**Enhancing Sentiment Analysis with Sarcasm Detection: A Machine Learning Method Using Text, Emoji, and Punctuation Features**

JayashriBAGADE1,\*, MrugankSHIRURKAR2,\*, ArpitaWAGHMARE3,\*, ShivamYELGIRE4,\*, SohamJADHAV5,\*

1Professor, Vishwakarma Institute of Information Technology, Pune, Maharashtra, India,

\*Email: jayashrihedaoo@rediffmail.com, jayashree.bagade@viit.ac.in

2Student, Vishwakarma Institute of Information Technology, Pune, Maharashtra, India,

\*Email: mrugank.22111161@viit.ac.in

3Student, Vishwakarma Institute of Information Technology, Pune, Maharashtra, India

\*Email: arpita.22220116@viit.ac.in

4Student, Vishwakarma Institute of Information Technology, Pune, Maharashtra, India

\*Email: shivam.22110713@viit.ac.in

5Student, Vishwakarma Institute of Information Technology, Pune, Maharashtra, India

\*Email: soham.21910380@viit.ac.in

**Abstract:** In the big data era, the explosion of user-generated content on social media and online forums offers an opportunity for sentiment analysis and sarcasm detection. However, traditional techniques frequently fail to take into account the rich expressiveness of punctuation and emoji, which restricts their capacity to fully represent the wide range of emotions found in textual data. This makes sentiment analysis models less applicable in the real world and produces erroneous sentiment predictions.

In order to improve sentiment analysis and sarcasm detection, this paper suggests a novel way to use punctuation and emoji. We present a framework to perform a more in-depth sentiment analysis of sentences using a decision tree model with feature connectors. Our approach is centered around three main points: We extract features such as sentiment (positive, negative, neutral), punctuation use (question marks, exclamation points, etc.), and emoji type/quantity to enhance sarcasm detection. Following that, each sentence's combined features are fed into a decision tree model via feature connectors. In order to improve sentiment analysis and sarcasm detection, this model examines sentence structure and context to comprehend how words, emoji, and punctuation interact.

**Keywords:** Sentiment analysis, Sarcasm detection, Emoji, Punctuation marks, Decision tree, Feature connectors

1. Introduction

Today's digital age offers opportunities for a variety of predictive analytics tasks, such as sentiment analysis and sarcasm detection, due to the massive amount of data generated on the internet on a daily basis. Rich language characteristics like emoji and punctuation have become usual in textual data due to the growth of social media platforms and online communication channels. Because human language expression is complicated, analyzing this data presents unique challenges in addition to offering insights into public opinions and attitudes.

Traditional sentiment analysis techniques have primarily focused on analyzing textual data based on lexical and syntactic features, often overlooking the expressive potential of emoji and punctuation marks. Similar to this, because sarcasm is context-dependent and frequently subtle, it has remained a difficult task to detect. Because of this, current sentiment analysis models might not be able to fully represent the range of emotions expressed in textual content, which would result in imprecise predictions and restricted applicability in real-world scenarios.

The increasing reliance on textual data for decision-making processes in various domains, including marketing, customer service, and public opinion analysis, underscores the need for more robust and nuanced sentiment analysis techniques. Furthermore, the prevalence of subtle expressions and sarcasm in online communication emphasizes how crucial it is to create precise and context-aware sarcasm detection models. Emoji and punctuation can be added to sentiment analysis and sarcasm detection algorithms to improve prediction granularity and accuracy, which will benefit these models in real-world scenarios.

The sentimental messages that emoji and punctuation marks convey, as well as the minute details they convey, are frequently overlooked by traditional sentiment analysis approaches. We can obtain a more comprehensive understanding of textual data by incorporating these components into models for sarcasm detection and sentiment analysis, which will capture both explicit and implicit user sentiments. This improves prediction accuracy and allows for a more thorough examination of user attitudes and emotions, leading to better decision-making across a range of industries.

In this paper, we propose a novel method to enhance sentiment analysis and sarcasm detection in textual data by utilizing emoji and punctuation marks. We introduce a decision tree-based model that uses connectors to break up sentences and perform more detailed sentiment analysis. We imply the proficiency and robustness of our approach in accurately predicting sentiments and detecting sarcasm in textual content through empirical evaluation and comparison with existing techniques.

