Hindawi
Computational Intelligence and Neuroscience
Volume 2022, Article ID 1883698, 26 pages
https://doi.org/10.1155/2022/1883698

Review Article

A Complete Process of Text Classification System Using State-of-the-Art NLP Models

Varun Dogra[1], **Sahil Verma**[2,3], **Kavita**[2,4], **Pushpita Chatterjee**[5], **Jana Shafi**[6], **Jaeyoung Choi**[7] and **Muhammad Fazal Ijaz**[8]

[1]*School of Computer Science and Engineering, Lovely Professional University, Phagwara, Punjab, India*
[2]*Department of Computer Science and Engineering, Chandigarh University, Mohali 140413, India*
[3]*Bio and Health Informatics Research Lab, Chandigarh University, Mohali 140413, India*
[4]*Machine Learning and Data Science Research Lab, Chandigarh University, Mohali 140413, India*
[5]*Tennessee State University, Nashville, TN, USA*
[6]*Department of Computer Science, College of Arts and Science, Prince Sattam Bin Abdul Aziz University, Wadi Ad-Dwasir 11991, Saudi Arabia*
[7]*School of Computing, Gachon University, Seongnam-si 13120, Republic of Korea*
[8]*Department of Intelligent Mechatronics Engineering, Sejong University, Seoul 05006, Republic of Korea*

Correspondence should be addressed to Jaeyoung Choi; jychoi19@gachon.ac.kr and Muhammad Fazal Ijaz; fazal@sejong.ac.kr

Received 4 March 2022; Revised 20 April 2022; Accepted 9 May 2022; Published 9 June 2022

Academic Editor: Sumarga Kumar Sah Tyagi

With the rapid advancement of information technology, online information has been exponentially growing day by day, especially in the form of text documents such as news events, company reports, reviews on products, stocks-related reports, medical reports, tweets, and so on. Due to this, online monitoring and text mining has become a prominent task. During the past decade, significant efforts have been made on mining text documents using machine and deep learning models such as supervised, semisupervised, and unsupervised. Our area of the discussion covers state-of-the-art learning models for text mining or solving various challenging NLP (natural language processing) problems using the classification of texts. This paper summarizes several machine learning and deep learning algorithms used in text classification with their advantages and shortcomings. This paper would also help the readers understand various subtasks, along with old and recent literature, required during the process of text classification. We believe that readers would be able to find scope for further improvements in the area of text classification or to propose new techniques of text classification applicable in any domain of their interest.

1. Introduction

In recent years, we have seen a growth in the amount of digital textual data available, which has generated new perspectives and so created new areas of research. With the emergence of information technology, the monitoring of such digital textual data is of great importance in many areas such as the stock market: gathering data from news sources to forecast the movement of underlying asset volatility [1], forecasting the stock prices of green firms in emerging markets [2], understanding the impact of tone of communications on stock prices [3], and determining indicators for stock prices volatility [4]; healthcare: disease surveillance [5, 6]; politics: developing a probabilistic framework on politics using short text classification [7]; education: understanding pedagogical aspects of the learners [8]; tourism: analyzing travelers sentiments [9]; and e-commerce: predicting success by evaluating users' reviews [10].

News is widely available in electronic format on the World Wide Web these days, and it has proven to be a valuable data source [11]. The volume of news, on the other hand, is enormous, and it is unclear how to use it most efficiently for domain-specific research. Therefore, a framework or architecture is required for a domain-specific

news monitoring system, as well as a classification mechanism for classifying relevant online news into distinct subject groups automatically [5]. News monitoring is a type of oversight system that monitors and ensures the quality of each news instance generated, used, and retained for a purpose. Processes for assessing news to guarantee its completeness, consistency, and correctness, as well as security and validity, are included in these methods. There is always a need for a methodology that can extract meaningful information from a pool of textual documents belonging to distinct subject groups intended for certain research as shown in Figure 1.

Indeed, the majority of the digital data are available in the form of text, but this is usually unstructured or semi-structured [12]. Thus, to make data useful for decision-making, structuring this textual data became a necessity [13, 14]. However, because of the high volume of data, it is quite impossible to process the data manually. Text classification has evolved due to this challenge. It is defined as assigning the text documents to one or more categories (called labels) according to their content and semantics. Traditionally, the majority of classification tasks were used solved manually, but it was expensive to scale. Classification can be thought of as writing rules for assigning a class to similar text documents. These rules include some related information that identifies a class. Handwritten rules can be performed well, but creating and maintaining them over time requires much manpower. A technical expert can frame rules by writing regular expressions that could maximize the accuracy of the classifier. The existing studies have proposed various techniques to automatically classify text documents using machine learning [15, 16]. In this approach, the set of rules or criteria for selecting a classifier is learned automatically from the training data. Under each class, it requires a lot of training documents and expertise to label the documents. The labeling is a process of assigning each document to its associated class. The labeling process was easier than writing handcrafted rules. Moreover, there exist variously supervised and semisupervised learning techniques that can even reduce the burden of manual labeling [17, 18]. This can be performed using automatic labeling. Automated text classification methods can be divided into three groups: rule-based methods, data-driven methods, and hybrid methods.

Using a set of predefined rules, rule-based techniques classify text into various categories as shown in Figure 2. For example, the "fruit" label is applied to any document with the words "apple," "grapes," or "orange." These techniques require a thorough knowledge of the domain, and it is difficult to maintain the systems. Data-driven methods, on the other hand, learn to make classifications based on previous data values. A machine learning algorithm can learn the inherent associations between pieces of text and their labels using prelabeled examples as training data. It can detect hidden patterns in the data, is more flexible, and can be applied to different tasks. As the title indicates, hybrid approaches use a mixture of rule-based and machine learning methods (data-driven) for making predictions.

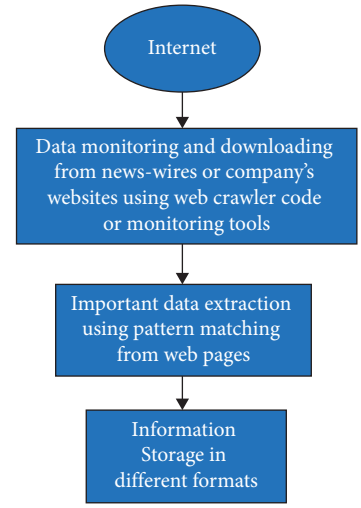

Figure 1: Monitoring and downloading relevant text documents to subject groups.

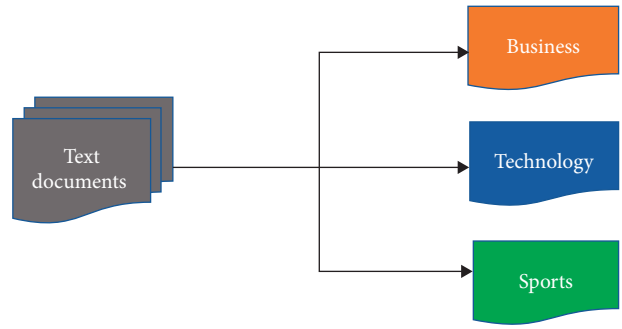

Figure 2: Labeling text documents with appropriate predefined classes or labels during the process of text classification.

In recent decades, models of machine learning have attracted a lot of interest [19, 20]. Most conventional models based on machine learning follow the common two-step method, where certain features are extracted from the text documents in the first step, and those features are fed to a classifier in the second step to make a prediction. The popular feature representation models are BOW (bag-of-words), TF-IDF (term frequency-inverse document frequency), and so on. And the common classifiers are naïve Bayes, KNN, SVM, decision trees, random forests, and so on. These models are discussed in detail in the following sections. Deep learning models have been applied to a wide variety of tasks in NLP, improving language modeling for more extended context [21–23]. These models are attempting, in an end-to-end fashion, to learn the feature representations and perform classification. They not only have the potential to uncover latent trends in data but also are far more transferable from one project to another. Quite significantly, in recent years, these models have become the mainstream paradigm for the various tasks of text classification. The following are some of the natural language challenges solved with the text classification.

Topic modeling is widely used to extract semantic information from text data. An unsupervised method of topic modeling learns the collection of underlying themes for a batch of documents as well as the affinities of each document to these topics.

News classification: online news reporting is one of the most significant sources of information. The task of finding and deriving structured information about news events in any text and assigning the relevant label is referred to as news classification.

Sentiment classification is an automatic technique of discovering views in text and classifying them as negative, positive, or neutral based on the emotions expressed in text. Sentiment classification, which uses NLP to evaluate subjective data, can help understand how people think about company's products or services.

Question answering has rapidly evolved as an NLP challenge that promises to deliver more intuitive means of knowledge acquisition. In contrast to the typical information retrieval approach of creating queries and perusing results, a question answering system simply takes user information requests stated in ordinary language and returns with a brief answer.

Language translation models have been attempting to translate a statement from one language to another resulting in perplexing and offensively inaccurate results. NLP algorithms through text classification may be trained on texts in a variety of languages, allowing them to create the equivalent meaning in another language. This approach is even applicable to languages such as Russian and Chinese, which have historically been more difficult to translate due to differences in alphabet structure and the use of characters rather than letters, respectively.

Nevertheless, it is observed that most text classification literature studies for solving NLP challenges are limited to showcasing the results of text classification using standard or state-of-the-art methods and focusing on specific research domains. For example, the authors mention the application of text analytics in the industry, but the task of monitoring and collecting text data was not detailed, and the scope of the proposed models appeared limited to particular domains [24]. In another study, the authors discuss the information extraction from tweets for monitoring trucks fleets to model truck trips, but it does not cover the feature selection or extraction methods to achieve information extraction [25]. Other studies [26, 27] focus on text classification for domain-specific search engine based on rule-based annotated data; however, it does not cover the semisupervised or unsupervised approaches of labeling data to achieve text classification [28]. Moreover, these works do not reveal the latest techniques being used in the area of natural language processing. The deep learning-based pretrained language representation model can be explored in information extraction and classification. These studies also lack in detailing the subtasks require to initiate the research in text classification, that is, data collection, data preprocessing, and semisupervised or unsupervised data labeling for training machine learning models. To the best of our knowledge, there are no similar review studies available that cover in-depth presentations of various subtasks of text classification.

In this paper, we focus to overcome the above-mentioned issues. We put a lot of effort to create qualitative research for text classification to help us understand its subtasks or elements. Moreover, this paper presents the old and latest techniques used in each subtask of text classification as shown in Figure 3 along with their benefits and limitations. It also presents the research gap in the area of text classification by examining various existing studies. The key contribution of the study is mentioned below:

(i) Discussing the subtasks of text classification

(ii) Presenting the most recent and former techniques used in each subtask

(iii) Presenting benefits and limitations of various models used in the process of text classification

(iv) Presenting the research scope for further improvements in existing techniques and proposing new techniques with their application in different domains

Section 2 presents the process of text classification along with the comprehensive literature on each subtask; Section 3 presents the evaluation methods of classification techniques; Section 4 presents the comparison of approaches or models used in the subtasks of the text classification system mentioning their benefits and limitations; Section 5 presents the research gap and further scope for research; and Section 6 concludes the existing studies.

2. Text Classification: Framework

Text classification is a problem formulated as a learning process where a classifier is used to train to differentiate between predefined classes based on features extracted from the collection of text documents [29]. The accuracy of the classifier depends upon the classification granularity and how well separated are the training documents among classes [30, 31]. In text classification, a set of labels or classes are given, and we need to evaluate which class/label a particular text document relates to. Usually, a class or label is a general topic such as sports or business. But it may be significantly more difficult to distinguish between documents that are about more similar classes such as networks and the internet of things. Certain features represent the potential overlap between classes; the learning task would be simplified by removing such overlapping features. If the gap between classes could be increased, the classification performance would increase. This can be achieved through features weighting and selecting valuable features. Text classification has been studied and applied by many researchers in real-world scenarios such as sentiment classification of stock market news and its impact [31], news classification for syndromic surveillance [5], microblog topic classification [32], domain adaptation for sentiment classification [33, 34], and brand promotion based on social media sentiments [35, 36].

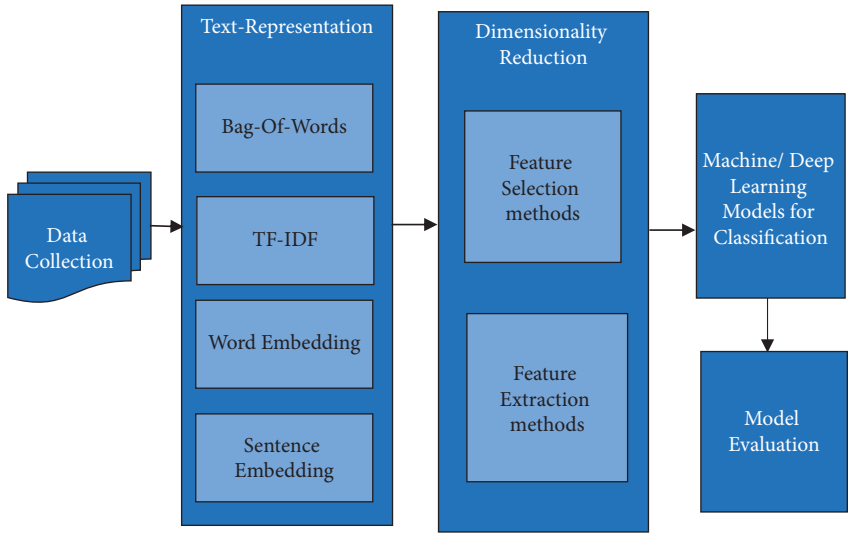

Figure 3: Subtasks of the text classification process cover state-of-the-art data collection, text representation, dimensionality reduction, and machine learning models for classifying text documents to an associated predefined class/label.

The text classification process is described as classifying a set of N documents; first, we build a classifier T. There is a collection of text documents D, and every text document is given a class/label by an expert. Secondly, we need to train a classifier for each class/label by giving input as a corresponding set of documents in D. Now we need to apply trained classifier C to classify N documents. We will get each document in N assigned to a predefined class/label by C. Text classification is a comprehensive process that includes not just model training but also several other steps including data preprocessing, transformation, and dimensionality reduction. The process starts with the collection of textual content from various sources. The textual content may be belonging to a domain(s) representing some events, business processes, or public information. Then these text documents require preprocessing to generate appropriate text representation for the learning model. This is done in two phases: in phase 1, the features are extracted from the processed text using any feature extraction algorithm, and in phase 2, the features are reduced by applying feature selection techniques. This reduction of features tends to decrease the dimensions of data required for the learning method. After these phases, the learning algorithms are chosen to train on data to generate the best classifier for recognizing a target category or class. This text data required to train a classifier is known as training data. The data is divided into two sets: the majority of data are taken for the training model, and the rest part of the data is taken for testing the classifier, known as testing data. Similarly, the model is trained to recognize each target class representing its data available in the associated text documents. During the testing phase, when a classification method is developed, it is executed on test data to define the target class of input text, and the result is produced in the form of weights or probabilities. Finally, the result is evaluated for its accuracy, of text classifier, using evaluation techniques. These are the main phases or subtasks of the text classification process, also shown in Figure 4. The different approaches have been used in each phase of text classification discussed in the next subsections of the study.

2.1. Data Collection. The initial stage in text classification is to acquire text data from different sources as per the research domain. There are several online open data sets available, for example, various newsgroups (Bloomberg, Reuters, Financial Express), Kaggle, and WebKB for solving a classification problem. Researchers have used such database architecture for their research purposes [37–39]. The corpus can also be built with data that could be anything from emails, language articles, company's financial reports, medical reports, to news events. In the study, the authors have created a fine-grained sentiment analysis corpus for annotating product reviews. However, they faced the most challenging tasks that had not been targeted in applications such as sentiment analysis, target-aspect pair extraction, and implicit polarity recognition, for recognizing aspects and searching polarity with nonsentiment sentences [40].

