

Ensemble Deep Learning Model to Evaluate Sentiments in Social media posts

Sapana Nikam¹ and Nisha Wandile Kimmatkar²

¹⁻²J.S.P. M's Rajarshi Shahu College of Engineering, Pune, India

Email: {sapanadeshmukh19,nishakimmatkar}@gmail.com

Abstract—The exponential rise in social media users makes it more difficult to comprehend what other netizens think about a certain topic. Therefore, sentiment analysis is essential for assessing the same through the use of cutting-edge technologies such as deep learning models, machine learning, and natural language processing (NLP). Numerous deep learning models are currently in use to examine the sentiments present in various social media datasets. For a thorough examination of sentiments, natural language processing and a single neural network are insufficiently powerful. Therefore, the suggested model uses the sentiment analysis ensembling learning mechanism to combine the two neural networks, which ultimately serves as a catalyst for the process of generating appropriate decisions in sentiment analysis. The proposed design combines two neural networks—a long short term memory (LSTM) neural network and a convolution neural network (CNN)—to train on data gathered from publicly accessible repositories, including posts from Facebook, Instagram, and X (formerly Twitter). To improve the sentiment analysis process utilizing deep learning models, the training data from the ensembled model is fed into the random forest classification model.

Index Terms— Sentiment analysis, Deep learning model, Convolution neural network, Long short term memory, Random Forest classification.

I. INTRODUCTION

By utilizing machine learning and deep learning algorithms, sentiment analysis can automatically mine unstructured data (such as social media, emails, and customer service tickets) for opinions and emotions. Machine learning has progressed to a new level with the advent of deep learning (DL). Sentiment analysis involves utilizing text analysis tools to categorize data according to positive, negative, or neutral emotions. We may train sentiment analysis models to read for context, sarcasm, etc., and comprehend the true mood and feeling of the data by harnessing the power of deep learning. This allows us to go beyond simple definitions of text. Just looking at the definitions of the words in the tweet wouldn't tell us much. A properly trained sentiment analyzer, however, would recognize this tweet for what it is.

When integrated with deep learning models, sentiment analysis technologies can unleash their full potential. Deep learning, a subfield of machine learning, mimics the way the human brain processes data by employing "artificial neural networks" (ANNs). Deep learning, a kind of hierarchical machine learning, allows us to accurately process large datasets with minimal human intervention by utilizing a cascade of algorithms to solve complicated issues.

At times, the terms deep learning and machine learning are thought to mean the same thing. As a more

sophisticated form of machine learning, deep learning is akin to it. It takes human intervention to fix basic machine learning's mistakes, alter the model's output, and "force" it to learn. On the other hand, deep learning allows neural networks to self-correct using its sophisticated algorithmic staircase. The good news is that after they master the basics, they'll be able to tackle more complex languages and maths on their own. When they have mastered the art of self-education, machine learning models are capable of extraordinary things. As an example, text analysis makes use of natural language processing (NLP) to deconstruct language and comprehend it in a human-like manner, identifying subjects, verbs, objects, etc. The computer can't go on to other analytical procedures until it has mathematically deconstructed a statement.

Of course, there's a lot more to it than just breaking down sentences into their component parts (subject, verb, and object) and going on. It has taken a long time to train successful NLP models. Nonetheless, deep learning models can parse entire texts, paragraphs, and even sentences into its component opinions with the help of natural language processing: Opinion Unit, Functionality, and Support are just a few examples of how the model may use subject categorization to arrange each statement into predetermined categories once it was broken down into opinion units. Then, the deep learning model may go through each statement and do topic-based sentiment analysis, such as "like the new update." "seems really slow" is a positive comment. "Can't get tech support on the phone" is a negative statement. At this point, our topic categories have undergone sentiment analysis of sectors Usability: Positive, Functionality: Negative and Support: Negative

Part 2 of this research study is devoted to reviewing the literature and conducting an in-depth analysis of prior efforts. While Part 3 delves into the details of the deployed steps, Part 4 presents the system's evaluation results, and Part 5 wraps up the paper with some suggestions for further research.