1. Related work

In a study by Gautam and Yadav [2], machine learning algorithms and semantic analysis were used to perform sentiment analysis on Twitter data. Using Naive Bayes, Maximum Entropy, and Support Vector Machines (SVMs), achieving the accuracy rates were 88.2%, 83.8%, and 85.5%, respectively. Furthermore, WordNet-based semantic orientation was added, resulting in an accuracy of 89.9%. The authors hypothesized that increasing the size of the training dataset would enhance the process of identifying sentences related to feature vectors. Additionally, they suggested expanding WordNet for review summarization in an effort to give users a more useful visual depiction of the content.

Researchers used multiple classifier layers in a cascaded machine learning approach to investigate multilingual sentiment analysis in another study [14]. This involved a combination of features such as discourse elements, negation cues, and unigrams, in addition to single classifiers. A dataset of blog entries, reviews, and forum discussions in the languages of English, Dutch, and French was used for the experiments. The most accurate language was English, with a maximum accuracy of 83.30% across all three classifier layers. It's interesting to note that a two-layer cascade for English produced almost identical results (83.10%), indicating that the most complex model may have redundant features. The Dutch performed worse, using the three-layer cascade to achieve a maximum accuracy of 69.03%. The results of the French accuracy tests were not made clear. The study emphasizes how difficult it is to perform sentiment analysis on single sentences, especially in the absence of additional context such as previous sentences or domain-specific knowledge. Errors in classification have a negative impact on recall and accuracy, particularly in tasks where retrieving information is a primary goal. It also emphasizes how crucial it is to use representative training data when developing sentiment analysis models.

A different study [13] compared machine learning algorithms with Twitter data to analyze sentiment. The highest accuracy (82.78%) was obtained by multinomial Naive Bayes using unigrams, or single words, as features. The lower performance was displayed by Bernoulli Naive Bayes (73.76% with unigrams), followed by Support Vector Machines (SVMs) (79.50%). This study draws attention to the problem of data sparsity in Twitter sentiment analysis, where the character count limit may limit tweet expressiveness and possibly degrade the efficiency of certain algorithms, such as support vector machines (SVMs). It's interesting that the study found no clear benefit for representing data in Twitter sentiment analysis using word frequency or sentiment polarity.

A different study examined into the effectiveness of machine learning classifiers performed on Twitter data for sentiment analysis [12]. The algorithms Random Forest, Support Vector Machine (SVM), and Naive Bayes (NB) were assessed; for a variety of datasets, they produced accuracies between 60% and 70%. A small improvement in accuracy was seen with larger datasets. Still, this study's restriction was its emphasis on categorizing sentiment into only three groups: neutral, negative, and positive. This method misses the opportunity to use more labels for advanced sentiment analysis.

Some studies concentrate on sarcasm detection in online communication in addition to sentiment analysis. In one such study, 1.3 million social media tweets were analyzed using machine learning algorithms like logistic regression, support vector machines, and BERT [10]. Despite the model's 73.1% accuracy rate, there may be space for improvement in some areas of sarcasm detection, according to the F1 score (72.4), precision (72.2), and recall (71.3) metrics. This demonstrates how difficult it is to reliably detect sarcasm in online text, even when using machine learning techniques.

The given excerpt [9] discusses a study that used contextual features and deep learning to detect sarcasm in online text. Although the precise deep learning model is not stated, the results indicate a promising 94% accuracy and 95% precision in sarcasm identification. This implies that sarcasm detection could benefit from the use of deep learning techniques. The recall metric (94%) highlights the continued difficulties in accurately capturing sarcasm in natural language, though, and suggests that there is still room for improvement.

Sentissi et al. [8] investigated sentiment analysis in the domain of software engineering using an emoji-aware model known as SEntiMoji. A variety of software development-related datasets, such as JIRA issues, Stack Overflow discussions, code reviews, and Java libraries, were used to assess this model. SEntiMoji's accuracy ranged from 0.876 on the Stack Overflow dataset to 0.891 on the JIRA dataset, indicating promising results. Nonetheless, this method's primary drawback is its reliance on labeled training data. Furthermore, the study only looked at English text, so it's unclear how useful SEntiMoji is for other languages.

Using an Attention-based Bidirectional Long Short-Term Memory (EA-Bi-LSTM) model, Wan et al. [6] examined emoji-based sentiment analysis on a Chinese microblog corpus. Although the model's accuracy was 0.764, the performance was negatively impacted by the imbalance in classes. In particular, the model performed worse at classifying neutral sentiments than it did positive and negative sentiments. This demonstrates how difficult it can be to handle unbalanced data in sentiment analysis tasks. Furthermore, the performance of the model might have been influenced by the caliber of the emoji labels in the dataset.