2.2. Text Document Representation: Features Construction and Weighting. Text classification is the most demanding area of machine learning for understanding texts written in any natural language. One of the most essential tasks that must be completed before any classification process is text representation. Moreover, the texts cannot be provided as input to the machine learning models because almost all algorithms take input in numbers as feature vectors with a predefined size instead of the textual data with variable length. To resolve this issue, first textual data need to be transformed to document vectors. This can be done in two different ways in general. The first is a context-independent approach in which a document is represented as a set of terms with their corresponding frequency in the document, but they are independent of the sequence of terms in the collection. The second approach is to represent text as strings, with each

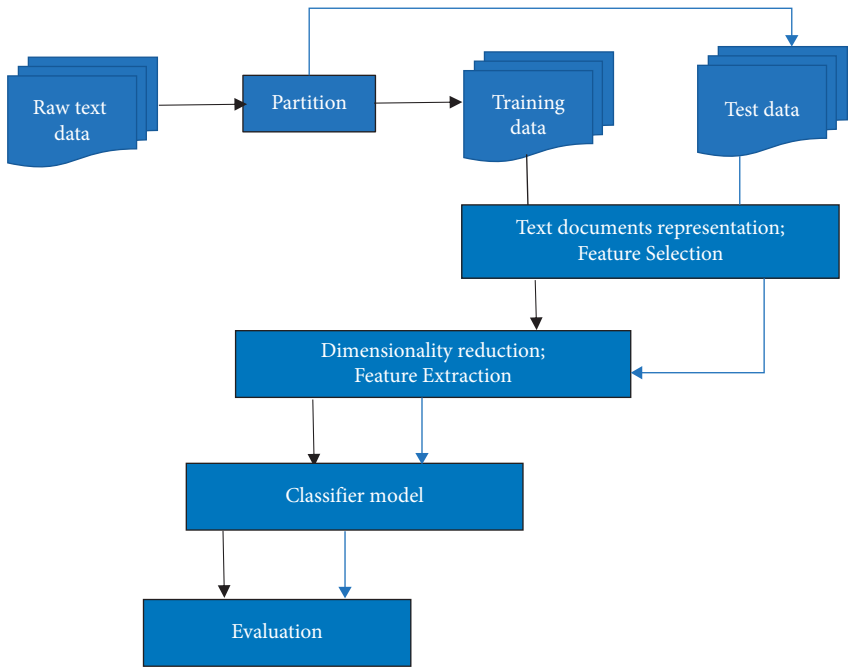

Figure 4: A text classification framework. Note: Black connecting lines represent training and blue connecting lines represent the testing phase.

document consisting of a sequence of terms. The following subtopic covers the various representations in natural language processing from the early days to the latest state-of-the-art models.

2.2.1. Context-Independent Approaches. The bag-of-words [41] is the most commonly used model in document transformation that considers every word in the text document as a token/feature although words' order and their context are ignored. Each word, sometimes tens or hundreds of dimensions, is represented by a real-valued vector called a one-hot representation [42]. The feature vector has the same length as the vocabulary size, and only one dimension is on as shown in Figure 5. However, the one-hot representation of a word suffers from data sparsity. On the other hand, the words with very high frequency may cause biases and dominate results in the model [44]. To overcome the weaknesses of BOW, the text documents are represented with weighted frequencies, a document-term matrix, where a column signifies a token and a row signifies a document. This scheme of assigning weights to token frequencies in the form of a matrix is called TF-IDF (term frequency-inverse document frequency). During the implementation of this model using a matrix, the value w_{ij} in each cell corresponds to the weight of t_j in d_i that is calculated as $tf\ \square t_j, d_i \mid n_{d_i} \square$, where $tf\ \square t_j, d_i \square$ represents the count of token t_j in text document and d_i and n_{d_i} represents the total quantity of token t_j in document d_i. Due to the simplicity of the model, this is preferably used in natural language processing. The improved features subset using this approach has been taken together with the characteristics of term frequency and document frequency [45–47]. However, even a small collection of documents may consist of a large number of meaningful words that leads to the problem of scalability or high dimensionality. This offers opportunities to find effective ways to decrease running time or reduce high dimensionality in the case of a large number of documents.

The primary alternative has emerged in the form of statistical language modeling for modeling complex text classification or other natural language tasks. In its beginning, however, it used to struggle with the curse of dimensionality when studying typical probability functions of language models [48]. This led to the inspiration to learn distributed representations of low-dimensional space terms. The distributed representations describe a co-occurrence matrix of terms × terms that considers the frequency of each term that appears in the context of another term, with a window size of k [49]. The singular value decomposition was used for text representation, where the matrix decomposition technique was used for reducing a given matrix to its constituent matrices via an extension of the polar decomposition with the idea of making subsequent matrix calculations simpler. It gives the top rank-k constituent parts of the original data. The singular value decomposition will break this into best rank approximation capturing information from most relevant to least relevant ones [49].

The big popularization of word embedding was possibly due to the continuous bag-of-words (CBOW) paradigm to create high-quality distributed vector representations effectively. A solution is designed to counter the curse of dimensionality where a distributed representation for each word is concurrently studied along with the probability distribution for word sequences represented in terms of such representations [21]. The continuous bag-of-words is a prediction-based model that directly learns word

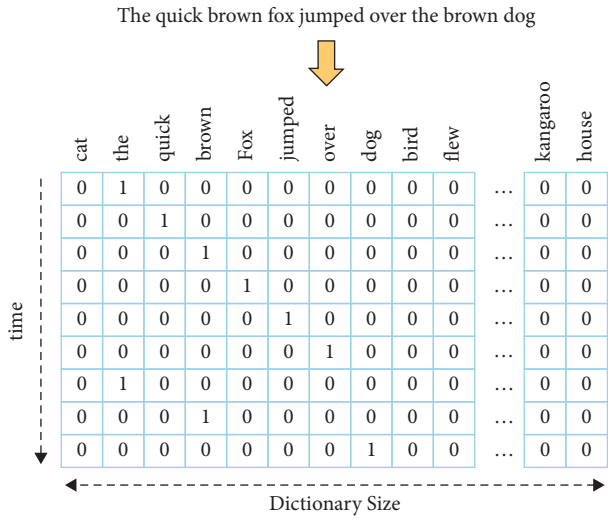

Figure 5: One-hot representation, a tensor that is used to represent each document. Each document tensor is made up of a potentially lengthy sequence of 0/1 vectors, resulting in a massive and sparse representation of the document corpus [43].

representation as shown in Figure 6(a). The distributed representations of context (or surrounding words) are combined in the CBOW model to predict the word in the middle. The CBOW has reshaped the word embedding [51]. The continuous bag of representation is applied with a neural network model to achieve improved accuracy in classification [52, 53]. Another model is designed called skip-gram that further reshaped the word embedding [54]; its architecture works in reverse of what the continuous bag-of-words model does. The model predicts each context word from the target word as shown in Figure 6(b). It iterates on the words of each sentence in the given corpus and uses the current word to predict its neighbors (its context); thus, the model is called "skip-gram" (local context window) [55].

Weighted words calculate document similarity directly from the word-count space, which takes longer to compute for large vocabularies. While counts of unique words give independent evidence of similarity, semantic similarities between words are not taken into consideration. Word embedding techniques solve this problem, but they are constrained by the need for a large corpus of text data sets for training. The word embedding algorithms were developed using the word and its closest neighbor. There was an approach suggested by authors for generating a word embedding GloVe (global vector, combines count- and predict-based methods) model for distributed word representation. The unsupervised learning algorithm, where a model is trained on overall statistics of word-word co-occurrence that how often it appears in a corpus, and the result obtains the vector representation of words with linear substructures of the word vector space [56]. GloVe's solution is to count how many times a term i (context word) in another term j (target word) occurs. The purpose is to establish a meaning for the word i and word j as to whether the two words occur close to N-word apart or not. The encoding vector includes the ratio of two words specifically recognized as a count-based system of co-occurrence probabilities. The prediction-based approach receives popularity, but GloVe's authors claim that the count-based methodology incorporates the global statistics and may be more effective because it outperforms word representation testing on word comparison, term similarity, and called entity recognition tasks.

The enhanced text document representation system was developed to work on the issues of traditional feature-based extraction techniques that included only nouns and nouns phrases to represent the important events called event detection techniques [57]. This technique has used fewer tokens or features than bag-of-words to handle the problem of scalability or high dimensionality of documents. Furthermore, this technique has led to another representation based on named entities. The authors have presented the classification of tweets using named entity recognition to filter out noiseless required information [30, 58]. It works by finding proper nouns in the documents that belong to predefined categories. This process involves systematically assigning categories to each term or entity while developing a corpus or labeling process. But this corpus-based representation was unable to represent some domain-specific words that are infrequent during training. The authors have proposed techniques to deal with infrequent or unseen words during labeling [59].

Previous representations were not considering the morphological relation of words to disambiguate the unseen words. Many studies have presented methods that automatically extract features from the documents. These have used infrequent words that produce a high variety of low anticipated relations between the text documents. This kind of information once aggregated provides potentially less obvious and hidden relations in the text documents. Using less-frequent words with lexical constraints has reduced the associated cost of knowledge re-engineering, and it was able to process many documents from a large number of domains [59, 60]. These methods help for better representations of text documents especially handling unseen or less-frequent words. And the problem of scalability was also controlled and associated with word's semantical approach. There is another approach that was proven most efficient in a domain-specific text representation, *proper nouns*, an intermediate solution between noun phrases and named entities. This technique has reduced the ambiguity that occurred due to particular associated nouns with more than one named entity category [31]. Recent approaches are concentrating on capturing context beyond the word level to produce performance by giving a more structured and semantic notion of text [61].

2.2.2. Context-Aware Approaches. Context-aware classification approaches essentially find and employ term association information to increase classification effectiveness. They allow the presence or absence of a term to impact how it contributes to a classification outcome. Context is a concise term referring to high-level semantics. It may be taken in several ways and used in a variety of dimensions. We categorize context-based classification systems according to how

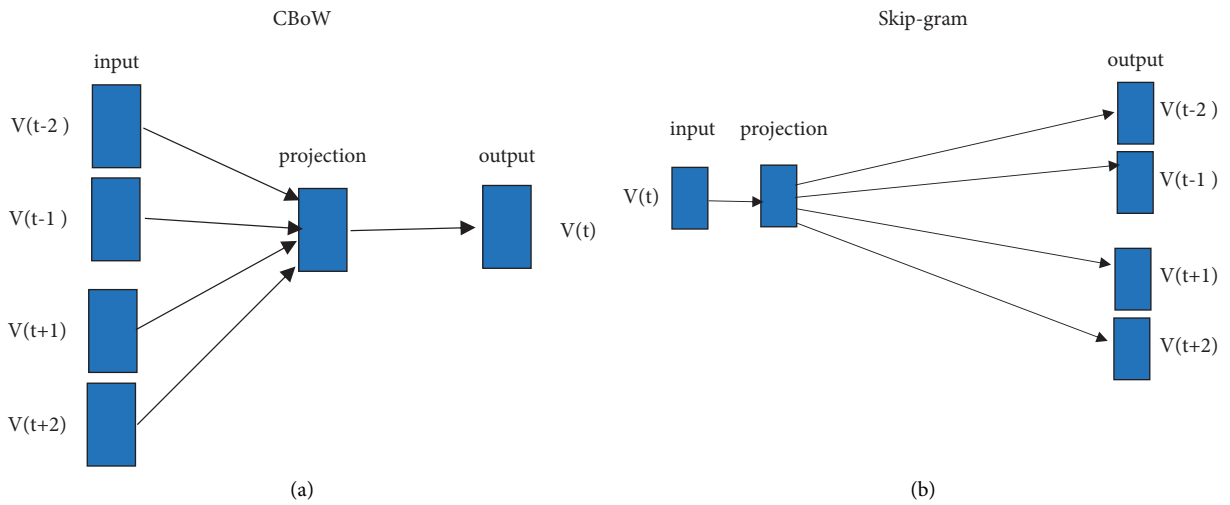

Figure 6: The word2vec algorithm uses two alternative methods: (a) continuous bag of words (CBoW) and (b) skip-gram (SG) [50].

the context was understood and what features were used to determine it.

The authors have come up with improved embedding for texts, such as *word2vec*, which transforms a word into an *n*-dimensional vector. To map the words into an Euclidean space, we can go through an approach to creating sequence embedding that brings a sequence into an Euclidean space. One of the sequential data function learning problems is called sequence embedding [62], where the aim is to convert a sequence into a fixed-length embedding. This approach is a highly strong tool for identifying correlations in text corpora as well as word similarity. However, it falls limited when it comes to capturing out-of-vocabulary words from a corpus. *RNN*-based models interpret the text as a sequence of words and are intended for text classification to capture word dependencies and text structures [63]. By adding a memory cell to remember values over arbitrary time intervals and three gates (input gate, output gate, and forget gate) to control the flow of information into and out of the cell, *LSTM* addresses gradient vanishing or exploding problems experienced by the RNNs. In a recursive method, the authors expand the chain-structured LSTM to tree structures, using a memory cell to store the background of multiple child cells or multiple descendants. They claim that the new model offers a realistic way to understand contact between hierarchies between long distances, for example, language parse structures [64]. To capture text features, the authors also incorporate a *bidirectional-LSTM* (Bi-LSTM) model with two-dimensional max-pooling [65]. The seq2seq model is used in various NLP applications [66, 67]. Most real-world problems have a data set with a substantial number of unusual words. The embeddings learned from these data sets are unable to produce the correct representation of the word. To do this, the data set needs to have a large vocabulary. Words that appear frequently help you create a large vocabulary. Second, when learning embeddings from scratch, the number of trainable parameters grows. As a result, the training process is slowed. Learning embeddings from scratch may also leave you confused about how the words are represented [68].

Pretrained word embeddings are the solution to all of the above difficulties. The studies have consistently shown the importance of transfer learning by pretraining a neural network model on an established problem in the field of computer vision and then doing fine-tuning utilizing the learned neural network as the foundation for a new purpose-specific model. It is demonstrated in recent years that a related approach may be effective in several tasks relating to natural language. This is another kind of word embedding; the classification algorithms provide a greater sense of learning the features of such embeddings. Yet such embeddings do not take the word order or the word meaning of each sentence into consideration. This is where *ELMo* (embedding from models of language) comes into action. ELMo is a contextual embedding that takes into consideration the terms that surround it. It models word use characteristics such as morphology and how it is used in different contexts. The term vectors are learned features of a deep bidirectional language model (biLM) internal state that is pretrained on a broad text corpus. The authors have demonstrated that these representations can be readily applied to current frameworks and greatly strengthen the state-of-the-art NLP issues such as addressing queries, textual entailment, and interpretation of emotions [62]. A Transformer [69] is another solution to working with long dependencies such as LSTM. LSTM is long short-term memory, a sort of neural network that has a "memory unit" capable of maintaining knowledge in memory over strong periods helping it to learn longer-term dependencies [22]. The Transformer is based through the encoder and decoder on an attention process as shown in Figure 7. The Transformer allows the use of this method to store knowledge about the specific meaning of a given term in its word vector.

