Journal of Computational Social Science
https://doi.org/10.1007/s42001-023-00236-5

RESEARCH ARTICLE

An empirical study of sentiment analysis utilizing machine learning and deep learning algorithms

Betul Erkantarci[1] · **Gokhan Bakal**[1]

Received: 19 July 2023 / Accepted: 2 November 2023

Abstract

Among text-mining studies, one of the most studied topics is the text classification task applied in various domains, including medicine, social media, and academia. As a sub-problem in text classification, sentiment analysis has been widely investigated to classify often opinion-based textual elements. Specifically, user reviews and experiential feedback for products or services have been employed as fundamental data sources for sentiment analysis efforts. As a result of rapidly emerging technological advancements, social media platforms such as Twitter, Facebook, and Reddit, have become central opinion-sharing mediums since the early 2000s. In this sense, we build various machine-learning models to solve the sentiment analysis problem on the Reddit comments dataset in this work. The experimental models we constructed achieve $F1$ scores within intervals of 73–76%. Consequently, we present comparative performance scores obtained by traditional machine learning and deep learning models and discuss the results.

Keywords Sentiment analysis · Machine learning · Deep learning · Text mining

Introduction

Sentiment analysis is a text-based study used for understanding people's emotions and opinions about a list of topics from texts. Technically, it is a computational examination in several domains, including marketing, social media, and customer satisfaction. Thus, sentiment analysis is exploited by companies to understand their customers' insight so that better business strategies can be developed [21]. In

✉ Gokhan Bakal
gokhan.bakal@agu.edu.tr
https://scholar.google.com/citations?user=RmUCPNwAAAAJ

Betul Erkantarci
betul.erkantarci@agu.edu.tr

[1] Department of Computer Engineering, Abdullah Gul University, Erkilet Bulvd., Kayseri 38080, Turkey

Published online: 09 December 2023

education systems, sentiment analysis is adopted to extract student opinions from feedback surveys. Social media platforms use the method of obtaining information about the emotions and thoughts of the users through sentiment analysis. Social media sites like Twitter, Facebook, and Instagram are massive data sources for users to share their opinions [18]. Reddit is also one of the social media platforms in which users can share their opinion. In this paper, we used Reddit comments as input data elements for sentiment analysis experiments. The target sentiments are divided into three categories positive, negative, and neutral. Considering the large amount of data produced on social media platforms, it is nearly impossible to analyze the sentiments without automated solutions. So, there are multiple sentiment analysis studies, including rule-based models combining NLP (Natural Language Processing) techniques, machine learning (ML) techniques, deep learning (DL) models, and semi-supervised methods like graph-based methods [15, 25]. In this article, two distinct approaches, which are traditional machine learning techniques and deep learning techniques, are employed to analyze the sentiments. Here, the essential motivation is *to analyze the Reddit comments with machine learning techniques (Logistic Regression and Random Forest) and deep learning models (CNN-Convolutional Neural Networks and LSTM-Long Short Term Memory)*. Eventually, we compare these methods by their $F1$ performance scores and running times. Within this study, we combined distinct features to get more accurate results and enhanced comparison. Particular contributions of this work are listed below.

- We built distinct traditional ML models (using logistic regression and random forest algorithms) using a unique set of n-gram features and their combinations, including unigram, bigram, and trigram.
- In addition, we built distinct deep-learning models using individual CNN and LSTM architectures along with a hybrid model with their combinations.
- Consequently, we tested the models and reported performance scores of both ten randomly-shuffled distinct test sets and 5-fold cross-validation results.

The remaining part of the paper is planned as follows. "Background and relevant literature" presents general background knowledge about sentiment analysis and particular machine learning techniques used for it. "Methodology" mentions the Reddit dataset and explains the methodologies utilized in the experiments. Finally, "Results and evaluations" demonstrates the comparative results, and "Conclusion" concludes the presented work with an overall summary.

Background and relevant literature

The popularity and usage of social media platforms have been dramatically increasing since they emerged due to their information dissemination power. In parallel to this situation, there have been many studies in different domains, such as health, finance, and politics, using social media data collected via available Twitter APIs [2, 3, 7]. These studies addressed the rise in social network platform usage and the challenge of extracting useful information from user-contributed

data. Also, they highlighted the relevance of sentiment analysis through distinct methods in this context.

The widely studied task is sentiment analysis, specifically on Twitter which is the most popular social media application in that sense. Technically, the tweets are classified into their sentiment in the tweet text. However, we chose to work on the Reddit comments dataset, unlike the typical works analyzing Twitter datasets.