Emoji data's impact on sentiment analysis was investigated in a different study [4]. Using a dataset that was gathered from Twitter, tweets classified into seven emotions (sad, angry, happy, etc.) were classified using Multinomial Naive Bayes (MNB) and Support Vector Machines (SVM). While MNB outperformed SVM in accuracy for larger vocabulary sets, both models' accuracy only slightly improved (by about 0.5%) when emoji data was included. This implies that while emoji alone might not greatly improve sentiment analysis, their combination with other features merits more research. The study also emphasizes how MNB is superior to SVM when handling high-dimensional data, as the addition of emoji increases the feature count.

Several studies use machine learning algorithms to analyze sentiment on Twitter data. Support Vector Machines (SVMs), Maximum Entropy, and Naive Bayes were investigated by Gautam et al. [1], who reported encouraging results (approximately 88% accuracy) for product sentiment analysis. The study does note certain difficulties, such as dealing with colloquial language and typos. Pang et al. [3] conducted a study wherein they compared multiple algorithms and observed that Naive Bayes exhibited a faster learning speed than J48 in the context of sentiment classification on product reviews.

Some studies investigate deep learning techniques in addition to machine learning approaches. Joshi et al.'s study [11] used a combination of hand-crafted contextual features and deep learning features to investigate sarcasm detection in English tweets. The difficulties in capturing sarcasm because of informal language and the scarcity of particular features in the data were emphasized, even though the accuracy wasn't stated directly.

The use of emoji and casual language in social media data presents particular difficulties for sentiment analysis. A social media dataset (size not specified) was used by Wang et al. [5] to study the effect of emoji on sentiment analysis. The model in question was probably created with social media complexity in mind, even though it isn't specifically mentioned. Emoji did, according to their analysis, slightly improve accuracy (by about 0.5%). This implies that, in addition to other features, emoji might have a minor impact that merits more research.

Emoji were investigated for sentiment analysis of photos by Sun et al. [7]. On a dataset of labeled Twitter images (1269 images), their SmileyNet model performed better than earlier models by translating emoji into numerical representations. However, SmileyNet's performance was hampered by issues like uneven training data and inconsistent labeling. These studies show how useful emoji can be in sentiment analysis, but they also stress the importance of weighing emoji against other variables and utilizing high-quality data to create reliable models.

An extensive review of sarcasm detection techniques can be found in "Automatic Sarcasm Detection"[15]. Statistical and deep learning techniques, rule-based classifiers, and other models are covered, with an emphasis on improving performance metrics like F-score and AUC. The study also looks at datasets, particularly shared task datasets like SemEval-2014 and SemEval-2015 and short text datasets like tweets. Notably, in SemEval-2015, the top-performing system achieved an astounding accuracy of 82.75%.

In order to detect sarcasm in tweets, "Contextualized Sarcasm Detection on Twitter"[16] presents a machine learning model. The study focuses on data from Twitter, though the exact dataset isn't made public. Among the important conclusions are that the accuracy starts at 47.4% and rises to 85.1% when all features are taken into account. The model attains an accuracy rate of 84.3% even when trained exclusively on essential features. The significance of contextual cues and varied features in enhancing sarcasm detection on Twitter is underscored by these findings.

The study examines the use of machine learning algorithms for sarcasm detection on Twitter[17], classifying 31 articles into AMLA and CMLA groups. In AMLA, SVM becomes a popular choice, and both CNN and SVM show good prediction performance. The performance of CMLA algorithms varies because of various text processing and classification characteristics. After attempting to find sarcastic characteristics that both groups share, the study produced eight different clusters. While particular model accuracy is not stated, the paper illustrates various methods for sarcasm detection.

The approach to sarcasm detection used in the paper[18] is multifaceted and includes machine learning, lexical, pragmatic, and syntactic tools. It combines deep learning methods for sarcasm detection in social data, including Attention Mechanisms, Recurrent Neural Networks, Convolutional Neural Networks, and Long Short-Term Memory (LSTM) models. Features such as capitalization, emoji/emoticons, interjections, bigrams, n-grams, hyperbole, punctuation, and unigrams are all examined. Furthermore, the study explores the difficulties and methods of locating satirical content in textual data. Though the paper provides detailed insights into various approaches for sarcasm detection, it does not provide the specific dataset or accuracy results.

