ISSN: 2583-9292 (ONLINE)
DOI: 10.xx/ijdsml.2024.xx

SENTIMENT ANALYSIS OF TWEETS USING DEEP LEARNING

M. Patel and R.V. Vikram
Department of Computer Science and Engineering, Rajarambapu Institute of Technology, India

Abstract

In the age of social media, Twitter has emerged as one of the largest platforms for expressing personal opinions and emotions. The vast volume of real-time data shared by users offers unique opportunities for analyzing public sentiment on various topics, including politics, entertainment, and social issues. However, extracting meaningful insights from such unstructured data presents significant challenges. Traditional sentiment analysis methods often struggle with the nuances of language, such as sarcasm, irony, and the context-dependent meaning of words. Thus, there is a need for more advanced techniques to improve the accuracy and efficiency of sentiment classification. This study proposes a novel approach to sentiment analysis of tweets using deep learning, specifically by leveraging the ResNet architecture. ResNet, a powerful deep convolutional neural network, has shown remarkable performance in image processing tasks and is adapted here for text-based sentiment analysis. The method involves preprocessing the tweet data, embedding it into word vectors, and passing it through a ResNet-based model. The model utilizes residual connections to overcome the vanishing gradient problem, enabling it to learn deeper representations of the tweet text and capture complex semantic features. The outcomes of this research demonstrate that the ResNet model outperforms traditional machine learning models, such as support vector machines (SVM) and logistic regression, in terms of both accuracy and efficiency. The proposed method achieves high classification performance, correctly categorizing tweets into positive, negative, or neutral sentiments. These results highlight the potential of deep learning techniques, particularly ResNet, in effectively analyzing social media content for sentiment detection, offering valuable insights for businesses, researchers, and decision-makers.

Keywords:

Sentiment Analysis, Twitter, Deep Learning, ResNet, Text Classification

1. INTRODUCTION

Social media platforms, especially Twitter, have become critical sources of real-time information, with millions of tweets generated daily on various topics ranging from politics to entertainment. Twitter's brevity, limited to 280 characters, makes it an ideal platform for expressing opinions, sentiments, and emotions. Given the massive volume of user-generated content, sentiment analysis has become a vital area of research for extracting valuable insights, such as identifying public opinion trends and making informed decisions in marketing, customer service, and public policy analysis. However, traditional sentiment analysis methods often struggle with accurately interpreting emotions in tweets, especially when faced with challenges such as sarcasm, slang, and context-dependent meanings. Deep learning models have shown promise in addressing these challenges by capturing complex semantic relationships in textual data [1-3].

Sentiment analysis of tweets presents several key challenges. One of the primary issues is the limited context available in short text, which makes it difficult to understand the underlying sentiment in isolation. The presence of irony, sarcasm, and informal language adds another layer of complexity. Traditional machine learning models rely on feature extraction techniques like bag-of-words or term frequency-inverse document frequency (TF-IDF), which may not fully capture the intricacies of natural language in tweets. Furthermore, tweets often contain emojis, abbreviations, hashtags, and other non-traditional elements that complicate analysis. These challenges highlight the need for more advanced models that can better process and understand textual nuances [4-6].

The problem addressed by this research is the effective sentiment classification of tweets using deep learning techniques, particularly leveraging the ResNet architecture. The task involves preprocessing tweets to extract meaningful features, followed by training a deep convolutional neural network (CNN) to classify the sentiments expressed in tweets as positive, negative, or neutral. While previous studies have applied various deep learning models such as recurrent neural networks (RNNs) and long short-term memory networks (LSTMs) for sentiment analysis, few have explored the potential of ResNet, which is traditionally used in image classification tasks. Moreover, the challenge remains to create a model that can handle the unique features of tweet data effectively [7-10].

This study aims to achieve the following objectives:

- To apply the ResNet deep learning architecture for sentiment analysis of tweets and evaluate its performance compared to traditional models.
- To develop a robust preprocessing pipeline that can effectively handle the unique features of tweet data, including slang, emojis, and hashtags.