II. LITERATURE REVIEW

It is described by Jiangtao He et al. [1] As a granular problem in natural language processing, Aspect-Based Sentiment Analysis (ABSA) seeks to forecast the sentiment polarity of several features in a phrase or text. The majority of previous research has concentrated on how aspect sentiment polarity correlates with regional circumstances. There has been a lack of focus on the significant deep relationships between aspect sentiment polarity and global context. In [2] The team led by Yassin S. Mehanna et al. lays out the text may convey its sentiments either subtly or overtly. So, finding hidden sentiments is the biggest problem with current sentiment analysis (SA) methods. Other common problems include incorrectly classifying opinion words, failing to take context into account, and mishandling short texts due to poor interpretation or insufficient data for analysis.

[3] Kim Cheong-Ghil et al. To find out if harsh language is considered "noise data" in sentiment analysis of Korean data, this paper presents the results of a study. Keras, TensorFlow's high-level deep learning API, was used to test the model's accuracy. After cutting out the profanity, the LSTM model performed emotion analysis and exhibited an accuracy of around 81%; this was found to be a 2% drop from the 83% accuracy achieved by the model trained without cutting out the profanity. In their description, Qian Yang et al. [4] In order to more precisely mine the sentiment polarity of users towards various aspects, aspect-based sentiment analysis (ABSA) seeks to determine the sentiment polarity of particular aspects in sentences. In most cases, the emotion features of individual aspects are derived by interactive learning of the connections between various context aspects. However, none of the aforementioned studies have attempted to enhance Graph Convolutional Networks' (GCNs) capacity to interactively capture sentiment relationships of inter-aspect words in various contexts by incorporating external affective commonsense information. Existing research also gives less weight to studies with a Chinese focus than ABSA studies conducted in English.

[5] Researchers Waqar Ali et al. present An attractive future development has been the amalgamation of AI and SIoT, which has coincided with the fast rise of AI technology. An improvement in communication and the ability for SIoT object peers to make autonomous decisions might be achieved by reviewing data based on user opinions. The author of this piece suggests a deep learning approach to entity sentiment prediction. The author's model employed a Bi-GRU architecture that included the GloVe word embedding layer to produce the feature vectors for words in opinion sentences.[6] Analyzed by Yanrong Zhang et al. In order to enhance product design and marketing strategy optimization, e-commerce platforms greatly benefit from cross-domain fine-grained sentiment analysis since it allows businesses to better understand customers' genuine feelings and reactions to items or services. Using features of both generative adversarial networks and capsule networks, this research suggests a model for fine-grained sentiment analysis of cross-domain e-commerce texts, SKEP_Gram-CDNN.

[7] The fast expansion of radio broadcasting services has generated a mountain of audio data that can shed light on public sentiment and opinion, according to Naman Dhariwal et al. In order to create a more effective real-time sentiment analysis model, this research expands the scope of sentiment analysis to audio data and proposes

a computational approach called bifurcate and mix. This approach uses advanced natural language processing techniques from audio sentiment analysis tools like Vokaturi, transcription services like AssemblyAI, and text sentiment analysis lexicons like VADER to extract and categorize sentiments from the audio data. The study's findings show trends and patterns in the opinions voiced on All India Radio's 'Akashwani,' a radio program run by the News Service Division. [8] According to Bashar M. A. Tahayna et al., people express themselves through the use of specific terms and phrases when interacting with social media. On the other hand, some people utilize indirect and implicit idioms and proverbs to create a bigger impact or to grab the audience's attention using humorous, caustic, or metaphorical language.