The document [19] indicate a variety of approaches to sarcasm detection in tweets, including discrete and neural models. Contextual features, such as historical tweet behavior, have proven effective. Recent studies emphasize the importance of contextual information from past tweets in improving accuracy. Neural network models, known for feature extraction, are gaining traction and outperforming traditional methods. Overall, integrating contextualized features enhances sarcasm detection algorithms.

1. Methodology
   1. Feature Extraction

Feature extraction plays a critical role in sarcasm detection with sentiment analysis, emoji, and punctuation marks. It allows us to extract relevant information from textual data that can be used to improve the accuracy of sarcasm identification. This section proposes a feature extraction technique that combines several textual features to enhance sarcasm detection effectiveness.

**3.1.1 Sentiment Analysis:**

Sentiment analysis aims to uncover the emotional tone of a sentence, which can be a strong indicator of sarcasm. Here, we extract sentiment features using either lexicon-based or machine learning approaches.

* **Lexicon-Based Approach:**
  + We employ a pre-defined sentiment lexicon, a dictionary containing words with positive, negative, or neutral sentiment scores.
  + Example: Let's consider the sentence "This movie was awful. It deserves 10 out of 10 stars!"
  + By matching words like "awful" with negative scores and "10" with positive scores in the lexicon, we can identify a potential sentiment mismatch, suggesting sarcasm.
* **Machine Learning Models:**
  + Alternatively, pre-trained sentiment analysis models trained on vast amounts of sentiment-labeled data can be used. These models assign sentiment scores or labels directly to the sentence.
  + Example: A pre-trained model might analyze the sentence "This movie was awful" and assign a negative sentiment score, further strengthening the suspicion of sarcasm when juxtaposed with the high rating of "10 out of 10 stars".

**3.1.2 Punctuation Marks:**

Punctuation marks often convey subtle nuances in tone and intent, making them valuable features for sarcasm detection. We extract features such as the presence and frequency of specific punctuation marks:

* Exclamation points (!) often indicate excitement or strong emotions, but their excessive use in sarcastic statements can be a giveaway.
* Question marks (?) can be used genuinely to ask questions, but they can also be used sarcastically to express disbelief or disapproval.
* Ellipses (...) can introduce hesitation or imply something left unsaid, potentially indicating sarcasm.
* Example: The sentence "This is the BEST movie EVER..." uses ellipses to create a sense of forced enthusiasm, hinting at sarcasm. Additionally, the capitalized "BEST" might be another indicator.

**3.1.3 Emoji:**

Emoji are a prevalent form of communication, adding emotional context and expressing feelings. We extract emoji features to capture this information:

* Emoji Type: We identify the type of emoji present, such as smiley faces, thumbs up/down, or wink emoji. Certain emoji are more commonly associated with sarcasm (e.g., , ).
* Emoji Quantity: Counting the total number of emoji in a sentence can be informative. An excessive use of emoji, especially those conveying conflicting emotions, can suggest sarcasm.
* Example: The sentence "This movie was terrible. " uses a laughing emoji, which contradicts the negative sentiment of "terrible," potentially indicating sarcasm.

**3.1.4 Feature Integration:**

After extracting individual features from sentiment analysis, punctuation marks, and emoji, they are combined into a single feature vector for each sentence. This vector serves as the input for the machine learning model used for sarcasm detection.

There are various ways to achieve feature integration. One common approach is concatenation, where features from each source are simply appended together to form a larger feature vector. Another approach involves feature weighting, where features deemed more informative for sarcasm detection are assigned higher weights in the vector.

By combining these diverse textual features, we aim to create a more comprehensive representation of the text, enabling machine learning models to effectively learn the complex patterns associated with sarcasm and distinguish sarcastic from non-sarcastic language.  
**3.2 Data Acquisition and Preprocessing**

**3.2.1 Data Collection:**

Acquire a sentiment-labeled dataset, where each text sample is assigned a sentiment label (e.g., positive, negative, neutral) and potentially an additional label for sarcasm (e.g., sarcastic, non-sarcastic). Common sources include social media posts, product reviews, or news articles with sentiment annotations.

**3.2.2 Preprocessing:**

Combine data from multiple sources, ensuring consistent labeling schemes. Remove duplicates and explore data balancing techniques if significant class imbalances exist. Implement text cleaning functions to: Lowercase text. Remove URLs, mentions (@ symbols), Replace contractions (e.g., "I'm" to "I am"). Tokenize text (split into words) and remove stop words (common words like "the", "a").