Unlike past deep contextualized language representation studies [62] that take into account the backward model as in the ELMo bidirectional LSTM, the authors proposed a new type of language representation named *BERT*, which stands for bidirectional transformer encoder representations. BERT uses a Transformer, a mechanism of *attention* that learns

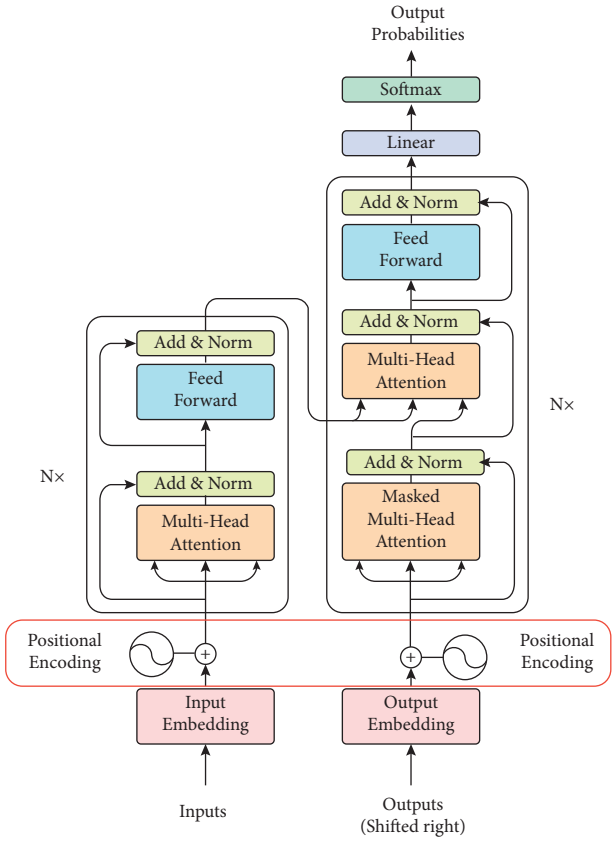

Figure 7: The transformer architecture.

contextual connections between words (or subwords) in a document. The authors have argued that conventional technologies limit the power of the pretrained representations, especially for the approaches to fine-tuning [70]. The main constraint is the unidirectional existence of modern language models, which restricts the range of frameworks that can be used during pretraining. BERT is structured to pretrain profound bidirectional representations from unlabeled text documents by jointly conditioning across both layers in both the left and right contexts [71]. In setting language modeling, transformers can acquire longer-term dependence but are constrained by a fixed-length context. The authors suggested a novel Transformer-XL neural architecture that allows learning dependence to interrupt temporal coherence beyond a fixed length. It consists of a recurrence function at the segment level and a novel positional encoding scheme [69]. Learning regarding the inductive transfer has significantly impacted computer vision, but current techniques in NLP also need complex task modifications and preparation from scratch. The authors suggested universal language model fine-tuning (ULMFIT), an important transfer learning approach that can be extended to any NLP task, and implemented techniques that are essential to fine-tuning a model of language [72].

A total of 15% of the words in every sequence are substituted with a mask token before feeding word sequences into BERT. The model then tries to determine the actual context of the masked words in the list, based on the context given by the other, unmasked, words. The BERT loss function only considers the estimation of the masked terms and excludes the estimation of the unmasked phrases. As a result, the model converges slower than directional ones, a feature offset by its enhanced understanding of the context. Centered on BERT's masking technique [71], the authors developed a novel language representation model enhanced by knowledge-masking technique named *ERNIE* (enhanced representation by knowledge integration) [73], which involves masking at the entity level and masking at the phrase level. Strategy at the entity level covers entities that typically consist of several terms. The phrase-level technique covers the whole phrase consisting of many words that serve as a cohesive entity together. Their experimental findings indicate that ERNIE outperforms other standard approaches by obtaining modern state-of-the-art outcomes on natural language processing activities, including natural language inference, conceptual similarity, named-entity identification, emotion analysis, and question answering. DistilBERT, a technique for pretraining a smaller general-purpose language representation model that can later be fine-tuned with high performance on a wide range of tasks like its bigger equivalents, is created. While most previous research focused on using distillation to build task-specific models [74, 75], this study uses knowledge distillation during the pretraining phase and demonstrates that it is possible to reduce the size of a BERT model by 40% while retaining 97% of its language understanding capabilities and being 60% faster [76].

The Transformer paradigm is generally popular in several tasks relating to natural language processing. Using a transformer is therefore also an expensive operation since it requires the method of self-attention. The Transformer employs an encoder-decoder design that includes stacked encoder and decoder layers. Two sublayers comprise encoder layers: self-attention and a positionwise feed-forward layer. Self-attention, encoder-decoder attention, and a positionwise feed-forward layer are the three sublayers that comprise decoder layers. Self-attention assumes we are conducting the task of attention on the sentence itself, as compared to two separate sentences. Self-attention allows defining the connection in a single sentence between the words. It is the function of self-attention that adds to the expense of utilizing a transformer. The quadratic structure of self-attention, however, constrains its operation on long text. Attention is described using the (query, key, and value) model. A query Q is a "context," and in previous equations, the prior concealed state is employed as the query context. Based on what we already know, we want to know what happens next. The value represents the features of the input. The phrase "key" is just an encoding of the word "value." To attract attention, the query's relevancy to the keys is established. The associated values that are unrelated to the query are then hidden. The authors follow a method of fine to coarse attention on multi-scale spans by binary partitioning (BP); they suggest *BP-Transformer*. BP-Transformer has a strong balance between the complexity of computations and the capability of models. The authors performed a series of experiments on text classification and language

processing, showing that BP-Transformer performs superior to previous self-attention models for long text [77]. The Binary-Partitioning Transformer attempts to boost the self-attention mechanism's usefulness by considering the transformer as a graph neural network. Any node in this graph represents an input token.

Another research suggests a variant of the Neural Attentive Bag-of-Entities, which is a neural network algorithm that uses entities in a knowledge base to conduct text classification. Entities include unambiguous and specific syntactic and semantic signs that are useful for catching semantics in documents. The authors put together easy high-recall dictionary-based entity recognition, with a neural attention system that helps the model concentrate on a limited number of unambiguous and specific entities in a text [78]. The model first identifies entities to whom this name might be addressed (e.g., Ap Inc., Apple (food)) and then describes the entity using the weighted average of all entities' embedding. The weights are measured using a modern method of neural attention that helps the model concentrate on a specific subset of entities that are less ambiguous in context and more important to the document.

It is the time for NLP when the transition started as we listed the unsupervised pretrained language models that had made breakthroughs in different tasks of understanding the natural language, such as named-entity identification, emotion interpretation, and question-answer records for art performance beginning one after another in that short period. These NLP models indicate that a lot more is yet to come, and the authors look forward to researching and implementing them. However, the authors proved that a single pretrained language model may be applied as "a zero-shot task transfer" to execute basic NLP tasks without the requirement for fine-tuning on a training example data set. While this was an encouraging proof of concept, the best case performance only equaled certain supervised baselines on a single data set. Performance on most tasks was still well behind even simple supervised baselines. Across one order of magnitude of scaling, the study found generally consistent log-linear trends in performance on both transfer tasks and language modeling loss. GPT-3 performs well on NLP tasks in the zero- and one-shot settings and, in the few-shot setting, is sometimes comparable with, and occasionally surpasses, the state of the art (although the state-of-the-art is held by fine-tuned models). This might imply that bigger models are better meta-learners [79].

GPT-3 approaches the performance of a fine-tuned RoBERT as a baseline on the "Challenge" version of the data set, which has been filtered to questions (multiple-choice questions on common sense reasoning collected from 3rd to 9th-grade science examinations) that standard statistical or information retrieval algorithms are unable to accurately answer. GPT-3 marginally outperforms the same fine-tuned RoBERT baseline on the "Easy" version of the data set. However, both of these findings are much inferior to MBART's overall SOTAs.

2.3. Data Preprocessing: Data Cleaning. There are a lot of text data available today, and data are being grown daily in structured, semiunstructured, or fully unstructured forms. To perform the text classification task, it is always required to process the raw corpus data. There are many steps involved in data processing; generally, data cleaning, that is, organizing the data as per the structure and removal of unneeded subtexts; tokenization, that is, breaking up text into words; normalization, that is, converting all texts into the same case, removing punctuation (stemming leaves out root forms of the verb and lemmatization); and substitution, that is, identifying candidate words for translation, performing word sense disambiguation [80]. In one of the studies [81], the researchers have also focused on how machine learning techniques are needed to design to recognize similar texts when text data are downloaded from multiple and heterogeneous resources. In the labeling task, the text documents are labeled with two commonly used approaches; one is to label each part of the text individually, and the second is to label the group of texts. The first approach includes different supervised learning methods, and the second is called multi-instance learning [82, 83].

2.4. Data Preprocessing: Dimensionality Reduction. Dimensionality reduction is a crucial approach in data preprocessing for preparing data for text classification. It is done to reduce the classifier's memory requirements and execution time, hence increasing the learning model's efficiency and efficacy. The dimensions of data are increasing as the volume of data grows. To map large dimensions to space with low dimensions, it becomes necessary to reduce the dimensions of the data [84, 85]. The purpose of decreasing high-dimensional space is to find a subdimensional space that is less complex and can adjust the learning model to the greatest extent possible. In some cases, several researchers have noticed that the number of features in samples is substantially larger than the number of samples. This leads to a problem known as overfitting [86]. As a result, dimensionality reduction becomes necessary to avoid overfitting. Feature selection and feature extraction are two significant subtasks in lowering dimensionality. The process of supplying the part of the original attributes that are necessary for the task is known as feature selection. Feature extraction is a technique for changing space with many dimensions into a new space with few dimensions, to increase data variance [87].

2.4.1. Standard Feature Selection Methods. The following are the goals of the feature selection method:

(i) To improve the predictability of the classifiers

(ii) To create a cost-effective classifier while also speeding up the process

(iii) To improve the clarity of the data-gathering approach

The approach of finding a portion of the original attributes that are significant for the training set is known as feature selection. It is used to create an effective classifier while keeping important lexical properties in vocabulary [88]. It removes the noisy attributes that tend to reduce

accuracy [89]. The goal of feature selection is to decrease the feature space by picking a subset of Kay attributes and minimizing overfitting while maintaining text classification performance [90]. For instance, a feature or attribute set $X = \{X_1, X_2, \ldots, X_N\}$

When N is a group of feature sets, $2N$ feature subsets are created, each of which is represented by a vector of size N. The methods identify a feature subset of size K, $K < N$ without losing the accuracy of the full feature set. It has been the subject of investigation, and the writers have so far provided many techniques. Filter, wrapper, and embedded are the three types of approaches available [91, 92]. Filter-based feature selection offers several ways to evaluate the information value of each feature. The filter approach picks the top-N features based on the results of various statistical tests to identify a connection with the target label or to establish which attributes are more predictive of the target label, and it is independent of any learning algorithms. Because this technique ignores feature dependencies, it is computationally efficient. The wrapper technique evaluates a subset of features based on their utility to a given class. It uses a learning model to assess the subset of features based on their predictive power, but it is significantly very costly owing to repetitive learning and cross-validation. Embedded techniques are analogous to wrapper methods, except that they include feature selection during the training phase [93].

(1) Filter methods

Univariate Feature Selection. To determine the link between the features and the target variable, univariate feature selection evaluates each feature independently. Because they pertain to linear classifiers created using single variables, the following univariate feature selection methods are linear.

The filter approach chooses attributes without focusing on the core goal of improving any classifier's performance. To score the attributes, it uses the data's most important attributes. If d features or attributes are identified as S, the goal of a filter-based approach is to pick a subset of $m < d$ features, T, which maximizes some function F:

$$\tau^* = \operatorname{argmax}_{\tau \sqsubseteq S} F(\tau), \text{s.t.} |\tau| = m. \tag{1}$$

It finally settles on the top-m rated features with the highest scores. This number is known as joint mutual information, and maximizing it is an NP-hard optimization problem since the number of potential feature combinations rises exponentially.

The following are the often used linear univariate filter techniques in text classification:

The *information gain* approach selects features based on the item's frequency concerning the class/label prediction. Researchers have demonstrated that by removing superfluous features without modifying the features, the approach may lower the vector dimensionality of text and enhance classification results [94]. It adds the most value when the text corresponds to a certain label or class, and the word is also present in the document. It can be written as follows:

$$\begin{aligned} \mathrm{IG}(t) = &-\sum_{i=1}^{m} P(C_i)\log p(C_i) + p(t)\sum_{i=1}^{m} P(C_i \mid t)\log p(C_i \mid t) \\ &+ p(\bar{t})\sum_{i=1}^{m} P(C_i \mid \bar{t})\log p(C_i \mid \bar{t}). \end{aligned} \tag{2}$$

The utility of feature t in the classification is measured by this formula. If IG is higher than the prior value without the feature t, the current feature t is more relevant for classification. In other words, the discriminating power of the term t increases as the value of the information gain IG increases. Here, IG stands for information gain; C_i is the i-th class; $P(C_i)$ is the probability of an i-th class; and m is several target classes. $P(t)$ is the probability the feature t appears in the documents and the probability $P(\bar{t})$ for feature; t does not appear in the document. $P(C_i|t)$ is the conditional probability of the feature t appearing in i-th class. $P(C_i \mid \bar{t})$ is the conditional probability of the feature t that does not appear in i-th class.

The *Chi-square test* is a statistical strategy for assessing the relationship between a set of categorical features using their frequency distribution and determining how much the findings differ from the predicted output [95, 96]. This may be determined given events A and B, which are considered to be independent if

$$p(AB) = p(A)p(B). \tag{3}$$

The occurrence of the term and the occurrence of the class are the two events in feature selection. The terms are then ranked according to the following value. Chi-square can be calculated from the following equation:

$$\mathrm{CHI}^2(t, C) = \sum_{t \in \{0,1\}} \cdot \sum_{C \in \{0,1\}} \frac{\left(N_{t,C} - E_{t,C}\right)^2}{E_{tC}}, \tag{4}$$

where t denotes the feature, C denotes the specific class, $N_{t,C}$ is the frequency of feature t and class C occurring together, and $E_{t,C}$ is the frequency of feature t occurring without class C. The chi-square between each feature and class is computed, and the features with the highest chi scores are chosen.

Fisher score calculates the variance of the predicted value from the actual value to get the information score or how much knowledge one variable has about the unknown parameter on which the variable depends, and when the variance is the smallest, the information score is the highest. For a support-vector-based feature ranking model, researchers employed Fisher's linear discriminant [97–99].

For instance, let μ_j^k and σ_j^k be the mean and standard deviation of the k-th class, concerning the j-th feature.

Let μ^j and σ^j represents the mean and standard deviation of the entire training data concerning the j-th feature. The Fisher equation for determining the j-th feature's score is stated as follows:

$$F\left(x^j\right) = \frac{\sum_{k=1}^{c} n_k\left(\mu_k^j - \mu^j\right)^2}{\sigma_j^2}, \tag{5}$$

where σ_j^2 is computed as $\sum_{k=1}^{c} n_k (\sigma_k^j)^2$. Top-$m$ features with higher fisher scores are chosen by the algorithms.

Pearson's correlation coefficient is used to measure linear dependency between two continuous variables by dividing their co-variance by the product of their standard deviation, and its value ranges from −1 to +1. In two variables, a–1 value signifies a negative correlation; a +1 value shows a positive correlation; and a 0 value represents no linear association [93].

Using vectors, Pearson's coefficient r can be computed as follows:

$$r = \frac{(x_1 - \overline{x}_1 L)(x_2 - \overline{x}_2 L)}{\left(|x_1||x_2|\right)}, \tag{6}$$

where $\overline{x}_1$ is the mean of the vector x_1 and similarly for x_2, L is the vector of 1s, and $|x|$ is the magnitude of vector x.

Variance threshold is a technique for reducing vector dimensionality by deleting all low-variance features. Features that have a lower training-set variance than the threshold will be deleted [100, 101].

$$s = \frac{\sqrt{\sum (x_i - \overline{x})^2}}{(n-1)}. \tag{7}$$

The equation may be used to find features that have a variation below a given threshold. When the feature does not vary much within itself, it is seen to have low predictive potential.

Multi-Variate Filter Methods. During the assessment of the multi-variate filter selection approach, the interdependencies of features are also taken into account to choose relevant features.