The novelty of this work lies in the adaptation of the ResNet architecture, typically used for image processing, to perform sentiment analysis on short text data like tweets. By leveraging residual connections, ResNet can learn deeper representations of tweet data, overcoming the vanishing gradient problem and enabling more effective learning. This study contributes to the field of sentiment analysis by demonstrating that ResNet can be a highly effective tool for this task, offering a new approach that outperforms traditional machine learning techniques. The proposed method could have significant applications in real-time sentiment tracking, enhancing the ability to process and analyze vast amounts of social media data.

2. RELATED WORKS

Sentiment analysis has been widely researched, with various approaches proposed to classify and analyze sentiments in textual data. Early methods, including rule-based systems and machine learning algorithms like support vector machines (SVM) and Naive Bayes, were primarily designed to handle sentiment classification tasks by relying on manually designed features such as n-grams and lexicons. However, these approaches struggle with the complexities of natural language, especially in informal contexts such as social media [7-8].

Deep learning models, particularly recurrent neural networks (RNNs) and long short-term memory networks (LSTMs), have gained significant attention due to their ability to capture sequential dependencies in text. In sentiment analysis, RNNs have been employed to model the sequential nature of sentences and their impact on sentiment classification. LSTMs, a more advanced type of RNN, have demonstrated improved performance by mitigating the vanishing gradient problem, allowing them to retain long-term dependencies in text. However, LSTMs often face challenges with long sequences or noisy data, as is common in tweets [9].

Recent studies have shown the potential of convolutional neural networks (CNNs) in sentiment analysis tasks. While CNNs are traditionally used for image processing, their ability to capture local patterns and spatial hierarchies has made them suitable for text classification. Convolutional layers can identify meaningful n-grams in text, and pooling operations help in downsampling and focusing on the most relevant features. Several studies have utilized CNNs for sentiment analysis, but these models often rely on manual feature extraction methods or require large amounts of labeled data for training [10-11].

The ResNet architecture, initially proposed for image classification, introduces residual connections that allow gradients to flow more effectively through deep networks, mitigating issues like the vanishing gradient problem. The use of residual connections in deep neural networks has been shown to improve performance in various tasks, including text classification. In the context of sentiment analysis, ResNet's ability to handle deeper architectures with improved gradient flow has made it a promising candidate for text-based sentiment classification [12].

Research on applying ResNet to sentiment analysis is still emerging, with few studies focusing on its adaptation for social media data. However, the potential of ResNet for understanding the intricate relationships in tweet data remains largely unexplored. This work seeks to fill that gap by evaluating the use of ResNet for sentiment analysis of tweets and demonstrating its effectiveness compared to traditional models. By adapting ResNet to work with tweet data, this study offers a promising alternative to current methods and explores its potential to improve sentiment classification accuracy on short, noisy text [12].

In summary, while deep learning techniques such as RNNs, LSTMs, and CNNs have been widely used in sentiment analysis, the application of ResNet to tweet sentiment classification is a novel approach that has the potential to offer significant improvements in performance and reliability.

3. PROPOSED METHOD

The proposed method for sentiment analysis of tweets using ResNet involves several key steps, beginning with data preprocessing and followed by the application of the ResNet deep learning architecture for sentiment classification.

- **Data Preprocessing**: The first step is to preprocess the raw tweet data. This involves cleaning the tweets by removing irrelevant elements such as URLs, special characters, and user handles. Additionally, tokenization is performed to break the text into individual words or subwords, and stopwords are removed to focus on meaningful terms. The tweets are then converted into numerical representations using word embeddings, such as Word2Vec or GloVe, to capture semantic relationships between words.
- **Model Architecture**: The preprocessed tweet data is fed into a ResNet-based architecture designed for text classification. The ResNet model consists of several convolutional layers that extract local features from the tweet text. These layers use residual connections to allow the network to learn deeper representations of the input data while mitigating the vanishing gradient problem. This enables the model to capture more complex semantic patterns that are crucial for accurately classifying sentiment in short, noisy texts like tweets.
- **Training**: During the training phase, the ResNet model is optimized using a loss function, such as categorical cross-entropy, that measures the discrepancy between predicted and actual sentiment labels. The model is trained on a labeled dataset of tweets, where each tweet is assigned a sentiment label (positive, negative, or neutral). The training process involves adjusting the model's weights using backpropagation and gradient descent to minimize the loss and improve the model's classification accuracy.
- **Sentiment Classification**: Once trained, the ResNet model is capable of classifying new, unseen tweets based on their sentiment. Each input tweet is passed through the trained model, which processes the text through the convolutional layers, extracting relevant features, and ultimately outputs a predicted sentiment label (positive, negative, or neutral).