As conventional machine learning models, sentiment analysis experiments also can be constructed by three approaches: unsupervised, semi-supervised, or supervised [19]. One of the unsupervised techniques is a lexicon-based method. This method typically uses a measurement to determine the polarity of the sentences. So, the predefined lexicons with annotated sentiment words are exploited to perform the analysis. On that basis, the work done by Diwali et al. [6] introduced a novel framework that combines Arabic dependency-based rules and deep learning models for sentiment analysis. Technically, dependency-based rules are established using linguistic patterns to map word meanings to concepts within sentence dependency structures. The semi-supervised learning technique involves unsupervised and supervised approaches, where both labeled and unlabeled data are used to conduct the sentiment analysis. For instance, Lee et al. [13] proposed a semi-supervised learning method to overcome the challenges of manual data labeling for sentiment analysis. Semi-supervised learning combines a small amount of available labeled data with a large pool of unlabeled data for training machine learning models. Lastly, the supervised approach employs labeled data (by supervision of known data examples) to build a classification model. The effort by Ye et al. [23] revealed that the SVM and character-based N-gram approach outperformed the Naïve Bayes approach in sentiment classification. Additionally, when the training datasets included a large number of reviews, all three approaches achieved accuracies of at least 80%. The qualitative finding underscores the importance of sentiment classification in mining travel-related reviews from blogs, with SVM and N-gram models being identified as effective approaches for accurate sentiment analysis. The findings have valuable implications for improving the usefulness of online travel information. Compared to the unsupervised learning approach, most machine learning and deep learning applications are built with the supervised learning approach [12]. The widely used machine learning algorithms are Naïve Bayes, Logistic Regression (LR), Random Forest (RF), and Support Vector Machines (SVM) to analyze the sentiments. Recently, Gulati et al. [10]'s work suggested that machine learning-based classifiers can be used to effectively identify the sentiment of COVID-19 tweets. This information can be used to track public opinion about the COVID-19 pandemic, identify areas of concern, and develop effective public health interventions. Similarly, the latest popular deep learning models are DNN (Deep Neural Network), CNN, and LSTM, exploited in sentiment analysis experiments. Specifically, [5] showed that deep learning models are a promising solution to the challenges of sentiment analysis. The experimented deep learning models have the potential to improve the efficiency and accuracy of sentiment analysis, which could lead to a wide range of applications. In Table 1, we demonstrate the fundamental approaches employed in sentiment analysis efforts.

Table 1 Sentiment analysis methods comparison

Approach	Models	Comparative details
Unsupervised	Rule-based	Domain specific Sophisticated linguistic resources are required Lower precision performance
Semi-supervised	Graph-based	Lower performance if the graph structure does not fit the data Require less time but not reliable and low precision performance in spite of taking less time
Supervised	Machine learning	Better classification scores Time-consuming Domain-dependent
	Deep learning	Enhance the precision High computational cost

Al Amrani et al. [1] conducted comparative research on sentiment analysis using machine learning methods, including SVM, random forest, and hybrid method RFSVM, which combines SVM and random forest algorithms. Hidayat et al. [11] performed a comparative sentiment analysis experiment on Twitter data to analyze the performances of Doc2Vec, SVM, and Logistic Regression. Dang et al. [5] tested deep learning approaches in sentiment analysis research, where they constructed DNN, CNN, and RNN models.

Unlike the aforementioned deep learning models, the transformer architecture, presented by Vaswani et al. [20], introduces a unique encoder-decoder model that enables the simultaneous processing of all input tokens- words, rather than sequential processing. Unlike traditional approaches, including other seq2seq architectures, transformers treat input sequences as unordered bags of words/tokens., The transformer model, technically, employs a self-attention mechanism to capture token dependencies. Furthermore, an initial encoding step, executed before the first encoder layer, guarantees distinct representations for the same word occurring at different positions within a sentence representation.

In this study, we constructed traditional machine learning models, including Logistic Regression and Random Forest, as well as deep learning models, such as CNN, LSTM, and a hybrid approach constructed by CNN and LSTM architectures. These models were utilized to determine the appropriate sentiment label for the Reddit comments dataset. Subsequently, we compared their performances through in-depth discussions and analysis.

Methodology

Machine learning, which is intensively based on mathematics and statistics, has had substantial significance in recent years. Basically, machine learning allows us to predict unknown test elements by analyzing previously known training data [17]. Here,

Table 2 Comprehensive statistical information about the dataset

Category	Initial count	After removing empty values	After lemmatizing and cleaning steps (removing stopwords, entries having less than five words)	Dataset distribution after balancing
−1	8277	8277	6801	5000
0	13,042	13,042	5632	5000
1	15,830	15,830	12,996	5000
Total	37,249	37,149	25,429	15,000

one widely studied application is text classification tasks. In that sense, sentiment analysis is known as a subtask under text classification.

In this study, we use the Reddit comments dataset, which is publicly available on the Kaggle platform [9]. In the dataset, there are 37K comments available along with its sentimental label. Those Reddit comments reflecting public opinions were collected in the context of general elections held in India in 2019. As mentioned in the previous sections, we built traditional ML models and DL models to evaluate their performances on the sentiment analysis task using the Reddit comments dataset (Table 2).

Traditional machine learning models

One of the conventional machine learning algorithms used is logistic regression. LR algorithm has a logistic function that converts the input space representations into the numeric values between the range of [0, 1] and yields the probability scores for the probabilistic assessments. Although it is widely associated with binary classification problems [4], it can be operated for multi-label classification tasks with the one-vs-rest option. We employed the LR algorithm from the popular Python-based ML library, scikit-learn [14] for multiclass classification purposes. The other conventional machine learning algorithm employed in the experiments is the Random Forest (RF) classifier. The RF models contain a group of decision trees classifying a new sample using its input vector. Technically, the RF algorithm exploits the power of various decision trees to achieve a final decision result. Thus, it is an ensemble classifier model [17].

