also provides interpretable results that support informed decision-making.

Table 13 Accuracies of different models overs different datasets

Models_ datasets	COVID_ Sentiments	Sentiment140	IMDB movie	Movie V2	Academic Yelp	World Cup2014
Our model	0.9515	0.9373	0.9897	0.9816	0.9897	0.9786
CNN	0.8246	0.8130	0.8108	0.7987	0.7631	0.7171
C4.5	0.6534	0.6384	0.6007	0.5912	0.5136	0.5465
Fuzzy C4.5	0.7965	0.7810	0.7732	0.7698	0.7364	0.7212
FRS	0.8650	0.8489	0.8365	0.8135	0.8007	0.7945

Practical Implications: This system is particularly valuable for organizations that need to monitor public opinion and sentiment in real-time. The scalable nature of the model, combined with its accuracy and explainability, makes it an effective tool for tracking evolving trends, such as public attitudes toward health policies during a pandemic.

Theoretical Implications: This study advances the field of sentiment analysis by introducing a novel hybrid model that combines machine learning and fuzzy logic for enhanced performance and interpretability. It also provides a framework for future research in applying such hybrid systems to other domains and datasets, particularly in low-resource language environments. The findings offer valuable insights into the development of scalable, real-time sentiment analysis systems that can be adapted to various applications.

Limitations: The proposed hybrid sentiment analysis model, despite its advancements, has several limitations. It requires careful hyper-parameter tuning, which can be resource-intensive, and its performance is dependent on the quality and diversity of labeled training data, potentially introducing biases. The model struggles with very short texts due to limited context, and while Hadoop's distributed computing enhances scalability, it demands robust infrastructure that may not be widely accessible. Additionally, the interpretability of the fuzzy rules generated can pose challenges for non-technical users, affecting its practical application in real-world scenarios.

Conclusions

In conclusion, the hybrid framework presented in this study for sentiment analysis of tweets showcases a comprehensive and effective approach. The six-stage model, involving data gathering, pretreatment, vectorization, extraction and selection, classification, and fuzzy reasoning, proved to be robust and outperformed existing classifiers in terms of error and accuracy rates. The integration of FastText for data representation, CNN for data extraction, and Gaussian fuzzy set function for classification contributed to achieving high accuracy levels of 95.15%, and 93.37% over dataset1 and dataset2, respectively.

The experiments assessing performance metrics such as TPR(94.72%, and 91.83%), FNR(5.28%, and 8.17%), TNR(95.74%, and 90.58%), FPR(4.26%, and 9.42%), PR(94.56%, and 92.31%), KS(95.12%, and 91.76%), and FS(94.63%, 92.06%) for both dataset 1 and dataset 2, respectively, further supported the superiority of the proposed approach. The model demonstrated favorable results in terms of time complexity, leveraging the Hadoop framework for data parallelization and exhibiting rapid convergence. While the hybrid model consumed more memory, it showcased a lower time complexity compared to other evaluated methods.

Additionally, the stability analysis revealed that the proposed approach outshines other models, showing higher average accuracy rates and lower mean standard deviations. The hybrid model's convergence to a threshold value of 0.0001 at the 75th and 254th learning iterations for dataset1 and dataset2, respectively, highlighted its efficiency.

In summary, the hybrid framework exhibits superior performance across various evaluation metrics, establishing its effectiveness in sentiment analysis of tweets and positioning it as a stable, accurate, and efficient solution compared to existing classifiers.

Our forthcoming research involves replacing the fuzzy C4.5 algorithm with a deep learning pattern for tweet classification. Additionally, we plan to explore and compare the effectiveness of various feature extractors and feature selectors. In this regard, we will also investigate the application of CNN to extract and select features in our contributions. Furthermore, we intend to explore the utilization of a Mamdani fuzzy system to handle the uncertainty and vagueness in the data, as an alternative to the fuzz rule-based model employed in this study.

Appendix A Fuzzy C4.5 equations

In general, each feature within any given dataset contains multiple values, all of which are represented by fuzzy sets in the theory of fuzzy logic. Within this theory, the membership function serves the purpose of describing each fuzzy set. Let us consider that E represents the set encompassing all dataset examples. The total count of existing features in the employed base is denoted by $F^{(n)} = \{n = 1, 2, ..., K\}$ and it is important to note that each feature can have several fuzzy values, which are in turn represented by several linguistic terms or fuzzy sets $F_c^{(n)} = \{c = 1, 2, ..., l\}$, the degree of membership of the vague collection $F_c^{(n)}$ is $D_{F_c^{(n)}}$, the linguistic words $Z_a = \{a = 1, 2, ..., m\}$ are employed to express the decision feature within the dataset in question. We can use the notation D_{Z_a} to represent the membership degree of the vague collection or the linguistic term Z_a.