3.1 DATA PREPROCESSING

The data preprocessing stage is crucial for transforming raw tweet data into a format suitable for deep learning models. Since tweets are typically noisy, containing a variety of informal language, emojis, hashtags, and URLs, preprocessing ensures that only the relevant information is retained for analysis. Below are the primary steps involved:

- **Cleaning and Tokenization**: The first step involves cleaning the raw tweet text by removing unnecessary elements such as URLs, special characters, and user handles. This is important as these elements do not contribute to sentiment analysis. For instance, in the tweet "I love this new phone! #tech @user #excited," the URL and user handle "@user" are removed. The cleaned tweet would look like: "I love this new phone! #tech #excited." After cleaning, the text is tokenized, meaning it is split into smaller components such as words or subwords. In this case, the tokenization of "I love this new phone" would result in the tokens: ["I", "love", "this", "new", "phone"].
- **Removing Stopwords**: Stopwords (such as "the," "is," "and," etc.) are commonly removed because they appear frequently across different tweets and do not carry significant meaning for sentiment classification. For instance, "I love this new phone!" would become ["love", "new", "phone"] after removing stopwords like "I" and "this."

- **Text Vectorization (Word Embedding)**: Since deep learning models require numerical input, the next step is converting words into vector representations. This is done using pre-trained word embeddings such as Word2Vec, GloVe, or FastText. These embeddings capture semantic relationships between words, ensuring that words with similar meanings (e.g., "happy" and "joyful") have similar vector representations. For example, the word "love" might be represented as a 300-dimensional vector like: [0.21, 0.34, -0.50, ..., 0.67]. The entire cleaned and tokenized tweet would be transformed into a sequence of such vectors. These vectors form the input to the model.

Table.1. Preprocessed Dataset

Original Tweet	Cleaned Tweet	Tokenized Tweet	Stopwords Removed	Word Embeddings (Vectorized)
I love this new phone! #tech @user #excited	I love this new phone! #tech #excited	["I", "love", "this", "new", "phone"]	["love", "new", "phone"]	["love": [0.21, 0.34, -0.50, ..., 0.67], "new": [...], "phone": [...]]

The resulting preprocessed data is now ready to be fed into the deep learning model.

4. MODEL ARCHITECTURE

The core of the proposed method involves using a ResNet-based architecture for tweet sentiment classification. ResNet, originally designed for image recognition, is adapted here for processing text. The key feature of ResNet is its use of residual connections, which help the model train deeper networks without suffering from the vanishing gradient problem. Below is an explanation of how the architecture works:

- **Input Layer (Word Embeddings)**: After preprocessing, the tweet text is transformed into a sequence of word embeddings. These embeddings represent the semantic meaning of each word in the tweet and serve as the input to the ResNet model. Each tweet, represented as a series of word vectors, is passed through an embedding layer, which turns the vectorized words into a fixed-size representation.
- **Residual Blocks**: ResNet is built using multiple residual blocks. A residual block consists of a series of convolutional layers, followed by a skip connection (also known as the residual connection). The skip connection adds the input of the block to the output, allowing the model to bypass certain layers if necessary. This architecture enables the model to train deeper networks by alleviating the vanishing gradient problem, where gradients become very small and hinder learning in deep networks. Each convolutional layer applies filters to the input data to detect important features (e.g., n-grams, sentiment-bearing words) in the text.

In the case of tweet sentiment analysis, these layers might detect patterns such as the presence of positive words like "love" or "great," or negative words like "hate" or "terrible." The residual connections allow the model to retain crucial information from earlier layers, ensuring that even deeper layers can capture more complex semantic patterns without losing important features.