Deep learning architectures

The artificial neural network (NN) algorithms are the approaches simulating human neural structure to analyze input data as a training operation. Hence, a usual neural network consists of inter-connected elements called artificial neurons, as shown in Fig. 1. Deep learning is a type of neural network with multiple layers. Recently, deep learning models are yielding superior performances in several areas, including NLP, image processing, and pattern recognition. Compared to traditional ML machine learning applications in which features are hand-crafted, deep learning

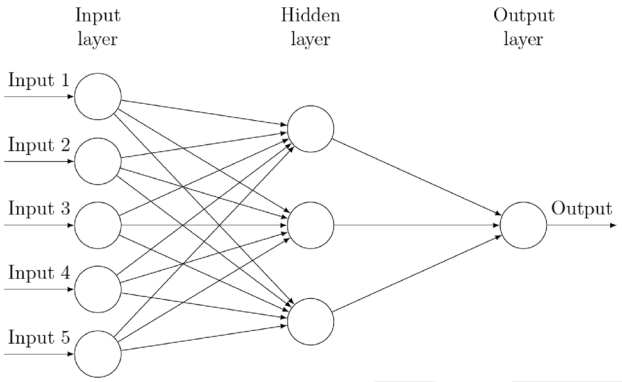

Fig. 1 An example representation of a usual feed-forward neural network

applications extract the features automatically; this functionality enables deep learning models to provide better accuracy and performance.

Convolutional neural networks

A standard CNN architecture is a subtype of a feed-forward neural network. Typically, it contains an input layer, multiple hidden layers, and an output layer. The hidden layers primarily consist of convolutional layers, an activation layer, possible pooling layers (via 2D operations), and followed by fully connected layers. The convolution layer is the core component of the CNN algorithm, which extracts contextual feature elements from the training instances [24]. For example, if we want to identify features for an input image, the convolution process will aggregate the underlying relationships between pixels by pre-defined two-dimensional convolution windows. From the text processing perspective, where a group of words is the input instead of images, we have a one-dimensional array representation of the instances. Thus, the CNN system will consist of a 1D convolutional and a pooling layer.

In this effort, the first deep learning experiment is a CNN model. In the model structure, feature extraction is executed within the convolution layers by filtering the given inputs with multiple filters and combining the outputs. Following the feature extraction, the dimensions of feature maps and the number of parameters during the learning process are reduced by the pooling layers. By doing that, CNN' s robustness and generalization power is enhanced by alleviating the overfitting issue. Consequently, the fully-connected layer performs the classification task, and the CNN network ends with an output layer containing a `softmax` function.

Long short-term memory networks

The LSTM model is a more sophisticated deep learning model based on the repetitive artificial neural networks (RNN) structure. Similar to standard RNN models, the LSTM model is widely used, too, on temporal and serial datasets, such as text sentences and DNA sequences [8, 16]. The most remarkable aspect of this model

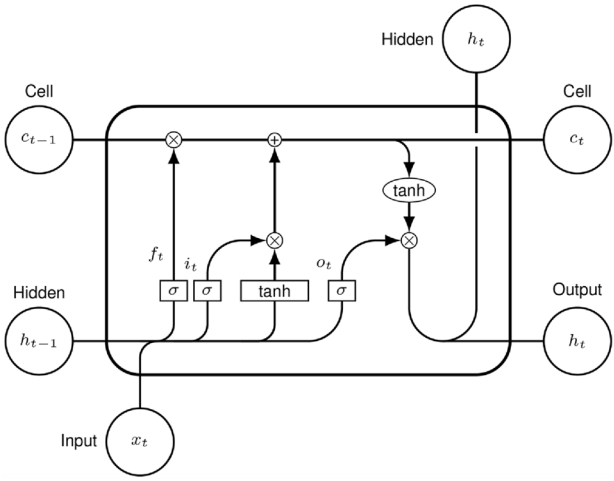

Fig. 2 An example of LSTM cell structure

is that it solves the vanishing gradient problem occurring over time in a standard RNN model. However, the LSTM model solves this problem by particular data gates (input, forget, and output) along with the cell state unit. A typical LSTM cell structure is represented in Fig. 2.

The cell state unit (c_t) serves as a memory area in the network and holds characteristic information to transmit to other cells. Thus, short-term information can be stored and transported throughout the network. The forget (f_t) and input (x_t) gates in the LSTM cell structure are responsible for determining the transmitted information to other cells. Technically, the forget gate erases and dismisses the information that is either 0 or close to 0 according to the `sigmoid` function result (typically a value between [0,1]) of the information existing in the cell and the value coming from the previous cell. The input gate performs the data update operation in the cell state unit according to the sigmoid function result of the data from the precedent cell's hidden state (h_{t-1}) and the data in the current cell (x_t). Finally, the output gate determines which information will be transmitted to other cells as previous cell information, according to the score generated by the multiplication of the `tanh` (a function generating value between $[-1, 1]$) activation result of the data in the cell state unit by the sigmoid of the previous and current cells data. Concisely, the LSTM architecture yields much higher classification performance thanks to its advanced structure compared to a standard RNN model.