The membership degree $F_c^{(n)}$ associated with the decision attribute YD of the cth linguistic expression within the nth attribute. The procedure of determining the ath fuzzy value of the decision feature Z_a is performed using the following eq. (6):

$$YD_{F_c^{(n)}}(Z_a) = \frac{\sum_{y \in Y_a} D_{F_c^{(n)}}(y^{(n)})}{\sum_{y \in Y} D_{F_c^{(n)}}(y^{(n)})} \tag{6}$$

Where: Y_a is the ath value of the nth feature which have the linguistic term $F_c^{(n)}$ in the set of training examples and also, it belongs to the instance that has the class value Z_a, Y is a set of the values of nth feature in the set of training examples, which possess the linguistic term $F_c^{(n)}$. $y^{(n)}$ is the value of nth feature of example y and $D_{F_c^{(n)}}(y^{(n)})$ is the computed membership degree of $y^{(n)}$ which is defined by the linguistic term $F_c^{(n)}$. As a result, the calculation of the fuzzy logic-based entropy (EF) for the cth linguistic expression in the nth attribute, denoted by $F_c^{(n)}$, is performed using the eq. (7):

$$EF_{F_c^{(n)}} = -\sum_{a=1}^{m} YD_{F_c^{(n)}}(Z_a) log YD_{F_c^{(n)}}(Z_a) \tag{7}$$

Where $YD_{F_c^{(n)}}(Z_a)$ denotes the membership degree of the feature of the class label YD of the cth linguistical expression of the nth attribute $F_c^{(n)}$ regarding the ath vague value of the class label feature Z_a is computed using the eq. (6). Moreover, fuzzy logic-based entropy (EF) of the nth attribute $F^{(n)}$ is calculated by adding the weighted values of $EF_{F_c^{(n)}}$

.

$$EF_{F^{(n)}} = \sum_{c=1}^{l} \frac{\sum_{y \in Y} D_{F_c^{(n)}}(y^{(n)})}{\sum_{j=1}^{l} \sum_{y \in Y} D_{F_c^{(n)}}(y^{(n)})} . EF_{F_c^{(n)}} \tag{8}$$

Where Y is a set of the values of nth feature in the set of training examples, which possess the linguistic term $F_c^{(n)}$, l is the total number of linguistic term $F_c^{(n)}$, $D_{F_c^{(n)}}(y^{(n)})$ is the computed membership degree of $y^{(n)}$, which is the value of the nth feature of instance y defined by the linguistic term $F_c^{(n)}$, and $EF_{F_c^{(n)}}$ is the fuzzy logic-based entropy (EF) of the cth linguistical expression of the nth attribute $F_c^{(n)}$ measured employing the eq. (8). however, the degree of membership of the class attribute YD of the collection of training tweets, regarding the ath fuzzy value of decision feature Z_a is obtained by employing the subsequent mathematical expression (9):

$$YD(Z_a) = \frac{\sum_{y \in Y_a} D_{Z_a}(y)}{\sum_{y \in Y} D_{Z_a}(y)} \tag{9}$$

Where Y represents a collection of the values of nth attribute in the collection of training instances, which possess the linguistic term $F_c^{(n)}$, Y_a is the ath value of the nth feature which have the linguistic term $F_c^{(n)}$ in the set of training examples and also, it belongs to the instance that has the class value Z_a, and $D_{Z_a}(y)$ is the measured degree-membership of the decision feature value a defined by the linguistic term Z_a in the example y. Hence, the fuzzy logic-based entropy (EF) of the collection of training instances is determined by the subsequent mathematical expression (10):

$$EF = -\sum_{a=1}^{m} YD(Z_a) log YD(Z_a) \tag{10}$$

Where $YD(Z_a)$ the is the degree of membership of the class attribute YD of the collection of tweets, regarding the ath fuzzy value of decision feature Z_a measured employing the eq. (9). Thus, the information fuzzy gain (IGF) of the nth attribute regarding the collection of training instances is calculated using the subsequent eq. (11):

$$IGF_{F^{(n)}} = EF - EF_{F^{(n)}} \tag{11}$$

Hence, the IGF equals the disparity of vague logic-based entropy (EF) of the collection of examples being trained, computed employing eq. (10) and the fuzzylogic-based entropy ($EF_{F^{(n)}}$) of the nth attribute determined using the eq. (8). Therefore, the split information $SI_{F^{(n)}}$ of the nth attribute is calculated by utilizing the subsequent mathematical expression (12):

$$SI_{F^{(n)}} = \sum_{c=1}^{l} -\left(\frac{\sum_{y \in Y} D_{F_c^{(n)}}(y^{(n)})}{\sum_{j=1}^{l} \sum_{y \in Y} D_{F_c^{(n)}}(y^{(n)})} \right) log \left(\frac{\sum_{y \in Y} D_{F_c^{(n)}}(y^{(n)})}{\sum_{j=1}^{l} \sum_{y \in Y} D_{F_c^{(n)}}(y^{(n)})} \right) \tag{12}$$

