Indonesian Journal of Electrical Engineering and Computer Science
Vol. 32, No. 3, December 2023, pp. 1736~1745
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v32.i3.pp1736-1745

Enhanced negation handling for sentiment analysis on Twitter using deep neural networks

Mamatha Mylarappa, Shiva Kumar B.N, Thriveni J. Gowda, Venugopal K. Rajuk
Department of Computer Science and Engineering, UVCE, Bangalore University, Bangalore, India

Article Info

Article history:

Received Jun 12, 2023
Revised Aug 30, 2023
Accepted Oct 8, 2023

Keywords:

Artificial neural network
Convolutional neural network
Machine learning
Negation handling
Support vector

ABSTRACT

Sentiment analysis is a tool to identify and measure the emotion in a piece of text. Negation handling is an important aspect of natural language processing (NLP) for Twitter data. It is a process of correctly interpreting the sentences containing negation words, such as, "never", "no", "neither" and so on. Negation words are used in machine learning to express negative sentiment or indicate the absence of something. In this article, a negation handling technique using deep learning models. Artificial neural networks (ANNs) and convolutional neural networks (CNNs) for classification is proposed. The system is evaluated on SemEval-2017 dataset. The classification performance is improved by using ANN and CNN on the negative tweets. The study aims to improve the classification accuracy by considering negation words in the text. The paper compares the performance of ANNs and CNNs in handling negation words and evaluates them on the tweets data. This study provides insights into the effectiveness of using deep learning techniques for negation handling in sentiment analysis and highlights the importance of considering negation words in the text for improved sentiment analysis performance. The proposed negation strategy attains a superior performance accuracy over machine learning models by preventing misclassified tweets.

Corresponding Author:

Mamatha Mylarappa
Department of Computer Science and Engineering, UVCE, Bangalore University
Bangalore, India
Email: mmtha.s@gmail.com

1. INTRODUCTION

Sentiment analysis also known as opinion mining, is a type of natural language processing (NLP) technique that involves identifying and extracting subjective information from textual data. Negation handling is the ability of NLP system to interpret the negation terms "no", "not", "bad", "shouldn't", and "nothing" in text. Morphological and syntactic are different forms. These words changes the meaning of the sentence completely. Due to the rapid growth of internet the use of social media has increased. Large amount of data in the form of blogs, comments and reviews, posts and tweets have been generated. Thus, to analyze this data sentiment analysis is necessary. Negation scope is always limited to the next word after negation or upto few other words. It can be performed as a process of preprocessing or in the classifier model using WordNet which contains synonyms and antonyms for a number of english words [1], [2]. Extraction of negation features is a difficult task in sentiment analysis. Negation pattern identification and scope detection is done using rule-based method [3]–[5]. Even after performing pre-processing, negation handling lacks in achieving good accuracy. It is resolved using different pre-processing techniques [6]–[8].

Negation is handled using long short-term memory (LSTM) model and it automatically learns negation features from labeled training data. The scope and cue to determine polarity shift in a negated sentence is identified. ConanDoyle, a pre-annotated dataset with negation information is used [9], [10]. The negations are based on linguistic features such as conjunctions, punctuations in the sentence and part of speech (POS) of negation [11]. The scope of negation is identified by the fixed number of words after negation. The polarities of such words are inverted. In syntactic negation, negative words or constructors are used in a sentence to express negation of a whole sentence. On other hand, diminishers (reducers) diminish the polarities of other words which they effect instead of inverting the polarity completely [12]. Machine learning techniques, such as deep learning models are used to handle negation in large annotated data to improve the accuracy [13]–[15]. These models identify the scope and negation cues automatically and handle complex negation structures efficiently. The opinion word from text is extracted and then their antonyms and synonyms are obtained from the dictionaries like SentiWordNet, and WordNet. The latter finds the opinion word in a context specific orientation. This method has a list of words, and finds other words in a huge corpus. Negation is analyzed on eight prominent corpus ranging six different natural language understanding tasks [16]–[18]. This has few negations which are not important to English language, and right predictions can be made by omitting such words or phrases. The main contributions are:

- A sentiment analysis system is presented with improvised negation handling methods to classify the tweets.
- Feature extraction and classification is performed on the input data to generate different sets of features and classify the data into positive and negative labels.
- The performance of the classifiers is improved by using efficient negation handling rules.
- Analysis of deep learning models to learn which performs better on which features and comparison with machine learning methods.
- Negation handling algorithm for handling tweets with negation occurrence on Twitter dataset.