- **Fully Connected Layer**: After passing through several residual blocks, the output of the convolutional layers is flattened and passed through a fully connected layer. This layer acts as a classifier, producing a vector of predictions. For sentiment analysis, this vector typically consists of three output values corresponding to the three sentiment categories: positive, negative, and neutral.
- **Softmax Activation**: The final output layer uses the softmax activation function to produce the probability distribution over the sentiment classes. This function converts the raw output values into probabilities, where the highest probability corresponds to the predicted sentiment class. For example, if the output probabilities are [0.1, 0.8, 0.1], the model predicts the sentiment as "negative" with 80% confidence.
- **Training and Optimization**: The ResNet model is trained using a loss function such as categorical cross-entropy, which calculates the difference between the predicted sentiment and the actual sentiment label. The model is optimized using backpropagation and gradient descent, adjusting the weights in the network to minimize the loss and improve classification accuracy.

By leveraging residual connections, the ResNet architecture allows the model to learn deeper representations of tweet data, overcoming challenges like noisy and sparse text data, and improving sentiment classification accuracy.

4.1 TRAINING

The training phase is a critical part of the model development process, as it enables the ResNet model to learn the patterns and features associated with sentiment in tweet data. The goal of training is to adjust the model's parameters (weights) to minimize the difference between the predicted sentiment and the actual sentiment labels, thus improving the model's accuracy over time. The training process involves several key steps:

- **Loss Function Calculation**: During training, the model's predictions are compared with the actual sentiment labels using a loss function. In the case of sentiment analysis with multiple classes (e.g., positive, negative, neutral), a common loss function is categorical cross-entropy. This function calculates the difference between the predicted probability distribution (output of the softmax function) and the actual one-hot encoded label. For example, if the actual label is "positive" (represented as [1, 0, 0]) and the model predicts the probabilities as [0.1, 0.8, 0.1], the loss would reflect the deviation between these values.
- **Backpropagation and Gradient Descent**: Once the loss is computed, backpropagation is used to calculate the gradients of the loss function with respect to each weight in the model. These gradients indicate how much each weight contributed to the error in the prediction. The gradients are then used to update the weights in the direction that minimizes the loss. This weight adjustment is done using an optimization algorithm such as stochastic gradient descent (SGD) or Adam. Adam, in particular, adapts the learning rate for each

weight, which helps speed up convergence and improve training stability.

- **Epochs and Batching**: The training data is divided into batches, and the model is trained for several epochs. An epoch is one complete pass through the entire training dataset. For each batch, the model makes predictions, computes the loss, performs backpropagation, and updates the weights. Multiple epochs allow the model to learn the patterns in the data more effectively, as each epoch provides the model with more opportunities to refine its understanding of sentiment.
- **Evaluation During Training**: After each epoch or at certain intervals, the model's performance is evaluated on a validation set that was not seen during training. This helps ensure that the model is generalizing well to new data and not overfitting to the training set. Common evaluation metrics for sentiment analysis include accuracy, precision, recall, and F1-score. These metrics provide insights into how well the model is performing across different sentiment classes.
- **Early Stopping**: To prevent overfitting, the training process may incorporate early stopping, which involves halting the training if the validation performance stops improving after a specified number of epochs. This helps save time and prevents the model from learning noise in the data rather than the underlying patterns.

Table.2. Results of Training

Epoch	Train	Valid	Precision	Recall	F1	Precision	Recall	F1
	Accuracy		Positive			Negative		
1	0.72	0.70	0.68	0.75	0.71	0.73	0.70	0.71
2	0.75	0.73	0.70	0.77	0.73	0.75	0.72	0.73
3	0.78	0.75	0.72	0.79	0.75	0.76	0.74	0.75
4	0.80	0.77	0.75	0.80	0.77	0.78	0.75	0.77
5	0.82	0.79	0.77	0.81	0.79	0.79	0.77	0.78

Explanation of Table Columns:

- **Epoch**: Represents the number of passes through the entire training dataset.
- **Training Accuracy**: The accuracy of the model on the training dataset after each epoch.
- **Validation Accuracy**: The accuracy of the model on the validation dataset after each epoch, showing how well the model generalizes.
- **Precision, Recall, F1-Score**: These metrics are provided separately for both positive and negative sentiments. Precision indicates how many of the predicted positive (or negative) sentiments were actually correct. Recall indicates how many of the true positive (or negative) sentiments were correctly identified by the model. F1-Score is the harmonic mean of precision and recall, offering a balance between the two.