It is based on mutual information that discovers the features in a feature set with the highest dependency with the target label. However, it is not appropriate for use when the goal is to achieve high accuracy with a small number of features.

Alternatively, it may utilize max relevance, which detects features with a dependency by averaging all mutual information values between all features x_i and target label c. S refers to features, and I represents mutual information; in the following equation, it is calculated between feature i and class c:

$$\max D(S,c), D = \frac{1}{|S|} \sum_{x_i \in S} \left(I\left(x_{i;}c\right)\right). \tag{8}$$

However, this results in a high level of redundancy, that is, a higher level of a dependency across features. As a result, to locate mutually exclusive features, minimum redundancy can be used [102].

$$\min R(S), R = \frac{1}{|S|^2} \sum_{x_i, x_j \in S} I\left(x_i, x_j\right), \tag{9}$$

where $I(x_i,x_j)$ is the mutual information between feature i and j.

Multi-Variate Relative Discriminative Criterion. The author offers a multi-variate selection strategy that takes into account both feature relevance and redundancy in the selection process. The RDC is used to assess the relevance, whereas Pearson's correlation is used to assess redundancy between features [103]. This measure boosts the rankings of terms that are exclusively found in one class or whose term counts in one class are much higher than in the other.

$$\mathrm{RDC}\left(w_i, tc_j(w_i)\right) = \frac{\left(\left|\mathrm{d}f_{\mathrm{pos}}(w_i) - \mathrm{d}f_{\mathrm{neg}}(w_i)\right|\right)}{\min\left(\mathrm{d}f_{\mathrm{pos}}(w_i), \mathrm{d}f_{\mathrm{neg}}(w_i)\right) * tc_j(w_i)}, \tag{10}$$

where $\mathrm{d}f_{\mathrm{pos}}(w_i), \mathrm{d}f_{\mathrm{neg}}(w_i)$ are the collection of positive and negative text documents, respectively, in which the term w_i is occurred. The word may be repeated several times in specific documents and represented by $tc_j(w_i)$. Instead of adding together RDC values for all term counts of a term, the area under the curve (AUC) for a difference graph is treated as term rank.

Researchers have been looking for novel approaches to increase classification accuracy while also reducing processing time. The author has provided a differentiate feature selector, a filter-based strategy that has picked unique features that have term properties while eliminating uninformative ones [92]. It provided efficiency by reducing processing time and improving classification accuracy.

(2) Wrapper Methods. Wrappers' approaches are bound to a certain classifier; the methods choose a subset of features based on their influence on the classifier by assessing the prediction performance of all potential feature subsets in a given space. It signifies that the features subset will be assessed by interacting with the classifier, which will improve the classification technique's accuracy. As the feature space expands, the computing efficiency suffers as a result of this method. Wrappers are used to choose features for other models as filters. The procedure may be accomplished in three ways: the first methodology employs a best-first search technique; the second methodology employs a stochastic approach such as random selection; and the third

methodology use heuristics such as forward and backward passes to include and omit features.

Multi-Variate Feature Selection. Univariate feature selection approaches are computationally efficient, but they eliminate features owing to a lack of interaction between features that, when combined, may have offered important information regarding classification [104, 105]. When evaluating the performance of features, multi-variate takes into account the interdependencies between them. "Linear multi-variate" employs linear classifiers made up of a subset of features, with the score of feature subsets being calculated based on classification performance. Nonlinear multivariate, on the other hand, use nonlinear classifiers to complete the task.

The following are the most often used *linear multivariate wrapper* approaches in in-text classification:

Recursive Feature Elimination. It is a recursive strategy that ranks features according to a key measure. During each cycle, the significance of features is assessed, and less relevant features are removed. To design ranking, the opposite process is utilized, in which features are rejected. From this rating, this technique extracts the top-N features [106]. This is a greedy optimization that seeks the highest performing feature subset.

Forward/Backward Stepwise Selection. It is an iterative procedure that begins with the examination of each feature and picks the one that produces the best performing model, based on some predetermined criteria (like prediction accuracy). The next step is to examine every potential combination of that selected feature and the following feature, and if it improves the model, the second feature is chosen. The model continuously appends the list of features that best improve the model's performance in each iteration until the requisite features subset is picked. In the backward feature selection approach, the method starts with the whole collection of features and discards the least relevant feature in each iteration, improving the method's speed. This method is repeated until no improvement is shown when features are removed, and the best subset of features is found. In comparison to other techniques, the researcher developed a rapid forward selection methodology for picking the optimal subset of features that required less computing work [107].

The *Genetic Algorithm* uses a feature set to generate a better subset of features that are free of noise. At each step, a new subset is formed by picking individual features in the correct sequence and merging those using natural genetics procedures. The result is cross-validated variance divided by the percentage of right predictions. The end outcome may be mutated. This procedure aids in the creation of a feature set of individual features that are more appropriate for the model than the initial feature set. The chaotic genetic algorithm was designed to simplify the feature selection procedure and improve the classification technique's accuracy [108, 109].

The commonly preferred *nonlinear multi-variate wrapper* methods in the text classification are discussed as follows:

Nonlinear kernel multiplicative updates entail iteratively training a classifier and rescaling the feature set by multiplying it by a scaling factor that lowers the value of less impacted features. Nonlinear techniques can outperform linear algorithms by selecting a subset of features [110].

Relief is based on instance-based learning. Each feature receives a value ranging from –1 to +1 based on how well it matches the desired label. The algorithm's scope is binary-classification-compatible [111, 112].

(3) Embedded Methods. In terms of computing, "embedded methods" outperform wrappers, but they conduct selection features as a subpart of the learning methodology, which is primarily exclusive to the learning model and may not function with any other classifier.

The commonly preferred embedded methods in the text classification are discussed as follows:

In social sciences, the *LASSO* method is generally used [113]. To alleviate the dimensionality problem, it penalizes features with large coefficients by inserting a penalty during the log-likelihood maximization procedure. By picking a correct weight and reducing dimensionality, LASSO assigns zero to some coefficients. When there is a strong correlation between some features, it creates a difficulty [114].

Ridge Regression lowers the complexity of a model by reducing coefficients while keeping all of its features. The issue with ridge regression is that features are retained. If the feature collection is huge, the problem remains complicated [115].

Elastic Net calculates a penalty that is a mix of LASSO and ridge penalties. The elastic net penalty may be readily handled to give LASSO or ridge penalties extra power. It has a grouping effect, with high correlation features tending to be in or out of the feature subset. It incorporates both L1 and L2 regularization techniques (LASSO and ridge). By fine-tuning the settings, it aids in the implementation of both strategies [116].

2.4.2. Text Feature Extraction Methods. After selecting the features and representing N documents by d-dimensional features *vectors* $\{X_1, X_2, \ldots, X_N\}$. Sometimes original terms in the form of features may not be optimal dimensions for text document representation. The text feature extraction methods try to solve these problems by creating new feature space Y or artificial terms [117]. It requires (a) a method to convert old terms to new terms and (b) a method to convert document representation from old to new. A popular example commonly used for this purpose is principal component analysis (PCA) in which a feature set Yi is selected in a manner that the variance of the original feature vectors is maximized in the direction of new feature vectors. This is

done by computing eigenvectors of the covariance matrix of the original vectors. The drawback of the PCA is the time it takes to evaluate eigenvalue decomposition to compute the principal component for each class/label when applied to a large data set. This is overcome by the researcher by using the power factorization method (PFM) to find a fixed quantity of eigenvectors from a data set [118]. Another commonly preferred method for feature extraction is latent semantic indexing (LSI) that uses singular value decomposition of the term correlation matrix computed from a large collection of text documents. This technique is used to address the problem of deriving from the use of synonymous and polysemous words as dimensions of the text document representation. But the disadvantage is to compute the correct number of latent components that proves computationally expensive. Another method that helps for optimal discrimination of data is linear discriminant analysis (LDA) [99]. It identifies the linear collection of features that best explain the data. It tries to find the model that can explicitly differentiate between the classes of data. Latent Dirichlet allocation is another method that explains that each text document is a mixture of latent topics and each word in that document is attributable to one of the topics of that document. This is most preferred for topic modeling [119]. This is a generative probabilistic model.

A newer approach is a simplified version of stochastic neighbor embedding that creates much better visuals by eliminating the potential to cluster points together in the map's centers known as t-SNE. It visualizes high-dimensional data by assigning a two- or three-dimensional map to each data point. When it comes to constructing a single map that displays structures of several sizes, t-SNE outperforms previous approaches. On almost all of the data sets, the analysis shows that t-SNE produces visuals that are much superior to those produced by the other approaches [120]. Furthermore, the authors provide a technique UMAP (uniform manifold approximation and projection) that is comparable to t-SNE in terms of visualization quality and, in certain ways, retains more of the global structure while providing better run time efficiency [121]. It is based on Laplacian eigenmaps as a mathematical foundation. Umap can scale to far bigger data sets than t-SNE. It is a general-purpose dimension reduction strategy for machine learning since it has no computational constraints on embedding dimensions. The approach is used in the study to evaluate the uniqueness of subjects, important phrases and features, information dissemination speed, and network behaviors for COVID-19 tweets and analysis [122]. To increase the detection of relevant themes in the corpus and analyze the quality of created topics, they use UMAP, which finds distinctive clustering behavior of separate topics [120]. Another study used UMAP to depict the matching word vector spaces of a pretrained language model using LaTeX mathematical formulations. In the LaTeX formula domain, they develop a state-of-the-art BERT-based text classification model augmented by unlabeled data (UL-BERT) [123].

2.5. Classifiers for Classification Task. The classifier is trained based on the selected features from the text documents. The selection of appropriate features in feature space decides the performance of learning models. Machines understand numbers more easily than texts as input. So texts as tokens are required to be converted into numbers (vectorization) for most of the learning algorithms. Vectors are combined to originate vector space to apply statistical methods for checking document relatedness. Each algorithm offers a different document representation for text classification. Researchers offer several classification methods that work on the vector representation of texts. The basic assumption for vector representation of texts is known as the contiguity hypothesis. It states that text documents belonging to the same class develop a contiguous region and regions of different classes do not overlap. The relatedness of the documents can be evaluated on 2D space based on cosine similarity or Euclidean distance.

The authors have presented naïve Bayes, a linear classifier, approach to vectorize the text documents according to probability distribution with two commonly used models: multi-variate Bernoulli event and multi-nomial event; the features with the highest probability were chosen to reduce the dimensionality [124].

$$PC_k|d_i = Pd_i|C_k * \frac{P(C_k)}{P(d_i)}. \tag{11}$$

The output of the classifier is the probability of the text document d_i is belonging to each class C_k, and it is a vector of C elements. In a way of text classification scenario, we could compute $Pd_i|C_k$ using bag-of-words as follows:

$$Pd_i|C_k = P(BoW(d_i)|C_k) = Pw_{1,i}w_{2,i},,w_{|V|,t}|C_k. \tag{12}$$

The problem can be reduced to compute the probability of each word $w_{j,i}$ in class C_k as follows:

$$Pd_i|C_k = \prod_{j}^{|V|} Pw_{j,i}|C_k. \tag{13}$$

The naïve Bayes has a high bias for a nonlinear problem because it can model one type of class, that is, a linear hyperplane. The bias is the statistical method of evaluating the performance of a classifier that how accurately a classifier classifies the texts into the correct class with less error. The learning task is the last activity in classification, but reducing the feature dimensions is more concerned with the efficiency of the classifier model. The possibility of salient feature reduction caused by using classifiers can be overcome by using the model SVM that identifies the best decision-boundary between feature vectors of the document with their categories [125]. The SVM entails an optimal classifier that guarantees the lowest classification error. The SVM computes a hyperplane that falls between the positive and negative example of the training set.

$$D = \left\{(x_i, y_i)\middle| x_i \in R^D, y_i \in \{-1, 1\}\middle|\right\} i=_i^n, \tag{14}$$

where the minus sign represents the negative hyperplane, the positive sign points to the positive hyperplane, i ranges from 1 to L (training examples), (x_i, y_i) represents feature vectors of each document, R^D is a vector space having a dimension of D. and D is evaluated to +1 and −1 for positive and negative hyperplanes, respectively.

The naïve Bayes itself results in the best classification model if it is trained on a high volume of data. However, feature reduction remains an issue. So naïve Bayes is used as a prestep to SVM that converts text documents into vectors before the classification task starts. This resulted in improving the whole system while spending quite an appropriate classification time by reducing to low-dimensional space. But, in certain cases, the majority of features are redundant with each other; the author has presented a divergence-based feature selection method without relying on a complex dependence model where the maximum marginal relevance-based feature selection was outperformed by the SVM [126]. The paper has suggested the need for novel criteria to measure the relevance and novelty of features separately and provided the linear combination as the metric.

The studies have mentioned the KNN, a nonlinear classifier, where the algorithm classifies the document by moving through all the training documents that are like that document. The KNN model frames the documents in the Euclidean space as points so that the distance between two-point u and v can be calculated as follows:

$$D(u, v)^2 = u - v^2 = (u - v)^T (u - v) = \sum_{i-1}^{d} (u_{i-} v_i)^2. \quad (15)$$

The classifier finds the K-value that is the factor that represents a collection of documents from all the documents closest to the selected document in that space [127]. If there are too many features, KNN may not operate effectively. As a result, dimensionality reduction techniques such as feature selection and principal component analysis may be used.

$$\hat{y}(x) = y_{n^*} \quad \text{where} \quad n^* = \arg\min_{n \in D} \text{dist}(x, x_n). \quad (16)$$

The study mentions that using KNN increases the overhead to calculate the K-value of all the documents with all other training documents with the largest similarity or closet to the selected document. Also, the variation in the number of training sample documents in different categories leads to a decline in accuracy. Due to the high variance and complex regions between classes, it becomes sensitive to noise documents. Sometimes, the document tends to misclassify if it occurs very relevant to a noise document in the training set and sometimes accurately classified if there is no presence of noise documents in the training set close to them. This ends up in high variance from the training set to the training set. High variance leads to overfitting. The goal of finding a good learning classifier is to decrease the learning error. Learning error is calculated from bias and variance or bias-variance trade-off. The traditional algorithms possess some limitations that attract the researchers to improve the efficiency by (a) reducing the computational overheads by establishing low-dimensional space, (b) speeding up the computational capacity of finding nearest neighbors in KNN or locating decision boundaries in SVM, and (c) increasing efficiency by not compromising accuracy [128]. The model is chosen that optimizes the fit to the training data.

$$H^* = \arg\max_H \text{fit}(H|D). \quad (17)$$

The supervised learning classification algorithms such as naïve Bayes, SVM, and KNN use a bag of words to design a classifier. None of these methods take the order of words into consideration that can lead to the loss of some valuable information. In natural-language-based problems, the order of the words, for example, multi-words, has a meaning (like names of organization or person) that is not considered by the learning models trained on individual words of the texts. The algorithm n-grams consider the sequence of n-adjacent words from the selected text phrases. The n-grams behave like the individual word's representation as feature vectors. The value of n may range from 1 to the upper value [129]. This proves very beneficial in short text documents where the number of n-grams is less in number [89]. The author proposes another representation for character n-grams to introduce the enhancement of the skip-gram model, which considers subword into account. It considers the word morphology while training the model, and words are represented by the sum of its character n-gram [55].

