Journal of Information Systems Engineering and Management
2025, 10(34s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Sentemonet: A Comprehensive Framework for Multimodal Sentiment Analysis from Text and Emotions

[1*]Mr. Bikshapathy Peruka, [2]Dr. K. Shahu Chatrapati

[1]Research Scholar, Department of Computer Science and Engineering, Jawaharlal Nehru Technological University, Kukatpally, Hyderabad, India.*

[2]Professor, Department of Computer Science and Engineering, Jawaharlal Nehru Technological University, Kukatpally, Hyderabad, India.

[1]Corresponding Author Email: peruka.bpathy@gmail.com*

ARTICLE INFO

Received: 18 Dec 2024

Revised: 10 Feb 2025

Accepted: 28 Feb 2025

ABSTRACT

Sentiment analysis, a crucial aspect of Natural Language Processing (NLP), plays a pivotal role in understanding public opinion, customer feedback, and user sentiments in various domains. In this study, we present a comprehensive approach to sentiment analysis that incorporates both textual and emoji data, leveraging diverse datasets from sources such as social media, customer reviews, and surveys. Our methodology consists of several key steps, including data collection, pre-processing, feature extraction, feature fusion, and feature selection. For data pre-processing, we apply techniques such as tokenization, lowercasing, stop word removal, and stemming to ensure uniform and meaningful text representation. We also extract emojis from text using regular expressions and convert them into textual representations, facilitating unified processing. Text data is transformed into feature vectors using Term frequency_ Inverse Term frequency (TF-IDF) weighting-based Bag of Words (TF-IDF_BoW) with word frequencies, pre-trained Word2Vec word embeddings, and n-grams to capture local word patterns. Emoji data, on the other hand, is processed using pre-trained emoji embeddings (Emoji2Vec) and emoji frequency counting. The Feature Fusion stage involves the combination of text and emoji features into a single feature vector, with a weighted concatenation approach. In the Feature Selection phase, introduced a Self-Improved Siberian Tiger Optimization (SI-STO), a novel feature selection technique, to identify the most relevant features for sentiment prediction. The sentiment classification model SentEmoNet combines LSTM with an Attention Mechanism and a CNN for capturing text patterns and extracting local features from text and emoji embeddings. We use standard sentiment analysis metrics to assess performance in the model evaluation phase. The proposed model has achieved an accuracy of 97.87%, which is better than the existing ones.

Keywords: Sentiment analysis, Natural Language Processing, Text and Emojis, Tokenization, Word embeddings, and Self-Improved Siberian Tiger Optimization.

INTRODUCTION

Sentiment categorization is the technique of reading a text to predict a person's attitude towards a situation or point of view. Based on text polarity, the sentiment is often assessed. Sentiment classifiers often classify as neutral, negative, or positive. With a great deal of research done, sentiment extraction forms the foundation of sentiment classification [1,2,3]. Sentiment analysis, which has grown significantly in recent years in tandem with the global expansion of textual data, is the next important stage. People today use electronic means to exchange opinions on a wide range of subjects, such as political commentary, book or movie reviews, and online product evaluations. It follows that assessing different points of view is crucial to deciphering people's motivations [4,5,6]. Generally speaking, sentiment encompasses two different types of thinking, either favourable or unfavourable, on many platforms where the consensus is valuable.

Sentiment analysis is a branch of natural language processing that looks for emotional content in written language [7]. For robots, it is a difficult assignment, and occasionally it may be difficult for humans as well. For someone lacking an innate comprehension of the language, the English language's structure, in conjunction with sarcastic tones and unclear syntax, may frequently be deceptive. Nonetheless, curiosity about emotion has increased in tandem with artificial intelligence, especially in light of the rise in emotional communication over the Internet [8,9]. More people than ever before are communicating about sensitive personal issues using both private and public message platforms because to the rise of social media in the last ten years. Sentiment analysis has drawn more attention from social media owners in an effort to pinpoint potentially hazardous sentiments on these sites, such abusive or threatening language [10,11,12]. Although reporting an abusive user might be a long and imprecise process, this kind of behaviour is traditionally handled by users. Improved control and more fruitful dialogue are possible with an automated system.

Deep learning (DL) methodologies, driven by increased processing power and the availability of extensive web data, have gained prominence. Within these DL techniques, word embedding has proven instrumental. Word embedding techniques, such as Word2Vec [13], GloVe [14], or FastText, represent words as continuous vectors, enhancing neural network efficiency and model performance in NLP tasks. By capturing semantic relationships between words, word embedding enables a deeper and more nuanced understanding of language, benefiting applications like sentiment analysis, text classification, and machine translation, among others.

This research examined an extensive sentiment dataset, including reviews from various domains like the internet and social networking sites, including politics, sports, products, and entertainment. The paper presents a robust framework for addressing the challenges in identifying and distinguishing individual emotions in online texts, known for their diversity and variability in syntax and semantics. To enhance the accuracy of sentiment analysis, the framework incorporates emojis, which serve as a rich source of individual sentiment expression in online conversations. The study assesses the impact of supplementing emojis as additional features through various popular sentiment analysis algorithms, including rule-based and classification methods. Emojis are transformed into corresponding sentiment words for feature construction, and the framework considers different functions of emojis in texts. Deep learning models, referred to as SentEmoNet, is developed for sentiment classification. These models incorporate both text and emoji feature and utilize LSTM with Attention Mechanism and CNN. LSTM captures sequential patterns in text data, while CNN focuses on extracting local features from text and emoji embeddings.

The remaining part of the paper is structured as follows, the detailed literature review is given in section 2, the proposed methodology for sentiment analysis is explained in section 3, the results obtained for the proposed model is compared with the existing techniques in section 4 and the detailed conclusion is given in section 5.

LITERATURE REVIEW

In this section, the recent papers related to the sentiment analysis is discussed with their drawbacks.

In 2021, Dashtipour, et. al., [16] have introduced a novel multimodal dataset for Persian language, which was derived from YouTube videos and includes utterances together with their sentiment polarity. Furthermore, a unique multimodal sentiment analysis framework in Persian was suggested, which enables the contextual combination of audio, visual, and textual elements. The amalgamation of Persian audio, visual, and textual characteristics surpasses all other unimodal classifiers, including text-, audio-, and video-based sentiment analysis models, as per our experimental findings.

In 2020, Xu, et. al., [17] have introduced a framework for Multimodal Data Augmentation (MDA) to improve sentiment/emotion classification task performance. That provide a MAD system to improve the accuracy on multimodal image-text classification tasks, addressing the shortage issue of labelled multimodal data. The multimodal synthetic dataset was created using that approach by learning a cross-modality matching network to choose image-text pairings from pre-existing unimodal datasets. Classifier performance was then improved using this dataset.

In 2021, Yan, et. al., [18] have create a sentence-level feature representation, the text was first separated into numerous sentences. Next, n-gram information of various granularities was extracted from each phrase using a CNN. Subsequently, the sentences undergo sequential integration using the bidirectional gated recurrent unit (BiGRU) in order to retrieve the text's contextual lexical characteristics. Lastly, the CNN-BiGRU model gains an attention mechanism, and by computing the attention score, the model was given various learning weights. To achieve sentiment classification, the softmax classifier receives the top-down text characteristics of word-sentence-text.

In 2020, Tang, et. al., [19] have recommended an optimising BERT for Multi-Label Analysis of Sentiment in Imbalanced Code-Switching Text. The research was a multi-label sentiment evaluation model for imbalanced

Journal of Information Systems Engineering and Management
2025, 10(34s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/
Research Article

code-switching text, based on BERT architecture. In order to test various pre-trained models and parameters, that employ BERT for this job. Which employ a multiple under sampling strategy and a translation-based data augmentation technique to provide balanced samples while taking the imbalanced dataset's features into account. Multiple BERT classifiers independently train these data, and the outputs of these classifiers were combined via ensemble learning.

In 2020, Lin, et. al., [20] have performed an evaluation of sentiment with contrast enhanced deep neural network. It presents a Comparative Enhanced Bi-LSTM with Multi-Head Attention (CE-B-MHA), which reformulates the classification job as a comparison issue to improve text sentiment analysis performance. In actuality, classifying via a comparison process is more efficient than using intricate calculations. In that model, relevant information was retrieved from various dimensions and depiction subspaces using Multi-Head Attention while bidirectional LSTM was employed for initial feature extraction.

In 2022, Loureiro, et. al., [21] have utilized data from Twitter conversations, that suggested an economic analysis that describes perceptions and responses to the wildfires that are occurring in Spain and Portugal. That examine the textual data using methods from natural language processing. That produce a hedonometer estimate of the relationship between attitudes towards wildfires and exposure, as determined by air quality and Euclidean distance from the catastrophic event. The research reveals that being in close proximity to wildfires dramatically lowers the stated emotion score and raises the expressions of fear and political unrest (protest).

In 2022, Tesfagergish, et. al., [22] have introduced Zero-Shot detection of emotions for semi-supervised evaluation of feelings by employing sentence transformers and ensemble learning. The research has evaluated three datasets to handle the binary (positive, negative) and three-class (positive, negative, neutral) sentiment assessment problem for the English language. The approach suggested a quite distinct from the typical methods used to accomplish these jobs. It designs the sentiment evaluation problem as a two-stage classification problem, whereby emotions were determined in the first stage and sentiments were determined in the second phase based on the feelings.