Where Y is a set of the values of nth feature in the collection of training examples, which possess the linguistic term $F_c^{(n)}$, l is the overall count of linguistic term $F_c^{(n)}$, and $D_{F_c^{(n)}}(y^{(n)})$ is the computed degree of membership of the value $y^{(n)}$ of the nth feature of example y determined by the linguistic term $F_n^{(n)}$. Therefore the information fuzzy gain ratio (IGRF) of the nth attribute is calculated using the subsequent eq. (13):

$$IGRF_{F^{(n)}} = \frac{IGF_{F^{(n)}}}{SI_{F^{(n)}}} \tag{13}$$

Where $IGF_{F^{(n)}}$ is the information fuzzy gain of the nth attribute concerning the collection of training examples computed employing the eq. (11), and $SI_{F^{(n)}}$ represents the split information measured by applying the eq. (12).

Appendix B Proposed fuzzy parallel C4.5 Algorithm

The algorithm 5 and the flowchart Fig. 12 describe our developed fuzzy parallel C4.5 used in this contribution for creating the fuzzy tree by implementing it on training tweet dataset.

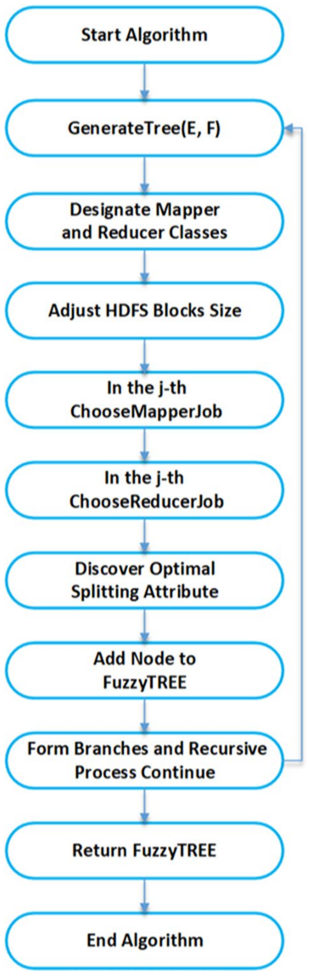

Start Algorithm
GenerateTree(E, F)
Designate Mapper and Reducer Classes
Adjust HDFS Blocks Size
In the j-th ChooseMapperJob
In the j-th ChooseReducerJob
Discover Optimal Splitting Attribute
Add Node to FuzzyTREE
Form Branches and Recursive Process Continue
Return FuzzyTREE
End Algorithm

Fig. 12 Fuzzy Parallel C4.5 Algorithm Flowchart with MapReduce

Algorithm 5 Our suggested Algorithm

```
Require: The dataset of training tweets $E$ which has been fuzzified; $F$ is a set of features.
Ensure: Fuzzy decision tree FuzzyTREE.
 1: GenerateTree($E$, $F$) : Generate a FuzzyTREE with a solitary base node.
 2: Define the job of Hadoop
 3: Designate the Mapper class as ChooseMapperJob
 4: Designate the Reducer class as ChooseReducerJob
 5: Altering the size of each block of HDFS is necessary to enable the division of dataset $E$ into $S$ subsets $E_j = \{j = 1, 2, \ldots, S\}$
 6: In the $j$-th ChooseMapperJob
 7:     Input: $E_j = \{x_1, x_2, \ldots, x_r\}$ where $x_r$ is the $r$-th instance with $K$ features
 8:     $F(n) = \{n = 1, 2, \ldots, K\}$
 9:     Output: (key,value)=(F(n), Ration(F(n), $E_j$))
10: for $n = 1$ to $K$ feature do
11:     Compute $YDF_c(n)(Z_a)$ using the equation (6).
12:     Compute $EFF_c(n)$ using the equation (7).
13:     Compute $EFF(n)$ using the equation (8).
14:     Compute $YD(Z_a)$ using the equation (9).
15:     Compute $EF$ using the equation(10).
16:     Compute $IGFF(n)$ using the equation (11).
17:     Compute $SIF(n)$ using the equation (12).
18:     Compute fuzzy information gain ratio $IGRFF(n)$ using the equation (13).
19:     Ration(F(n), $X_j$) = $IGRFF(n)$
20: end for
21: MapperOutput: (key,value)=(F(n), Ration(F(n), $E_j$))
22: In the $j$-th ChooseReducerJob
23:     Input: (key,value)=(F(n), list([Ration(F(n), $E_j$)], ($j = 1, \ldots, S$)))
24:     Output: (key,value)=(F(n)*, Ration*(E))
25: for $k = 1$ to $n$ Attribute do
26:     Ration(E) = $\sum_{j=1}^{m}$ Ration(F(n), $E_j$).
27: end for
28: Discover the optimal splitting attribute $F(n)*$ with the highest ratio of fuzzy information gain $F(n)* = \arg\max_A \{\text{Ration}(E)\}_{k=1}^{n}$
29: ReducerOutput : (key,value)=(F(n)*, Ration*(E))
30: Add $F(n)*$ to FuzzyTREE;
31: for values $v \in F(n)*$ do
32:     Form a branch for the current node, where $X_v$ denotes a subset of tweets in $E$ with the feature $F(n)*$ being $v$.
33:     if $S_v$ is void then
34:         Label the branch node as a leaf node, and assign it the class that corresponds to the most substantial count of examples in $E$.
35:         return ;
36:     else
37:         Recursive process of GenerateTree($E_v, A \setminus \{F(n)*\}$) continues
38:     end if
39: end for
```