The authors in [9] Narisa Zhao et al. talks about a lot of focus on studying internet reviews, and ABSA has long been considered a crucial task in the field of natural language processing. Hence, investigating reviews with varying degrees of positive and negative emotion is crucial. Thus, the Author suggests a CNN-GRU hybrid deep neural network model for aspect-based sentiment analysis. In [10] V. Diviya Prabha et al. This study compiles thoughts on the COVID-19 vaccine from Indian tweets sent during the shutdown. Across all the datasets, public sentiment is neutral. To dispel this misconception, these results call for more public and governmental pressure on the COVID-19 vaccine. Using six different Twitter datasets, experimental results show that a new CamCapsNet delivers good classification accuracy. This work sheds light on people's vaccination opinions and emphasizes the need to take action to change this neutral mood. This is according to the work of Byung-Won On et al. [11] Building a sentiment dictionary utilizing current web crawling methods is the main emphasis of this effort. Using this approach results in a sentiment dictionary that is laborious to construct and inefficient in terms of database space utilization. The author first suggests a new sentiment aware task to supplement the current web crawling approach in order to address these issues. The work involves finding emotion terms in a sentiment dictionary that match the words found in the crawled web page. The authors [12] Mohd Norhisham Razali et al. of the article discuss Discovering consumer preferences, thoughts, and feedback, sentiment analysis is a potent tool in culinary tourism. Businesses and destinations can use these insights to make data-driven decisions that enhance the tourist experience based on lexicons, emotional analysis, and sentiment analysis based on machine learning.

III. PROPOSED APPROACH

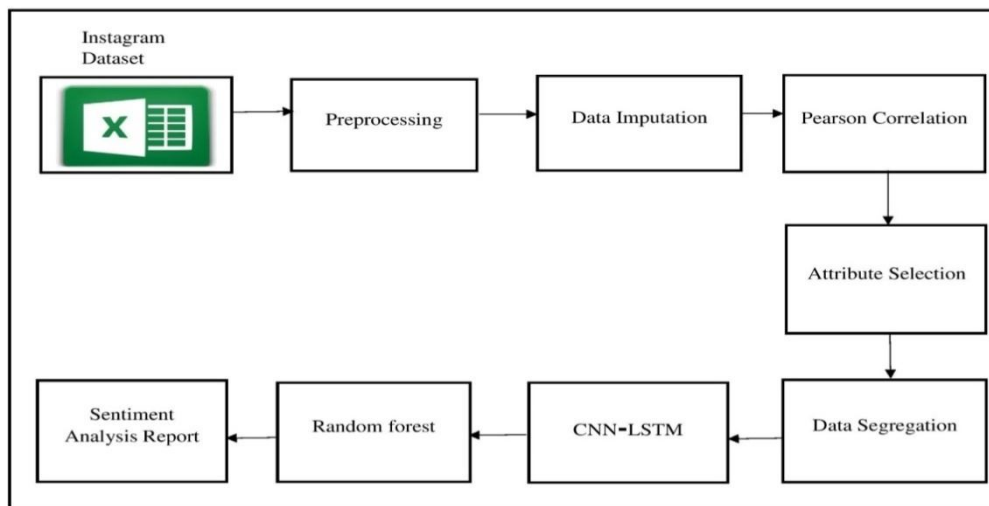


Figure 1. Overview of the proposed approach

In Figure 1 above, the proposed system for Sentiment analysis in social media posts. A series of actions are followed as mentioned below in order to deploy the model.

Phase 1: Dataset Collection and Preprocessing- For the process of deploying the model, a dataset has been downloaded from the URL <https://www.kaggle.com/code/muhammadyasirsaleem/sentiment-analysis-of-instagram-text-classificaton>. This dataset is basically a collection of posts from Instagram on wine, which consists of some 14 attributes like Sr no., country, description, designation, points, price, province, region_1, region_2, title, variety, and winery. This dataset is collected in the form of a workbook sheet that is then fed to the system by providing its path. The Pandas library of the Python programming language has been used to read

the dataset in the form of a double-dimensional list. From this double-dimension list, the “description” attributes contain the major textual part that indicates the posts of the user on the specific wine. So the system needs to evaluate the sentiment on the basis of this “description” attributes.

The obtained text from the ‘description’ attribute is subjected to preprocessing process to eliminate the redundant data and make the text lighter based on the following three steps.

Special Symbol removing - The system will remove any special symbols from the string. Incorporating grammatical errors and other nuances into written language is the goal of using certain symbols. This portion of our strategy is superfluous since removing it would have no negative effect on the post.

Stemming- "Stemming" is the next step in the pre-processing methodology, and it requires taking the de-stripped post as an input. Here we focus on lightening the load on the system, which drastically cuts down on processing time. Words in English are mostly defined by their suffixes, although many of them have a similar etymological root. Stemming, which reduces the words to their base form while preserving their semantic content, is then applied to the previously given terms. The phrase "sleeping" will be shortened to "sleep," for example, so that the post is shorter without losing any of its semantic significance.