As the model progresses through epochs, the training accuracy increases, and the validation accuracy typically improves as well, although there might be slight fluctuations. If the validation accuracy plateaus or starts decreasing, this could be an indicator that the model is overfitting to the training data and should stop training.

4.2 SENTIMENT CLASSIFICATION

The sentiment classification stage is the final step in the proposed method, where the trained ResNet model is used to predict the sentiment of unseen tweet data. This phase involves applying the learned patterns from the training process to new input data and classifying each tweet into one of the sentiment categories: positive, negative, or neutral. The model makes predictions based on the features extracted during training and the patterns it has learned.

The working of sentiment classification can be broken down into the following steps:

- **Input Data (Preprocessed Tweets)**: The input to the sentiment classification stage consists of tweets that have undergone the same preprocessing steps as the training data. These tweets are cleaned, tokenized, and converted into word embeddings, which represent the semantic meaning of the words in the tweet. The embeddings are then passed through the trained ResNet model.
- **Prediction**: The ResNet model processes the word embeddings through its layers (including the residual blocks) and outputs a probability distribution across the three sentiment categories (positive, negative, and neutral). This is done using the softmax activation function in the final layer, which converts the raw output scores into probabilities. For instance, for a given tweet, the output might be a vector like [0.1, 0.8, 0.1], indicating that the model predicts a "negative" sentiment with an 80% probability.
- **Classification Decision**: After obtaining the probability distribution from the softmax layer, the model selects the class with the highest probability. In the example above, the sentiment would be classified as "negative" since it has the highest probability (0.8). This classification result is then outputted as the final sentiment prediction for the tweet.
- **Output and Post-processing**: After the classification is completed, the results are typically displayed in a user-friendly format, such as a table or a graph, that shows the predicted sentiment for each tweet along with associated probabilities. This step may also include generating a report or visualizing the overall sentiment distribution (e.g., the percentage of positive, negative, and neutral sentiments across a large dataset of tweets).

Table.3. Results of Classification

Tweet	Actual Sentiment	Predicted Sentiment	Probability (Positive)	Probability (Negative)	Probability (Neutral)
"I love this new phone! #tech"	Positive	Positive	0.85	0.10	0.05
"This movie was terrible and boring!"	Negative	Negative	0.05	0.90	0.05
"The weather is okay, not great, but fine"	Neutral	Neutral	0.15	0.20	0.65
"Best pizza ever, so delicious!"	Positive	Positive	0.80	0.15	0.05

"I hate waiting in long lines"	Negative	Negative	0.05	0.88	0.07

- For the tweet "I love this new phone! #tech", the model predicts a positive sentiment with 85% confidence, which aligns with the actual sentiment, making it a correct prediction.
- For "This movie was terrible and boring!", the model correctly classifies it as negative with 90% confidence.
- The tweet "The weather is okay, not great, but fine" is predicted as neutral with 65% confidence, reflecting the ambiguous nature of the sentiment.
- "Best pizza ever, so delicious!" is classified as positive with 80% confidence, matching the sentiment accurately.
- "I hate waiting in long lines" is correctly classified as negative with 88% confidence, again making it a correct prediction.

These results show that the model performs well in distinguishing between positive, negative, and neutral sentiments, with high confidence in its predictions. The model's accuracy can be further assessed by calculating metrics such as overall accuracy, precision, recall, and F1-score based on the confusion matrix derived from these results.

Results and Discussion

In this experiment, a deep learning-based sentiment analysis model using ResNet architecture is implemented to classify the sentiment of tweets. The simulation tool used for model training and evaluation is TensorFlow with Keras API for building and training the deep learning model.