Where:

- E: The training dataset of tweets (Dataset1 or Dataset2), denoted as E, has undergone fuzzification using a Gaussian fuzzifier.
- F: It is a set of features extracted using the CNN, serving as essential elements that guide the construction of fuzzy decision trees.
- *Altering the the size of each block of HDFS*: To ensure an optimal balance in data distribution across computing nodes during the division of our dataset (**E**) into subsets (**S**).
- *Parameter* K **:** K represents the number of features in each instance x_r of a subset E_j. We have chosen K based on the nature of the tweet data and the relevant features for sentiment analysis.
- (key,value)=($F^{(n)}, Ratio_n(F^{(n)}, E_j)$): The output from the MapperJob enables the storage of computed fuzzy information gain in HDFS as a key/value pair, with the instance to classify represented as the key ($F^{(n)}$) in the MapReduce function and the corresponding fuzzy information gain ratio as the value.
- *Equation Selection:* The equations (6 to 13) utilized in the Mapper class play a pivotal role in computing various metrics, such as fuzzy information gain ratio ($IGRF_{F^{(n)}}$). The selection of these equations is grounded in their effectiveness in capturing the nuances of sentiment expressed in individual tweets. Additionally, these equations are chosen based on empirical evidence presented in cited references [39, 54, 58].

Appendix C Flowchart of our proposed parallel hybrid model

See Fig. 13

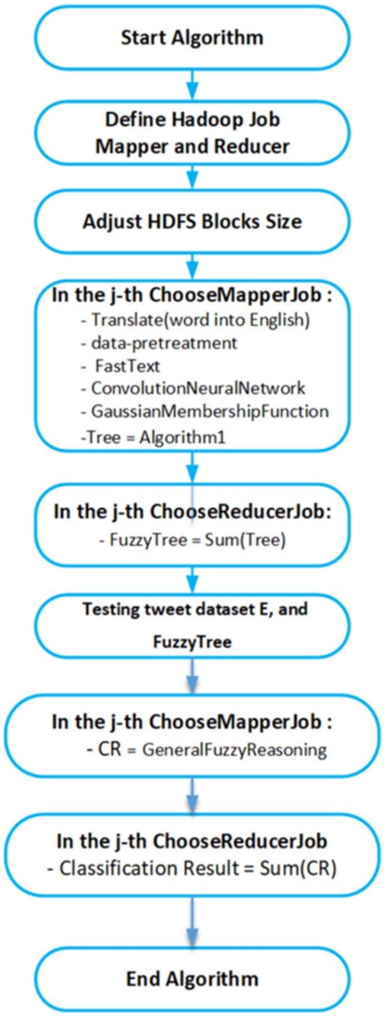

Start Algorithm
Define Hadoop Job Mapper and Reducer
Adjust HDFS Blocks Size
In the j-th ChooseMapperJob :
- Translate(word into English)
- data-pretreatment
- FastText
- ConvolutionNeuralNetwork
- GaussianMembershipFunction
-Tree = Algorithm1
In the j-th ChooseReducerJob:
- FuzzyTree = Sum(Tree)
Testing tweet dataset E, and FuzzyTree
In the j-th ChooseMapperJob :
- CR = GeneralFuzzyReasoning
In the j-th ChooseReducerJob
- Classification Result = Sum(CR)
End Algorithm

Fig. 13 Parallel hybrid model flowchart for tweet classification using Hadoop

Table 14 The results obtained from the experiments include metrics such as TPR, FNR, TNR, FPR, PR, KS, and FS

Name of dataset	Techniques	TPR	FNR	TNR	FPR	PR	KS	FS
dataset2	Our model	94.72	5.28	95.74	4.26	94.56	95.12	94.63
	Paper [14]	60.75	39.25	61.04	38.96	61.29	59.43	61.01
	Paper [22]	74.19	25.81	73.64	26.36	75.10	74.89	74.64
	Paper [34]	62.79	37.21	63.51	36.49	64.07	61.42	63.42
	Paper [39]	85.72	14.28	86.51	13.49	86.67	87.69	85.54
dataset1	Our model	91.83	8.17	90.58	9.42	92.31	91.76	92.06
	Paper [14]	41.02	58.98	40.69	59.31	39.96	40.15	40.48
	Paper [22]	58.97	41.03	59.02	40.98	58.99	58.75	58.97
	Paper [34]	44.98	55.02	45.34	54.66	46.00	44.87	45.48
	Paper [39]	81.41	18.59	82.33	17.67	83.04	84.58	83.87