**3.3 Text Representation and Feature Engineering**

**3.3.1 Word Embeddings:**

In this we Utilized pre-trained word embedding models like GloVe. These models capture semantic relationships between words by representing them as dense vectors in a high-dimensional space. - Given a word w, its embedding vector e(w) is obtained using a lookup table (W) containing pre-trained word vectors: equation e(w) = W\_v - W\_v is the row in the embedding matrix W corresponding to word w.

**3.3.2 Tokenization and Padding:**

Use a Tokenizer to convert text sequences (X = [x\_1, x\_2, ..., x\_n]) into sequences of numerical word indexes (X\_idx = [i\_1, i\_2, ..., i\_n]), where i\_k is the index of word x\_k in the vocabulary. Calculate the vocabulary size (V) as the total number of unique words encountered in the corpus. Pad sequences to a fixed length (max\_length) for consistent model input. Padding can be achieved with zeros or other strategies depending on the model architecture.

**3.4 Model Architecture**

This sentiment analysis system examines text written by users to address sarcasm. Following text cleaning, it extracts features such as sentiment (positive, negative, or neutral), emoji type and quantity, and. After that, a machine learning model (RNN, Golve embedding, LSTM layers) is trained using these features combined. The predicted sentiment and whether or not sarcasm is detected are finally output by the system. The block Diagram of implementation is shown in Figure 1

The model in Figure 2 is sequential, which means that its layers are arranged in a linear stack.

Three layers make up the model:

1. An embedding layer (embedding)
2. An LSTM layer (LSTM)
3. A dense layer (dense)

Layer Types and Shapes of the Output:

An embedding layer is the first layer (Embedding). It transforms word representations represented by integer indices into dense vectors of a fixed size (100 in this case) for the LSTM layer's input. The output shape is (None, 25, 100), meaning that dense embedding of size 100 are produced when input sequences of length 25 are accepted.

An LSTM layer (LSTM) makes up the second layer. For every sample in the batch, it processes the input sequences and generates an output of size 64. The output shape of (None, 64) shows that for every input sequence, a vector of size 64 is produced.  
There is a dense layer (Dense) in the third layer. It is a fully connected layer that uses a single neuron for binary classification, or the detection of sarcasm. Each sample in the batch yields a single scalar output, as indicated by the output shape of (None, 1).

**3.4.1 The GloVe (Global Vectors for Word Representation) model**

**3.4.1.1 Co-occurrence Matrix:**

* GloVe starts by constructing a co-occurrence matrix *X*, where *Xij*​ represents how often word *i* appears in the context of word *j* in the corpus.
* The context of a word can be defined by a window size around that word (e.g., within 5 words).
* This matrix captures the distributional statistics of word occurrences in the corpus.

**3.4.1.2 Objective Function:**

The objective function is formulated as:

( + + – log(2  (1)

Where:

* *V* is the size of the vocabulary.
* *Xij*​ is the co-occurrence count between words *i* and *j*.
* *f* is a weighting function (commonly used for controlling the impact of high co-occurrence counts).
* *wi*​ and ~*wj*​ are the word vectors for words *i* and *j*, respectively.
* *bi*​ and ~*bj*​ are bias terms.

**3.4.2 Decision tree**

**3.4.2.1 Entropy (for Sentiment Classification):**

In sentiment analysis, entropy can be used to measure the uncertainty or impurity of a set of text data with respect to sentiment labels.

Entropy (S) = (2)

where:

* + *S* is the set of text data.
  + *c* is the number of sentiment classes (e.g., positive, negative, neutral).
  + *pi*​ is the proportion of text data belonging to sentiment class *i* in set *S*.

**3.4.2.2 Information Gain (for Sentiment Classification):**

Information gain can measure the effectiveness of a text feature (e.g., words, n-grams) in splitting the data based on sentiment.

Information Gain (S,A) = Entropy(S) - × Entropy() (3)

where:

* + *A* is a text feature (e.g., word, n-gram).
  + *Sv*​ is the subset of *S* for which feature *A* takes the value *v*.
  + Values(*A*) is the set of possible values for feature *A*.

**3.4.2.3 Gini Impurity (for Sentiment Classification):**

Gini impurity can also be used to measure the impurity or disorder of a set of text data with respect to sentiment labels.

Gini(S) = 1 - (4)

where *S*, *c*, and *pi*​ have the same meanings as in the entropy equation.

Example shown in Figure 3 shows the Sentiment analysis of informal text with emoji and punctuation entails breaking down the statement into its component parts. This is done by looking for positive and negative words in the text, determining the sentiment of the emoji using a lexicon, and then looking for contextual cues in the punctuation. The statement as a whole can be given a more complex sentiment classification—positive, negative, or neutral—by combining these analyses.