The model is compared with three existing sentiment analysis methods to assess its performance:

- **Support Vector Machine (SVM)**: A classical machine learning approach often used for text classification tasks. It works by finding the optimal hyperplane that separates data points from different classes (sentiment categories in this case).
- **Convolutional Neural Network (CNN)**: A popular architecture for text classification that uses convolutional layers to extract features from the input data. CNNs are often effective at capturing local patterns in text data.
- **Long Short-Term Memory (LSTM)**: A type of recurrent neural network (RNN) that is well-suited for sequence prediction tasks like sentiment analysis. LSTM models can capture long-term dependencies in text sequences, making them effective for sentiment classification.

Table.4. Experimental Setup/Parameters

Parameter	Value
Dataset	Tweets Dataset (20,000 tweets)
Training Data Split	80% Training, 20% Testing
Preprocessing Method	Tokenization, Lowercasing, Stop-word Removal, Lemmatization
Word Embedding	GloVe 100-dimensional embeddings
Maximum Sequence Length	100 tokens per tweet
Batch Size	64
Number of Epochs	10
Optimizer	Adam (learning rate = 0.001)
Loss Function	Categorical Cross-Entropy
Activation Function	ReLU (for hidden layers), Softmax (for output)
Dropout Rate	0.5

Table.5. Performance Evaluation on various Epochs

Epoch	Method	Accuracy	Precision	Recall	F1	AUC
2	Proposed ResNet	78.5	76.4	80.1	78.2	0.82
	SVM	70.2	68.4	72.5	70.3	0.75
	CNN	73.4	71.8	75.2	73.5	0.77
	LSTM	75.1	73.6	77.1	75.3	0.78
4	Proposed ResNet	80.2	78.1	82.3	80.1	0.84
	SVM	72.5	71.2	74.3	72.7	0.76
	CNN	76.1	74.2	77.8	75.9	0.79
	LSTM	78.3	76.5	79.9	78.1	0.80
6	Proposed ResNet	82.4	80.3	84.2	82.2	0.86
	SVM	73.8	72.6	75.6	74.1	0.77
	CNN	77.5	75.3	78.4	76.8	0.80
	LSTM	80.1	78.7	81.3	79.9	0.82
8	Proposed ResNet	84.1	82.2	86.4	84.2	0.88
	SVM	74.3	73.0	76.5	74.7	0.78
	CNN	79.2	77.1	80.6	78.8	0.81
	LSTM	81.5	79.8	82.1	80.9	0.83
10	Proposed ResNet	86.3	84.1	88.2	86.0	0.90
	SVM	75.4	74.1	77.0	75.5	0.79
	CNN	80.3	78.2	81.4	79.7	0.82
	LSTM	83.2	81.4	84.5	82.9	0.85

The proposed ResNet model consistently outperforms existing methods (SVM, CNN, and LSTM) across all performance metrics. By epoch 10, the proposed method achieves an accuracy of 86.3%, which is higher than SVM (75.4%), CNN (80.3%), and LSTM (83.2%). The precision and recall for ResNet are also superior, reflecting better performance in both identifying positive tweets and minimizing false positives. The F1-score for ResNet reaches 86.0%, demonstrating a strong balance between precision and recall. Finally, AUC for ResNet is 0.90, indicating excellent model discriminability, compared to 0.79-0.85 for the other methods.

Table.6. Performance Evaluation on Training and Testing Split

Method	Dataset	Accuracy	Precision	Recall	F1	AUC
Proposed ResNet	Train	88.7	85.4	90.1	87.7	0.91
	Test	86.3	84.1	88.2	86.0	0.90

SVM	Train	80.1	77.8	82.5	79.9	0.80
	Test	75.4	74.1	77.0	75.5	0.79
CNN	Train	83.4	80.2	85.3	82.7	0.84
	Test	80.3	78.2	81.4	79.7	0.82
LSTM	Train	85.6	82.9	88.3	85.5	0.87
	Test	83.2	81.4	84.5	82.9	0.85

The proposed ResNet method outperforms existing methods on both training and testing datasets. On the training set, ResNet achieves 88.7% accuracy, 85.4% precision, 90.1% recall, 87.7% F1-score, and 0.91 AUC, which is significantly higher than the SVM, CNN, and LSTM models. On the test set, ResNet still leads with 86.3% accuracy and 0.90 AUC, showcasing its robust generalization capability. In contrast, SVM, CNN, and LSTM have lower metrics, particularly on the test set, indicating a higher risk of overfitting or suboptimal performance compared to the proposed model.