Many researchers have used a decision-tree-based algorithm (decision support tool) that represents a tree-structure-based graph of decisions [130]. The commonly used decision tree algorithms are ID3, C4.5, and C5. The algorithm presents each intermediate node (labeled as terms and branches represent weight) that can split into subtrees and ends at leaf nodes (represents the class/label/outcome of the problem). Decision tree structures are rule-based solutions. A rule can be designed by forming a conjunct of every test that occurs on the path between the root node and the leaf node of the tree. The rules are formed after traversing every path from a root to the leaf node. Once the decision tree and rule are framed, it helps assign the class/label for a new case [129, 131]. It was evaluated in the study that decision trees result better than naïve Bayes in terms of accuracy but a little worse than KNN methods [132]. As a result, the authors introduce the boosting data classification approach. The boosting algorithm is a method for combining many "poor" classifiers into a single, powerful classifier. The boosting technique is used on decision trees in the study, and the boosted decision tree performs better than an artificial neural network [133]. It is expected to find widespread use in a variety of fields, particularly text classification. Gradient tree boosting is another boosting strategy that builds an additive regression model using decision trees as the weak learner. Trees in stochastic gradient boosting are trained on a randomly selected portion of the training data and are less prone to overfitting than shallow decision trees [134].

Neural networks or deep learning systems use several processing layers to learn hierarchical data representations and have reached state-of-the-art outcomes in most domains. In the sense of natural language processing (NLP), several model designs and approaches have recently

progressed. In areas such as computer vision and pattern recognition, deep learning architectures and algorithms have also made remarkable progress. Recent NLP research is now primarily focused on the application of new deep learning approaches, continuing this development. In areas such as computer vision and pattern recognition, deep learning architectures and algorithms have also made remarkable progress. Recent NLP research is now primarily focused on the application of new deep learning approaches, continuing this development. The performance of word embedding and deep learning strategies referred to in the following section is driving this development.

In the late 90s, the researcher found an application of nonlinear neural networks to text classification or topic modeling [135]. In this model, a three-layered neural network was designed to learn a nonlinear mapping from training documents to each class/label. Later on, the researcher proposes convolutional neural network (CNN) for text classification by considering the order of words in the phrases, and this outperforms SVM in terms of error rate [136]. CNN uses the vector representation of text data considering the order of words. Each word is considered a pixel, and the document is treated as an image. Then the image is taken into $|D| \times 1$ pixels, and each pixel represent a word as a $|V|$ dimensional vector. For instance, vocabulary $V = \{$"classification", "course", "I", "love," "NLP'," "text"$\}$, and words are taken as a dimension of vectors in alphabetical order. And document $D =$ "I love NLP course". Then the document vector would be $X = [0010000 \mid 000100 \mid 00001 \mid 010000]$.

The researcher mentions that to reduce the dimensionality of vector space, the vocabulary size must be kept low. Also, the n-gram algorithm ignores the fact that some n-grams share the constituent words; this is overcome by CNN that learns the embedding of text regions by providing CNN with the constituent words as input, and this technique provides higher accuracy. To construct an informative latent semantic representation of the sentence for downstream activities, CNNs have the potential to extract salient n-gram features from the input sentence. The author proposes a three-way enhanced CNN for classification for sentiment analysis, where decisions are divided into three parts accept, reject, and delay. The instances in boundary regions that are neither in accept nor reject are reclassified by another classification model. This guarantees the enhancement in CNN to deal with boundary regions in a better way, resulting in model 3W–CNN [137]. Another study has shown the application of CNN in document modeling for personality detection based on text in the context of sentiment analysis [34]. Overall, in contextual windows, CNNs are highly successful in mining semantic hints. They are very data-heavy models, though. They have a huge range of parameters that are trainable and need tremendous training data. This raises a concern as data shortage happens. Another unresolved concern with CNNs is their failure to model contextual long-distance data and maintain sequential order in their representations [138].

Deep neural networks are difficult to train on data; it requires a lot of resources to get high performance. The feed-forward neural network commonly known as multi-layer perceptron (MLP) is the most preferred technique in classification problems. In a feed-forward neural network, the information travels in one direction from the input layer to hidden layers and then followed by the output layer. They have no memory of the input received previously so lack in predicting what comes next. To overcome this, RNN (recurrent neural network) is preferred where information moves through the loop. In this paragraph, we discuss the fundamental characteristics that have favored the popularization of RNNs in a variety of NLP tasks. Since an RNN performs sequential processing in sequence by modeling units, it may have the ability to generate the intrinsic sequential structure present in language, where characters, words, or even phrases are units. Based on the previous words in the sentence, words in a language establish their semantic meaning. The disparity in interpretation between "computer" and "computer vision" is a clear example that states this. RNNs are perfect for language and related sequence modeling activities to predict certain context dependencies, which turned out to be a clear incentive for researchers to use RNNs over CNNs in these fields.

RNNs were originally three-layer networks in the NLP sense [139]. While deciding on the current input layer, it considers what it has learned from the previous inputs. Basically, in the architecture of simple RNN, the hidden units create internal representations for the input patterns and recode these patterns in feed-forward networks using hidden units and a learning algorithm in a way that allows the network to generate the appropriate output for a given input. Typically, the hidden state of the RNN is assumed to be the most important feature. It can be regarded as the memory portion of the network that accumulates data from other steps.

The formula of the current state in RNN can be written as mentioned below. A nonlinear transformation such as tanh, or ReLU, is taken to be the function f.

$$h_t = f(h_{t-1}, x_t), \tag{18}$$

where h_t is the new state, h_{t-1} is the previous state, and X_t is the input at time t.

The tanh function is commonly used as an activation function. The weights can be defined as the matrix W_{hh}, and input is defined by the matrix W_{xh}:

$$h_t = \tan h(W_{hh}h_{t-1} + W_{xh}X_t). \tag{19}$$

The output can be calculated during test time as follows:

$$y_t = W_{hy}h_t. \tag{20}$$

The output is then compared to the actual output, and then the error value is computed. The network learns by backpropagating the error through the network to update the weights. But usual RNN has a short-term memory. These basic RNN networks suffer from the issue of vanishing gradient, which makes it very difficult to understand and adjust the parameters of the previous layers in the network. It is used in combination with LSTM, which has long-term memory, and it gives an extension to the memory of the usual RNN. Over the basic RNN, LSTM has additional "forget" gates, allowing the error to backpropagate over an

infinite amount of time steps. Comprising three gates: input, forget, and output gates, taking a combination of these three gates, it determines the hidden state [22]. The applications based on RNN and LSTM have been used in solving many NLP problems due to their capacity of capturing complex patterns within the text [140]. It has also been used in sequence labeling tasks in POS (part of speech) activity. It is preferably used in topic modeling for fake news [141, 142], sentiment analysis [143, 144], and negative speech detection on social media. More recently, authors have suggested another type of recurrent unit, which they refer to as a gated recurrent unit (GRU) [145]. It has been shown that RNNs employing any of these recurrent units perform well in tasks (such as machine translation, speech recognition, or depending parsing in text documents for NER) requiring long-term dependency capture. The application of gated RNN is not limited to the mentioned tasks, but it can be applied to different NLP challenges [146]. GRU is a form of recurrent neural network (RNN) that can process sequential data using its recurrent architecture. The fundamental issue in text classification is how to improve classification accuracy, and the sparsity of data, as well as semantics sensitivity to context, frequently impedes text classification performance. The study introduces a unified framework to evaluate the impacts of word embedding and the gated recurrent unit (GRU) for text classification to overcome the flaw [147].

Recurrent neural networks, in particular long short-term memory [22], and gated recurrent neural networks [62] are firmly known as state-of-the-art approaches in sequence modeling. Within training examples, the inherently sequential nature of recurring models prevents parallelization, which becomes important at longer sequence lengths, as memory limitations restrict batching through examples. Recent work, through factorization tricks [148] and conditional computation [149], has achieved substantial improvements in computational efficiency while also improving model output in the case of the latter. The authors have presented two simple ways of reducing the number of parameters and speeding up the training of large long short-term memory networks: the first is the "matrix factorization by design" of the LSTM matrix into two smaller matrices, and the second is the division into separate classes of the LSTM matrix, its inputs, and its states. However, the essential restriction of sequential computation persists.

Attention mechanisms have become an integral part of sequence modeling in different applications, allowing dependencies to be modeled regardless of their gap in the input or output sequences. The following paragraph examines some of the most prominent models of attention that have created new state-of-the-art tasks for text classification. The study's conclusion was largely focused on text classification experiments that could not be extended to many other NLP tasks [150]. The authors mentioned that attention offers a reliable explanation for model predictions, expecting these properties to hold (a) attention weights should align with feature-relevant measurements (e.g., gradient-based measurements) and (b) alternative (or counterfactual) weight configurations should result in corresponding prediction changes. They stated that in the sense of text classification, neither property is consistently observed by a Bi-LSTM with a standard attention mechanism. The layers of attention specifically weigh the representations of the input elements; it is also often believed that attention can be used to classify information that was considered relevant by models. In another study, the authors evaluate if that assumption holds by modifying weights of attention in already trained text classification methods and examining the resulting variations in their predictions. Although experimenting with text classification, the authors note several cases in which higher attention weights correlate with a greater impact on model predictions, they also notice several cases this does not hold, that is, where gradient-based attention weight rankings predict their effects better than their magnitudes [151]. In contrast to the current work on interpretability, the authors in another research reported that they examined the attention mechanism on a more diverse set of NLP tasks that included text classification, pairwise text classification, and tasks for generating text such as neural machine translation [152].

Although the hidden vectors represented by an attention model through encoding can be interpreted as internal memory entries for the model, memory-augmented networks integrate neural networks with an external memory type that the model could learn from it and respond to. For text classification, the study proposes a memory-augmented neural network called the neural semantic encoder [153]. In another research, the authors introduce neural network architecture, the dynamic memory network (DMN), which processes input sequences and questions, shapes episodic memories, and generates appropriate responses. The model possesses an iterative mechanism of attention that allows the model to condition its attention on the inputs and outcomes of previous iterations. In a hierarchical recurrent sequence model, these outcomes are then reasoned over to produce answers [154]. The authors also stated that it is possible to train the DMN end-to-end and obtain state-of-the-art text classification results using the Stanford Sentiment Treebank data collection.

Sequential processing of text is one of the system inefficiencies experienced by RNNs. Transformers address this constraint by applying self-attention to measure an "attention ranking" in parallel for each word in a phrase or document to model the impact each word has on another. Because of this feature, Transformers allow for far more parallelization than CNNs and RNNs, allowing very large models to be efficiently trained on large quantities of data on GPU stacks [155]. The Transformer architecture is especially suitable for pretraining large text corpus, leading to significant accuracy improvements in downstream tasks, including text classification [69]. They propose a novel Transformer-XL neural architecture that allows learning dependence to interrupt temporal coherence beyond a fixed length. We have seen the emergence of several large-scale transformer-based pretrained language models in the current scenario. As stated in Section 2.2.2, these Transformers are pretrained to learn contextual text representations in much greater volumes of text corpora by predicting terms that are trained on their context. These pretrained models

were fine-tuned using task-specific tags and in many subsequent NLP tasks, especially text classification produced new state-of-the-art. Fine-tuning is supervised learning, while pretraining is unsupervised.

The authors design the largest model, OpenGPT, 1.5 B parameter Transformer that ensures state-of-the-art results and comprises 12 layers of Transformer frames, each composed of a masked multi-head attention unit, followed by a standardization layer and a forward feed layer in place. With the addition of task-specific linear classifiers and fine-tuning with task-specific tags, OpenGPT can be extended to the text classification. Unlike OpenGPT that predicts words based on previous predictions, there is another model that comes into use, that is, BERT, intended to pretrain deep bidirectional representations from the unlabeled text by conditioning in all layers on both the left and right context together [71]. For text classification, BERT variants have been fine-tuned [156]. ALBERT decreases memory usage and improves BERT's training speed [157]. Another variant of BERT, SpanBERT [158], is a pretraining method designed to accurately represent and forecast spans of text. It improves BERT by (1) masking consecutive random spans, rather than random tokens, and (2) training the span boundary representations to estimate the entire content of the masked span without relying on the individual token representations within it. Deep learning provides a way to manage massive volumes of processing and data with next to no engineering by hand [23]. Unsupervised learning has had a catalytic impact on the growing interest in deep learning, but the contributions of solely supervised learning have since been overshadowed. In the longer term, we expect unsupervised learning to become even more significant.

3. Evaluation

We have discussed several supervised and unsupervised text classification methods based on machine and deep learning models so far; however, the shortage of uniform data collection procedures is a big issue when testing text classification techniques. Even if there is a standard collection method available, it can generate differences in model results by simply selecting different training and test sets [146]. Moreover, to compare various performance measures used during separate tests, there may be another difficulty related to process evaluation. In general, performance metrics assess attributes of the performance of the classification task and therefore do not necessarily present similar information. Although the underlying mechanics of various measurement metrics vary, it is important to consider precisely what each of these metrics describes and what kind of data they are attempting to express for comparability. Some examples of such performance measures include precision, recall, accuracy, microaverage, macroaverage, and F-measure. These calculations are based on a "confusion matrix" composed of true positive, false positive, false negative, and true negative [47]. Accuracy is considered the fraction of accurate predictions in overall predictions. The fraction of known positives accurately estimated is referred to as recall. The fraction of positives accurately estimated for all positives is called precision.

Another technique for evaluating how well our machine learning models perform on unknown data is cross-validation. If we expose the model to entirely new, previously unknown data, it may not be as accurate in predicting and may fail to generalize over the new data. Overfitting is the term for this issue. Because it is unable to discover patterns, the model does not always train effectively on the training set. It would not do well on the test set in this situation. Underfitting is the term for this issue. We employ cross-validation to solve overfitting issues. A cross-validation is a resampling approach in which the data set is split into two parts: training data and test data. The model is trained using training data, and the model is predicted using test data that has yet to be observed. If the model performs well on the test data and has a high level of accuracy, it has not overfitted the training data and may be used to forecast. *K*-fold cross-validation is the most basic type of cross-validation. Other types of cross-validation include variations on *k*-fold cross-validation or entail repeating *k*-fold cross-validation rounds. The data is initially partitioned into *k* equally sized segments or folds in *k*-fold cross-validation. Following that, *k* iterations of training and validation are undertaken, with each iteration holding out a different fold of the data for validation and the remaining *k*-1 folds being employed for learning [147]. For each of the *k* "folds", the following approach is used:

(i) The folds are used as training data to train a model

(ii) The generated model is validated using the remaining data (i.e., it is used as a test set to compute a performance measure such as accuracy)

In cases where there is uncertainty, entropy is especially appealing as a predictor of classification quality. It denotes how well the class membership probabilities are distributed throughout the specified classes. Its usefulness as a predictor of classification accuracy is predicated on the notion that in an accurate classification, each sentence has a high likelihood of belonging to just one class [159]. Cross-entropy loss is a key indicator for evaluating the performance of a classification issue. The prediction is a probability vector, which means that it reflects the anticipated probabilities of all classes, which add up to one. In a neural network, this is commonly accomplished by activating the final layer with a softmax function, but anything goes, it just has to be a probability vector. Maximum entropy is used in our text classification scenario to estimate the conditional distribution of a class label given a document. A collection of word-count characteristics represents a document. On a class-by-class basis, the labeled training data is utilized to estimate the anticipated value of these word counts. Iterative scaling is improved to obtain a text classifier with an exponential shape that is compatible with the limitations of the labeled data [160].

4. Comparative Analysis

The following section summarizes the benefits and limitations of feature extraction, feature selection methods, and supervised and unsupervised machine and deep learning models used for a text classification task.

Table 1: Benefits and limitations of text representation or feature extraction methods.

Method	Benefits	Limitations
Bag-of-words	Works well with unseen words and is easy to implement as it is based on the most frequent terms in a document	Does not cover the syntactic and semantic relation of the words, common words impact classification
TF-IDF	In like bag-of-words approach, common words are excluded due to IDF so does not impact the result	Does not cover the syntactic and semantic relation of the words
Word2vec	Covers the syntactic and semantic relation of the words in the text	Does not cover the words' polysemy
GloVe	As the same as word2vec but performs better, eliminates common words, trained on a large corpus	Does not cover the words' polysemy and does not work well for unseen words
Context-aware representation	Covers the context or meaning of the words in the text	Huge memory is required for storage and does not work well for unseen words

Table 2: Benefits and limitations of feature selection methods.