In 2021, Lu and Zhang [23] Improved AT-BiGRU Model-Based Sentiment Analysis Method for Network Text. This research presents an enhanced AT-BiGRU model-based approach for Internet text sentiment analysis. Before text preprocessing, the textblob package is first loaded to fix spelling mistakes. Second, data is extracted using a two-way gated recurrent network, the attention mechanism is utilised to draw attention to the important information in the word vector, and pad_sequences are used to fill in the input layer with a fixed length. Ultimately, an enhanced BiGRU that can adjust to the recursive network topology is built by transforming the GNU memory unit.

In 2022, Bharti, et. al., [24] Deep Learning-Based Text-Based Emotion Recognition Technique. a text-based model for emotion recognition was proposed in this work. Deep learning and machine learning techniques are combined in the suggested model. Three different datasets—WASSA, Emotion-Stimulus, and ISEAR—are combined in this suggested hybrid technique. Numerous benefits come with the suggested approach, including its ability to operate on multi-text phrases, tweets, dialogues, keywords, and readily identifiable vocabulary terms of emotions.

In 2021, Sharaf Al-deen, et. al., [25] An Improved Model for Analysing Textual Emotion Based on a Deep Neural Network Employing Multi-Head Attention Process. We initially create an enhanced deep neural network for the DNN–MHAT model by fusing recurrent bidirectional long short-term memory units (Bi-LSTM) with a CNN in order to gather the local features of position constants and retain the text's true context. Second, to add a distinct focus to the information generated from the hidden layers of Bi-LSTM, we provide a multi-head attention technique to identify words in the text that are highly connected to long space and encoding dependencies.

1.1. Problem Statement

In recent years, sentiment analysis and emotion recognition have made significant progress with new techniques, models, and datasets. However, challenges persist. One critical challenge is expanding multimodal sentiment analysis to languages with fewer resources, such as Persian. This work is pioneering but highlights the need for further research in underrepresented languages. Data augmentation remains relevant, especially for imbalanced datasets and code-switching text, requiring efficient augmentation strategies. Sentence-level feature representation, like the CNN-BiGRU model, is crucial for capturing contextual lexical characteristics in sentiment analysis. Further investigation is needed for optimal architectures and feature extraction methods. Comparative models, such as CE-B-MHA, need research to assess scalability and generalizability across different languages and domains.

Emotion analysis is vital for understanding real-world events' impact on sentiments, especially on platforms like Twitter. Research should explore the relationship between emotions, sentiment, and external events in various contexts. Zero-shot emotion detection approaches need adaptability testing for different languages and cultural contexts. Enhanced sentiment analysis models for internet text, like the AT-BiGRU model, are essential for handling dynamic and noisy online content. Research is required to improve their accuracy and efficiency. Hybrid models combining deep learning and machine learning for emotion recognition show promise but need assessment for various text types. The integration of attention mechanisms, like the DNN-MHAT model, enhances sentiment analysis by understanding context and word dependencies. Ongoing research is necessary to explore optimal application in diverse languages and domains. Addressing these challenges will lead to more accurate and robust sentiment analysis systems with broader applications.

PROPOSED METHODOLOGY

The proposed approach for sentiment analysis using text and emoji data involves a comprehensive process that begins with data collection and pre-processing. A diverse dataset of text data with associated sentiment labels is gathered from sources like social media, customer reviews, or surveys, ensuring that both text and emoji data are included. Text data is pre-processed through tokenization, lowercasing, stop word removal, and stemming, while emoji data is extracted from the text using regular expressions (Regex) and converted into textual representations. Feature extraction is a crucial step, with various techniques employed for both text and emoji data. Text features are obtained through TF-IDF_BoW, word embeddings (Word2Vec), and N-grams. Emoji features are extracted using pre-trained emoji embeddings (Emoji2Vec) and emoji frequency counting. Feature fusion combines the extracted text and emoji features into a unified feature vector using weight-based concatenation. Feature selection is implemented to identify the most relevant features, enhancing the efficiency of sentiment prediction with techniques like SI-STO. The Sentiment Classification Model, SentEmoNet, is introduced to perform sentiment analysis. SentEmoNet combines LSTM with Attention Mechanism and CNN models to leverage both sequential patterns in text data and local features from text and emoji embeddings. The LSTM with Attention Mechanism captures word importance and sequential context, while the CNN model focuses on identifying specific patterns and representations is shown below the figure 1.

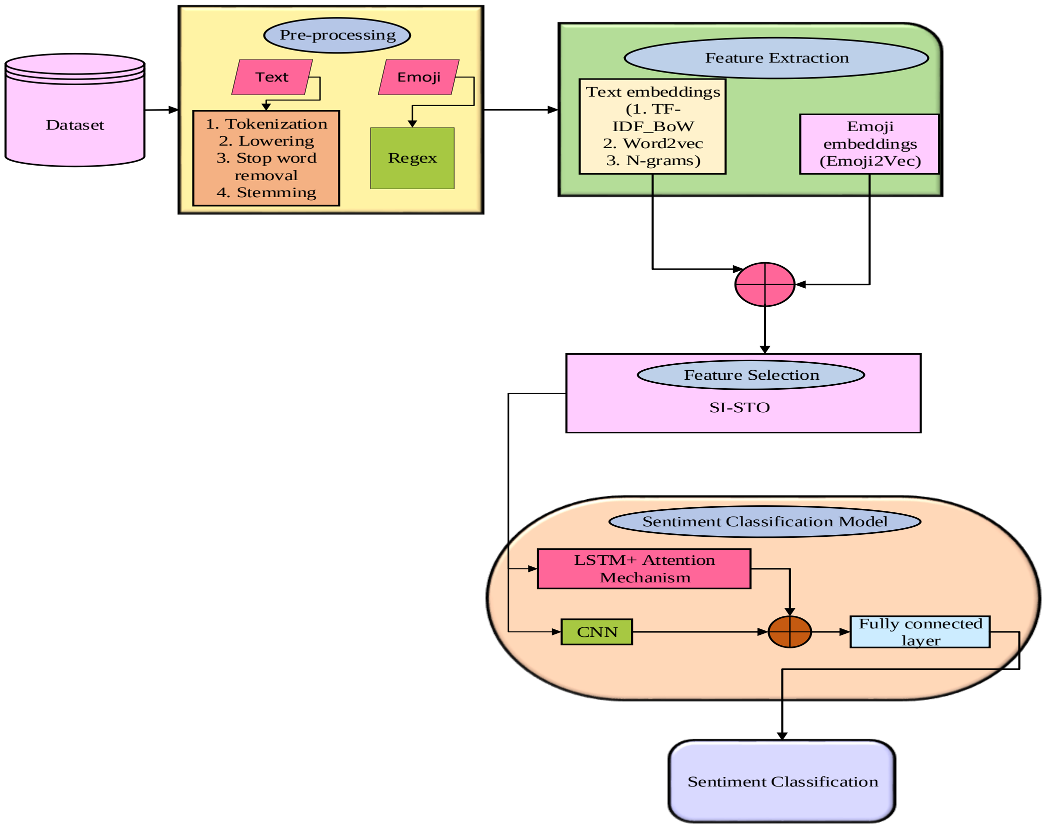

Figure 1: Block diagram of the proposed sentiment analysis model

1.2. Data Collection

1.2.1. Dataset 1 (Emoji Sentiment Analysis)

The "Emoji Sentiment Analysis" dataset offers a comprehensive exploration of the emotional connotations associated with a diverse array of emojis. This dataset comprises several key columns, each serving a unique role in unravelling the sentiment behind these expressive symbols. The "emoji" column lists emojis represented by their Unicode code points, facilitating the identification of individual symbols. Sentiment scores, denoted by "s.writer" and "s.reader," provide insights into the emotional responses of the author (writer) and the reader to each emoji, ranging from -1 (indicating negativity) to 1 (indicating positivity). The "count" column quantifies the frequency of emoji usage, shedding light on the prevalence of specific symbols. "sd" remains enigmatic, potentially holding additional sentiment-related information. Human-readable emoji descriptions in the "description" column aid in understanding the meaning of each symbol. Lastly, the "diff" column reveals the disparity between writer and reader sentiment scores for each emoji. While this dataset is geared towards sentiment analysis of emojis, further contextual information is essential for a more comprehensive interpretation of its contents.

1.2.2. Dataset 2 (Twitter Sentimental Analysis)

It has the compilation of 1,600,000 tweets accessed via the Twitter API, each thoughtfully annotated with sentiment labels that categorize them into negative (0), neutral (2), or positive (4) sentiments. This dataset is specifically tailored for sentiment analysis, enabling the exploration of sentiment in textual data. It encompasses six crucial fields: "target" for sentiment polarity, "ids" for unique tweet identification, "date" for timestamp information, "flag" indicating associated queries, "user" for Twitter user attribution, and "text," which contains the tweet's content, serving as the primary source of sentiment data. What sets this dataset apart is its innovative approach to sentiment labeling, which leveraged emoticons as sentiment indicators. For instance, tweets with positive emoticons like ":)" were considered positive, and those with negative emoticons like ":(" were labeled as negative. The dataset was curated using the Twitter Search API, making it a valuable resource for sentiment analysis tasks and research in the realm of natural language processing, offering a wealth of real-world tweets with sentiment annotations.