Stop word Removal- At this point in the pre-processing process, which takes the stemmed post as input, stop words are extracted. Stop words are words used in English to bring together two separate assertions or parts of the same statement that are otherwise unrelated. Eliminating the stop words does not compromise the reader's comprehension in any kind, and our method does not require this. So, to achieve the pre-processed post, which is sent to the next phase for more extensive treatment, words like and, is, and others like them are eliminated.

Phase 2: Pearson Correlation – In order to calculate the Pearson correlation coefficient—a linear link between two variables—numerical codes for each variable's category are required. In order to assess this correlation, we estimate the correlation for each attribute with its own set of values, such as x and y. The technique is able to determine which attributes are most strongly correlated using this Pearson correlation, which takes into account correlation coefficients between 0 and 1. Stringent correlation is denoted by any number closer to 1, while loose correlation is denoted by 0. Important linked attributes are chosen for further estimation of the sentiments based on this. The Pearson correlation is calculated using Equation 1, which is shown below.

$$r = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum(x_i - \bar{x})^2 \sum(y_i - \bar{y})^2}} \text{----- (1)}$$

Where, x_i =values of x (independent) variable, y_i = values of y (Dependent) variable, $\bar{x}$ = mean of x variable values and $\bar{y}$ = mean of y variable values

Phase 3: Attribute Selection and Data Segregation- After selecting the best attributes based on pearson correlation system is set to train with the net step of the process. By using the `train_test_split()` method of Sklearn library, we can divide our data into two sets: one for training and one for testing. First, we must sort our information by characteristics (X) and then by labels (y). The four parts of the data are y_train, y_test, X_train, and X_test. With the help of the X_train and y_train sets, the model is trained and fitted. Using the X_test and y_test sets, we may evaluate how well the model predicted the outcome. It is possible to explicitly test the train and test sets' sizes. It is advised to maintain a larger train set than test set. We use 67% of our data for training and 33% for test sets in the proposed system. A subset of the training set is used as a validation set when fine-tuning the model's hyper parameters. After that, we scale the data such that the minimum value for each feature is zero and the maximum value is one. The MinMaxScaler is used to scale the data such that it falls inside a given range, often from 0 to 1. The data is changed when features are scaled to a certain range. The original distribution pattern is maintained while the numbers are adjusted to a preset range. The Sklearn preprocessor defines the MinMaxScaler() method for this exact reason. The MinMaxScaler() method generates output for lists like train_X, test_X, train_Y, and test_Y. The phase explains how to use these four lists in the application of the CNN-LSTM. Equations 2 and 3 can be used to express the minmaxscaler transformation.

$$x_{std} = \frac{(x - x.min(axis=0))}{(x.max(axis=0) - x.min(axis=0))} \text{-----(2)} \quad x_{scaled} = x_{std} * (max - min) + min \text{---(3)}$$

where min, max = feature_range.

Phase 4: CNN-LSTM - The model parameters of the neural network are as follows: case category, test description attribute, numerical object of normalization, train x, train y, test x, and test y. To ensure that words

are properly capitalized, we can utilize the capitalize() method. An additional 32-bit filter using the ReLU activate function and a one-kernel capacity is added to the conv1D. An extra layer is added to the one-dimensional representation using the same features and activation function as tanh. Under this scenario, a max-pooling one-dimensional layer with a pool size of 1 is employed. The final hidden layer and output layer both use a 30% dropout, in contrast to the additional variables. This is accomplished by means of the symbol (or 0.3) between the two layers.

The model is constructed using multiple layers, with distinct combinations of characteristics being created for each layer. Using tanh as the activation function, the second layer that follows uses a 64-size filter with a 1-value kernel and similar padding. By combining a pool size of one with a dropout layer of thirty-five percent, or 0.35, the maximum amount of pooling can also be achieved. A final one-dimensional convolution layer is also included, with a filter value of 128; this is in addition to the maximum pooling layer. Everything else remains unchanged from the previous layers' variables. Four percent, or 0.4, is the rate at which a new dropout layer is formed.