Method		Benefits	Limitations
Univariate filter method	Information gain	Results into the relevance of an attribute or feature	Biased towards multi-valued attributes and overfitting
	Chi-square	Reduces training time and avoids overfitting	Highly sensitive to sample size
	Fishers' score	Evaluates features individually to reduce the feature set	Does not handle features redundancy
	Pearson's correlation coefficient	Is simplest and fast and measures the linear correlation between features	It is only sensitive to a linear relationship
	Variance threshold	Removes features with variance below a certain cutoff	Does not consider the relationship with the target variable
Multi-variate filter method	mRMR (minimal redundancy maximum relevance)	Measures the nonlinear relationship between feature and target variable and provides low error accuracies	Features may be mutually as dissimilar to each other as possible
	Multi-variate relative discriminative criterion	Best determines the contribution of individual features to the underlying dimensions	Does not fit for a small sample size
Linear multi-variate wrapper method	Recursive feature elimination	Considers high-quality top-N features and removes weakest features	Computationally expensive and correlation of features not considered
	Forward/backward stepwise selection	Is computationally efficient and greedy optimization	Sometimes impossible to find features with no correlation between them
	Genetic algorithm	Accommodates data set with a large number of features and knowledge about a problem not required	Stochastic nature and computationally expensive
Nonlinear multi-variate wrapper methods	Nonlinear kernel multiplicative	De-emphasizes the least useful features by multiplying features with a scaling factor	The complexity of kernel computation and multiplication
	Relief	Is feasible for binary classification, based on nearest neighbor instance pairs and is noise-tolerant	Does not evaluate boundaries between redundant features, not suitable for the low number of training data sets
Embedded methods	LASSO	L1 regularization reduces overfitting, and it can be applied when features are even more than the data	Random selection when features are highly correlated
	Ridge regression	L2 regularization is preferred over L1 when features are highly correlated	Reduction of features is a challenge
	Elastic net	Is better than L1 and L2 for dealing with highly correlated features, is flexible, and solves optimization problems	High computational cost

4.1. Comparative Analysis of Standard Text Representation or Feature Extraction Methods. The two major feature extraction methods are highlighted: weighted words and embedding of words. By considering their frequency and co-occurrence details, word embedding approaches learn through sequences of words. These strategies are also unsupervised models to create word vectors. In comparison, the properties of weighted terms are based on counting words in documents and can be used as a basic word representation ranking scheme. Each approach poses specific constraints. Weighted words explicitly quantify text similarities from the word-count space, which enhances the computational time for large vocabulary. Although counting unique terms offers

Table 3: Benefits and limitations of the machine and deep learning model.

Model	Benefits	Limitations
Naïve Bayes	It needs less training data; probabilistic approach handles continuous and discrete data; and it is not sensitive to irrelevant features, easily updatable	Data scarcity can lead to loss of accuracy because it is based on assumption that any two features are independent given the output class.
SVM	It is possible to apply to unstructured data also such as text, images, and so on; kernel provides strength to the algorithm and can work for high-dimensional data	It needs long training time on large data sets and is difficult to choose good kernel function, and choosing key parameters varies from problem to problem.
KNN	It can be implemented for classification and regression problems and produces the best results if large training data is available or even noisy training data, preferred for multi-class problems	Cost is high for computing distance for each instance; finding attributes for distance-based learning is quite a difficult task; imbalanced data causes problems; and no treatment is required for missing value.
Decision tree	It reduces ambiguity in decision-making; implicitly performs feature selection, easy representation, and interpretation; and requires fewer efforts for data preparation	It is unstable due to the effect of changes in data requires changes in the whole structure, is not suitable for continuous values, and causes overfitting problem.
Boosted decision tree	It is highly interpretable and prediction accuracy is improved. It can model feature interactions and execute feature selection on its own. Gradient boosted trees are less prone to overfitting since they are trained on a randomly selected subset of the training data.	These are computationally expensive and frequently need a large number of trees (>1,000), which can take a long time and consume a lot of memory.
Random forest	In contrast to other methods, clusters of decision trees are very easy to train, and the preparation and preprocessing of the input data do not require.	More trees in random forests increase the time complexity in the prediction stage, and high chances of overfitting occur.
CNN	It provides fast predictions, is best suited for a large volume of data, and requires no human efforts for feature design.	Computationally expensive requires a large data set for training.
RNN	It implements feedback model so considers best for time series problems and makes accurate predictions than other ANN models.	Training of model is difficult and takes a long time to find nonlinearity in data, and gradient vanishing problem occurs.
LSTM, Bi-LSTM	Adds short- and long-term memory components into RNN so it considers best for applications that have a sequence and uses for solving NLP problems such as text classification and text generation, and computing speed is high. Bi-LSTM solves the issue of predicting fixed sequence to sequence.	It is expensive and complex due to the backpropagation model, increases the dimensionality of the problem, and makes it harder to find the optimal solution. Since Bi-LSTM has double LSTM cells so it is expensive to implement.
Gated RNN (GRU)	In natural language processing, GRUs learn quicker and perform better than LSTMs on less training data. As it requires fewer training parameters. GRUs are simpler and hence easier to modify and do not need memory units, such as by adding extra gates if the network requires more input.	Slow convergence and limited learning efficiency are still issues with GRU.
Transformer with an attention mechanism	The issue with RNNs and CNNs is that when sentences are too long, they are not able to keep up with context and content. By paying attention to the word that is currently being operated on this limitation was resolved, the attention strategy is an effort to selectively concentrate on a few important items while avoiding those in deep neural networks to execute the same operation, enabling much more parallelization than RNNs and thus reduces training times.	At inference time, it is strongly compute-intensive.

independent confirmation of similarity, semantic comparisons between words are not taken into consideration [148]. Word embedding techniques solve this challenge but are constrained by the need for a large corpus of text data sets to train [21]. Therefore, researchers tend to use vectors with pretrained word embedding [149]. Table 1 presents the advantages and limitations of each technique of text representation or feature extraction.

4.2. Comparative Analysis of Standard Feature Selection Methods. Some studies have preferred Fisher's linear discriminant for the support-vector-based feature ranking model [99, 100]. The author has mentioned that the filter-based method provides distinguish feature selector that has further selected distinctive features that possess term characteristics during the elimination of uninformative ones [92]. It offered performance by decreasing processing time and

increasing classification accuracy. Another technique in the wrapper method has gained popularity fast forward selection technique for selecting the best subset of a feature that demanded less computational effort as compared to other methods [107]. In the genetic algorithmic approach, a chaos genetic algorithm was proposed to simplify the feature selection method and obtained higher accuracy of classification technique [108, 109, 161]. Embedded methods were preferred over wrappers in many studies. Researchers mentioned that embedded methods have performed better than wrapper computationally, but these algorithms perform selection features as a subpart of the learning technique. Furthermore, these were followed by hybrid feature selection approaches where both filter and wrapper methods were combined, and these approaches were proven more computationally effective than the performance of a single selection technique. It was observed by the researchers that sometimes original features may not be optimal dimensions for text document representation. They provided text feature extraction methods to solve the problem by creating new feature space or artificial terms [117]. Table 2 presents the benefits and limitations of feature selection methods.

4.3. Comparison of State-of-the-Art Machine and Deep Learning Models for Text Classification. The performance of the classifier depends on the selection of feature selection and extraction method. The supervised and unsupervised machine learning techniques have offered a variety of classifiers that performed well in a variety of domain-specific classification problems. The following Table 3 presents their pros and cons. Recent studies have focused on deep learning or neural network-based classifiers such as CNN, RNN, and RNN with LSTM that have shown better results than conventional algorithms such as SVM and KNN in solving a different range of problems. RNN and LSTM have been used in many NLP applications due to their capacity of capturing complex patterns within the text [151]. It has also been used in sequence labeling tasks in POS (part of speech) activity. But it has offered a lot of future scope in resolving complexities involved in the backpropagation technique used in RNN and making the learning model cheaper and faster [104].

5. Research Gap and Future Direction

From this review of existing studies, we identified that there exist certain gaps, which we can plan to fill in the future. While labeling unstructured text data manually, it takes a lot of time to understand the data to categorize it. Also, it needs specialists for understanding domain-specific data. In such scenarios, the machine learning algorithms do not produce the expected accuracy [12]. Extraction of meaning or finding semantic relations between words of unstructured data is a complex task, which has been tried by several authors using NLP techniques for years [24, 162]. However, these methods prove inefficient when pursuing building a high-quality classification system. It offers opportunities to design semisupervised machine learning models to label some parts of training data manually, and the rest data can be trained using machine learning algorithms [163].

Apart from conventional methods used for representing data sets for extracting patterns from the text data using vector representation based on word-embedding and paragraph-embedding [38, 58], the authors have presented deep neural network models for NLP-based applications using character-embedding [164], unsupervised techniques based on transfer learning, Bert, with fine-tuning with domain-specific data [156]. However, these representations and globally available representations cannot be generalized to unseen texts, which are very specific to a particular domain [59]. It offers opportunities to design methodology for extending the vocabulary of existing representations for specific domains.

The deep learning algorithms are proven good in decision-making for NLP-based applications, and these models cannot handle symbols directly [165]. Also, the computational cost of training such algorithms is very high. It offers scope for designing deep neural network-based architecture, which can be inputted with linguistic knowledge, lexical knowledge, and word knowledge from different domains.

6. Conclusions

In this paper, we have provided a detailed review of the complete process of the text classification system. This paper covered various algorithms or methods used in subtasks of classification. It has presented the techniques for data collection from several online sources. The documents were represented with basic techniques and followed by recent research in document representation for different areas of machine learning and natural language processing. To provide suitable and fast classifiers the higher dimensional space of data was reduced to a lower space using feature selection and feature extraction methods. Different algorithms perform differently depending on the domain-specific data collections and to train machine learning text classifiers. The authors have used these algorithms based on the problem statement, and none of the algorithms has proven perfect for all types of problems and data dimensionality.

It is observed that in recent years, some studies focused on new applications of text classification such as multi-label classification [166, 167], and hierarchical classification [168, 169] in the field of natural language processing or machine translation and medical sciences, respectively. The neural network-based algorithms are commonly used in NLP-based problems [136, 137, 142]. However, these algorithms mainly focus on generic data. Moreover, CNN is preferably used for image processing, and RNN with LSTM is preferred for time-series problems. Initially, these algorithms were not applied to text data, but in recent years, CNN with character-embedding is being used for document representation and feature selection in text documents [58]. The transformers-based unsupervised models also overcome the issue of RNN and LSTM, but these techniques are proven computationally expensive. With the advancement of deep neural networks, we may find in the future that these deep

neural networks will be applied efficiently in the automatic monitoring of web-based text data and classifying unseen data into automated labels [7].

Data Availability

The data used in this study are publicly available.

Conflicts of Interest

The authors declare that there are no conflicts of interest.

Acknowledgments

Jana Shafi would like to thank the Deanship of Scientific Research, Prince Sattam bin Abdul Aziz University, for supporting this work. This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korean government (MSIT; No. 2022R1C1C1004590).

References

[1] A. Atkins, M. Niranjan, and E. Gerding, "Financial news predicts stock market volatility better than close price," *The Journal of Finance and Data Science*, vol. 4, no. 2, pp. 120–137, 2018.

[2] J. Robinson, A. Glean, and W. Moore, "How does news impact on the stock prices of green firms in emerging markets?" *Research in International Business and Finance*, vol. 45, pp. 446–453, 2018.

[3] G. Löffler, L. Norden, and A. Rieber, *Salient News and the Stock Market Impact of Tone in Rating Reports*, Ssrn, Rochester, NY, USA, pp. 1–41, 2016.

[4] M. N. Elagamy, C. Stanier, and B. Sharp, "Stock market random forest-text mining system mining critical indicators of stock market movements," in *Proceedings of the 2018 Second International Conference on Natural Language and Speech Processing (ICNLSP)*, pp. 1–8, IEEE, Algiers Algeria, June 2018.

[5] Y. Zhang, Y. Dang, H. Chen, M. Thurmond, and C. Larson, "Automatic online news monitoring and classification for syndromic surveillance," *Decision Support Systems*, vol. 47, no. 4, pp. 508–517, 2009.

[6] J. C. West, D. E. Clarke, F. F. Duffy et al., "Availability of mental health services prior to health care reform insurance expansions," *Psychiatric Services*, vol. 67, no. 9, pp. 983–989, 2016.

[7] M. Ali, S. Khalid, M. I. Rana, and F. Azhar, "A probabilistic framework for short text classification," in *Proceedings of the IEEE Eighth Annu Comput Commun Work Conf CCWC 2018*, pp. 742–747, Las Vegas, NV, USA, February 2018.

[8] C. Romero and S. Ventura, "Educational data mining: a survey from 1995 to 2005," *Expert Systems with Applications*, vol. 33, no. 1, pp. 135–146, 2007.

[9] Q. Ye, Z. Zhang, and R. Law, "Sentiment classification of online reviews to travel destinations by supervised machine learning approaches," *Expert Systems with Applications*, vol. 36, no. 3, pp. 6527–6535, 2009.

[10] D. Thorleuchter and D. Van Den Poel, "Predicting e-commerce company success by mining the text of its publicly-accessible website," *Expert Systems with Applications*, vol. 39, no. 17, Article ID 13026, 2012.

[11] G. Yang, M. A. Jan, A. U. Rehman, M. Babar, M. M. Aimal, and S. Verma, "Interoperability and data storage in Internet of multimedia things: investigating current trends, research challenges and future directions," *IEEE Access*, vol. 8, Article ID 124382, 2020.

[12] A. Gandomi and M. Haider, "Beyond the hype: big data concepts, methods, and analytics," *International Journal of Information Management*, vol. 35, no. 2, pp. 137–144, 2015.

[13] Y. Sun, A. K. C. Wong, and M. S. Kamel, "Classification of imbalanced data: a review," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 23, no. 4, pp. 687–719, Lyon, France, 2009.

[14] S. Manne and S. S. Fatima, "A novel approach for text categorization of unorganized data based with information extraction," *International Journal of Computational Science and Engineering*, vol. 3, pp. 2846–2854, 2011.

[15] B. S. Harish, D. S. Guru, and S. Manjunath, "Representation and classification of text documents: a brief review," *IJCA, Spec Issue Recent Trends Image Process Pattern Recognit*, pp. 110–119, 2010.

[16] Y. Liang, Y. Liu, C. K. Kwong, and W. B. Lee, "Learning the "Whys": discovering design rationale using text mining - an algorithm perspective," *Computer-Aided Design*, vol. 44, no. 10, pp. 916–930, 2012.

[17] B. Liu, X. Li, W. S. Lee, and P. S. Yu, "Text classification by labeling words," *Artificial Intelligence*, vol. 34, pp. 425–430, 2004.

[18] D. Y. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf, "Learning with local and global consistency," *Advances in Neural Information Processing Systems*, vol. 16, pp. 321–328, 2004.

[19] H. U. Rehman, R. Rafique, and M. C. M. Nasir, "Forecasting CO2 emissions from energy, manufacturing and transport sectors in Pakistan: statistical vs. Machine learning methods," *Mach Learn Methods*, vol. 28, p. 21, 2017.

[20] N. Kundu, G. Rani, V. S. Dhaka et al., "IoT and interpretable machine learning based framework for disease prediction in pearl millet," *Sensors*, vol. 21, no. 16, p. 5386, 2021.

[21] Y. Bengio, R. Ducharme, and P. Vincent, "A neural probabilistic language model," *Advances in Neural Information Processing Systems*, vol. 3, pp. 1–11, 2001.