1.3. Text Data Pre-processing

To standardise text data, use stemming, lemmatization, lowercasing, tokenization, and stop word removal.

1.3.1. Tokenization

The process of breaking up complex content, such as paragraphs, into manageable chunks known as tokens is known as tokenization. There are two groups,

- Sentence tokenization**-** The sent tokenize () function may be utilised to segment a paragraph into a list of phrases.
- Word tokenization: The tokenize () function may be utilised to separate a statement into a list of words.

Activate every library required to perform tokenization on the supplied data.

1.3.1.1. Sentence tokenization

Dissecting a text paragraph into its constituent phrases is necessary. The punctuation used in English and other languages to denote sentences and paragraphs makes this task easier: commas and full stops. The process is not straightforward, though, as English abbreviations also use the same symbol.

1.3.1.2. Word tokenization

Word tokenization, sometimes called word segmentation, is the act of breaking down a text paragraph into its individual words. When delimiters like word space are used in certain languages, like English, it might be easy to tell when a new word is beginning. However, compound words employ the same spaces, making the process much more difficult.

1.3.2. Lowercasing

The reason text in lowercase is important is that the computer might read the same word twice if it is written in capital letters or in a different font style. The terms "love" and "love" may be seen differently and assigned different vectors, for example, when the text is vectorized for feature extraction. Despite being the simplest and most efficient pre-processing method that maximises the chance of accurate results, this step is sometimes disregarded.

1.3.3. Stop word removal

The pre-processing method that is most commonly used in different NLP applications is stop word removal. Simply removing phrases that appear often in all corpus texts is the goal. Pronouns and articles are common stop words. Stop words, such as "a," "an," "the," "in," "has," and so on, are frequently found in languages.

1.3.4. Stemming

Words can be shortened and returned to their original form through the process of stemming. The process becomes so much shorter that the semantics remain the same but the meaning is lost in some situations. While "connect" will be stemmed as "conect," "trouble" will be stemmed as "troubl." To decide whether or not to use stemming for pre-processing, it is necessary to do a comprehensive investigation of the matter. Stemming searches on Google might be advantageous since it displays all relevant results for the phrases that are input. Porter Stemmer is the most popular algorithm for stemming.

1.4. Emoji Data Preprocessing

1.4.1. Emoji Extraction

Utilizing Regular Expressions (Regex), extract emojis from text and transform them to text representations so they may be processed. The data collected for analysis includes a lot of non-essential information such as emoticons (😊 , 🙄), hashtags, usernames (@), hyperlinks, RT (the symbol used when a tweet is retweeted), and other information. These indicators and expressions are meaningless while processing the data since they are unrelated to sentiment analysis. All the information they include is user data, other people's tags, or specifics about the various kinds of tweets. Formatting raw Twitter data is necessary to produce a dataset that several classifiers can easily access. Using the Regex Module, the data is cleaned. Regular expressions, or Regex, are special grammatical snippets that enable pattern matching in other strings. Regular expressions are worked with in Python using the "re" package. Hash tags, URLs, retweets, hyperlinks, emojis, and other strategies for eliminating supplementary data are all included.

Emoticons, often known as Emojis, were created as a means of representing a range of emotions metaphorically. In order to convey their emotions, people use a range of different emoticons in their tweets. Emoticons may represent any emotion—positive, negative, or neutral. replacing each emoticon with a suitable word as a result. The quantity, extremeness, and usage of an emoji as the final tweet badge are all considered highlights. For example: " 😊 used to convey a cheerful mood, but replaced with a specific word".

1.5. Feature Extraction

Text feature extraction techniques encompass several methodologies for deriving meaningful information from textual data. The Improved Bag of Words (BoW) technique, introduced as a proposal, represents text as a vector of word frequencies, where individual words serve as features, and their counts within the document are used as feature values. Word embeddings, such as Word2Vec, offer dense vector representations of words, enabling the capture of semantic relationships among words. N-grams, on the other hand, involve considering sequences of adjacent words as features, which helps capture local word patterns within the text.

In parallel, emoji feature extraction techniques focus on harnessing information from emojis in text data. Emoji embeddings, exemplified by pre-trained models like Emoji2Vec, provide dense representations of emojis, leveraging their usage in extensive text corpora to encapsulate their meanings and associations. Additionally, emoji frequency analysis involves counting the occurrences of each emoji in a text document, treating this count as a feature to understand the prevalence and significance of emojis in the text. These techniques collectively enable the extraction of valuable insights from both textual content and the inclusion of emojis, enriching the analysis of multimodal data.

1.5.1. Text Feature Extraction

1.5.1.1. TF-IDF weighting based BoW (TF-IDF_BoW)

Text is represented as a vector of word frequencies, with each word serving as a feature and its count within the document as its feature value. The Bag of Words model is a method for obtaining textual characteristics that may be applied to modelling, such as in our case using machine learning algorithms to classify tweet sentiment. Put simply, it's a collection of words that characterise a sentence in a word count text. There are two components to it: a list of well-known words and a measure to assess whether or not those terms are present. Moreover, BoW does not maintain their appearance order. Using every unique word in our tweets data set, the first step is to create a vocabulary. Listing each of these unique terms and keeping an eye on their appearance in each and every tweet is the next step. When the model is ready for training, finally give it to the numerical matrix. Balance the BoW representation using TF-IDF weighting. Word counts are adjusted using this method by taking into account a word's uniqueness over the whole corpus as well as its frequency in a given text. By doing so, familiar words may have less of an impact and more weight given to useful concepts.

The initial stage in this feature extraction process is to calculate the document's $TF - IDF$ score. The most significant characteristic throughout the whole document collection is shown by the $TF - IDF$ score. The dataset receives the application of the $TF - IDF$ formula. The formula for $TF - IDF$ is expressed as follows:

$$TF - IDF = TF * IDF \tag{1}$$

The frequency with which a feature occurs in a document relative to the total number of features in the text document is defined as TF. Concurrently, IDF assesses a feature's capacity to discriminate between groups. Keep in mind that the text document's defined class label is represented by the categories below. The following formula represents $TF\ and\ IDF$:

$$TF = \frac{FFTD}{TFTD} \tag{2}$$

$$IDF = \log \frac{NDF}{TD} \tag{3}$$

In a written document, $FFTD$ represents the frequency of a feature's appearance, while $TFTD$ denotes the total number of times a phrase appears. Regarding the IDF, NDF denotes the quantity of documents that include the characteristic, whereas TD represents the total number of documents. For a given text document, a feature's greater $TF - IDF$ score indicates its greater significance.

1.5.1.2. N-grams

Take neighbouring word sequences and treat them as characteristics that represent local word patterns. We initially create word N-grams from the phrases in order to construct the vector. Words from a phrase combined to generate a Markovian process is called an N-gram. This is typically employed to forecast the following word in a string of words. Additionally, the Markovian process produces word co-occurrence, which is a crucial factor in determining a text's emotion. In this instance, an online review or a sentence are divided into many portions using N-gram sequences to represent the whole content. This is so because, compared to a simple bag of words (BoW), N-grams display word co-occurrence in a text in a more thorough manner. N is a variable that determines the partition's size. Considering the statement S that is provided as an example,

$$S = \{w_1, w_2, w_3, w_4, \ldots\ldots, w_n\} \tag{4}$$

Where, w_i are words. By varying the values of N, the N-gram values are calculated as,

N=1 means $N_1 = \{w_1, w_2, w_3, w_4, \ldots\ldots, w_n\}$

N=2 means $N_2 = \{w_1\_w_2, w_2\_w_3, w_3\_w_4, \ldots\ldots, w_{n-1}\_w_n\}$

The main notion is that a portion of the whole input text may be chosen using the set of N-grams. This guarantees that, while creating text vectors for sentiment analysis, we utilise the most important terms. After identifying the N-gram(s) in the text, it is converted back to a list of words.

1.5.1.3. Word Embeddings (Word2Vec)

Words are represented as dense vectors in pre-trained word embeddings, which capture the semantic connections between words.

Word2Vec is a popular word embedding technique. To determine the semantic similarity of the word context, it employs a neural network. Word2Vec implements a Skip-Gram and a continuous bag of words (CBOW), two designs that are inversely connected. Depending on the environment, the Skip-Gram architecture for unsupervised learning is employed to identify semantic ideas. In order to determine the greatest average logarithmic probability, Skip-Gram operates using Eq. (5):

$$E = -\frac{1}{v}\sum_{v=1}^{V}\sum_{-c\le m\le c, m\ne o} \log\log[p(w_{v+m} | w_v)] \tag{5}$$

It utilises the training set of $w_1, w_2, w_3, w_4, \ldots\ldots, w_n$ that is supplied. The context size, or window size, is indicated by the letter c. The embedding size is denoted by E. Eq. (6) may be used to compute the probability $(w_n + m|w_n)$:

$$p(o) = \frac{\exp\,(u_i^T . u'_o)}{\sum_{v\in V} exp exp(u_z^T . u'_o)} \tag{6}$$

In this case, V stands for the vocabulary list, u for "input," and u' for the corresponding vector representations of $i\ and\ o$ as "output." Using the semantic data found in a particular text collection, CBOW predicts the target word. Distributed continuous contextual representations are used. CBOW takes a word sequence and builds a static window. Next, the model guesses the centre word of the window using a log-linear classifier trained on incoming and previous words. Eq. (7)'s bigger value indicates a higher likelihood of the term w_v being inferred:

$$\frac{1}{V}\sum_{v\in V}\log(p\ (w_{v-c},\dots,w_{v-2},w_{v-1,v+1,v+2,..},w_{v+c} \qquad (7)$$

The Skip-Gram model parameters are represented by $V\ and\ c$ in this instance. Both models are shown in Figure 2.