Appendix D Empirical findings

In order to showcase the effectiveness of our suggested hybrid proposal, we conduct a follow-up experiment where we compute the TPR, TNR, KS, FNR, FS, PR, and FPR metrics. This comparison involves four alternative methods chosen from the existing body of literature. Table 14 presents the experimental results Attained through the implementation of our proposal, along with the other selected classifiers, in terms of TPR, TNR, KS, FNR, FS, PR, and FPR metrics.

As shown in Table 14, our proposed hybrid model demonstrates superior performance compared to all other approaches across various metrics, including True Positive Rate (TPR), True Negative Rate (TNR), Kappa statistic (KS), False Negative Rate (FNR), Precision (PR), and False Positive Rate (FPR). Specifically, our model achieves an impressive TPR of 91.83% and 94.72% on Dataset 1 and Dataset 2, respectively, indicating high accuracy in predicting positive tweets.

Moreover, our model shows a decreased FNR, with values of 8.17% and 5.28% on Dataset 1 and Dataset 2, respectively. A lower FNR reflects fewer misclassifications of positive tweets as negative. Furthermore, our model attains TNR values of 90.58% and 95.74% on Dataset 1 and Dataset 2, respectively, signifying accurate classification of negative tweets.

Regarding the False Positive Rate (FPR) for negative tweets, our model demonstrates significantly lower values compared to other approaches, with rates of 9.42% and 4.26% on Dataset 1 and Dataset 2, respectively. This indicates high accuracy in correctly classifying negative tweets and outperforms alternative methods in this regard.

The Precision (PR) metric, which measures the closeness of predicted positive results to actual positive instances, shows that our model achieves 92.31% and 94.56% on Dataset 1 and Dataset 2, respectively. This surpasses the performance of all other evaluated methods.

Additionally, our model exhibits elevated KS values of 91.76% and 95.12% on Dataset 1 and Dataset 2, respectively. The KS statistic measures the separation between the distributions of observed and expected accuracies, with higher values indicating better performance.

Lastly, the F-Score (FS), which evaluates both accuracy and robustness, yields values of 92.06% and 94.63% on Dataset 1 and Dataset 2, respectively. Based on all computed evaluation metrics, our model outperforms all other classifiers in terms of both power and efficiency.

Acknowledgements
We would like to thank the anonymous reviewers for their constructive feedback, which helps improve our paper signifcantly.

Author contributions
Author A proposed the idea. Authors A and B implemented the idea using Python. Authors A, B, and C wrote the main manuscript and prepared Figs. 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12. All authors (A,B,C,D,E) reviewed the manuscript

Funding
This work is supported by the eVida Research Group, University of Deusto, Bilbao, Spain, under Grant IT 905-16.

Availability of data and materials
Sentiment140 dataset with 1.6 million tweets. https://www.kaggle.com/datasets/ kazanova/sentiment140 covid-19 sentiments India[20/03/20–31/05/20]. https://www.kaggle.com/datasets/abhaydhiman/covid19-sentiments.

Declarations

Ethics approval and consent to participate
Not applicable.

Consent for publication
Not applicable.

Competing interests
The authors declare no competing interests.

Received: 2 July 2024 Accepted: 13 October 2024
Published online: 18 December 2024