[22] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[23] Y. Lecun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[24] A. Ittoo, L. M. Nguyen, and A. Van Den Bosch, "Text analytics in industry: challenges, desiderata and trends," *Computers in Industry*, vol. 78, pp. 96–107, 2016.

[25] F. Da Costa Albuquerque, M. A. Casanova, J. A. F. De Macedo, M. T. M. de Carvalho, and C. Renso, "A proactive application to monitor truck fleets," in *Proceedings of the 2013 IEEE Fourteenth International Conference on Mobile Data Management*, vol. 1, pp. 301–304, IEEE, Milan, Italy, July 2013.

[26] S. Schmidt, S. Schnitzer, and C. Rensing, "Text classification based filters for a domain-specific search engine," *Computers in Industry*, vol. 78, pp. 70–79, 2016.

[27] A. Hussain, S. Nazir, F. Khan et al., "A resource efficient hybrid proxy mobile IPv6 extension for next generation IoT networks," *IEEE Internet of Things Journal*, p. 1, 2021.

[28] M. Kumar, P. Mukherjee, K. Verma, S. Verma, and D. B. Rawat, "Improved deep convolutional neural network based malicious node detection and energy-efficient data transmission in wireless sensor networks," *IEEE Transactions on Network Science and Engineering*, p. 1, 2021.

[29] P. Rani, V. S. Kavita, S. Verma, and G. N. Nguyen, "Mitigation of black hole and gray hole attack using swarm inspired algorithm with artificial neural network," *IEEE Access*, vol. 8, Article ID 121755, 2020.

[30] B. Billal, A. Fonseca, and F. Sadat, "Named entity recognition and hashtag decomposition to improve the classification of tweets," in *Proceedings of the Second Workshop on Noisy User generated Text*, pp. 64–73, Osaka, Japan, December 2016.

[31] R. P. Schumaker and H. Chen, "A quantitative stock prediction system based on financial news," *Information Processing & Management*, vol. 45, no. 5, pp. 571–583, 2009.

[32] Y. Chen, Z. Li, L. Nie, X. Hu, and X. Wang, "Supervised bayesian network model for microblog topic classification," vol. 1, pp. 561–576, 2012.

[33] R. Xia, C. Zong, X. Hu, and E. Cambria, "Feature ensemble plus sample selection: domain adaptation for sentiment classification," in *Proceedings of the IJCAI Int Jt Conf Artif Intell 2015*, pp. 4229–4233, AAAI Press, California, CA, USA, January 2015.

[34] N. Majumder, S. Poria, A. Gelbukh, E. Cambria, and E. Cambria, "Deep learning-based document modeling for personality detection from text," *IEEE Intelligent Systems*, vol. 32, no. 2, pp. 74–79, 2017.

[35] E. Cambria, "Affective computing and sentiment analysis," *IEEE Intelligent Systems*, vol. 31, no. 2, pp. 102–107, 2016.

[36] L. Gaur, G. Singh, A. Solanki et al., "Disposition of Youth in Predicting Sustainable Development Goals Using the Neuro-Fuz," *Human-Centric Computing and Information Sciences*, vol. 11, pp. 2192–1962, 2021.

[37] R. H. W. Pinheiro, G. D. C. Cavalcanti, and I. R. Tsang, "Combining dissimilarity spaces for text categorization," *Information Sciences*, vol. 406-407, pp. 87–101, 2017.

[38] T. Sabbah, A. Selamat, M. H. Selamat et al., "Modified frequency-based term weighting schemes for text classification," *Applied Soft Computing*, vol. 58, pp. 193–206, 2017.

[39] V. Dogra, S. Verma, K. Verma, N. Ghosh, U. Le, and D.-N. au, "A comparative analysis of machine learning models for banking news extraction by multiclass classification with imbalanced datasets of financial news: challenges and solutions," *International Journal of Interactive Multimedia and Artificial Intelligence*, vol. 7, no. 3, p. 35, 2022.

[40] Y. Zhao, B. Qin, and T. Liu, "Creating a fine-grained corpus for Chinese sentiment analysis," *IEEE Intelligent Systems*, vol. 30, no. 1, pp. 36–43, 2015.

[41] Y. Zhang, R. Jin, and Z. H. Zhou, "Understanding bag-of-words model: a statistical framework," *International Journal of Machine Learning and Cybernetics*, vol. 1, no. 1-4, pp. 43–52, 2010.

[42] T. Joseph and Y. B. Lev Ratinov, "Word representations: a simple and general method for semi-supervised learning," in *Proceedings of the Forty Eighth Annu Meet Assoc Comput Linguist Assoc Comput Linguist*, pp. 384–394, AAAI Press, California, CA, USA, July 2010.

[43] R. Silipo and K. Melcher, *Text Encoding: A Review*, Kdnuggets, 2019, https://www.kdnuggets.com/2019/11/text-encoding-review.html.

[44] Z. Sivic and Zisserman, "Video Google: a text retrieval approach to object matching in videos," in *Proceedings of the Ninth IEEE International Conference on Computer Vision*, vol. 2, pp. 1470–1477, IEEE, New York, NY, USA, October 2003.

[45] Li-P. Jing, H.-K. Huang, and H.-Bo Shi, "Improved Feature Selection Approach TFIDF in Text Mining," in *Proceedings of the International Conference on Machine Learning and Cybernetics*, vol. 2, pp. 944–946, IEEE, New York, NY, USA, November 2002.

[46] E. Montañés, I. Díaz, J. Ranilla, E. Combarro, and J. Fernandez, "Scoring and selecting terms for text categorization," *IEEE Intelligent Systems*, vol. 20, no. 3, pp. 40–47, 2005.

[47] Z. Li, S. Verma, and M. Jin, "Power allocation in massive MIMO-HWSN based on the water-filling algorithm," *Wireless Communications and Mobile Computing*, vol. 2021, Article ID 8719066, 11 pages, 2021.

[48] Y. Bengio, R. Ducharme, V. Pascal, and J. Christen, "A neural probabilistic language model," *Journal of Machine Learning Research*, vol. 3, pp. 1137–1155, 2003.

[49] Y. Xiong, S. Chen, H. Qin et al., "Distributed representation and one-hot representation fusion with gated network for clinical semantic textual similarity," *BMC Medical Informatics and Decision Making*, vol. 20, no. S1, pp. 72–77, 2020.

[50] K. Kowsari, K. Jafari Meimandi, M. Heidarysafa, S. Mendu, L. Barnes, and D. Brown, "Text classification algorithms: a survey," *Information*, vol. 10, no. 4, pp. 150–168, 2019.

[51] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," in *Proceedings of the First Int Conf Learn Represent ICLR 2013 - Work Track Proc 1–12*, Scottsdale, Arizona, May 2013.

[52] B. Liu, "Text sentiment analysis based on CBOW model and deep learning in big data environment," *Journal of Ambient Intelligence and Humanized Computing*, vol. 11, no. 2, pp. 451–458, 2020.

[53] H. Schwenk, "Continuous space language models," *Computer Speech & Language*, vol. 21, no. 3, pp. 492–518, 2007.

[54] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in Neural Information Processing Systems 26*, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, Eds., pp. 3111–3119, Curran Associates Inc, Red Hook, NY, USA, 2013.

[55] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 135–146, 2017.

[56] J. Pennington and C. D. M. Richard Socher, "GloVe: global vectors for word representation," in *Proceedings of the 2014 Conf Empir Methods Nat Lang Process (EMNLP)*, pp. 1532–1543, Doha, Qatar, October 2014.

[57] C.-P. Wei and Y.-H. Lee, "Event detection from online news documents for supporting environmental scanning," *Decision Support Systems*, vol. 36, no. 4, pp. 385–401, 2004.

[58] Q. Zhu, X. Li, A. Conesa, and C. Pereira, "GRAM-CNN: a deep learning approach with local context for named entity recognition in biomedical text," *Bioinformatics*, vol. 34, no. 9, pp. 1547–1554, 2018.

[59] V. Prokhorov, M. T. Pilehvar, P. Lio, and N. Collier, "Unseen word representation by aligning heterogeneous lexical semantic spaces," in *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence*, Honolulu, HI, USA, February 2019.

[60] A. Makazhanov and D. Rafiei, "Predicting political preference of Twitter users," in *Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pp. 298–305, Association for Computing Machinery, New York NY USA, August2013.

[61] C. Bosco, V. Patti, and A. Bolioli, "Developing corpora for sentiment analysis: the case of irony and senti-TUT," *IEEE Intelligent Systems*, vol. 28, no. 2, pp. 55–63, 2013.

[62] M. E. Peters, M. Neumann, M. Iyyer et al., "Deep contextualized word representations," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, vol. 1, pp. 2227–2237, New Orleans, LA, USA, June 2018.

[63] S. Minaee, N. Kalchbrenner, E. Cambria, N. Nikzad, M. Chenaghlu, and J. Gao, "Deep learning--based text classification," *ACM Computing Surveys*, vol. 54, no. 3, pp. 1–40, 2022.

[64] X. Zhu, P. Sobihani, and H. Guo, "Long short-term memory over recursive structures," in *Proceedings of the 32nd International Conference on International Conference on Machine Learning*, pp. 1604–1612, Lille, France, July 2015.

[65] P. Zhou, Z. Qi, S. Zheng, J. Xu, and H. Bao, "Text classification improved by integrating bidirectional lstm with two-dimensional max pooling," 2016, https://arxiv.org/abs/1611.06639.

[66] W. Wei, J. Li, L. Cao, Y. Ou, and J. Chen, "Effective detection of sophisticated online banking fraud on extremely imbalanced data," *World Wide Web*, vol. 16, no. 4, pp. 449–475, 2013.

[67] S. I. Alfuraih, N. T. Sui, and D. McLeod, "Using Trusted Email to Prevent Credit Card Frauds in Multimedia Products," *World Wide Web*, vol. 5, pp. 245–256, 2004.

[68] D. Yan and S. Guo, "Leveraging contextual sentences for text classification by using a neural attention model," *Computational Intelligence and Neuroscience*, vol. 2019, Article ID 8320316, 11 pages, 2019.

[69] Z. Dai, Z. Yang, Y. Yang, C. Jaime, V. L. Quoc, and S. Ruslan, "Transformer-XL: Attentive language models beyond a fixed-length context," in *Proceedings of the ACL 2019 - Fifty Seventh Annu Meet Assoc Comput Linguist Proc Conf*, pp. 2978–2988arXiv:1901.02860, Florence, Italy, July 2020.

[70] Y. Hu, J. Ding, Z. Dou, and H. Chang, "Short-text classification detector: a bert-based mental approach," *Computational Intelligence and Neuroscience*, vol. 2022, Article ID 8660828, 11 pages, 2022.

[71] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," 2018, https://arxiv.org/abs/1810.04805.

[72] J. Howard and S. Ruder, "Universal Language Model Fine-tuning for Text Classification," 2018, https://arxiv.org/abs/1801.06146.

[73] Y. Sun, S. Wang, Y. Li et al., "Enhanced representation through knowledge integration," 2019, https://arxiv.org/abs/1904.09223.

[74] M. T. Varun Dogra Assvknj and, "Analyzing DistilBERT for sentiment classification of banking financial news," in *Intelligent Computing and Innovation on Data Science*, S.-L. Peng, S.-Y. Hsieh, S. Gopalakrishnan, and Balaganesh, Eds., p. 582, Springer Singapore, Singapore, 2021.

[75] V. Dogra, S. Verma, and A. Singh, "Banking news-events representation and classification with a novel hybrid model using DistilBERT and rule-based features," *Computer Science*, vol. 12, pp. 3039–3054, 2021.

[76] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter," pp. 2–6, 2019, https://arxiv.org/abs/1910.01108.

[77] Z. Ye, Q. Guo, Q. Gan, X. Qiu, and Z. Zhang, "BP-transformer: modelling long-range context via binary partitioning," 2019, https://arxiv.org/abs/1911.04070.

[78] I. Yamada and H. Shindo, "Aip R Neural Attentive Bag-Of-Entities Model for Text Classification," arXiv:1909.01259, 2019.

[79] T. B. Brown, B. Mann, N. Ryder et al., "Language models are few-shot learners," in *Proceedings of the Adv Neural Inf Process Syst 2020*, December 2020.

[80] B. Magnini, C. Strapparava, G. Pezzulo, and A. Gliozzo, "The role of domain information in Word Sense Disambiguation," *Natural Language Engineering*, vol. 8, no. 4, pp. 359–373, 2002.

[81] M. Bilenko, R. Mooney, W. Cohen, P. Ravikumar, and S. Fienberg, "Adaptive name matching in information integration," *IEEE Intelligent Systems*, vol. 18, no. 5, pp. 16–23, 2003.

[82] S. J. Huang, W. Gao, and Z. H. Zhou, "Fast multi-instance multi-label learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 11, pp. 2614–2627, 2019.

[83] M.-L. Zhang and Z.-H. Zhou, "ML-KNN: a lazy learning approach to multi-label learning," *Pattern Recognition*, vol. 40, no. 7, pp. 2038–2048, 2007.

[84] J. Jun Yan, B. Benyu Zhang, N. Ning Liu et al., "Effective and efficient dimensionality reduction for large-scale and streaming data preprocessing," *IEEE Transactions on Knowledge and Data Engineering*, vol. 18, no. 3, pp. 320–333, 2006.

[85] X. Xu, T. Liang, J. Zhu, D. Sun, and T. au, "Review of classical dimensionality reduction and sample selection methods for large-scale data processing," *Neurocomputing*, vol. 328, pp. 5–15, 2019.

[86] N. Armanfard, J. P. Reilly, and M. Komeili, "Local feature selection for data classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 6, pp. 1217–1227, 2016.

[87] S. Pölsterl, S. Conjeti, N. Navab, and A. Katouzian, "Survival analysis for high-dimensional, heterogeneous medical data: exploring feature extraction as an alternative to feature selection," *Artificial Intelligence in Medicine*, vol. 72, pp. 1–11, 2016.

[88] D. Mladenić and M. Grobelnik, "Feature selection on hierarchy of web documents," *Decision Support Systems*, vol. 35, no. 1, pp. 45–87, 2003.

[89] A.-C. Haury, P. Gestraud, and J.-P. Vert, "The influence of feature selection methods on accuracy, stability and interpretability of molecular signatures," *PLoS One*, vol. 6, no. 12, Article ID e28210, 2011.

[90] J. Yang, Y. Liu, X. Zhu, Z. Zhang, Z. Liu, and X. Zhang, "A new feature selection based on comprehensive measurement both in inter-category and intra-category for text categorization," *Information Processing & Management*, vol. 48, no. 4, pp. 741–754, 2012.

[91] I. Inza, P. Larrañaga, R. Blanco, and A. J. Cerrolaza, "Filter versus wrapper gene selection approaches in DNA microarray domains," *Artificial Intelligence in Medicine*, vol. 31, no. 2, pp. 91–103, 2004.

[92] A. K. Uysal and S. Gunal, "A novel probabilistic feature selection method for text classification," *Knowledge-Based Systems*, vol. 36, pp. 226–235, 2012.

[93] D. Roobaert, G. Karakoulas, and N. V. Chawla, "Information gain, correlation and support vector machines," *Feature Extraction*, vol. 470, pp. 463–470, 2008.

[94] S. Lei, "A feature selection method based on information gain and genetic algorithm," in *Proceedings of the 2012 International Conference on Computer Science and Electronics*

Engineering, vol. 2, pp. 355–358, Hangzhou, China, March 2012.

[95] X. Jin, A. Xu, R. Bie, and P. Guo, "Machine learning techniques and chi-square feature selection for cancer classification using SAGE gene expression profiles," *Lecture Notes in Computer Science*, Springer, Berlin, Germany, pp. 106–115, 2006.