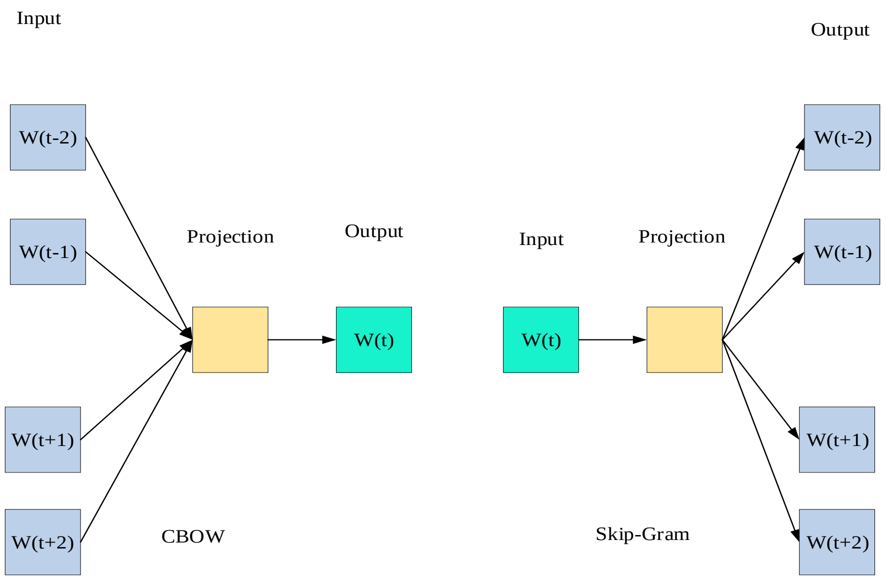

Figure 2: Structure of the CBOW and Skip-gram methods

1.5.2. *Emoji Feature Extraction Techniques*

1.5.2.1. Emoji Embeddings

Pre-trained emoji embeddings (Emoji2Vec) provide dense representations of emojis based on their usage in large text corpora. The utilization of Emoji2Vec in this research serves as a valuable tool for capturing the meanings and associations of emojis effectively. Emoji2Vec is a pre-trained model specifically designed to represent emojis in a way that enables machines to understand their semantics and relationships. Additionally, every emoji in the remark is represented as a numeric vector using pre-trained emoji2vec embedding.

$$e_i \rightarrow emoji2vec(e_i) \rightarrow \widetilde{e_i} \qquad (1)$$

Where e_i stands for the emoji in the remark with index i and $\widetilde{e_i}$ stands for the emoji embedding for e_i emoji. If pre-trained embedding models do not contain the e_i emoji, the matching $\widetilde{e_i}$ is given a zero vector. Using Eq. (1) the V dictionary, the emoji embedding matrix is produced. The Eq. (1) is utilised to generate the embedding matrix if there is a token emoji in the V lexicon. Count the frequency of each emoji in a text document as a feature.

1.6. Feature Fusion

Combine text and emoji embeddings to create a fusion representation that successfully combines both modalities. Text and emoji embeddings together provide a powerful and coherent depiction that effectively captures the spirit of both communication modalities. This approach facilitates the more comprehensive interpretation of sentiment expression by taking into account both the textual and emotive information conveyed by emojis. By using these embeddings, the model is able to analyse and comprehend text and emojis simultaneously. The fusion procedure enhances the sentiment analysis model's ability to assess and classify sentiment from a broader perspective.

1.7. Feature Selection using Self Improved-Siberian Tiger Optimization

The intelligence exhibited by Siberian tigers in their natural hunting and combat behaviours with brown bears can serve as a model for the development of a novel metaheuristic algorithm. In order to create the suggested SI-STO, the Cauchy operator is introduced in the basic STO to enhance the prey attacking behaviour of the STO. The Cauchy operator is a non-local operator, which means that it can take into account the information from all over the search space, not just from the immediate neighborhood of the current prey position. This allows the SI-STO to search for prey more efficiently and effectively. The mathematical modelling of these two intriguing Siberian tiger techniques is used.

1.7.1. Mathematical modelling

Based on the normal habits of this species, the method of updating the position of Siberian tigers in the SI-STO is divided into two stages.

1.7.1.1. Initialization

Through the use of an iterative process and the searching capability of its population members, the suggested SI-STO can offer a viable solution to the problem. Siberian tigers make up the SI-STO population, and they move about in the search space in an attempt to find better answers. Every Siberian tiger belongs to the population of SI-STOs. It so presents a potential fix for the issue. Values for the problem's variables are represented by its location in the search space. Thus, from a mathematical perspective, a vector may be used to describe each Siberian tiger, and a matrix can be used to model the population of Siberian tigers, as stated in Eq. (8).

$$X = \begin{bmatrix} X_1 \\ \vdots \\ X_i \\ \vdots \\ X_N \end{bmatrix}_{N\times m} = \begin{bmatrix} x_{1,1} & \cdots & x_{1,j} & \cdots & x_{1,j} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ x_{i,1} & \cdots & x_{i,j} & \cdots & x_{i,m} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ x_{N,1} & \cdots & x_{N,j} & \cdots & x_{N,m} \end{bmatrix}_{N\times m} \quad (8)$$

When N is the total number of Siberian tigers, X_i is the i^{th} Siberian tiger (a possible solution), and X is the population matrix of Siberian tigers' locations. When SI-STO is implemented, the starting random location of Siberian tigers in the search space is established using Eq. (9).

$$x_{i,j} = lb_j + r_{ij}.\left(ub_j - lb_j\right), i = 1,2, \ldots., N \text{ and } j = 1,2, \ldots, m \quad (9)$$

Where m is the number of issue variables, r_{ij} are random values in the interval [0, 1], lb_j and ub_j are the lower and upper bounds of the jth problem variable, respectively, and $x_{i,j}$ is the j^{th} dimension of X_i in the search space (problem variable). As previously shown, the values of the problem's variables depend on where each Siberian tiger is located inside the search space. Thus, a value of the problem's objective function may be determined for every Siberian tiger. As per Eq. (10), a vector known as the objective function vector may be utilised to exhibit the collection of computed values for the objective function.

$$F = \begin{bmatrix} F_1 \\ \vdots \\ F_i \\ \vdots \\ F_N \end{bmatrix}_{N\times 1} = \begin{bmatrix} F(X_1) \\ \vdots \\ F(X_i) \\ \vdots \\ FX(_N) \end{bmatrix}_{N\times 1} \quad (10)$$

The objective function value for the i^{th} Siberian tiger is represented by F_i, whereas F is the vector of objective function values. An important piece of knowledge on the calibre of these potential solutions is provided by the objective function's acquired values. The optimal candidate solution, X_{best} is determined by maximising the value obtained for the objective function. To update the best member in each iteration of the algorithm, the objective function value must be compared with the new values as they are calculated for the objective function.

1.7.1.2. Phase 1: Cauchy Updated Prey hunting

Siberian tigers are strong predators who hunt and devour a variety of animals. In order to simulate the hunting tactics of Siberian tigers, SI-STO members are modified throughout the first phase. This tactic involves the Siberian tiger first choosing its victim, then attacking it and chasing it until it is killed. Consequently, two steps of simulating the prey hunting phase are used. Based on the selection and attack on the prey, population members' positions are updated in the first step. This step leads to abrupt and significant shifts in the SI-STO members' positions, which improves the algorithm's capacity for accurate search space scanning through global search and exploration. All Siberian tigers have suggested prey places in the SI-STO design, which are chosen from among population members with higher objective function values than the tiger in question. This suggested prey's set of placements is displayed in Eq. (11).

$$PP_i = \{X_k | k \in \{1,2, \ldots., N\} \Lambda F_k < F_i\} \cup \{X_{best}\} \quad (11)$$

Journal of Information Systems Engineering and Management
2025, 10(34s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Where N is the total number of SI-STO members and X_{best} is the best candidate solution (the best SI-STO member). Next, we choose one member (TP_i) at random from this set PP_i to be the target of the i^{th} Siberian tiger assault (i.e., the ith population member). Its new location is then determined by simulating the attack on the prey using Eq. (12).

$$x_{i,j}^{P1S1} = x_{i,j} + r_{i,j}.(TP_{i,j} - I_{i,j}.x_{i,j}) \quad (12)$$

$i = 1,2,\dots,N$ and $j = 1,2,\dots,m$

Where the number of problem variables is m, the j^{th} dimension of X_i^{P1S1} is its j^{th} dimension, $r_{i,j}$ are random numbers in the interval [0, 1], and $I_{i,j}$ are random numbers from the set {1, 2}. $TP_{i,j}$ is the j^{th} dimension of TP_i. The new position of the i^{th} member is based on the first stage of the first phase of SI-STO. If the updated computed location increases the value of the objective function, then SI-STO members can accept it. This procedure is represented by Eq. (13),

$$X_i = \begin{cases} X_i^{P1S1}, & F_i^{P1S1} < F_i \\ X_i, else \end{cases} \quad (13)$$

Where F_i^{P1S1} is objective function value of the i^{th} member X_i^{P1S1}.