Proposed solution approach

Traditional ML and DL models require particular preprocessing steps, such as lemmatization and data cleaning, before performing the main experiments. So, as a first step, the lemmatization step, which converts words to their base forms, is administered over the dataset to establish the unification of the distinct word representations.

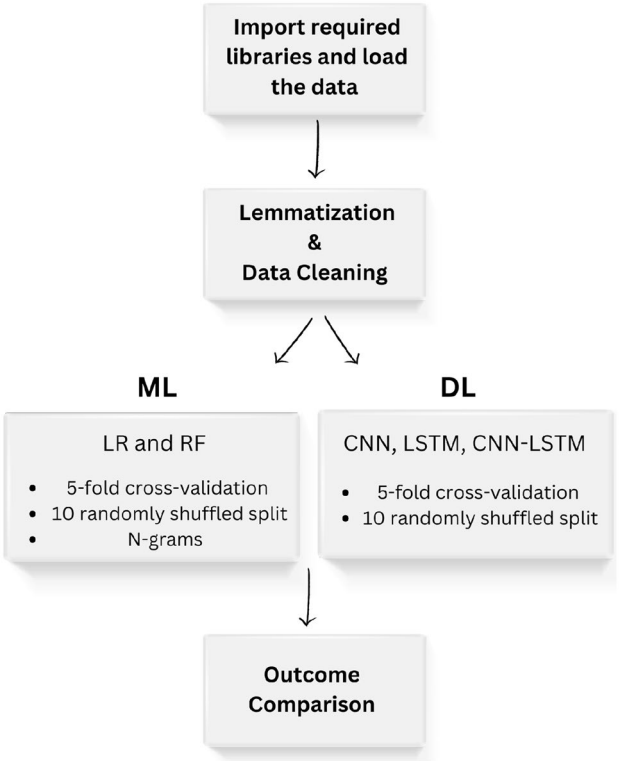

Fig. 3 Overall experimental steps

In this way, lemmatization helps the models match the root words and to provide more reliable classification results. The term forms are converted to base forms by considering part-of-speech tags (adjective, verb, noun, or adverb) to have more precise lemmas. After lemmatization, the cleaning step removes the stop words, punctuation symbols (characters), and meaningless words. The overall graphical representation of the experimental steps is illustrated in Fig. 3.

In addition to the cleaning step, the comments containing fewer words than five were omitted to retain only the most discriminative comments. The data distribution in the dataset with three categories (positive, negative, and neutral) is imbalanced, but we evenly distributed data with random sampling to compose a balanced dataset scenario. Consequently, there are 15,000 evenly distributed examples in the balanced dataset.

Experimental details

We have two distinct assessment configurations to evaluate the models' performances: *ten randomly shuffled split datasets* and *5-fold cross-validation* to derive average scores and confidence intervals (CIs). Before building the models, we split

Table 3 N-gram extraction through the sample sentence

N-gram type	Extracted n-grams
Unigrams ($n = 1$)	[Take], [a], [moment], [to], [appreciate]
Bigrams ($n = 2$)	[Take a], [a moment], [moment to], [to appreciate]
Trigrams ($n = 3$)	[Take a moment], [a moment to], [moment to appreciate]

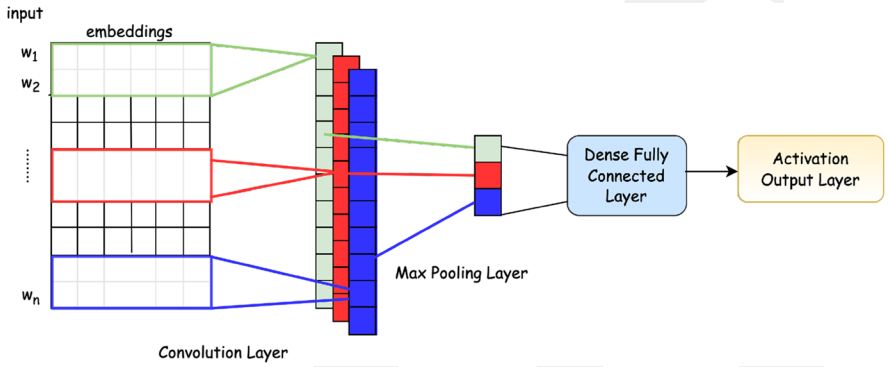

Fig. 4 Our CNN model architecture

each shuffled dataset into two major subsets; 80% as the training set and 20% as the test set. So, there are different train-test sets in each split. The text expressions in the dataset are transformed into TF-IDF (Term Frequency-Inverse Document Frequency) numeric vector representations to be utilized in traditional machine learning models. The purpose of TF-IDF values is to show how each n-gram element is relevant to a document. N-gram is a probabilistic language model in which n-ordered word phrases are present. Mainly, it is used extensively in the statistical natural language processing field to predict the next word [22]. In this study, five n-gram groups are generated and used as the contextual feature sets from the dataset. These n-gram groups consist of the list as in [unigrams], [bigrams], [unigrams ∪ bigrams], and [unigrams ∪ bigrams ∪ trigrams]. In Table 3, the extracted n-grams (where n is the number of consecutive words and can take a value as 1, 2, and 3 in the experiments) from the sample sentence "*Take a moment to appreciate*" describe the whole process with a better illustration.