[96] Y.-T. Chen and M. C. Chen, "Using chi-square statistics to measure similarities for text categorization," *Expert Systems with Applications*, vol. 38, no. 4, pp. 3085–3090, 2011.

[97] Q. Gu, Z. Li, and J. Han, "Generalized Fisher score for feature selection," *A brief review of Fisher score. Ratio*, AUAI Press, Arlington, VA, USA, 2010.

[98] J. Weston, S. Mukherjee, O. Chapelle, M. Pontil, T. Poggio, and V. Vapnik, "Feature selection for SVMs," *Advances in Neural Information Processing Systems*, vol. 13, pp. 668–674, 2000.

[99] E. Youn, L. Koenig, M. K. Jeong, and S. H. Baek, "Support vector-based feature selection using Fisher's linear discriminant and Support Vector Machine," *Expert Systems with Applications*, vol. 37, no. 9, pp. 6148–6156, 2010.

[100] Y. Wang and X.-J. Wang, "A new approach to feature selection in text classification," in *Proceedings of the 2005 International Conference on Machine Learning and Cybernetics*, vol. 6, pp. 3814–3819, Guangzhou, China, August 2005.

[101] A. Genkin, D. D. Lewis, and D. Madigan, "Large-scale bayesian logistic regression for text categorization," *Technometrics*, vol. 49, no. 3, pp. 291–304, 2007.

[102] H. Peng, F. Long, and C. Ding, "Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 8, pp. 1226–1238, 2005.

[103] M. Labani, P. Moradi, F. Ahmadizar, and M. Jalili, "A novel multivariate filter method for feature selection in text classification problems," *Engineering Applications of Artificial Intelligence*, vol. 70, pp. 25–37, 2018.

[104] K. A. Norman, S. M. Polyn, G. J. Detre, and J. V. Haxby, "Beyond mind-reading: multi-voxel pattern analysis of fMRI data," *Trends in Cognitive Sciences*, vol. 10, no. 9, pp. 424–430, 2006.

[105] L. Wang, Y. Lei, Y. Zeng, L. Tong, and B. Yan, "Principal feature analysis: a novel voxel selection method for fMRI data," *Computational and Mathematical Methods in Medicine*, vol. 2013, Article ID 645921, 7 pages, 2013.

[106] P. M. Granitto, C. Furlanello, F. Biasioli, and F. Gasperi, "Recursive feature elimination with random forest for PTR-MS analysis of agroindustrial products," *Chemometrics and Intelligent Laboratory Systems*, vol. 83, no. 2, pp. 83–90, 2006.

[107] K. Z. Mao, "Fast orthogonal forward selection algorithm for feature subset selection," *IEEE Transactions on Neural Networks*, vol. 13, no. 5, pp. 1218–1224, 2002.

[108] H. Chen, W. Jiang, C. Li, and R. Li, "A heuristic feature selection approach for text categorization by using chaos optimization and genetic algorithm," *Mathematical Problems in Engineering*, vol. 2013, Article ID 524017, 6 pages, 2013.

[109] R. Leardi, R. Boggia, and M. Terrile, "Genetic algorithms as a strategy for feature selection," *Journal of Chemometrics*, vol. 6, no. 5, pp. 267–281, 1992.

[110] I. Guyon, H.-M. Bitter, Z. Ahmed, M. Brown, and J. Heller, "Multivariate non-linear feature selection with kernel multiplicative updates and gram-schmidt relief," in *Proceedings of the BISC Flint-CIBI 2003 Workshop*, pp. 1–11, Berkeley, CA, 2003.

[111] R. J. Urbanowicz, M. Meeker, W. L Cava, R. S. Olson, and J. H. Moore, "Relief-based feature selection: introduction and review," *Journal of Biomedical Informatics*, vol. 85, pp. 189–203, 2018.

[112] R. J. Urbanowicz, R. S. Olson, P. Schmitt, M. Meeker, and J. H. Moore, "Benchmarking relief-based feature selection methods for bioinformatics data mining," *Journal of Biomedical Informatics*, vol. 85, pp. 168–188, 2018.

[113] N. Mimouni and T. Y. Yeung, "Comparing Performance of Text Pre-processing Methods for Predicting a Binary Position by LASSO Experiment with Textual Data of European Union Public Consultation," in *Proceedings of the Workshop on Women in Data Science 2018*, pp. 18–21, 2018.

[114] B. J. Marafino, W. J. Boscardin, and R. A Dudley, "Efficient and sparse feature selection for biomedical text classification via the elastic net: application to ICU risk stratification from nursing notes," *Journal of Biomedical Informatics*, vol. 54, pp. 114–120, 2015.

[115] P. Taylor, A. E. Hoerl, and R. W. Kennard, *Technometrics Ridge Regression: Biased Estimation for Nonorthogonal Problems Ridge Regression : Biased Estimation Nonorthogonal Problems*, pp. 37–41, 2012.

[116] H. Zou and T. Hastie, "Regularization and variable selection via the elastic net," *Journal of the Royal Statistical Society: Series B*, vol. 67, no. 2, pp. 301–320, 2005.

[117] F. Sebastiani, "Machine learning in automated text categorization," *ACM Computing Surveys*, vol. 34, no. 1, pp. 1–47, 2002.

[118] J. C. Gomez and M. F. Moens, "PCA document reconstruction for email classification," *Computational Statistics & Data Analysis*, vol. 56, no. 3, pp. 741–751, 2012.

[119] D. M. Blei, A. Y. Ng, and M. I. Jordan, *Latent Dirichlet Allocation*, vol. 3, pp. 993–1022, 2003.

[120] C. Ordun, S. Purushotham, and E. Raff, "Exploratory Analysis of Covid-19 Tweets Using Topic Modeling, UMAP, and DiGraphs," 2020, https://arxiv.org/abs/2005.03082.

[121] L. Mcinnes, J. Healy, and J. Melville, "UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction," 2020, https://arxiv.org/abs/1802.03426.

[122] H. U. Rehman, M. Shafiq, S. Baig, and U. Manzoor, "Analyzing the epidemiological outbreak of COVID-19," *A Vis Explor data Anal approach J Med Virol*, vol. 92, no. 6, 2021.

[123] H. Cheng and R. Yu, *Text Classification Model Enhanced by Unlabeled Data for LaTeX Formula*, 2021.

[124] J. Chen, H. Huang, S. Tian, and Y. Qu, "Feature selection for text classification with Naïve Bayes," *Expert Systems with Applications*, vol. 36, no. 3, pp. 5432–5435, 2009.

[125] B. Trstenjak, S. Mikac, and D. Donko, "KNN with TF-IDF based framework for text categorization," *Procedia Engineering*, vol. 69, pp. 1356–1364, 2014.

[126] C. Lee and G. G. Lee, "Information gain and divergence-based feature selection for machine learning-based text categorization," *Information Processing & Management*, vol. 42, no. 1, pp. 155–165, 2006.

[127] L. Khreisat, "A machine learning approach for Arabic text classification using N-gram frequency statistics," *Journal of Informetrics*, vol. 3, no. 1, pp. 72–77, 2009.

[128] J. Liu, T. Jin, K. Pan, Y. Yang, Y. Wu, and X. Wang, "An improved KNN text classification algorithm based on Simhash," in *Proceedings of the 2017 IEEE 16th Int Conf Cogn Informatics Cogn Comput ICCI∗CC*, pp. 92–95, Oxford UK, July 2017.

[129] C. Apté and S. Weiss, "Data Mining with Decision Trees and Decision Rules," *Future Generation Computer Systems*, vol. 13, 1997.

[130] V. N. Phu, V. T. N. Tran, V. T. N. Chau, N. D. Duy, and K. L. D. au, "A decision tree using ID3 algorithm for English semantic analysis," *International Journal of Speech Technology*, vol. 20, no. 3, pp. 593–613, 2017.

[131] C. N. Mahender, *TEXT CLASSIFICATION AND CLASSIFIERS*, vol. 3, pp. 85–99, 2012.

[132] Y. Yang, "An evaluation of statistical approaches to text categorization," *Inf Retr Boston*, vol. 1, pp. 69–90, 1997.

[133] B. P. Roe, H. J. Yang, J. Zhu, Y. Liu, I. Stancu, and G. McGregor, "Boosted decision trees as an alternative to artificial neural networks for particle identification," *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, vol. 543, no. 2-3, pp. 577–584, 2005.

[134] J. Ye, J. H. Chow, J. Chen, and Z. Zheng, "Stochastic gradient boosted distributed decision trees," in *Proceedings of the Eighteenth ACM conference on Information and knowledge management - CIKM '09*, pp. 2061–2064, Hong Kong China, November 2009.

[135] E. Wiener, J. Pedersen, and A. Weigend, "A neural network approach to topic spotting," in *Proceedings of the Annu Symp Doc Anal Inf Retr*, pp. 317–332, 1995.

[136] R. Johnson and T. Zhang, "Effective Use of Word Order for Text Categorization with Convolutional Neural Networks," pp. 103–112, 2014, https://arxiv.org/abs/1412.1058.

[137] Y. Zhang, Z. Zhang, D. Miao, and J. Wang, "Three-way enhanced convolutional neural networks for sentence-level sentiment classification," *Information Sciences*, vol. 477, pp. 55–64, 2019.

[138] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, "A convolutional neural network for modelling sentences," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, vol. 1, pp. 655–665, Baltimore, Maryland, June 2014.

[139] J. L. Elman, "Finding structure in time," *Cognitive Science*, vol. 14, no. 2, pp. 179–211, 1990.

[140] V. Makarenkov, I. Guy, N. Hazon, T. Meisels, B. Shapira, and L. Rokach, "Implicit dimension identification in user-generated text with LSTM networks," *Information Processing & Management*, vol. 56, no. 5, pp. 1880–1893, 2019.

[141] L. Sarmento, P. Carvalho, M. J. Silva, and E. de Oliveira, "Automatic creation of a reference corpus for political opinion mining in user-generated content," *Attitude*, vol. 29, 2009.

[142] N. Ruchansky, S. Seo, and Y. Liu, "CSI: A Hybrid Deep Model for Fake News Detection," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, Singapore, November 2017.

[143] S. Dong and C. Liu, "Sentiment classification for financial texts based on deep learning," *Computational Intelligence and Neuroscience*, vol. 2021, Article ID 9524705, 9 pages, 2021.

[144] Y. Zhu, W. Zheng, and H. Tang, "Interactive dual attention network for text sentiment classification," *Computational Intelligence and Neuroscience*, vol. 2020, Article ID 8858717, 11 pages, 2020.

[145] J. Chung, "Gated Recurrent Neural Networks on Sequence Modeling," pp. 1–9, 2014, https://arxiv.org/abs/1412.3555.

[146] D. Tang, B. Qin, and T. Liu, "Document modeling with gated recurrent neural network for sentiment classification," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 1422–1432, Lisbon, Portugal, September 2015.

[147] M. Zulqarnain, R. Ghazali, M. G. Ghouse, and M. F. Mushtaq, "Efficient processing of GRU based on word embedding for text classification," *JOIV: International Journal on Informatics Visualization*, vol. 3, no. 4, 2019.

[148] O. Kuchaiev and B. Ginsburg, "Factorization tricks for LSTM networks," in *Proceedings of the Fifth Int Conf Learn Represent ICLR 2017 - Work Track Proc 1–6*, Toulon, France, April 2019.

[149] N. Shazeer, A. Mirhoseini, K. Maziarz et al., "Outrageously large neural networks: the sparsely-gated mixture-of-experts layer," 2017, https://arxiv.org/abs/1701.06538.

[150] B. C. W. Jain, "Attention is not Explanation," 2019, https://arxiv.org/abs/1902.10186.

[151] S. Serrano and N. A. Smith, "Is attention interpretable?" in *Proceedings of the ACL 2019 - 57th Annu Meet Assoc Comput Linguist Proc Conf*, pp. 2931–2951, Florence, Italy, July 2020.

[152] S. Vashishth, S. Upadhyay, G. S. Tomar, and M. Faruqui, "Attention Interpretability Across NLP Tasks," pp. 1–10, 2019, https://arxiv.org/abs/1909.11218.

[153] T. Munkhdalai and H. Yu, "Neural semantic encoders," in *Proceedings of the 15th Conference of the European Chapter of the*, Valencia, Spain, April 2017.

[154] T. Commissariat, "Ask me anything," *Physics World*, vol. 33, no. 3, pp. 53–58, 2020.

[155] A. Vaswani, N. Shazeer, N. Parmar et al., "Attention is all you need," in *Proceedings of the Adv Neural Inf Process Syst 2017*, pp. 5999–6009, Long Beach, CA, USA, December 2017.

[156] C. Sun, X. Qiu, Y. Xu, and X. Huang, "How to fine-tune BERT for text classification?" *Lect Notes Comput Sci (Including Subser Lect Notes Artif Intell Lect Notes Bioinformatics)*, Springer, Berlin, Germany, pp. 194–206, 2019.

[157] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "ALBERT: A Lite BERT for Self-supervised Learning of Language Representations," pp. 1–17, 2019, https://arxiv.org/abs/1909.11942.

[158] M. Joshi, D. Chen, Y. Liu, D. S. Weld, L. Zettlemoyer, and O. Levy, "SpanBERT: Improving Pre-training by Representing and Predicting Spans," 2019, https://arxiv.org/abs/1907.10529.

[159] C. Holzhey, F. Larsen, and F. Wilczek, "Geometric and renormalized entropy in conformal field theory," *Nuclear Physics B*, vol. 424, no. 3, pp. 443–467, 1994.

[160] K. Nigam, J. Lafferty, and A. Mccallum, "Using maximum entropy for text classification," *Computet Science*, vol. 80, 1999.

[161] J. Atkinson-Abutridy, C. Mellish, and S. Aitken, "Combining information extraction with genetic algorithms for text mining," *IEEE Intelligent Systems*, vol. 19, no. 3, pp. 22–30, 2004.

[162] A. Bhardwaj, Y. Narayan, Vanraj, Pawan, and M. Dutta, "Sentiment analysis for Indian stock market prediction using sensex and nifty," *Procedia Computer Science*, vol. 70, pp. 85–91, 2015.

[163] D. Kim, D. Seo, S. Cho, and P. Kang, "Multi-co-training for document classification using various document representations: TF-IDF, LDA, and Doc2Vec," *Information Sciences*, vol. 477, pp. 15–29, 2019.

[164] M. H. Steinberg, "Clinical trials in sickle cell disease: adopting the combination chemotherapy paradigm," *American Journal of Hematology*, vol. 83, no. 1, pp. 1–3, 2008.

[165] Z. Xu and J. Sun, "Model-driven deep-learning," *National Science Review*, vol. 5, no. 1, pp. 22–24, 2018.

[166] Y. Chen, B. Xiao, Z. Lin, C. Dai, Z. Li, and L. Yan, "Multi-label text classification with deep neural networks," in *Proceedings of the 2018 Sixth IEEE Int Conf Netw Infrastruct Digit Content*, pp. 409–413, IEEE, Guiyang China, November 2018.

[167] R. B. Pereira, A. Plastino, B. Zadrozny, and L. H. C. Merschmann, "Categorizing feature selection methods for multi-label classification," *Artificial Intelligence Review*, vol. 49, no. 1, pp. 57–78, 2018.

[168] R. A. Stein, P. A. Jaques, and J. F. Valiati, "An analysis of hierarchical text classification using word embeddings," *Information Sciences*, vol. 471, pp. 216–232, 2019.

[169] R. A. Sinoara, C. V. Sundermann, R. M. Marcacini, M. A. Domingues, and S. R. Rezende, "Named Entities as Privileged Information for Hierarchical Text Clustering," in *Proceedings of the Eighteenth International Database Engineering & Applications Symposium, Association for Computing Machinery*, pp. 57–66, New York, NY, USA, July 2014.