Based on the pursuit process, the population members' positions are updated in the second step. The Siberian tiger shifts its posture in the region where it is attacking the victim during this phase. Through this process, the algorithm's capacity for local search and exploitation is enhanced, leading to more effective solutions. Using Eq. (14), a new position for the Siberian tiger close to the assault site is initially determined in order to recreate the pursuit process. Then, in accordance with Eq. (15), this newly computed position takes the place of the associated member's prior position if the value of the goal function is increased.

$$x_{i,j}^{P1S2} = x_{i,j} + \frac{r_{i,j}.(ub_j - lb_j)}{t} \quad (14)$$

Where $i = 1,2,\dots.,N$, $j = 1,2,\dots,m$ and $t = 1,2,\dots.,T$

$$X_i = \begin{cases} X_i^{P1S2}, F_i^{P1S2} < F_i \\ X_i, else \end{cases} \quad (15)$$

Where X_i^{P1S2} is the new position of the i^{th} Siberian tiger based on the second stage of the first phase of SI-STO, $x_{i,j}^{P1S2}$ is its j^{th} dimension, F_i^{P1S2} is its objective function value, $r_{i,j}$ are random numbers in the interval [0, 1], and t is the iteration counter of the algorithm.

In this prey hunting strategy, the Cauchy Mutation (CM) operator is included for enhance the hunting behaviour of the SI-STO. Large mutations with a high probability can be produced using the potent CM operator. This can aid in the SI-STO's exploration of a wider search space and the discovery of more effective prey solutions. When the prey solutions are grouped together, the Cauchy mutation operator can be a useful addition to the SI-STO prey hunting approach. The Cauchy mutation operator can assist the SI-STO in escaping local optima and locating more effective prey solutions.

In Eq. (14), the CM operator is updated for improve the prey attacking stratergy, which is given by,

$$x_{i,j}^{P1S2} = x_{i,j} + CM.\frac{r_{i,j}.(ub_j - lb_j)}{t} \quad (16)$$

$$CM = \tan(\pi \times (\varepsilon - \tfrac{1}{2})) \quad (17)$$

The probability density function of the standard Cauchy distribution may be expressed as:

$$f(x) = \frac{1}{x} \times \frac{1}{1+x^2} \quad (18)$$

The related distribution function is written as follows where $-\infty < x < \infty$:

$$F(x) = \frac{1}{\pi} * arc\tan(x) + \frac{1}{2} \quad (19)$$

1.7.1.3. Phase 2: Fighting with a bear

Siberian tigers engage in combat with brown and black bears in their native habitat, ostensibly to settle disagreements over prey and to defend their own lives. As a result, in the second stage, SI-STO members receive updates based on a simulation of the Siberian tigers' bear-fighting tactics. The Siberian tiger ambushes the bear first in this battle before attacking it. The ongoing combat between these two creatures will go on the battlefield until the bear is killed by the Siberian tiger. As a result, two stages of the Siberian tiger's combat strategy—attack and conflict—are modelled. The other members of the community are regarded as the set of bears in the first stage, which simulates the attack of the i^{th} Siberian tiger on the bear. The assaulted bear's position (designated as k) is chosen at random from this group of potential bears. The location of SI-STO members changes significantly and abruptly as a result of this stage, which can improve the global search and exploration capacity of the suggested strategy. As a result, using Eq. (20), a new location is first computed for the i^{th} SI-STO member, $i = 1,2,\ldots,N$, in order to emulate the aforementioned notions.

$$x_{i,j}^{P2S1} = \begin{cases} x_{i,j} + r_{i,j}.\left(x_{k,j} - I_{i,j}.x_{i,j}\right), & F_k < F_i \\ x_{i,j} + r_{i,j}.\left(x_{i,j} - I_{i,j}.x_{k,j}\right), & else \end{cases} \tag{20}$$

Where $j = 1,2,\ldots,m$, and k is chosen at random from the set $\{1,2,\ldots,i-1,i+1,\ldots,N\}$, and $x_{k,j}$ is the j^{th} dimension of a bear position. According to the first stage of the second phase of SI-STO, X_i^{P2S1} represents the i^{th} member's new location; $x_{i,j}^{P2S1}$ denotes its j^{th} dimension; $r_{i,j}$ are random integers in the range [0, 1], and $I_{i,j}$ are random numbers from the set {1, 2}. Then, in line with Eq. (21), this newly computed position takes the place of the associated member's prior position if the value of the goal function is increased.

$$X_i = \begin{cases} X_i^{P2S1}, F_i^{P2S1} < F_i \\ X_i, else \end{cases} \tag{21}$$

Where F_i^{P2S1} is the value of X_i^{P2S1}'s objective function and F_k is the bear's (the k^{th} member of SI-STO) objective function value. The population's position is updated in the second stage based on the modelling of conflicts during warfare. This behaviour leads to slight shifts in the positions of population members, which improves SI-STO's local search and exploitation capability. In order to simulate this behaviour, Eq. (22) is first used to choose a random position close to the combat site.

$$x_{i,j}^{P2S2} = x_{i,j} + \frac{r_{i,j}}{t}.\left(ub_j - lb_j\right) \tag{22}$$

Where X_i^{P2S2} is the new location of the i^{th} Siberian tiger based on the second stage of the second phase of SI-STO, $x_{i,j}^{P2S2}$ is its j^{th} dimension, and t is the algorithm's iteration clock. When the goal function's value increases as per Eq. (23), the new position is then appropriate for the update process.

$$X_i = \begin{cases} X_i^{P2S2}, & F_i^{P2S2} < F_i \\ X_i, & else \end{cases} \tag{23}$$

Where F_i^{P2S2} is the objective function value of X_i^{P2S2}.

After updating every Siberian tiger based on the first and second stages, the first iteration of SI-STO is finished. After then, the algorithm moves on to the next iteration using the newly acquired data, and so on until the last iteration of the method, which is based on Eq. (8) to Eq. (23). During this time, the Siberian tigers' positions are updated. The optimal candidate solution discovered during all algorithm iterations is included in the output as the problem's solution following the complete application of SI-STO.

1.8. Sentiment Classification using SentEmoNet

The Sentiment Classification Model, SentEmoNet, is an innovative deep learning framework designed to excel in the task of sentiment classification by effectively leveraging both textual and emoji features. This model introduces a dual approach, combining two fundamental neural architectures: LSTM with an Attention Mechanism and CNN. The LSTM component is instrumental in capturing sequential patterns within text data while affording special attention to the relative importance of individual words, thus enhancing the understanding of context and nuance. Simultaneously, the CNN model plays a pivotal role in extracting localized features from text and emoji embeddings, concentrating on the identification of specific patterns and representations that contribute to a more comprehensive sentiment analysis. SentEmoNet is poised to offer an advanced and versatile solution for sentiment classification, embodying a fusion of these two neural approaches to maximize the utilization of both text and emoji features in the pursuit of accurate and nuanced sentiment analysis.

1.8.1. LSTM

The loop structure of the RNN produces a kind of continuous memory. However, with time, it becomes easy to forget about events that occurred a long time ago. By enhancing the intricacy and precision of information processing at the neural network layer, the LSTM partially addresses this problem. Unlike RNN, the LSTM records not just the hidden state $h(t)$, but also another memory vector, the cell state, labelled $c(t)$. Additionally, the LSTM directly controls the updating of $h(t)$ and $c(t)$ through the use of the input gate $i(t)$, the forgetting gate $f(t)$, and the output gate $o(t)$. To find the three gate vectors, utilise the following Eq. (24) to Eq. (27),

$$i(t) = \sigma(W_i h(t-1) + V_i x(t) \tag{24}$$

$$f(t) = \sigma\left(W_f h(t-1)\right) + V_f x(t) \tag{25}$$

$$o(t) = \sigma(W_o h(t-1) + V_o x(t) \tag{26}$$

The hidden state $h(t)$ and cell state $c(t)$ are computed using the following Eq. (27), Eq. (28), and Eq. (29),

$$\tilde{c}(t) = \tanh(W_c h(t-1) + V_c x(t)) \tag{27}$$

$$\tilde{c}(t) = f(t) \odot c(t-1) + i(t) \odot \tilde{c}(t) \tag{28}$$

$$h(t) = o(t) \odot c(t) \tag{29}$$

Based on cell state and gates, the LSTM can retain memory information for a longer amount of time and has been used for a number of natural language processing tasks in the past.