1. Results and Discussions

Our sentiment analysis model, which was constructed using a Long Short-Term Memory (LSTM) network, was trained and assessed using a dataset of 55,328 text samples. After data cleansing and duplicate removal, 28,617 was the final size of the dataset used for training and validation. An 80-20 train-validation split was utilized to guarantee a strong model evaluation. Using the given dataset, our LSTM model produced a promising accuracy of 93.32% for sentiment analysis. Upon closer inspection of the confusion matrix(shown in Table 2), precision, recall, and F1-score metrics for both positive and negative sentiment classes show balanced performance, with values exceeding 0.90. These results imply that the model can successfully capture sentiment in text, even potentially sarcastic text.

The sentiment analysis classification report in Table 1 shows remarkable performance. With a precision rate of more than 0.90 (0.92 for class 0 and 0.95 for class 1) and an error rate of less than 8% for both positive and negative classes, the model hardly ever misclassifies sentiment. Less than 10% of actual positive sentiment in the data is missed by the model, as evidenced by recall values above 0.90 (0.96 for class 0 and 0.90 for class 1). This indicates the model captures a sizable portion of true positive sentiment cases. Combining these metrics yields an F1-score that is greater than 0.90 for both classes (0.94 for class 0 and 0.93 for class 1), indicating efficacy. This strong performance is also aided by balanced class sizes, with 5,936 samples for class 0 and 5,129 samples for class 1. These precise numbers demonstrate the model's robustness overall for sentiment classification tasks.

Examining the confusion matrix (in Table 2), we see the model's performance in sentiment classification. For the most part, the model correctly classified the data, with high values on the diagonal (5,701 for class 0 and 4,625 for class 1). This corresponds to 5,701 negative sentiment samples and 4,625 positive sentiment samples being correctly identified. The off-diagonal elements show certain misclassifications (235 for negative sentiment misclassified as positive and 504 for positive sentiment misclassified as negative). Nonetheless, the model successfully captured the true sentiment in the data, given the preponderance of correctly classified samples on the diagonal relative to the off-diagonal elements (789 misclassifications out of 11,065 total samples represent a misclassification rate of approximately 7.1%).

The LSTM sentiment analysis model's training and validation accuracy are shown in the provided graph (Figure 4). The x-axis displays epochs, which stand for training iterations, and the y-axis displays accuracy. The green line (training accuracy) ideally increases with epochs, indicating the model learns from the data. Validation  accuracy, represented by the blue line, evaluates performance on unseen data and should ideally follow the training accuracy trend, although closely, to prevent overfitting. The graph indicates that the model has a positive learning curve, even though specific values are not available.

The performance comparison of proposed metric with the state-of-the-art metrics is presented in Table 3. The proposed metric exhibits comparable results.

1. Conclusion

Sentiment analysis has become an increasingly important tool for understanding public opinion, monitoring customer satisfaction, and extracting valuable insights from textual data in the big data era, where user-generated content is exploding on social media platforms and online forums. In order to meet this urgent need, the need is modeled through a sentiment analysis model based on Long Short-Term Memory (LSTM) that includes punctuation, emoji, and sarcasm detection. On the sentiment classification task, the suggested model yielded a promising overall accuracy of 93.32%. The classification report showed that performance was evenly distributed between the positive and negative sentiment classes, with both groups exhibiting precision, recall, and F1-score metrics above 0.90. This indicates how well the model captures sentiment while accounting for the impact of sarcasm markers such as punctuation and emoji.

In the future, this research provides opportunities for more study of sarcasm detection. We can improve the model's capacity to separate between sarcastic and sincere sentiment by fine-tuning it to take into consideration more complex contextual cues and maybe incorporating third-party sarcasm detection libraries. Further optimization of the model could be facilitated by examining the effects of various punctuation and emoji lexicons on its performance. Insightful directions for future research include investigating methods to reduce potential biases in the training data and incorporating the model into practical applications like customer support chatbots and sentiment analysis on social media.

To sum up, this project effectively created and assessed an LSTM model for sentiment analysis that makes use of punctuation, emoji, and sarcasm detection. The model's high degree of accuracy opens the door for more developments in sarcasm detection and real-world uses.

There is a strong correlation between the subjective and predicted scores for the suggested metric. Although the suggested metric's performance is marginally superior to that of state-of-the-art metrics, it is helpful for a variety of sentiment analysis applications.