Following the TF-IDF vectorization, the vectorized data is transferred to the pipeline process building the logistic regression and random forest classifier models. Then, the models are fitted by using the training examples. When the models are learned, the test examples are subjected to the trained models to obtain the prediction results. Considering the deep learning experiments, we have a CNN, an LSTM model, and a hybrid CNN-LSTM model. In the CNN model, we have an embedding layer as input followed by three 1D convolution layers (filter sizes: 32, 64, and 128, respectively), along with 1D max pooling (by pooling size of 2) layers. Then a dense layer with class sizes is used as output as shown in Fig. 4.

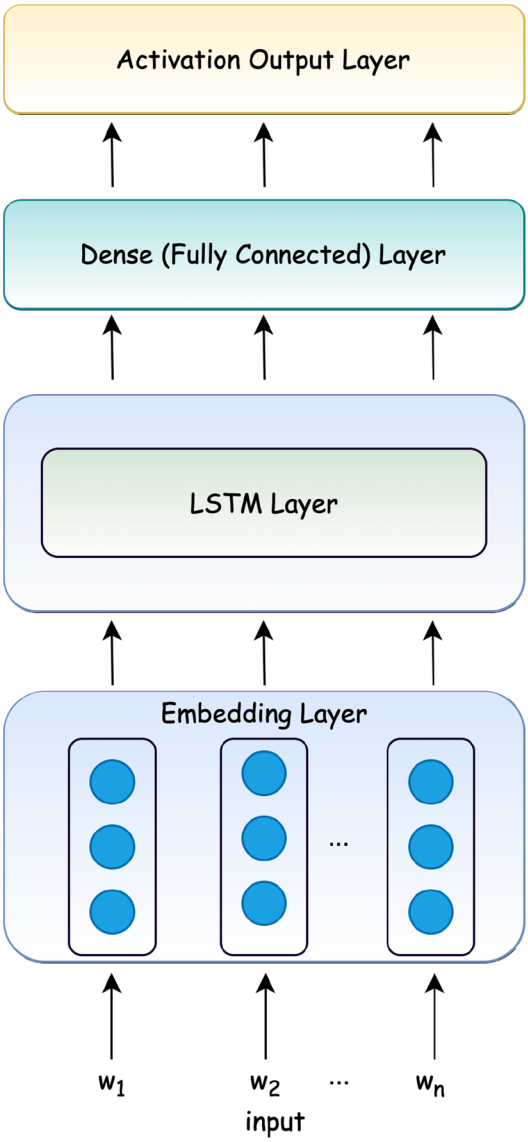

Fig. 5 Our LSTM model architecture

In the LSTM model, as presented in Fig. 5, we use an embedding layer as input to the architecture. Then, we utilize an LSTM layer with 100 units and a dropout rate of 0.2. Next, the LSTM model ends with a dense layer with the number of classes as output. As demonstrated in Fig. 6, the hybrid model constructed by CNN and LSTM algorithms starts with an embedding layer as input, then two 1D convolution layers (by the filter size of 32), and an immediate 1D max pooling layer. Then, the hybrid network includes an LSTM (with 100 units and a dropout rate of 0.2) layer and ends with a dense layer containing the number of classes

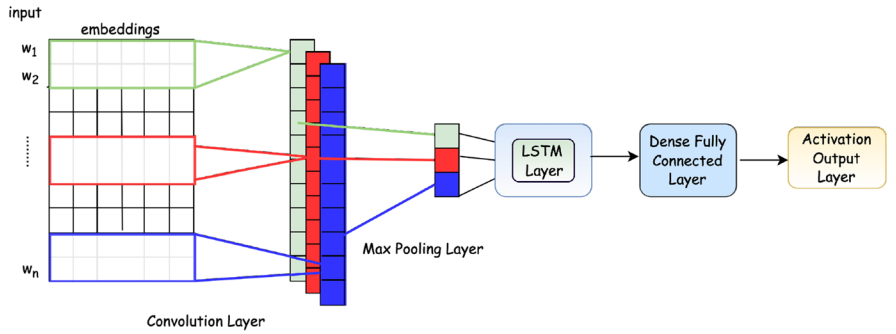

Fig. 6 Our hybrit (CNN + LSTM) model architecture

as output. In all DL models, the batch size is selected as 128, and the embedding dimension is picked as 100.

Performance evaluation metrics

Precision, recall, and specifically the $F1$ score (the harmonic mean of precision and recall scores) will be used to reveal the classification performances for each model. The motivation for using the $F1$ score[1] as a principal evaluation metric is to show how erroneous False Positive (FP) and False Negative (FN) cases vary against True Positive (TP) classification examples. The performance scores will be calculated according to the formulas in Eqs. (1)–(4).

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{1}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{2}$$

$$F1 = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{3}$$

$$F1 \text{ on Avg. Scores} = \frac{2 \times \text{Avg. Precision} \times \text{Avg. Recall}}{\text{Avg. Precision} + \text{Avg. Recall}} \tag{4}$$

[1] We have calculated the main $F1$ metric score by using both average precision score and average recall score.