- **Attention mechanism (AM)**

The human brain often selectively highlights important aspects while disregarding unimportant information while interacting with objects. Important information gains more impact when AM is introduced. By gaining knowledge of the mapping between weigh and parameter, AM can assign different weights to the given LSTM states. In order to identify the relevance of input and output for the distribution of features with a greater weight, AM aims to emulate the attention allocation mechanism found in the human brain. When AM is combined with LSTM, the approach's accuracy is increased by concentrating on the characteristics that significantly affect the output variables. The $x_1, x_2, \dots, x_T$ represents the input of the LSTM, while $h_1, h, \dots, h_T$ represents the output through the hidden layer of the LSTM, that is used to obtain the attention weight distribution as the AM's input. The importance of the state parameter is indicated by its weight. The AM formula may be stated as:

$$PD_i = u \tanh(wh_i + b) \tag{30}$$

$$\alpha_i = \frac{\exp(PD_i)}{\sum_i \exp(PD_i)} \tag{31}$$

$$C = \sum_i \alpha_i h_i \tag{32}$$

Where PD_i represents the probability distribution of h_i's attention at i^{th} instant. The terms $u \ and \ w$ stand for the weighting coefficients. The bias coefficient is b, and the weighted feature is C.

1.8.2. CNN

In addition to lowering computation time and complexity, CNN, a multi-layer feed-forward neural network, enhances the error in backpropagation networks (BP). Due to its ability to automatically learn these features for classification, it has been utilised recently for sentiment classification. It does this by employing a convolution kernel to detect local characteristics. Convolution, pooling, and fully linked layers are the three primary layers that make up the CNN model. Convolutional layer input is given sentences that have been transformed into a matrix of integers. A row or vector on the matrix table corresponds to each token, which are the words that make up a phrase. Embedding methods like Word2Vec and GloVe model are commonly used to build these vectors. After receiving vector input, the CNN model applies filters to extract local features. In the convolutional layer of CNN, the most computation-intensive feature layer, the majority of features are computed. With a function known as the convolution kernel, the convolutional layer generates feature maps. The pooling layer recovers the most significant information following the convolution procedure. Local adequate statistics are computed by the pooling layer. This approach keeps the model from overfitting, enables the pooling layer to lower feature dimensions, and helps CNN achieve computational speed and cost reduction. Ultimately, a probability distribution is generated by the fully connected layer to categorise sentiment outcomes.

RESULT AND DISCUSSION

This section compares the suggested sentiment analysis model's performance to the methods already in use and assesses its efficacy using performance indicators. Two different kinds of datasets are employed in this context to assess sentiment. The datasets utilised for assessment are Twitter Sentimental Analysis [27] and Emoji Sentiment Analysis using Python [26]. The PYTHON platform uses the dataset, which is split into training (70%) and testing (30%) categories.

Journal of Information Systems Engineering and Management
2025, 10(34s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/
Research Article

1.9. Overall Comparison of The Proposed Sentiment Analysis Model

The performance of the proposed model is compared with the existing models like Bi-LSTM [20], CNN_BiGRU[18], CNN and Deep Neural Network (DNN). The table 1 give the comparison of the performance metric.

Table 1: Comparison of performance metrics for dataset 1

Testing Metrics for Dataset 1					
Performance metrics	**Bi-LSTM [20]**	**CNN_BiGRU[18]**	**CNN**	**DNN**	**Proposed**
Accuracy	69.7765	76.8775	74.7664	82.89855	97.87884
Precision	73.775	75.7774	73.6664	84.8775	95.7774
Sensitivity	72.3456	74.4664	76.88923	84.6475	97.4434
Specificity	71.664	75.55663	72.98745	80.87877	94.77474
F-Measure	74.76774	70.8775	75.7774	78.7775	90.7744
MCC	73.98774	78.89885	71.898774	79.8885	92.434343
NPV	76.8774	81.7775	73.7774	85.7875	91.646489
Recall	67.7748	83.7755	70.8984	81.98885	93.7677
FPR	0.45653	0.34455	0.34545	0.21983	0.09884
FNR	0.40098	0.3234	0.32111	0.21103	0.08988

In the evaluation of sentiment analysis models on Dataset 1, several performance metrics were used to assess their effectiveness. The Proposed model stands out with remarkable results, achieving an impressive accuracy of 97.88%, demonstrating its proficiency in correctly classifying sentiments within the dataset. It also excels in precision, with a value of 95.78%, indicating its ability to make precise positive predictions, and it exhibits high sensitivity (97.44%) and specificity (94.77%), showcasing its strength in identifying both positive and negative sentiments accurately. The Proposed model strikes a balance between precision and sensitivity, as reflected in its high F-Measure and Matthews Correlation Coefficient (MCC) values, further affirming its excellent binary classification capabilities. Additionally, it demonstrates the lowest False Positive Rate (FPR) and False Negative Rate (FNR) among all models, highlighting its efficiency in minimizing incorrect predictions. These exceptional results position the Proposed model as a top-performing choice for sentiment analysis in Dataset 1. In a similar manner, table 2 provides the metrics values for dataset 2.

Table 2: Comparison of performance metrics for dataset 2

Testing Metrics for Dataset 2					
Performance metrics	**Bi-LSTM [20]**	**CNN_BiGRU[18]**	**CNN**	**DNN**	**Proposed**
Accuracy	68.87877	78.67567	75.7774	86.885	96.84545
Precision	69.8775	70.77775	76.87774	87.8839	97.7774
Sensitivity	67.5353	77.877876	79.898774	80.7774	93.787545
Specificity	72.7664	76.8775	67.8774	83.9883	92.898854
F-Measure	72.7664	74.78776	69.7774	84.8884	94.885454
MCC	70.77774	71.77766	73.78854	85.77755	90.884535
NPV	70.7774	74.7775	70.85656	86.8884	91.84545
Recall	62.7774	77.88844	73.0094	78.88895	93.77455
FPR	0.5643	0.390099	0.31223	0.1233	0.07877
FNR	0.51212	0.39987	0.36555	0.23232	0.068888

In the assessment of sentiment analysis models on Dataset 2, the performance metrics reveal the proficiency of each model in classifying sentiments within this specific dataset. Notably, the Proposed model stands out as the top-performing model, with an exceptional accuracy of 96.85%, indicating its ability to make correct sentiment predictions. This model also exhibits the highest precision at 97.78%, showcasing its precision in identifying positive sentiments accurately. Furthermore, it demonstrates the highest sensitivity (true positive rate) and recall, underscoring its effectiveness in correctly identifying positive cases. Specificity, measuring the ability to identify negative cases, is also commendable for the Proposed model, ranking second in this category. The F-Measure, MCC, and NPV highlight the model's ability to balance precision and sensitivity effectively. Additionally, the Proposed model achieves the lowest FPR and FNR+, reflecting its excellence in minimizing incorrect predictions. These results collectively position the Proposed model as the leading choice for sentiment analysis in Dataset 2, with its remarkable performance across multiple key metrics.

1.9.1. Accuracy

When comparing the accuracy metrics from Table 1 (Dataset 1) and Table 2 (Dataset 2), the Proposed model consistently outperforms the other sentiment analysis models in both datasets. In Table 1, the Proposed model achieves an impressive accuracy of 97.88%, showcasing its exceptional ability to make correct sentiment predictions within Dataset 1. Similarly, in Table 2, the Proposed model maintains its superiority with an accuracy of 96.85% in Dataset 2. These results underscore the Proposed model's remarkable and consistent performance across different datasets, positioning it as a strong choice for sentiment analysis tasks. The comparison of Accuracy metrics for both datasets are shown in figure 3.

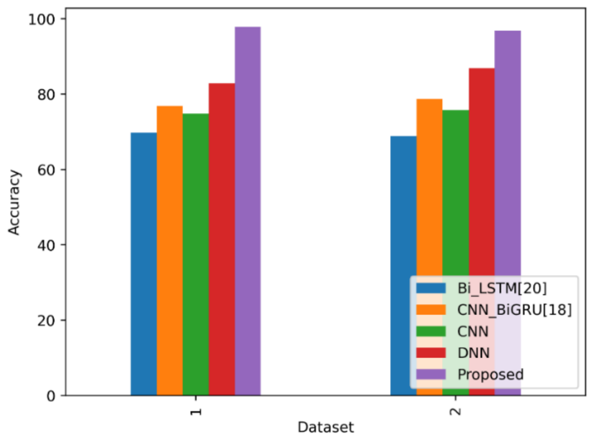

Figure 3: Comparison of the accuracy values

1.9.2. Precision

A comparison of the precision metrics from Table 1 (Dataset 1) and Table 2 (Dataset 2) reveals significant differences in the performance of sentiment analysis models. In both datasets, the Proposed model consistently stands out with the highest precision values. In Table 1, it achieves a precision of 95.78%, showcasing its exceptional ability to make precise positive sentiment predictions within Dataset 1. Similarly, in Table 2, the Proposed model maintains its superiority with a precision of 97.78% in Dataset 2. Figure 4 displays the precision metrics comparison for both datasets.

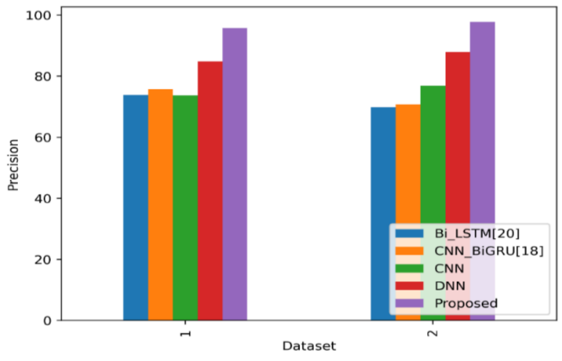

Figure 4: Comparison of the precision metric

1.9.3. Sensitivity

A comparison of the sensitivity metrics from Table 1 (Dataset 1) and Table 2 (Dataset 2) reveals variations in the performance of sentiment analysis models, with the Proposed model consistently emerging as a top performer in both datasets. In Table 1, the Proposed model achieves an extraordinary sensitivity of 97.44%, underlining its exceptional ability to accurately identify positive sentiments within Dataset 1. Likewise, in Table 2, the Proposed model continues to excel with a sensitivity of 93.79% in Dataset 2, confirming its robustness in recognizing positive sentiments. Figure 5 compares the sensitivity metrics for the two datasets.