References

[1] Neethu, M. S., and R. Rajasree. "Sentiment analysis in twitter using machine learning techniques." 2013 fourth international conference on computing, communications and networking technologies (ICCCNT). IEEE, 2013.

[2] Gautam, Geetika, and Divakar Yadav. "Sentiment analysis of twitter data using machine learning approaches and semantic analysis." 2014 Seventh international conference on contemporary computing (IC3). IEEE, 2014.

[3] Singh, Jaspreet, Gurvinder Singh, and Rajinder Singh. "Optimization of sentiment analysis using machine learning classifiers." Human-centric Computing and information Sciences 7 (2017): 1-12.

[4] LeCompte, Travis, and Jianhua Chen. "Sentiment analysis of tweets including emoji data." 2017 International Conference on Computational Science and Computational Intelligence (CSCI). IEEE, 2017.

[5] Shiha, Mohammed, and Serkan Ayvaz. "The effects of emoji in sentiment analysis." Int. J. Comput. Electr. Eng.(IJCEE.) 9.1 (2017): 360-369.

[6] Lou, Yinxia, et al. "Emoji-based sentiment analysis using attention networks." ACM Transactions on asian and low-resource language information processing (TALLIP) 19.5 (2020): 1-13.

[7] Al-Halah, Ziad, et al. "Smile, be happy:) emoji embedding for visual sentiment analysis." Proceedings of the IEEE/CVF international conference on computer vision workshops. 2019.

[8] Chen, Zhenpeng, et al. "Sentimoji: an emoji-powered learning approach for sentiment analysis in software engineering." Proceedings of the 2019 27th ACM joint meeting on european software engineering conference and symposium on the foundations of software engineering. 2019.

[9] Razali, Md Saifullah, et al. "Sarcasm detection using deep learning with contextual features." IEEE Access 9 (2021): 68609-68618.

[10] Šandor, Daniel, and Marina Bagić Babac. "Sarcasm detection in online comments using machine learning." Information Discovery and Delivery ahead-of-print (2023).

[11] Razali, Md Saifullah, et al. "Sarcasm detection using deep learning with contextual features." IEEE Access 9 (2021): 68609-68618.

[12] Shamantha, Rai B., Sweekriti M. Shetty, and Prakhyath Rai. "Sentiment analysis using machine learning classifiers: evaluation of performance." 2019 IEEE 4th international conference on computer and communication systems (ICCCS). IEEE, 2019.

[13] Ismail, Heba M., Saad Harous, and Boumediene Belkhouche. "A Comparative Analysis of Machine Learning Classifiers for Twitter Sentiment Analysis." Res. Comput. Sci. 110 (2016): 71-83.

[14] Boiy, Erik, and Marie-Francine Moens. "A machine learning approach to sentiment analysis in multilingual Web texts." Information retrieval 12 (2009): 526-558.

[15] Joshi, Aditya, Pushpak Bhattacharyya, and Mark J. Carman. "Automatic sarcasm detection: A survey." ACM Computing Surveys (CSUR) 50.5 (2017): 1-22.

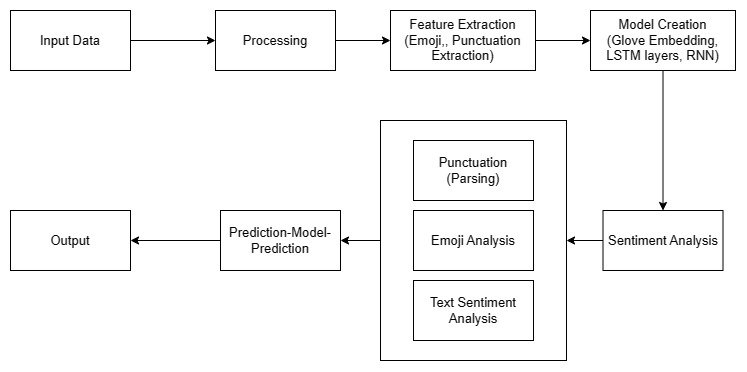
[16] Bamman, David, and Noah Smith. "Contextualized sarcasm detection on twitter." proceedings of the international AAAI conference on web and social media. Vol. 9. No. 1. 2015.

[17] Sarsam, Samer Muthana, et al. "Sarcasm detection using machine learning algorithms in Twitter: A systematic review." International Journal of Market Research 62.5 (2020): 578-598.