All one hundred blocks that make up the LSTM architecture have their activation functions set to tanh and their return sequencing to true. Then, the 25% dropout layer (or 0.25% of the total) was added. Fifty blocks make up the LSTM's secondary layer, and a dropout frequency of 20% (or 0.2) is already in place. The other features, on the other hand, have stayed unchanged. After this, the network is flattened and an extra dropout layer and a dense layer with parameters of 0.2 and 20% are added, respectively. Both the activation function and the kernel dimension of the dense layer are set to softmax, with the former set to 6. At startup, the activation function is not provided to the last dense layer, and it is given a kernel capacity of 1. Entire architecture is represented in Figure 2, which comprises design schematics.

The used tanh and ReLU activation function are depicted in equation 4 and 5.

$$\tanh = \frac{(e^x - e^{-x})}{(e^x + e^{-x})} \quad (4) \quad \text{Relu} = \max(0, x) \quad (5)$$

Where x is the input attributes values

Training the network would involve a total of 100 epochs and a batch size of 10, following its construction using the Adam optimizer the trained data will be stored in extension.h5. The architecture for the CNN-LSTM can be seen in the figure 2 below along with the accuracy graph of training for 100 epochs in figure 3.

Phase 5- Random forest classification for Sentiment analysis - To assign labels to the posts which are trained by the past step, this step use a random forest classification model. A random forest is a meta-estimator that controls over-fitting and improves predictive accuracy by averaging the results of numerous decision tree classifier fits to diverse sub-samples of the dataset. The optimal split approach is used by forest trees, which is equivalent to passing "best" as a parameter to the Decision Tree Regressor. Random forest use the complete data to build each tree, or it changes the sub-sample size with the max_samples option to set default bootstrap=True. This parameter is specific to every tree. This analyses the data and produces a set of predicted rows with the labels like positive, negative, and neutral sentiments.

CNN LSTM	
Layer	Activation
CONV 1D 32 Samples,Kernel=1	relu
CONV 1D 32 Samples,Kernel=1	relu
Max Pooling 1D	
Dropout 30%	
CONV 1D 64 Samples,Kernel=1	relu
Max Pooling 1D	
Dropout 35%	
CONV 1D 128 Samples,Kernel=1	Tanh
Max Pooling 1D	
Dropout 40%	
LSTM 50	
Dropout 25%	
Flatten	
Dense 6	Tanh
Dropout 20%	
Dense 1	None
Adam Optimizer	
Batch size 10	
Epochs 250	

Figure 2: Architecture for CNN LSTM

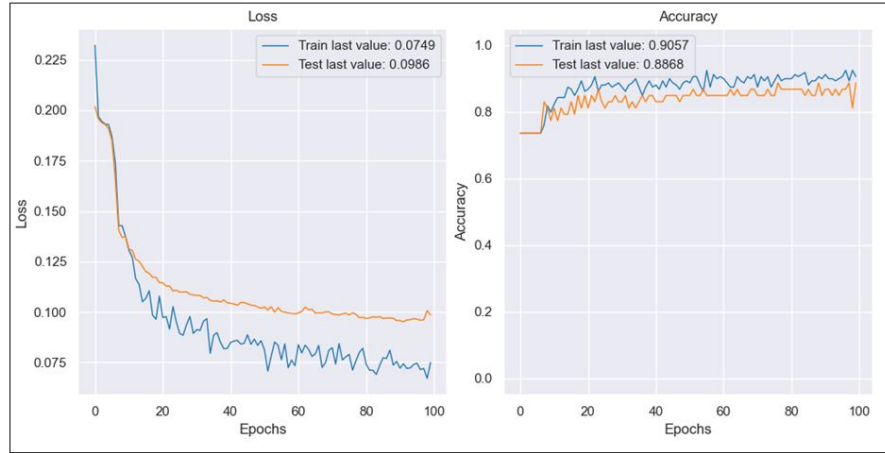


Figure 3: Accuracy and loss graph for training the dataset for 100 epochs

IV. RESULTS AND DISCUSSIONS

A Windows-based, Intel Core i5-powered PC with 16 GB of RAM is utilized to assess the deployed model. The confusion matrix's accuracy score parameter, which is expressed by the following equation, is used to test the deployed system. Accuracy, Precision, Recall and Macro F1 can be presented in equation 6,7,8 and 9 respectively.