A negation handling method using artificial neural network (ANN) and convolutional neural network (CNN) is used solve the problem and achieve a better accuracy. Data cleaning is done and synsets are used to find the word similarity. Similarity score is calculated.

The rest of the paper is organized as follows: section 2 describes the proposed method. The construction of neural network model is explained in section 3. Section 4 presents our experimental results and necessary analysis. Finally, we conclude the paper in section 5.

2. METHOD

In this section improved negation handling for Twitter sentiment analysis is presented. It is a major research area to improve the classification performance. The dataset is distributed and labeled with positive, negative and neutral tag. Next, the system works on many different tasks namely, pre-processing, normalization, classification and finally evaluation. Figure 1 shows the workflow of the sentiment detection model.

2.1. Pre-processing

Various pre-processing steps are carried over to clean the dataset. POS tagging is performed using Twitter specific tagger and adjectives, adverb, nouns are used as keywords for negation. Entities such as punctuations, URL's, and hashtags are removed. A composition of POS and tweet tokens are obtained in a sentence after tokenization and passed as input. This is to divide the text into unigrams, bigrams and n-grams.

Normalization is done to convert out-of-vocabulary words to their canonical form and stemming and lemmatization is also performed to convert the words to their root form. The source is converted into lowercase. Stop word and noise removal are some of the common steps in pre-processing. Punctuations, white space removal, hashtags and URL's are removed from tweets. Duplicate characters are removed from the dataset. Example: asiiiiikkkkkkk into asik.

2.2. Feature engineering for deep neural networks

All the words from the training data are extracted in the previous stages. In this phase, morphological, negation, POS and lexicon features are extracted from tweets. A standard approach is used to experiment on each module. Word patterns and combination of morphemes are formed. The features extracted are: count of capitalized words, presence of duplicate characters in words, exclamation, punctuation and question marks, existence of subjective emoticons, existence of prefixes, suffixes and URL's and existence of question marks at the end of the sentence. Slang is not considered in negation handling as it detects sarcasm.

Sentiment-140 has tweets in English that automatically classifies the sentiment of particular product, topic or brand. It is used to extract the lexicon features. Based on the polarity the emotion in the text is determined. The tokens with non-zero sentiment score, the number of sentiment scores and maximum of score are generated. The features for all these tokens are obtained, positive tokens whose score is greater than zero, negative token score with '-ve' value and then unigrams and bigrams.

The negation extracted features are combined in column and the final vector is obtained. It is used to train the classifier and based on correlation a subset of features is selected using the filtering approach. Feature optimization is done using feature extraction technique.