References

1. Salur MU, Aydin I. A novel hybrid deep learning model for sentiment classification. IEEE Access. 2020;8:58080–93. https://doi.org/10.1109/ACCESS.2020.2982538.
2. Antonakaki D, Fragopoulou P, Ioannidis S. A survey of twitter research: data model, graph structure, sentiment analysis and attacks. Expert Syst Appl. 2021;164: 114006. https://doi.org/10.1016/j.eswa.2020.114006.
3. Almatrafi O, Parack S, Chavan B. Application of location-based sentiment analysis using twitter for identifying trends towards Indian general elections 2014. In: Proceedings of the 9th International Conference on Ubiquitous Information Management and Communication. 2015. pp. 1–5. https://doi.org/10.1145/2701126.2701129.
4. Sarlan A, Nadam C, Basri S. Twitter sentiment analysis. In: Proceedings of the 6th International Conference on Information Technology and Multimedia. IEEE; 2014. pp. 212–6. https://doi.org/10.1109/ICIMU.2014.7066632.
5. Sharma P, Moh T-S. Prediction of Indian election using sentiment analysis on Hindi twitter. In: 2016 IEEE International Conference on Big Data (big Data). IEEE; 2016. pp. 1966–71. https://doi.org/10.1109/BigData.2016.7840818.
6. Madjid MF, Ratnawati DE, Rahayudi B. Sentiment analysis on app reviews using support vector machine and naïve bayes classification. Politek Ganesha Medan. 2023. https://doi.org/10.33395/sinkron.v8i1.12161.
7. Aslan S, Kızıloluk S, Sert E. TSA-CNN-AOA: twitter sentiment analysis using CNN optimized via arithmetic optimization algorithm. Neural Comput Appl. 2023;35(14):10311–28. https://doi.org/10.1007/s00521-023-08236-2.
8. Wang X, Lyu J, Kim B-G, Parameshachari BD, Li K, Li Q. Exploring multimodal multiscale features for sentiment analysis using fuzzy-deep neural network learning. IEEE Trans Fuzzy Syst. 2024. https://doi.org/10.1109/TFUZZ.2024.3419140.
9. Rahman A, Hossen MS. Sentiment analysis on movie review data using machine learning approach. In: 2019 International Conference on Bangla Speech and Language Processing (ICBSLP), IEEE; 2019. pp. 1–4. https://doi.org/10.1109/ICBSLP47725.2019.201470.
10. Krishna A, Akhilesh V, Aich A, Hegde C. Sentiment analysis of restaurant reviews using machine learning techniques. In: Emerging Research in Electronics, Computer Science and Technology: Proceedings of International Conference, ICERECT 2018. Springer; 2019. pp. 687–96. https://doi.org/10.1007/978-981-13-5802-9_60.
11. Noor F, Bakhtyar M, Baber J. Sentiment analysis in e-commerce using svm on roman Urdu text. In: Emerging Technologies in Computing: Second International Conference, iCETiC 2019, London, UK, August 19–20, 2019, Proceedings 2. Springer; 2019. pp. 213–22. https://doi.org/10.1007/978-3-030-23943-5_16.
12. Ngoc PV, Ngoc CVT, Ngoc TVT, Duy DN. A C4. 5 algorithm for English emotional classification. Evol Syst. 2019;10(3):425–51. https://doi.org/10.1007/978-3-030-23943-5_16.
13. Sridharan K, Komarasamy G. Retracted article: sentiment classification using harmony random forest and harmony gradient boosting machine. Soft Comput. 2020;24(10):7451–8. https://doi.org/10.1007/s00500-019-04370-z.
14. Xu F, Pan Z, Xia R. E-commerce product review sentiment classification based on a naïve bayes continuous learning framework. Inform Process Manag. 2020;57(5): 102221. https://doi.org/10.1016/j.ipm.2020.102221.
15. Ruas T, Ferreira CHP, Grosky W, França FO, Medeiros DMR. Enhanced word embeddings using multi-semantic representation through lexical chains. Inform Sci. 2020;532:16–32. https://doi.org/10.1016/j.ins.2020.04.048.
16. Alfreihat M, Almousa OS, Tashtoush Y, AlSobeh A, Mansour K, Migdady H. Emo-SL framework: emoji sentiment lexicon using text-based features and machine learning for sentiment analysis. IEEE Access. 2024;12:81793–812. https://doi.org/10.1109/ACCESS.2024.3382836.
17. Ramzy M, Ibrahim B. User satisfaction with Arabic COVID-19 apps: Sentiment analysis of users' reviews using machine learning techniques. Inform Process Manag. 2024;61(3): 103644. https://doi.org/10.1016/j.ipm.2024.103644.
18. Chamberlain BP, Rossi E, Shiebler D, Sedhain S, Bronstein MM. Tuning word2vec for large scale recommendation systems. In: Proceedings of the 14th ACM Conference on Recommender Systems. 2020. pp. 732–7. https://doi.org/10.1145/3383313.3418486.
19. Young JC, Rusli A. Review and visualization of Facebook's fasttext pretrained word vector model. In: 2019 International Conference on Engineering, Science, and Industrial Applications (ICESI). IEEE; 2019. pp. 1–6. https://doi.org/10.1109/ICESI.2019.8863015.
20. Sakketou F, Ampazis N. A constrained optimization algorithm for learning glove embeddings with semantic lexicons. Knowl-Based Syst. 2020;195: 105628. https://doi.org/10.1016/j.knosys.2020.105628.
21. Feng Y, Cheng Y. Short text sentiment analysis based on multi-channel CNN with multi-head attention mechanism. IEEE Access. 2021;9:19854–63. https://doi.org/10.1109/ACCESS.2021.3054521.
22. Liao S, Wang J, Yu R, Sato K, Cheng Z. CNN for situations understanding based on sentiment analysis of twitter data. Proced Computer Sci. 2017;111:376–81. https://doi.org/10.1016/j.procs.2017.06.037.