Results and evaluations

In the following subsections, we first presented the traditional models' performance scores. Then, we reported the deep learning models' performances. Finally, we shared the comparative evaluations using the best-performing model scores for both configurations.

Traditional machine learning models evaluations

In Table 4, we demonstrated the performance evaluation results of various model configurations constructed as traditional machine learning approaches. When we inspect at the result table, the interesting point is that the performance scores of both assessment configurations are nearly identical for each machine-learning model. Among all distinct experiments, the trigram models yielded the lowest performance scores.

The potential reason for this outcome is that the trigrams are the least frequent elements compared to other n-grams across the dataset. The 5-fold cross-validation assessment on the model utilizing the combination of unigram, bigram, and trigram gave the best performance in the logistic regression experiments, even if there is a slight difference. Besides, the model using the unigram features yielded the second-best performance on the ten-shuffled split assessment.

In the random forest experiments, the 5-fold cross-validation assessment gained the best $F1$ score on the model employing the union of unigram and bigram features. However, the model exploiting unigram, bigram, and trigram yielded the second-best (*by a tiny difference of 0.001119*) $F1$ score through the ten-shuffled split assessment. This consequence is not surprising because both model configurations use the most frequent n-grams (unigrams and bigrams) extracted from the dataset, contrary to the models empowered by individual n-grams. Overall, the logistic regression model configurations slightly outperformed the random forest models even though their performance scores are close to each other. When we calculated the confidence intervals for the RF 10-RS models, we acquired 0.004, 0.007, 0.007, 0.005, and 0.004 CI values for the [unigrams], [bigrams], [unigrams ∪ bigrams], and [unigrams ∪ bigrams ∪ trigrams] model configurations, respectively. Since these CI values are less than 0.01, we can strongly entail that the performance scores are statistically significant and not obtained by chance.

Deep learning models' evaluations

To build and evaluate the models, we have divided the dataset into three subsets: 70% as training, 20% as testing, and 10% as validation to use in the deep learning experiments. As in traditional machine learning experiments, the performance scores are derived by the average scores of ten distinct shuffled data splits. Here, we have three central model configurations; CNN, LSTM, and CNN-LSTM. The first model was constructed by three CNN layers followed by a final dense layer

Table 4 Performance scores of distinct model configurations for traditional machine learning experiment

Algorithm	Assessment case	Model	Avg. precision	Avg. recall	*F*1-score on avg. precision & Avg. recall
Logistic regression	Ten randomly shuffled split datasets	Unigram	0.755663	0.752267	**0.753961**
		Bigram	0.511373	0.504300	0.507810
		Trigram	0.520431	0.361300	0.426347
		Unigram ∪ Bigram	0.748264	0.746433	0.747347
		Unigram ∪ Bigram ∪ Trigram	0.748016	0.746300	0.747157
	5-fold cross-validation	Unigram	0.761442	0.760328	0.760884
		Bigram	0.522354	0.511464	0.516852
		Trigram	0.513634	0.359618	0.423044
		Unigram ∪ Bigram	0.740327	0.740008	0.740167
		Unigram ∪ Bigram ∪ Trigram	0.768889	0.768211	**0.768550**
Random forest	Ten randomly shuffled split datasets	Unigram	0.758421	0.758800	0.758610
		Bigram	0.512650	0.502800	0.507670
		Trigram	0.526206	0.352933	0.421950
		Unigram ∪ Bigram	0.759292	0.759200	0.759245
		Unigram ∪ Bigram ∪ Trigram	0.759712	0.759467	**0.759589**
	5-fold cross-validation	Unigram	0.753796	0.755663	0.754728
		Bigram	0.501584	0.495762	0.498656
		Trigram	0.533786	0.358123	0.428656
		Unigram ∪ Bigram	0.759795	0.761624	**0.760708**
		Unigram ∪ Bigram ∪ Trigram	0.751822	0.752967	0.752394

Bold scores indicate better classification results

Table 5 Performance scores of deep-learning models

Model	Avg. precision	Avg. recall	$F1$-score on avg. precision & Avg. recall
CNN_{10-RS}	0.738874	0.735000	0.736923
$LSTM_{10-RS}$	0.742441	0.740400	**0.741416**
$CNN-LSTM_{10-RS}$	0.733699	0.731433	0.732559
CNN_{CV}	0.744321	0.735289	0.739764
$LSTM_{CV}$	0.748223	0.747328	0.747775
$CNN-LSTM_{CV}$	0.754165	0.753562	**0.753863**

Bold scores indicate better classification results

before the output. Similarly, the second model was built by using an LSTM layer followed by a final fully-connected dense layer. In the final deep learning model, two CNN layers (to extract the contextual features) and one LSTM layer were employed to create a hybrid model configuration to inspect the performance tradeoff between individual and hybrid models. The performance scores are demonstrated in Table 5 obtained by the deep learning experiments over both ten randomly shuffled datasets (*indicated by the subscript of 10-RS*) and a 5-fold cross-validation set (*referred by the subscript of CV*).