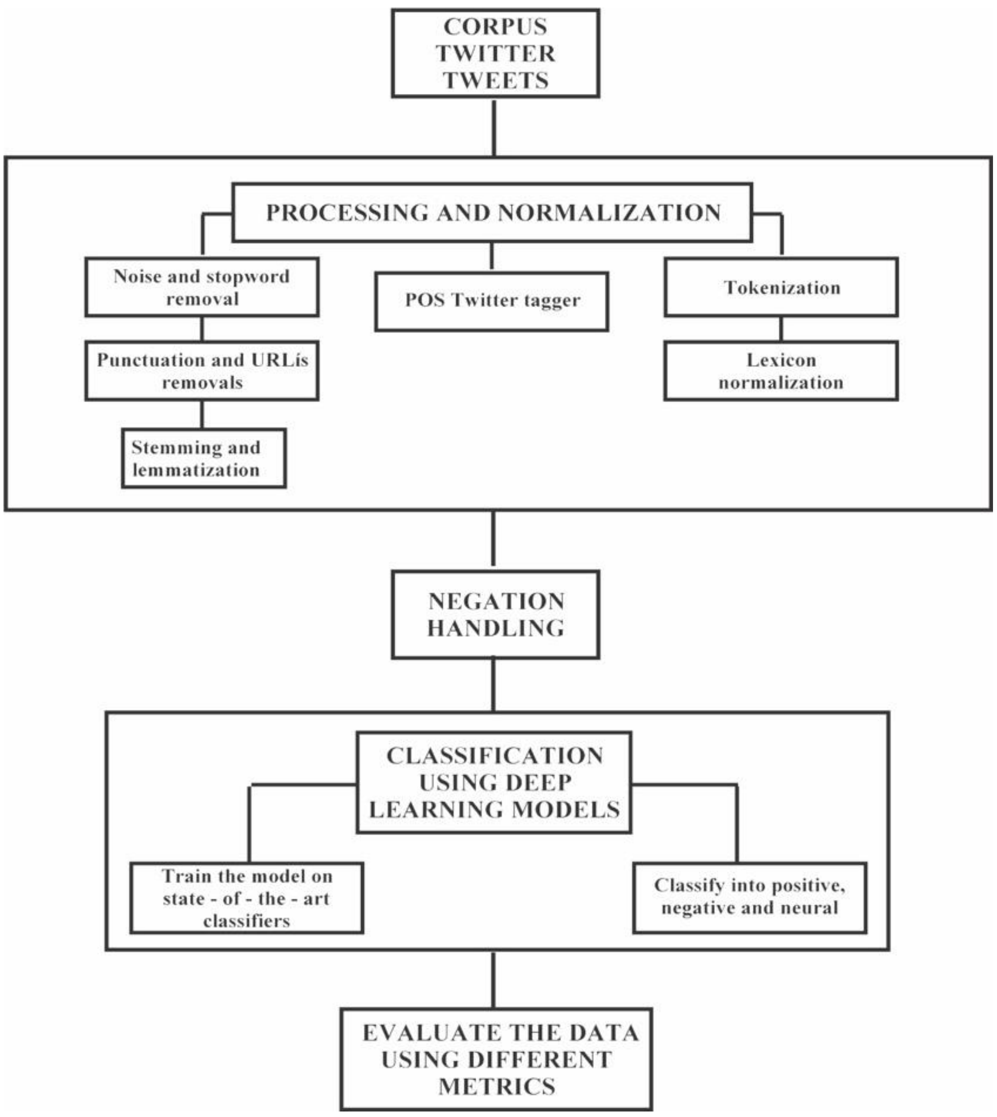
CORPUS TWITTER TWEETS
PROCESSING AND NORMALIZATION
Noise and stopword removal
POS Twitter tagger
Tokenization
Punctuation and URLís removals
Lexicon normalization
Stemming and lemmatization
NEGATION HANDLING
CLASSIFICATION USING DEEP LEARNING MODELS
Train the model on state - of - the - art classifiers
Classify into positive, negative and neural
EVALUATE THE DATA USING DIFFERENT METRICS

Figure 1. Workflow of the Twitter sentiment detection model

2.3. Negation handling

Because negation terms thoroughly alter the polarity of a sentence, handling negation is an important NLP application. It is a two-class classification problem where the objects are labeled as either positive or negative. When the object's containing negation is present, their meaning is reversed. An example of negation in text classification is represented in Table 1. The word to be negated is easily recognized in this example: the negative word "not" before the term "good" in the second sentence shows negation of positive term. This leads to negative/reversed classification where the predicted class of a sentence is flipped.