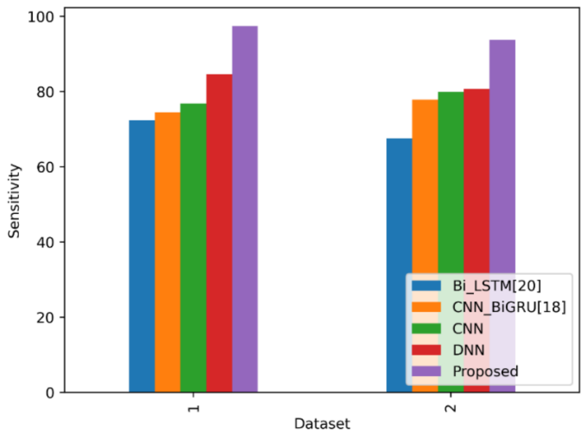

Figure 5: Sensitivity values comparison

1.9.4. Specificity

In both datasets, the Proposed model consistently achieves one of the highest specificity values, indicating its ability to correctly identify negative sentiment cases. In Table 1, it boasts a specificity of 94.77%, showcasing its exceptional performance in recognizing negative sentiments accurately within Dataset 1. Similarly, in Table 2, the Proposed model maintains a strong performance with a specificity of 92.90% in Dataset 2, further emphasizing its proficiency in accurately identifying negative sentiment. Figure 6 displays the comparison of the specificity metrics for the two datasets.

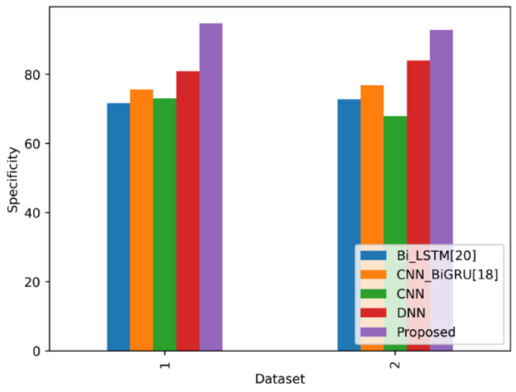

Figure 6: Comparison of the specificity values

1.9.5. Recall

In both datasets, the Proposed model consistently achieves one of the highest recall values, indicating its effectiveness in correctly identifying positive sentiment cases. In Table 1, it demonstrates a recall of 93.77%, showcasing its ability to accurately recognize positive sentiments within Dataset 1. Similarly, in Table 2, the Proposed model maintains a strong performance with the same recall value of 93.77% in Dataset 2, emphasizing its proficiency in accurately identifying positive sentiment. Figure 7 compares the recall metrics for the two datasets.

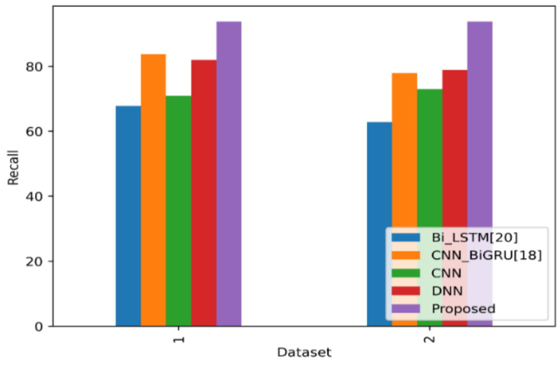

Figure 7: Recall metric comparison

1.9.6. *F-Measure*

In both datasets, the Proposed model consistently achieves the highest F-Measure, indicating its ability to balance precision and sensitivity effectively. In Table 1, it demonstrates an F-Measure of 90.77%, showcasing its capacity to achieve a harmonious balance between precision and sensitivity in sentiment analysis within Dataset 1. Similarly, in Table 2, the Proposed model maintains a strong performance with an F-Measure of 94.89% in Dataset 2, further emphasizing its proficiency in achieving a balanced approach to sentiment classification. The F-Measure values for the two datasets are compared in figure 8.

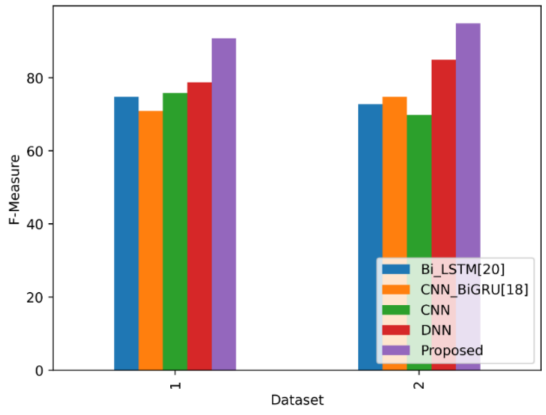

Figure 8: Comparison of the F-Measure values

1.9.7. *MCC*

A comparison of the MCC metrics from Table 1 (Dataset 1) and Table 2 (Dataset 2) reveals differences in the performance of sentiment analysis models in terms of binary classification quality. In both datasets, the Proposed model consistently stands out by achieving the highest MCC, highlighting its proficiency in providing high-quality binary sentiment classifications. In Table 1, the Proposed model attains an MCC of 92.43, demonstrating its ability to deliver exceptionally strong binary classification results within Dataset 1. Likewise, in Table 2, the Proposed model maintains a robust performance with an MCC of 90.88 in Dataset 2, further underscoring its reliability in maintaining high-quality binary sentiment classification. The comparison of MCC values for both datasets are shown in Figure 9.

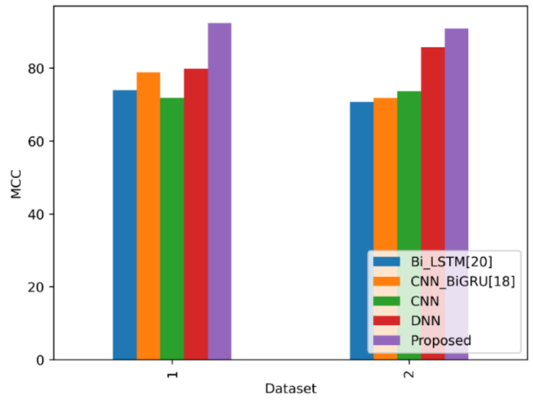

Figure 9: Comparison of the MCC for dataset 1 and dataset 2

1.9.8. NPV

In both datasets, the Proposed model consistently achieves high NPV, indicating its effectiveness in correctly predicting negative sentiment cases. In Table 1, it demonstrates an NPV of 91.65%, showcasing its capacity to accurately predict negative sentiments within Dataset 1. Similarly, in Table 2, the Proposed model maintains a strong performance with an NPV of 91.85% in Dataset 2, emphasizing its proficiency in correctly predicting negative sentiment. The NPV values for both datasets are compared in Figure 10.

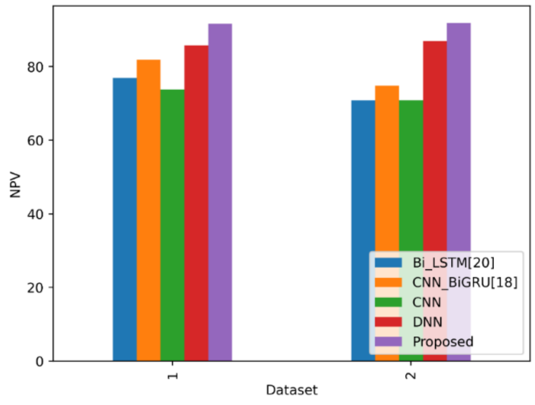

Figure 10: Comparison of the NPV for dataset 1 and dataset 2

1.9.9. FPR

In both datasets, the Proposed model consistently achieves the lowest FPR, indicating its efficiency in minimizing incorrect predictions of negative sentiment cases as positive. In Table 1, it stands out with an FPR of 0.09884, emphasizing its exceptional performance in correctly classifying negative sentiments with a high degree of accuracy within Dataset 1. Similarly, in Table 2, the Proposed model continues to exhibit the lowest FPR with a value of 0.07877 in Dataset 2, underscoring its proficiency in minimizing incorrect predictions of negative sentiment. Figure 11 compares the FPR values for the dataset 1 and dataset 2.