$$\text{Accuracy} = \frac{TP+TN}{TP+FP+TN+FN} \quad - (6) \quad \text{Precision}(P) = \frac{TP}{TP+FN} \quad - (7)$$

$$\text{Recall}(R) = \frac{TP}{TP+FP} \quad - (8) \quad \text{Macro - F1} = \frac{2 \cdot P \cdot R}{P+R} \quad - (9)$$

Here, TP is True positive cases, TN is True Negative cases, FP is False positive cases and FN is False Negative cases.

The obtained accuracy and macro-F1 scores are compared with that of [5], Which is written by Waqar Ali et al. present An attractive future development has been the amalgamation of AI and SIoT, which has coincided with the fast rise of AI technology. An improvement in communication and the ability for SIoT object peers to make autonomous decisions might be achieved by reviewing data based on user opinions. The author of this piece suggests a deep learning approach to entity sentiment prediction. The author's model employed a Bi-GRU architecture that included the GloVe word embedding layer to produce the feature vectors for words in opinion sentences. By utilizing the data found in the tags before and after each word, the feature vector is obtained from the input by use of two GRU layers: the forward GRU and the reverse GRU.

The Accuracy and Macro F1 score of the proposed model using CNN-LSTM is compared with the LSTM and Bi-GRU model as mentioned in [5] for the restaurant dataset. The results are tabulated in the table 1 and the comparative graph can be seen in figure 4.

TABLE I: ACCURACY AND MACRO F1 SCORE COMPARISON TABLE

	Accuracy	Macro F1 Score
LSTM	75.45	66.57
Bi-GRU	82.05	72.53
CNN-LSTM	86.12	76.88

The graph clearly shows that the proposed CNN-LSTM ensembling model outperforms the LSTM and Bi-GRU methods discussed in [5] with an accuracy of 86.12 and a Macro F1 score of 76.88. The reasoning behind this is that combining two deep learning models strengthens the sentiment analysis process, making the resulting model credible enough to demonstrate its validity.

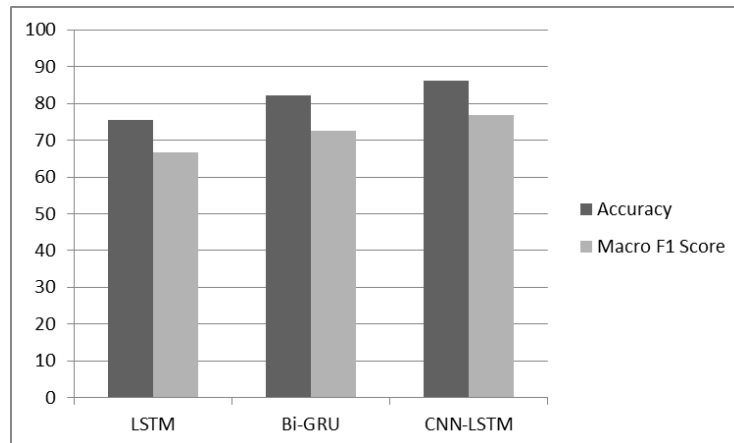


Figure 4: Accuracy and Macro F1 Score comparison Graph

V. CONCLUSION AND FUTURE SCOPE

In order to achieve optimal results while analyzing the sentiment of social media posts, this research article suggested a system that combines the two neural networks, LSTM and CNN. At first, the model uses the stopword and stemming procedure post-to-string tokenization approach to remove unnecessary data from the input dataset. This is done in order to estimate the sentiment analysis. To obtain additional information for the following phase of estimating the Pearson correlation for the provided attributes, Data is separated for training purposes by vomiting out the attributes with the lowest correlation. The CNN-LSTM neural networks are trained using ensemble learning to achieve fine-tuned training accuracy, as described in earlier sections. The Random Forest classification model is employed to categorize the feelings using the fine-grained training data. The obtained results with an accuracy of 86.12 and a Macro F1 score of 76.88 are clearly having upper hand than that of single deep learning models.