Table 1. Example of negation in text data determining whether the sentence is positive or negative

Sentence	Classification label
"It was a good movie"	Positive
"It was not a good movie"	Negative

2.3.1. Synset

It is a central part of WordNet (lexical database), also a part of natural language toolkit (NLTK) Python library used for English language. A collection of words is grouped together based on their meaning. Synsets can be used to identify negations in text and find the antonyms for the negation word. Instead of training complex models or applying word embeddings, that takes much longer processing time and power to identify the relation between different words, synset is a simple approach.

For the word next to negation term ("non-", "no", "never", and "un-"), a collection of subjective synonyms are generated. These synsets are semantically and lexically interlinked to each other in the WordNet lexical database by means of conceptual-semantic and lexical relations. Once the synsets are created, it checks for corresponding antonyms in the corpus. If 'not' is present, it is left as it is as there is no opposite entity. If present, a list of antonyms is created and dissimilarity co-efficient is found. It is shown in (1).

$$DS = (1 - w1.wup\_similarity(w2)) \tag{1}$$

where, DS-dissimilarity, and w1, w2-dissimilar words.

The wup_similarity method stands for Wu-Palmer similarity. It is a scoring method which is calculated based on the depths of two different synsets in the WordNet dictionary. It depends on how similar are the word senses and where the synsets are placed in the hypernym tree. It also considers the depth of least common subsume (LCS). The equation for Wu_Palmer is shown in (2).

$$Wu\_Palmer = 2 \times \frac{depth(lcs(syn1, syn2))}{depth(syn1)+depth(syn2)} \tag{2}$$

The similarity score cannot be zero as the depth of LCS is never zero, it can be 0<score<=1. Antonym with highest similarity is identified and substituted to the word followed by negation. Then, a sentence with inverted polarity is obtained after eliminating the negation word. By substituting antonym with maximum similarity, maximum polarity inversion is ensured. It prevents tshe words by being replaced with indirect/irrelevant antonyms. Algorithm 1 depicts the process of how synsets are used to handle negation on text data.

Tokenization is done to break the sentence into groups of words. The tokenization is divided into unigram, and bigrams. In this study unigram and bigram tokenization process is tested to obtain the most accurate SA process. It matches any word character in the document.

Algorithm 1. Synsets to find word similarity using synonyms and antonyms

```
Input: Tokenized sentence (a set of word tokens)
Output: Tokenized sentence after negation handled (a      list of words) by
replacing with antonym
1.  Initialize temp to integer value 0
2.  BEGIN:
3.    for sentence[i] within the dataset do
4.       if sentence[i-1] is a negation word 'not' or "n't"
5.          create empty list of antonyms
6.    for sentence[i] in the WordNet do
7.       return an array of synsets for sentence[i]
           w1 = syns[0].name() used to access one of the word elements
8.    for l in syn.lemmas() do
9.       if l.antonyms()
10.         all the antonym names are obtained and appended to the list antonmys = []
11. Initialize max_ dissimilarity = 0
12.   for ant in antonyms do
13.         syns = wordnet.synsets(ant)
14.        w2 = syns[0].name first antonym of current word
15.        syns = wordnet.synsets(sentence[i])
16.        w1 = syns[0].name() the current word itself
17.        if wup_ similarity() is an integer or float value
              temp= 1-word1.wup_similarity(word2)
18.       if temp > max_ disimilarity
19.            assign the antonym with highest similarity score to current word
20.   while sentence still contains 'not' or "n't" do
21.      remove using sentence.remove()
22.         return sentence after replacing with antonym
23. END
```

2.4. Training and classification

In the proposed system we have used CNN and ANN to classify the sentiments in both training and testing phase. The data is divided into 80-20 or 90-10 train and test percent ratio and evaluated using different metrics. In this phase, classification is performed on neural network classifiers. CNN and ANN are trained on the feature vectors generated in the previous step. The parameters are tuned on both the models with learning rate of 0.0002 and beta value of 0.9. This improves the accuracy of the proposed system. Deep-learning classifiers take less time to train compared to machine learning methods.