23. Behera RK, Jena M, Rath SK, Misra S. Co-LSTM: convolutional LSTM model for sentiment analysis in social big data. Inform Process Manag. 2021;58(1): 102435. https://doi.org/10.1016/j.ipm.2020.102435.
24. Cheng Y-Y, Chen Y-M, Yeh W-C, Chang Y-C. Valence and arousal-infused bi-directional LSTM for sentiment analysis of government social media management. Appl Sci. 2021;11(2):880. https://doi.org/10.3390/app11020880.
25. Basiri ME, Nemati S, Abdar M, Cambria E, Acharya UR. ABCDM: an attention-based bidirectional CNN-RNN deep model for sentiment analysis. Future Gener Comput Syst. 2021;115:279–94. https://doi.org/10.1016/j.future.2020.08.005.
26. Baktha K, Tripathy B. Investigation of recurrent neural networks in the field of sentiment analysis. In: 2017 International Conference on Communication and Signal Processing (ICCSP). IEEE; 2017. pp. 2047–50. https://doi.org/10.1109/ICCSP.2017.8286763.
27. Zhao N, Gao H, Wen X, Li H. Combination of convolutional neural network and gated recurrent unit for aspect-based sentiment analysis. IEEE Access. 2021;9:15561–9. https://doi.org/10.1109/ICCSP.2017.8286763.
28. Vashishtha S, Susan S. Fuzzy interpretation of word polarity scores for unsupervised sentiment analysis. In: 2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT). IEEE; 2020. pp. 1–6. https://doi.org/10.1109/ICCCNT49239.2020.9225646.
29. Neelakandan S, Paulraj D, Ezhumalai P, Prakash M. A Deep Learning Modified Neural Network(DLMNN) based proficient sentiment analysis technique on Twitter data. J Exp Theor Artif Intell. 2024;36:415–34. https://doi.org/10.1080/0952813X.2022.2093405.
30. Rakshit P, Sarkar A. A supervised deep learning-based sentiment analysis by the implementation of Word2Vec and GloVe embedding techniques. Multimed Tools Appl. 2024. https://doi.org/10.1007/s11042-024-19045-7.
31. Hossain MR, Hoque MM, Siddique N, Dewan MAA. AraCovTexFinder: leveraging the transformer-based language model for Arabic COVID-19 text identification. Eng Appl Artif Intell. 2024;133: 107987. https://doi.org/10.1016/j.engappai.2024.107987.
32. Hossain MR, Hoque MM, Siddique N, Sarker IH. CovTiNet: Covid text identification network using attention-based positional embedding feature fusion. Neural Comput Appl. 2023;35(18):13503–27. https://doi.org/10.1007/s00521-023-08442-y.
33. Liu H, Cocea M. Fuzzy rule based systems for interpretable sentiment analysis. In: 2017 Ninth International Conference on Advanced Computational Intelligence (ICACI). IEEE; 2017. pp. 129–36. https://doi.org/10.1109/ICACI.2017.7974497.
34. Maheswari SU, Dhenakaran S. Aspect based fuzzy logic sentiment analysis on social media big data. In: 2020 International Conference on Communication and Signal Processing (ICCSP). IEEE; 2020. pp. 971–5. https://doi.org/10.1109/ICCSP48568.2020.9182174.
35. Nguyen T-L, Kavuri S, Lee M. A multimodal convolutional neuro-fuzzy network for emotion understanding of movie clips. Neural Netw. 2019;118:208–19. https://doi.org/10.1016/j.neunet.2019.06.010.
36. Es-Sabery F, Hair A. A MapReduce C4. 5 decision tree algorithm based on fuzzy rule-based system. Fuzzy Inform Eng. 2019;11(4):446–73. https://doi.org/10.1080/16168658.2020.1756099.
37. Mehra R, Bedi MK, Singh G, Arora R, Bala T, Saxena S. Sentimental analysis using fuzzy and naive bayes. In: 2017 International Conference on Computing Methodologies and Communication (ICCMC). IEEE; 2017. pp. 945–50. https://doi.org/10.1109/ISCO.2017.7855960.
38. Ali F, Kwak K-S, Kim Y-G. Opinion mining based on fuzzy domain ontology and support vector machine: a proposal to automate online review classification. Appl Soft Comput. 2016;47:235–50. https://doi.org/10.1016/j.asoc.2016.06.003.
39. Es-Sabery F, Es-Sabery K, Qadir J, Sainz-De-Abajo B, Hair A, Garcia-Zapirain B, De La Torre-Díez I. A MapReduce opinion mining for covid-19-related tweets classification using enhanced ID3 decision tree classifier. IEEE Access. 2021;9:58706–39. https://doi.org/10.1109/ACCESS.2021.3073215.
40. Krouska A, Troussas C, Virvou M. The effect of preprocessing techniques on twitter sentiment analysis. In: 2016 7th International Conference on Information, Intelligence, Systems & Applications (IISA). IEEE; 2016. pp. 1–5. https://doi.org/10.1109/IISA.2016.7785373.
41. Song S, Johnson AP. Predicting drug review polarity using the combination model of multi-sense word embedding and fuzzy latent Dirichlet allocation (FLDA). IEEE Access. 2023. https://doi.org/10.1109/ACCESS.2023.3326757.
42. Phan HT, Nguyen NT. A fuzzy graph convolutional network model for sentence-level sentiment analysis. IEEE Trans Fuzzy Syst. 2024;32(5):2953–65. https://doi.org/10.1109/TFUZZ.2024.3364694.
43. Essameldin R, Ismail AA, Darwish SM. An opinion mining approach to handle perspectivism and ambiguity: Moving toward neutrosophic logic. IEEE Access. 2022;10:63314–28. https://doi.org/10.1109/ACCESS.2022.3183108.
44. Sentiment140 dataset with 1.6 million tweets. https://www.kaggle.com/datasets/kazanova/sentiment140. Accessed 15 Nov 2023.
45. Covid-19_sentiments India [20/03/20–31/05/20]. https://www.kaggle.com/datasets/abhaydhiman/covid19-sentiments. Accessed 15 Feb 2023.
46. Chandrasekar P, Qian K. The impact of data preprocessing on the performance of a naive bayes classifier. In: 2016 IEEE 40th Annual Computer Software and Applications Conference (COMPSAC), vol. 2. IEEE; 2016. pp. 618–9. https://doi.org/10.1109/COMPSAC.2016.205.
47. Es-Sabery F, Hair A. Big data solutions proposed for cluster computing systems challenges: A survey. In: Proceedings of the 3rd International Conference on Networking, Information Systems & Security. 2020. pp. 1–7. https://doi.org/10.1145/3386723.3387826.
48. Church KW. Word2vec. Nat Lang Eng. 2017;23(1):155–62. https://doi.org/10.1017/S1351324916000334.
49. Jogin M, Madhulika M, Divya G, Meghana R, Apoorva S, et al. Feature extraction using convolution neural networks (CNN) and deep learning. In: 2018 3rd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT). IEEE; 2018. pp. 2319–23. https://doi.org/10.1109/RTEICT42901.2018.9012507.
50. Farooq M, Sazonov E. Feature extraction using deep learning for food type recognition. In: Bioinformatics and Biomedical Engineering: 5th International Work-Conference, IWBBIO 2017, Granada, Spain, April 26–28, 2017, Proceedings, Part I 5. Springer; 2017. pp. 464–72. https://doi.org/10.1007/978-3-319-56148-6_41.