The first noticeable outcome is that the performance scores are nearly identical regardless of model configurations in each assessment option. Considering the ten randomly shuffled split datasets, the best $F1$ score, along with the best precision and recall scores, was achieved by the LSTM model. However, the best-performing model in the cross-validation assessment is the hybrid model constructed by CNN and LSTM algorithms. Another interesting point is that the CNN model underperformed compared to other models regardless of the assessment options. The more spectacular consequence is that the $F1$ score difference between the best and worst performing models is less than 2% for each evaluation option. Nevertheless, we achieved the best classification performance scores using the hybrid model using the 5-fold cross-validation approach. Similar to the traditional ML models, we computed the CIs for the DL models with the $F1$ scores for RS-10 configurations. So, we got the CI scores as 0.005, 0.007, and 0.01 for the CNN, LSTM, and CNN-LSTM models, respectively. Based on the CI scores, the $F1$ scores are statistically significant because the 95% CI widths are reasonably small.

Beyond the performance metrics evaluation, another critical point is to compare the models' running times. From that perspective, we also measured the elapsed time values (in minutes) and reported them in pie chart illustrations. In Fig. 7, the elapsed time values of the ML models are presented in minutes, while the elapsed times of the deep learning models are shown in Fig. 8.

Overall, the traditional ML models took more time to be built compared to deep learning models. The RF model tested by ten randomly shuffled sets took the longest time, while the CNN model on 5-fold cross-validation assessment lasted one minute, which is the shortest time among other models.

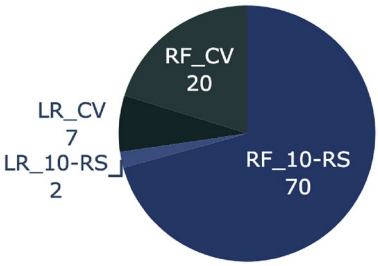

Fig. 7 Elapsed times of the traditional machine learning models (*LR: logistic regression, RF: random forest, 10-RS: ten random shuffles, and CV: cross-validation*)

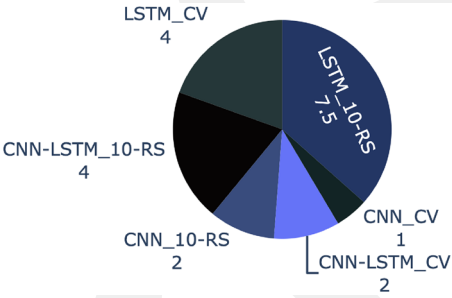

Fig. 8 Elapsed times of the deep learning models (*LSTM: long short-term memory network, CNN: convolutional neural network, 10-RS: ten random shuffles, and CV: cross-validation*)

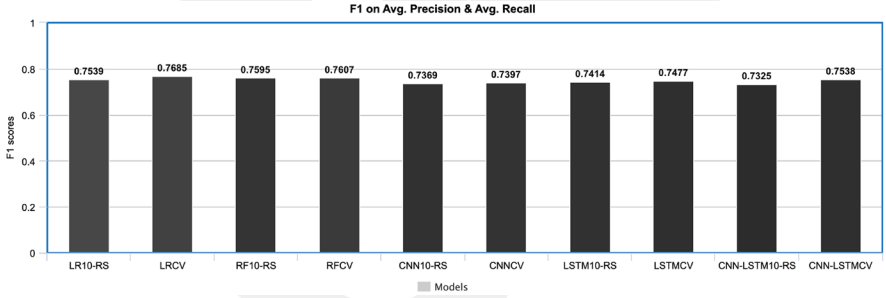

Fig. 9 Overall $F1$ scores of all models

Evaluation of the best performing models

When we consider traditional ML and DL models examined over each assessment procedure, we show their $F1$ performance scores in a bar-graph representation for a better comparison in Fig. 9. Generally, the models tested by the cross-validation method yielded better $F1$ scores against the other models assessed by ten randomly shuffled splits. The best-performing model is the Logistic Regression model with cross-validation (76.8%), while the lowest-performing model configuration is the hybrid deep learning model tested by ten shuffled splits (73.2%). Due to the relatively small data size, it is reasonable to observe this outcome, as traditional machine learning models are more suitable for handling such cases. The results also highlight

an intriguing finding: among the experimental models, the hybrid deep learning model exhibited the highest performance gap (~2%) between cross-validation and ten shuffled split assessments compared to other experimental models. Inevitably, the most striking consequence is that the ML models outperformed the DL models. However, this outcome is understandable because the dataset is not large enough for the DL models to learn the contextual associations among the data elements.

Conclusion

Sentiment analysis has been a widely studied research area by many researchers since digitalization began and opinion-sharing concepts emerged. In this article, we have built multiple ML (composed of logistic regression and random forest algorithms) and DL models (constructed by CNN and LSTM architectures) and tested them using ten randomly shuffled splits along with 5-fold cross-validation over the Reddit comments dataset. Among distinct ML and DL configurations, we achieved the $F1$ score of 76% with the Logistic Regression model with 5-fold cross-validation as the superior performance. The comparative results proved that the traditional ML models are relatively more successful in sentiment analysis tasks. This outcome is conceivable because the deep learning models may not learn the contextual associations in a small-sized dataset, even though they are more sophisticated architectures compared to traditional ML models.

Author contributions BE carried out the experiments and wrote the initial manuscript draft. GB supervised the study and conceived the original idea. Also, GB wrote the final manuscript.

Funding No funds, grants, or other types of support were received.