Label encoding is done to convert categorical variables, such as characters or strings into numerical form that can be used in machine learning algorithms. In this process, each unique category is assigned a unique integer value. For example, if we have a categorical feature like "color" with three possible values: "orange", "white", and "green", we can assign them integer values like 1, 2, and 4 respectively. However, it is important to note that label encoding can introduce an arbitrary order to categorical variables, which may not always be appropriate, and can lead to suboptimal model performance.

In the context of a Twitter dataset, the TF-IDF vectorizer can be used to convert the text of the tweets into numerical representations. This allows the machine learning methods to process the textual content, as they can only understand numerical inputs. It first computes the term frequency (TF), which is occurrence count of a word in a document divided by the total number of words. The inverse document frequency (IDF) is then calculated using log function. It is the ratio of total documents to number of documents containing a given term. The TF-IDF score for a word in a document is formulated as:

$$TF - IDF(w,c,M) = 1 + \log f_{w,M} \times \log \frac{|M|}{c \in M : w \in M} \quad (3)$$

where,
$f_{w,M}$-number of occurrences of word ’w’ in document ’M’
$|M|$-total number of documents in the corpus
$(c \in M : w \in M)$-number of documents in the dictionary, which has word ’w’.

The TF-IDF vectorizer takes tweets as input from a set of documents and returns a matrix of term frequency-inverse document frequency (TF-IDF) scores. Each row in this matrix represents a document and column shows a word. Machine learning algorithms use this matrix as input to represent text data in numerical form.

3. NEURAL NETWORK MODEL CONSTRUCTION

3.1. Convolutional neural networks

This model converts the data into tokens and performs encoding. The first layer in the network is embedding. By using to_categorical() method from the keras API, integer data can be encoded to binary vector form. As all the values of the class labels are represented, the to_categorical() function is used directly. Next, padding is performed to ensure that all sequences in the list are of same length. It is done using pad_sequence() by default, zero is added in the beginning until each sequence has the same length as the longest one. Padding in our work is done at the end of the sequence as the argument is set to 'post'. The input is a list of integers obtained from one_hot encode and the maximum text sequence length. Binary classification is performed using padded sequences as input in the neural network.

Embedding is the first hidden layer added to the model to represent input text data as dense vectors of fixed size, and these vectors are further processed by rest of the neural network. The idea is to learn a continuous representation for every word in the vocabulary, where similar words have same representations. Words such as "excellent" and "amazing" have positive sentiment and the vectors representing these words in embedding layer should be close together so the phrases such as "excellent job" and "amazing job" has similar score.

A 1D convolution layer with 64 filters and size of the convolutional kernel (stride) is 2 is added. Kernel size is used to slide over the input and compute dot products between the weights and the input. Rectified linear unit (ReLu) activation function is passed in all the CNN layers to provide a non-linear relationship for the output. Padding specifies whether to add padding to the input in order to control the spatial size of the output. In this case, "same" padding is used, which means that zero padding is added to the input such that the output has the same spatial size as the input.

Pooling layer reduces the dimensional complexity and still maintains the convolution information. It combines the vectors from the convolution into single dimensional vector. Max pooling is used in our model to represent most relevant features in the sentence and also help reduce overfitting. A pool size of 2×2 is selected to reduce the dimension of feature map and then the result is flattened and forwarded to the fully-connected layer for classification. Two fully-connected layers with 1,024 and 512 neurons and ReLu activation function is used to improve the network performance. All of the neurons in the flattening layer are fully connected to every neuron in this layer.

3.2. Artificial neural networks

The basic ANN architecture classify the input data, process it in hidden layers and produce output results. The weights are calculated and distributed amongst the layers. Each layer has multiple nodes and number of nodes are connected to the features in the data. Every layer has separate weights assigned to them, and it is combined with the input features and moves forward to the next layer. By back-propagating the predicted error to the layers before it, optimized weight values for each layer is attained by ANN. It is intended to improve the model's accuracy and reduce error so that optimal results are achieved.