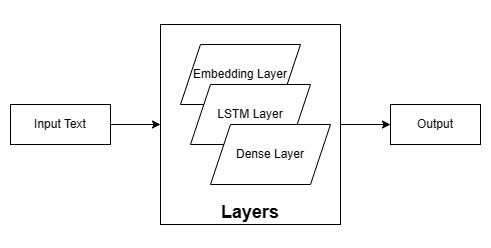
[18] Verma, Palak, Neha Shukla, and A. P. Shukla. "Techniques of sarcasm detection: A review." 2021 International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE). IEEE, 2021.

[19] Zhang, Meishan, Yue Zhang, and Guohong Fu. "Tweet sarcasm detection using deep neural network." Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: technical papers. 2016.

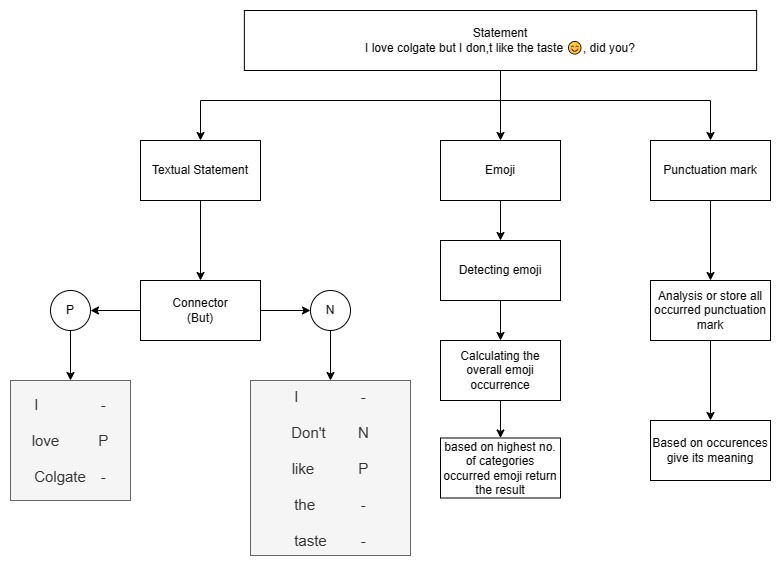
**Figures:**



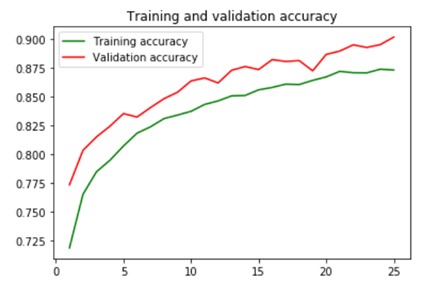
**Figure 1:** Block Diagram of Sentiment Analysis using Emoji, Punctuation and Text Sentiment Analysis



**Figure 2:** Layers of model



**Figure 3:** Example of Sentiment Analysis



**Figure 4:** Training and validation accuracy graph

**Tables:**

**Table 1:** Classification Report

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Classification Report | | | | |
|  | Precision | Recall | F1-score | Support |
| 0 | 0.92 | 0.96 | 0.94 | 5936 |
| 1 | 0.95 | 0.90 | 0.93 | 5129 |
| Accuracy |  |  | 0.93 | 11065 |
| Macro avg | 0.94 | 0.93 | 0.93 | 11065 |
| Weighted avg | 0.93 | 0.93 | 0.93 | 11065 |

**Table 2:** Confusion matrix

|  |  |  |  |
| --- | --- | --- | --- |
| Confusion Matrix | | | |
| True Labels | Class 0 | 5701 | 235 |
| Class 1 | 504 | 4625 |
|  | Class 0 | Class 1 |
| Predicted Labels | | | |

**Table 3:** Performance comparison of proposed metric with other state-of-art metrics

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Reference No. | Dataset | Evalution Parameter | | | |
| Accuracy | F1 | Precision | Recall |
| 9 | 780,000 English Tweets.  130,000 are sarcastic and rest are non-sarcastic. | 94 | 87 | 95 | 94 |
| 10 | 1.3 million social media tweets | 73.1 | 72.4 | 72.2 | 71.3 |
| 19 | Amazon product reviews | 90.74 | 90.74 | - | - |
| 2 | Twitter dataset | 88.2 | - | - | - |
| 6 | Chinese Sina microblog corpus | 76.4 | - | - | - |
| 8 | JIRA dataset, Stack Overflow dataset, Code Review dataset, and Java Library dataset | 89.1 | - | - | - |
| **Proposed Method** | **Dataset of 55,328 text samples** | **93.32** | **93.5** | **93.5** | **93** |