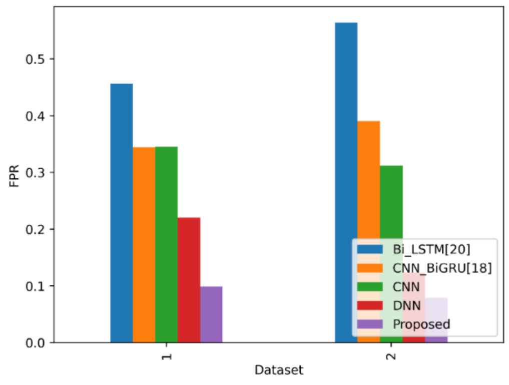

Figure 11: FPR values for dataset 1 and dataset 2

1.9.10. FNR

In both datasets, the Proposed model consistently achieves the lowest FNR, indicating its proficiency in minimizing incorrect predictions of negative sentiment cases. In Table 1, it stands out with an FNR of 0.08988, emphasizing its exceptional performance in accurately classifying negative sentiments within Dataset 1. Similarly, in Table 2, the Proposed model continues to exhibit the lowest FNR with a value of 0.068888 in Dataset 2, underscoring its proficiency in minimizing incorrect predictions of negative sentiment. The FNR values for datasets 1 and 2 are shown in Figure 12.

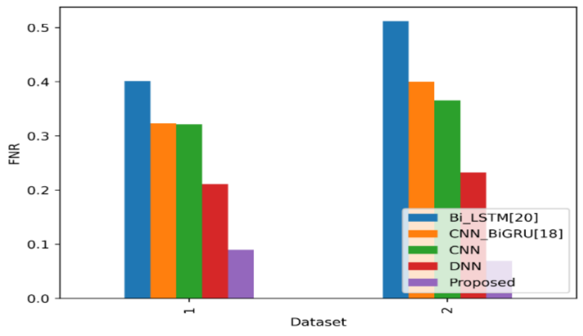

Figure 12: FNR values for dataset 1 and dataset 2

CONCLUSION

In conclusion, the SentEmoNet approach to sentiment analysis, which incorporates both textual and emoji data, presents a robust framework for understanding and predicting sentiment in diverse text sources, including social media, customer reviews, and surveys. Through a systematic methodology encompassing data collection, pre-processing, feature extraction, fusion, and selection, addressed the complexity of sentiment analysis in the digital age. Our data pre-processing steps, such as tokenization, lowercasing, stop word removal, stemming, and emoji extraction, have ensured that the input data is transformed into a uniform and meaningful representation. Leveraging TF-IDF_BoW, pre-trained Word2Vec word embeddings, and n-grams for text, and pre-trained emoji embeddings and emoji frequency counting for emojis, the feature extraction techniques capture the essence of both words and emojis.

The feature fusion approach, which combines text and emoji features with weighted concatenation, acknowledges the unique emotional context conveyed by emojis, resulting in a more holistic sentiment analysis model. Furthermore, the introduction of SI-STO as a feature selection technique enables the identification of the most relevant features for accurate sentiment prediction. The sentiment classification model, SentEmoNet, showcases the fusion of LSTM with an Attention Mechanism and a CNN to capture text patterns and extract local features from both text and emoji embeddings. The model evaluation phase, using standard sentiment analysis metrics, demonstrates the effectiveness of the approach in achieving high accuracy in sentiment prediction. As the digital landscape continues to evolve, understanding sentiment in textual and emoji-rich data is becoming increasingly important. The approach not only provides a state-of-the-art sentiment analysis framework but also opens doors to a wide range of applications, including social media sentiment monitoring, customer feedback analysis, and user sentiment tracking. With its ability to leverage both text and emojis, our approach is well-equipped to handle the nuances and intricacies of modern communication, making it a valuable asset for various domains and industries.

REFERENCE

[1]. Akhtar, M.S., Garg, T. and Ekbal, A., 2020. Multi-task learning for aspect term extraction and aspect sentiment classification. Neurocomputing, 398, pp.247-256.
[2]. Farkhod, A., Abdusalomov, A., Makhmudov, F. and Cho, Y.I., 2021. LDA-based topic modeling sentiment analysis using topic/document/sentence (TDS) model. Applied Sciences, 11(23), p.11091.
[3]. Rajput, A., 2020. Natural language processing, sentiment analysis, and clinical analytics. In Innovation in health informatics (pp. 79-97). Academic Press.
[4]. Pathak, A.R., Pandey, M. and Rautaray, S., 2021. Topic-level sentiment analysis of social media data using deep learning. Applied Soft Computing, 108, p.107440.
[5]. Ruz, G.A., Henríquez, P.A. and Mascareño, A., 2020. Sentiment analysis of Twitter data during critical events through Bayesian networks classifiers. Future Generation Computer Systems, 106, pp.92-104.

[6]. Neogi, A.S., Garg, K.A., Mishra, R.K. and Dwivedi, Y.K., 2021. Sentiment analysis and classification of Indian farmers' protest using twitter data. International Journal of Information Management Data Insights, 1(2), p.100019.
[7]. Wang, Y., Liu, J., Ruan, Q., Wang, S. and Wang, C., 2021. Cross-subject EEG emotion classification based on few-label adversarial domain adaption. Expert Systems with Applications, 185, p.115581.
[8]. Adikari, A., Gamage, G., De Silva, D., Mills, N., Wong, S.M.J. and Alahakoon, D., 2021. A self structuring artificial intelligence framework for deep emotions modeling and analysis on the social web. Future Generation Computer Systems, 116, pp.302-315.
[9]. Shankar, S. and Tewari, V., 2021. Understanding the emotional intelligence discourse on social media: Insights from the analysis of twitter. Journal of Intelligence, 9(04), p.56.
[10]. Samuel, J., Rahman, M.M., Ali, G.M.N., Samuel, Y., Pelaez, A., Chong, P.H.J. and Yakubov, M., 2020. Feeling positive about reopening? New normal scenarios from COVID-19 US reopen sentiment analytics. Ieee Access, 8, pp.142173-142190.
[11]. Essameldin, R., Ismail, A.A. and Darwish, S.M., 2022. Quantifying Opinion Strength: A Neutrosophic Inference System for Smart Sentiment Analysis of Social Media Network. Applied Sciences, 12(15), p.7697.
[12]. Caratù, M., Brescia, V., Pigliautile, I. and Biancone, P., 2023. Assessing Energy Communities' Awareness on Social Media with a Content and Sentiment Analysis. Sustainability, 15(8), p.6976.
[13]. Chen, Q. and Sokolova, M., 2021. Specialists, scientists, and sentiments: Word2Vec and Doc2Vec in analysis of scientific and medical texts. SN Computer Science, 2, pp.1-11.
[14]. Abid, F., Li, C. and Alam, M., 2020. Multi-source social media data sentiment analysis using bidirectional recurrent convolutional neural networks. Computer Communications, 157, pp.102-115.
[15]. Wang, Y., Huang, G., Li, J., Li, H., Zhou, Y. and Jiang, H., 2021. Refined global word embeddings based on sentiment concept for sentiment analysis. IEEE Access, 9, pp.37075-37085.
[16]. Dashtipour, K., Gogate, M., Cambria, E. and Hussain, A., 2021. A novel context-aware multimodal framework for persian sentiment analysis. Neurocomputing, 457, pp.377-388.
[17]. Xu, N., Mao, W., Wei, P. and Zeng, D., 2020. MDA: Multimodal data augmentation framework for boosting performance on sentiment/emotion classification tasks. IEEE Intelligent Systems, 36(6), pp.3-12.
[18]. Yan, W., Zhou, L., Qian, Z., Xiao, L. and Zhu, H., 2021. Sentiment analysis of student texts using the CNN-BiGRU-AT model. Scientific Programming, 2021, pp.1-9.
[19]. Tang, T., Tang, X. and Yuan, T., 2020. Fine-tuning BERT for multi-label sentiment analysis in unbalanced code-switching text. IEEE Access, 8, pp.193248-193256.
[20]. Lin, Y., Li, J., Yang, L., Xu, K. and Lin, H., 2020. Sentiment analysis with comparison enhanced deep neural network. IEEE Access, 8, pp.78378-78384.
[21]. Loureiro, M.L., Alló, M. and Coello, P., 2022. Hot in Twitter: Assessing the emotional impacts of wildfires with sentiment analysis. Ecological Economics, 200, p.107502.
[22]. Tesfagergish, S.G., Kapočiūtė-Dzikienė, J. and Damaševičius, R., 2022. Zero-shot emotion detection for semi-supervised sentiment analysis using sentence transformers and ensemble learning. Applied Sciences, 12(17), p.8662.
[23]. Lu, X. and Zhang, H., 2021. Sentiment analysis method of network text based on improved AT-BiGRU model. Scientific Programming, 2021, pp.1-11.
[24]. Bharti, S.K., Varadhaganapathy, S., Gupta, R.K., Shukla, P.K., Bouye, M., Hingaa, S.K. and Mahmoud, A., 2022. Text-Based Emotion Recognition Using Deep Learning Approach. Computational Intelligence and Neuroscience, 2022.
[25]. Sharaf Al-deen, H.S., Zeng, Z., Al-sabri, R. and Hekmat, A., 2021. An improved model for analyzing textual sentiment based on a deep neural network using multi-head attention mechanism. Applied System Innovation, 4(4), p.85.
[26]. Dataset 1 is taken from https://www.kaggle.com/code/rtanuj/emoji-sentiment-analysis-using-python/input?select=votes_cleaned.csv dated on 10/10/2023.
[27]. Dataset 2 is taken from https://www.kaggle.com/code/karan842/twitter-sentimental-analysism dated on 10/10/2023.