A linear stack of layers is defined using the sequential class from Keras API. The architecture contains 3 dense layers defined with 1,024, 512 and 256 as dimensionality of output vectors and ReLu activation function for first three layers. The dense layer is created as the first layer in the network with input_dim features. Input to the model is the number of features extracted during training. Before training the model, the learning process is configured. This is done using compile() method with loss calculated using categorical cross entropy function, where the targets are one-hot encoder and Adam optimizer, which stands for adaptive moment estimation as arguments. beta1 and beta2 are the two exponential decay rates for the first and second moments (mean and variance) in the Adam optimizer, respectively. These parameters control the rate at which the moving averages of the gradient and squared gradient are updated. Once the model has been compiled, it is ready to be trained. The parameter settings in both the neural network model is presented in Table 2.

Table 2. Model parameter comparison

Layers	CNN		Layers	ANN		Activation function
	Parameters	Size		Parameters	Size	
Convolution 1D	64	2	Dense	1,024	2	ReLu
Pooling 1D	Max-pool	2×2	Dense	512	2	ReLu
Flatten			Dense	256	2	ReLu
Dense	1,024 neurons					ReLu
Dense	512 neurons					ReLu
Dense	2 neurons		Dense	2 neurons		Softmax

4. RESULT AND ANALYSIS

This section presents information on results and analysis conducted to determine the efficacy of our system. The proposed algorithm is implemented in Python 3.11 using jupyter notebook IDE. Keras with tensorflow is used to implement the deep learning models with fine-grained analysis at each step.

4.1. Corpus description

In this work, SemEval-2017 Task 4 Twitter dataset is used for evaluation of sentiment detection. For comparison we work on SemEval-2013 Task 2 Twitter data and show that our system outperforms the machine learning classifiers. SemEval-2017 is publicly available corpus with message-level polarity classification with both training and testing data. The dataset contains 49,496 tweets with positive, negative and neutral labelled sentences. The corpus is imbalanced with more number of neutral tweets. Figure 2 clearly shows the number of positive, negative and neutral tweets in the SemEval-2017 dataset.

4.2. Experiments

To demonstrate the effectiveness of the proposed method, we compare it with baseline machine learning methods for text sentiment classification [19]. Experiment was conducted on CNN and ANN models with maximum features of 20,001, one hot encoding for words and embedded vector of size 64 dimensions. According to the results we can conclude that CNN classifies the sentiment in twitter data better than the other models by first handling negation. In this work, the number of epochs is fixed to 50 and batch size 100. In ANN, the learning rate is set to 0.0002, and beta1 and beta2 are set to 0.9 and 0.999 respectively.

TensorFlow is an open-source machine learning framework designed to facilitate the development and deployment of machine learning models. It is an excellent alternative for experimenting after pre-processing, since it allows for fine tuning model parameters at each layer of the sequential model. Table 3 shows comparison of output of neural networks models with the machine learning models [20], [21] on different test datasets. Deep learning models perform better with highest accuracy as feature extraction is handled automatically within the model. But, machine learning models need to decide upon the features useful to perform classification when a text is given. CNN outperforms the other methods with an accuracy of 98.6%.

Table 4 compares the proposed method to the most recent and baseline classifiers for sentiment detection in message and term level tasks. It is a benchmark for the other classifiers and SVM model. The comparison is shown using macro-averaged F1-score metric which is average of both positive and negative class, as it was selected as a primary metric by winning team of SemEval-2013. An F1-score of 95.6% on test data is achieved, which is better than the other models.