Availability of data and materials The dataset we used is publicly available via the link: https://www.kaggle.com/ds/429085.

Declarations

Conflict of interest The authors have no competing interests to declare that are relevant to the content of this article.

Ethical approval Not applicable.

References

1. Al Amrani, Y., Lazaar, M., & El Kadiri, K. E. (2018). Random forest and support vector machine based hybrid approach to sentiment analysis. *Procedia Computer Science, 127*, 511–520.
2. Arias, M., Arratia, A., & Xuriguera, R. (2014). Forecasting with twitter data. *ACM Transactions on Intelligent Systems and Technology (TIST), 5*(1), 1–24.
3. Bakal, G., & Kavuluru, R. (2017). On quantifying diffusion of health information on twitter. In *2017 IEEE EMBS international conference on biomedical & health informatics (BHI)* (pp. 485–488). https://doi.org/10.1109/BHI.2017.7897311

4. Bakal, G., Talari, P., Kakani, E. V., et al. (2018). Exploiting semantic patterns over biomedical knowledge graphs for predicting treatment and causative relations. *Journal of Biomedical Informatics, 82*, 189–199.
5. Dang, N. C., Moreno-García, M. N., & De la Prieta, F. (2020). Sentiment analysis based on deep learning: A comparative study. *Electronics, 9*(3), 483.
6. Diwali, A., Dashtipour, K., Saeedi, K., et al. (2022). Arabic sentiment analysis using dependency-based rules and deep neural networks. *Applied Soft Computing, 127*(109), 377.
7. Elghazaly, T., Mahmoud, A., & Hefny, H. A. (2016). Political sentiment analysis using twitter data. In *Proceedings of the international conference on internet of things and cloud computing* (pp. 1–5).
8. Gers, F. A., Schraudolph, N. N., & Schmidhuber, J. (2002). Learning precise timing with LSTM recurrent networks. *Journal of Machine Learning Research, 3*(Aug), 115–143.
9. Gowda, C., Anirudh, Pai, A., et al. (2019). Twitter and reddit sentimental analysis dataset. https://doi.org/10.34740/KAGGLE/DS/429085.
10. Gulati, K., Kumar, S. S., Boddu, R. S. K., et al. (2022). Comparative analysis of machine learning-based classification models using sentiment classification of tweets related to covid-19 pandemic. *Materials Today: Proceedings, 51*, 38–41.
11. Hidayat, T. H. J., Ruldeviyani, Y., Aditama, A. R., et al. (2022). Sentiment analysis of twitter data related to Rinca island development using doc2vec and svm and logistic regression as classifier. *Procedia Computer Science, 197*, 660–667.
12. Jiang, T., Gradus, J. L., & Rosellini, A. J. (2020). Supervised machine learning: A brief primer. *Behavior Therapy, 51*(5), 675–687.
13. Lee, V. L. S., Gan, K. H., Tan, T. P., et al. (2019). Semi-supervised learning for sentiment classification using small number of labeled data. *Procedia Computer Science, 161*, 577–584.
14. Pedregosa, F., Varoquaux, G., Gramfort, A., et al. (2011). Scikit-learn: Machine learning in python. *Journal of Machine Learning research, 12*, 2825–2830.
15. Punetha, N., & Jain, G. (2023). Bayesian game model based unsupervised sentiment analysis of product reviews. *Expert Systems with Applications, 214*(119), 128.
16. Ranjan, M. N. M., Ghorpade, Y., Kanthale, G., et al. (2017). Document classification using LSTM neural network. *Journal of Data Mining and Management, 2*(2), 1–9.
17. Shah, K., Patel, H., Sanghvi, D., et al. (2020). A comparative analysis of logistic regression, random forest and KNN models for the text classification. *Augmented Human Research, 5*(1), 1–16.
18. Shaik, T., Tao, X., Dann, C., et al. (2022). Sentiment analysis and opinion mining on educational data: A survey. *Natural Language Processing Journal*, *2*, 100003.
19. Vashishtha, S., & Susan, S. (2019). Fuzzy rule based unsupervised sentiment analysis from social media posts. *Expert Systems with Applications, 138*(112), 834.
20. Vaswani, A., Shazeer, N., Parmar, N., et al. (2017). Attention is all you need. In *Advances in neural information processing systems* (pp. 5998–6008).
21. Verma, S. (2022). Sentiment analysis of public services for smart society: Literature review and future research directions. *Government Information Quarterly*, 101708.
22. Yazdani, A., Safdari, R., Golkar, A., et al. (2019). Words prediction based on n-gram model for free-text entry in electronic health records. *Health Information Science and Systems, 7*(1), 1–7.
23. Ye, Q., Zhang, Z., & Law, R. (2009). Sentiment classification of online reviews to travel destinations by supervised machine learning approaches. *Expert Systems with Applications, 36*(3), 6527–6535.
24. Zeiler, M. D., Krishnan, D., Taylor, G. W., et al. (2010). Deconvolutional networks. In *2010 IEEE computer society conference on computer vision and pattern recognition* (pp. 2528–2535). IEEE.
25. Zhang, L., Wang, S., & Liu, B. (2018). Deep learning for sentiment analysis: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 8*(4), e1253.