5. CONCLUSION

In this study, the proposed sentiment analysis model using ResNet demonstrated superior performance compared to traditional methods like SVM, CNN, and LSTM. Through comprehensive evaluation across training and testing datasets, the ResNet model consistently achieved higher accuracy, precision, recall, F1-score, and AUC, indicating its effectiveness in capturing complex patterns in tweet data for sentiment classification. The ResNet’s ability to generalize well across unseen data was evidenced by its consistent test set performance, where it outperformed the existing methods, particularly in terms of recall and AUC. The results emphasize the advantages of leveraging deep learning, particularly ResNet, in sentiment analysis tasks, as it offers robust feature extraction and learning capabilities that traditional machine learning models struggle to match. Additionally, the experiment highlighted the significance of choosing the appropriate model architecture to balance between high training performance and generalization. Thus, the findings suggest that the proposed method can be effectively applied to real-world sentiment analysis tasks, particularly in the dynamic domain of social media, where accurate and efficient sentiment detection is critical. Future work may involve further optimization of the model and testing on diverse datasets to validate its scalability and adaptability across various applications.

REFERENCES

[1] Mondher Bouazizi and Tomoaki Ohtsuki, “Sarcasm Detection in Twitter”, *Proceedings of International Conference in Global Communications*, pp. 331-334, 2015.

[2] E. Riloff, A. Qadir, P. Surve, L. De Silva, N. Gilbert and R. Huang, “Sarcasm as Contrast between a Positive Sentiment and Negative Situation”, *Proceedings of Conference on Empirical Methods in Natural Language Processing*, pp 704-714, 2013.

[3] P. Reyes and T. Veale, “A Multidimensional Approach for Detecting Irony in Twitter”, *Proceedings of International Conference on Language Resources and Evaluation*, pp. 1-30, 2012.

[4] I. Habernal and J. Hong, “Sarcasm Detection on Czech and English Twitter”, *Proceedings of International Conference on Computational Linguistics*, pp. 121-127, 2014.

[5] Kim Schouten and Flavius Frasincar, “Survey on AspectLevel Sentiment Analysis”, *IEEE Transactions on Knowledge and Data*, Vol. 28, No. 3, pp. 813-830, 2015.

[6] Lillian Lee and Shivakumar Vaithyanathan, “Thumbs Up? Sentiment Classification using Machine Learning Techniques”, *Proceedings of International Conference on Empirical Methods in Natural Language Processing*, pp. 79-86, 2002.

[7] Bo Pang and Lillian Lee, “Opinion Mining and Sentiment Analysis”, *Computational Linguistics*, Vol. 35, No. 2, pp. 311-312, 2009.

[8] Jenifer Jothi Mary, S. Santiago and L. Arockiam, “A Methodological Framework to Identify the Students Opinion using Aspect based Sentiment Analysis”, *International Journal of Engineering Research and Technology*, Vol. 5, No. 2, pp. 642-645, 2018.

[9] Ioannis Dematis, Eirini Karapistoli and Athena Vakali, “Fake Review Detection via Exploitation of Spam Indicators and Reviewer Behavior Characteristics”, *Proceedings of International Conference on Current Trends in Theory and Practice of Informatics*, pp. 581-595, 2017.

[10] E.E. Luneva, P.I. Banokin, A.A. Yefremov and T. Tiropanis, “Method of Evaluation of Social Network User Sentiments Based on Fuzzy Logic”, *International Journal of Key Engineering Materials*, Vol. 685, pp. 847-851, 2016.

[11] V. Reshma and Ansamma John, “Aspect Based Summarization of Reviews Using Naive Bayesian Classifier and Fuzzy Logic”, *Proceedings of IEEE International Conference on Control, Communication and Computing*, pp. 617-621, 2015.

[12] B.N. Supriya, V. Kallimani, S. Prakash and C.B. Akki, “Twitter Sentiment Analysis Using Binary Classification Technique”, *Proceedings of International Conference on Nature of Computation and Communication*, pp. 391-396, 2016.