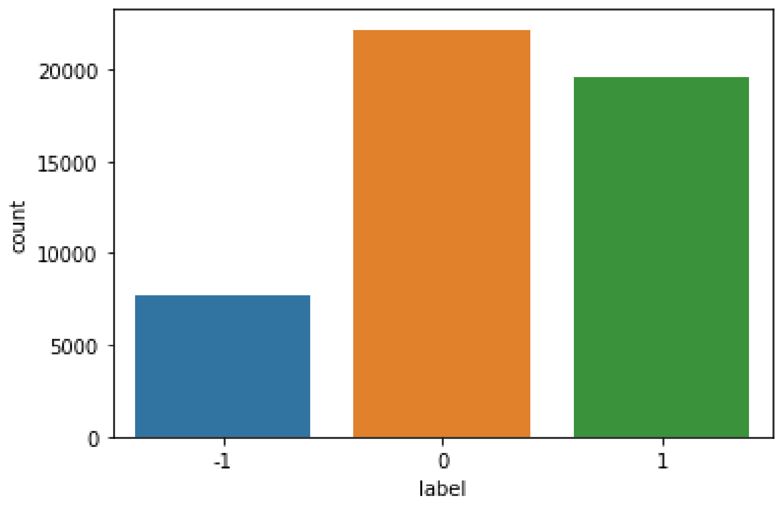

Figure 2. Dataset distribution

Table 3. Performance comparison of state-of-the-art classifiers

Model	SemEval 2013 (%)	SemEval 2017 (%)
SVM	71.4	75.3
Naïve Bayes	62	64
Decision tree	65.4	68.2
ANN	80.6	94.1
CNN	87.3	98.6

Table 4. Macro-avg F1 score obtained on the test dataset

System	Macro-avg F1-score (%)
NRC-Canada [22]	69.02
SVM	68
CNN	95.6

Now, we experimented with the trained network models to demonstrate the significance of negation handling during the process of sentiment analysis. Table 5 shows a comparison of the system with and without negation being handling in terms of accuracy [23]–[25]. The results show that the performance of CNN and ANN classifier increases by 3% to 4% points when negation is handled.

Table 5. Significance of negation handling in Twitter sentiment classification of text

Models	Accuracy (%)
CNN+negation	98.6
CNN (without negation)	92.5
ANN+negation	94.1
ANN (without negation)	90.6

Figure 3 shows the results of test accuracy and loss with learning epochs. The accuracy of CNN reaches 98.6% at 30 epochs. The Adam optimizer parameter finds the best weights to minimize the loss and thus accuracy increases. Figure 4 plots the test accuracy gain and loss in ANN against the epochs. For 21 epochs ANN achieves 94% classification accuracy. The loss is reduces gradually from epoch 1 to epoch 21.