The suggested model can be enhanced for use with generic datasets in the future by identifying attributes for sentiment analysis using a hybrid neural network model that makes use of the transformer technique to guarantee more accurate results when dealing with real-time data.

REFERENCES

- [1] J. He, A. Wumaier, Z. Kadeer, W. Sun, X. Xin and L. Zheng, "A Local and Global Context Focus Multilingual Learning Model for Aspect-Based Sentiment Analysis," in *IEEE Access*, vol. 10, pp. 84135-84146, 2022, doi: 10.1109/ACCESS.2022.3197218.
- [2] Yassin S. Mehanna, Massudi Bin Mahmuddin, "A Semantic Conceptualization Using Tagged Bag-of-Concepts for Sentiment Analysis," in *IEEE Access*, Received July 8, 2021, accepted August 17, 2021, date of publication August 24, 2021, date of current version September 1, 2021.
- [3] C. -G. Kim, Y. -J. Hwang and C. Kamyod, "A Study of Profanity Effect in Sentiment Analysis on Natural Language Processing Using ANN," in *Journal of Web Engineering*, vol. 21, no. 3, pp. 751-766, May 2022, doi: 10.13052/jwe1540-9589.2139.
- [4] Q. Yang, Z. Kadeer, W. Gu, W. Sun and A. Wumaier, "Affective Knowledge Augmented Interactive Graph Convolutional Network for Chinese-Oriented Aspect-Based Sentiment Analysis," in *IEEE Access*, vol. 10, pp. 130686-130698, 2022, doi: 10.1109/ACCESS.2022.3228299.
- [5] W. Ali, Y. Yang, X. Qiu, Y. Ke and Y. Wang, "Aspect-Level Sentiment Analysis Based on Bidirectional-GRU in SIoT," in *IEEE Access*, vol. 9, pp. 69938-69950, 2021, doi: 10.1109/ACCESS.2021.3078114.
- [6] Y. Zhang, C. Zhu and Y. Xie, "Fine-Grained Sentiment Analysis of Cross-Domain Chinese E-Commerce Texts Based on SKEP Gram-CDNN," in *IEEE Access*, vol. 11, pp. 74058-74070, 2023, doi: 10.1109/ACCESS.2023.3296447.
- [7] N. Dhariwal, S. C. Akunuri, Shivama and K. S. Banu, "Audio and Text Sentiment Analysis of Radio Broadcasts," in *IEEE Access*, vol. 11, pp. 126900-126916, 2023, doi: 10.1109/ACCESS.2023.3331226.
- [8] B. M. A. Tahayna, R. K. Ayyasamy and R. Akbar, "Automatic Sentiment Annotation of Idiomatic Expressions for Sentiment Analysis Task," in *IEEE Access*, vol. 10, pp. 122234-122242, 2022, doi: 10.1109/ACCESS.2022.3222233.

- [9] N. Zhao, H. Gao, X. Wen and H. Li, "Combination of Convolutional Neural Network and Gated Recurrent Unit for Aspect-Based Sentiment Analysis," in IEEE Access, vol. 9, pp. 15561-15569, 2021, doi: 10.1109/ACCESS.2021.3052937.
- [10] V. D. Prabha and R. Rathipriya, "Competitive Capsule Network Based Sentiment Analysis on Twitter COVID'19 Vaccines," in Journal of Web Engineering, vol. 21, no. 5, pp. 1583-1601, July 2022, doi: 10.13052/jwe1540-9589.2159.
- [11] B. -W. On, J. -Y. Jo, H. Shin, J. Gim, G. S. Choi and S. -M. Jung, "Efficient Sentiment-Aware Web Crawling Methods for Constructing Sentiment Dictionary," in IEEE Access, vol. 9, pp. 161208-161223, 2021, doi: 10.1109/ACCESS.2021.3129187.
- [12] M. N. Razali, S. A. Manaf, R. B. Hanapi, M. R. Salji, L. W. Chiat and K. Nisar, "Enhancing Minority Sentiment Classification in Gastronomy Tourism: A Hybrid Sentiment Analysis Framework With Data Augmentation, Feature Engineering and Business Intelligence," in IEEE Access, vol. 12, pp. 49387-49407, 2024, doi: 10.1109/ACCESS.2024.3362730.