51. Es-Sabery F, Hair A. An improved ID3 classification algorithm based on correlation function and weighted attribute. In: 2019 International Conference on Intelligent Systems and Advanced Computing Sciences (ISACS). IEEE. 2019. pp. 1–8. https://doi.org/10.1109/ISACS48493.2019.9068891.
52. Ghazi MR, Gangodkar D. Hadoop, MapReduce and HDFS: a developers perspective. Proced Comput Sci. 2015;48:45–50. https://doi.org/10.1016/j.procs.2015.04.108.
53. Memiş S, Enginoğlu S, Erkan U. Fuzzy parameterized fuzzy soft k-nearest neighbor classifier. Neurocomputing. 2022;500:351–78. https://doi.org/10.1016/j.neucom.2022.05.041.
54. Es-Sabery F, Hair A, Qadir J, Sainz-De-Abajo B, García-Zapirain B, De La Torre-Díez I. Sentence-level classification using parallel fuzzy deep learning classifier. IEEE Access. 2021;9:17943–85. https://doi.org/10.1109/ACCESS.2017.2694446.
55. Memiş S, Enginoğlu S, Erkan U. A classification method in machine learning based on soft decision-making via fuzzy parameterized fuzzy soft matrices. Soft Comput. 2022;26(3):1165–80. https://doi.org/10.1007/s00500-021-06553-z.
56. Memiş S, Enginoğlu S, Erkan U. A new classification method using soft decision-making based on an aggregation operator of fuzzy parameterized fuzzy soft matrices. Turk J Electr Eng Comput Sci. 2022;30(3):871–90. https://doi.org/10.55730/1300-0632.3816.
57. Memiş S, Enginoğlu S, Erkan U. Numerical data classification via distance-based similarity measures of fuzzy parameterized fuzzy soft matrices. IEEE Access. 2021;9:88583–601. https://doi.org/10.1109/ACCESS.2021.3089849.
58. Zadeh LA. Fuzzy logic= computing with words. IEEE Trans Fuzzy Syst. 1996;4(2):103–11. https://doi.org/10.1109/91.493904.
59. Seddiq Y, Alotaibi YA, Selouani S-A, Meftah AH. Distinctive phonetic features modeling and extraction using deep neural networks. IEEE Access. 2019;7:81382–96. https://doi.org/10.1109/ACCESS.2019.2924014.
60. Demšar J. Statistical comparisons of classifiers over multiple data sets. J Mach Learn Res. 2006;7:1–30.