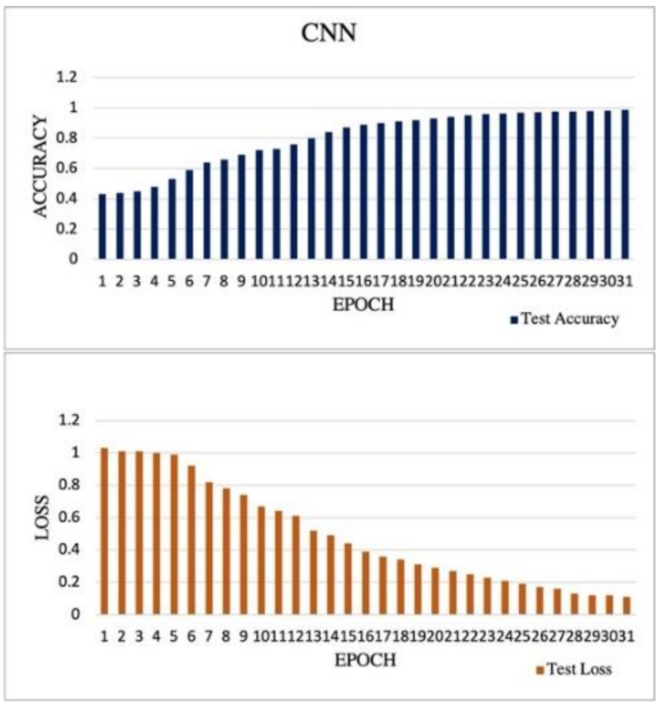

Figure 3. Test classification accuracy and loss on CNN

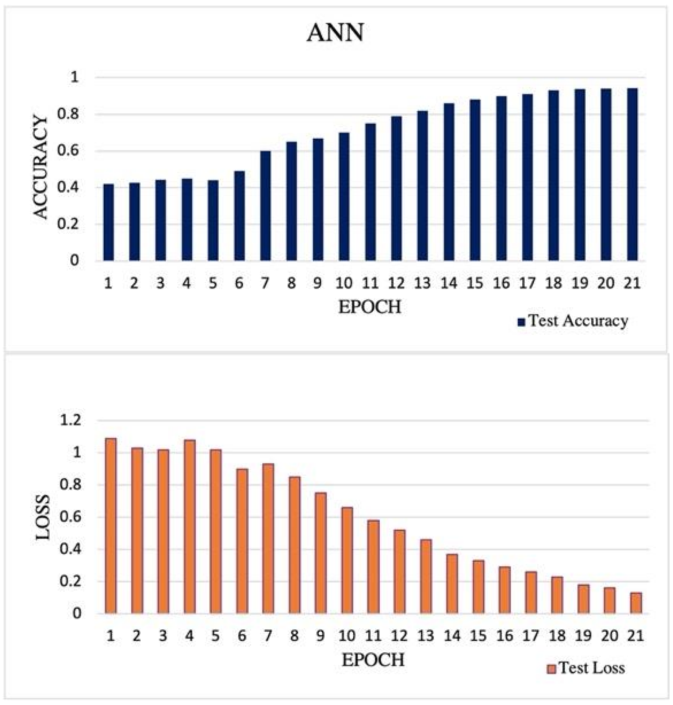

Figure 4. Test classification accuracy and loss on ANN

5. CONCLUSION

The proposed work shows a performance comparison of ANNs and CNNs for sentiment classification using negation handling on Twitter dataset SemEval-2017. The results showed that CNNs outperformed ANNs in handling negation words and achieved a higher accuracy in sentiment analysis. It highlights the importance of considering negation words in sentiment analysis and demonstrates the effectiveness of using deep learning models for negation handling. This shows that the sentiment classification can be improved by performing too many preprocessing techniques on the data. This study provides insights for future research in sentiment analysis and negation handling, particularly in improving the structure of data and performance of deep learning models.

REFERENCES

[1] J. Kocon, A. Janz, and M. Piasecki, "Context-sensitive sentiment propagation in WordNet," in *GWC 2018 - 9th Global WordNet Conference*, 2018, vol. 2018-Janua, pp. 329–334.

[2] P. K. Singh and S. Paul, "Deep learning approach for negation handling in sentiment analysis," *IEEE Access*, vol. 9, pp. 102579–102592, 2021, doi: 10.1109/ACCESS.2021.3095412.

[3] R. Amalia, M. A. Bijaksana, and D. Darmantoro, "Negation handling in sentiment classification using rule-based adapted from Indonesian language syntactic for Indonesian text in Twitter," *Journal of Physics: Conference Series*, vol. 971, no. 1, p. 12039, Mar. 2018, doi: 10.1088/1742-6596/971/1/012039.

[4] T. G. Prahasiwi and R. Kusumaningrum, "Implementation of negation handling techniques using modified syntactic rule in Indonesian sentiment analysis," *Journal of Physics: Conference Series*, vol. 1217, no. 1, p. 12115, May 2019, doi: 10.1088/1742-6596/1217/1/012115.

[5] A. Khandelwal and S. Sawant, "NegBERT:a transfer learning approach for negation detection and scope resolution," in *LREC 2020 - 12th International Conference on Language Resources and Evaluation, Conference Proceedings*, 2020, pp. 5739–5748.

[6] I. Gupta and N. Joshi, "Tweet normalization: a knowledge based approach," in *2017 International Conference on Infocom Technologies and Unmanned Systems: Trends and Future Directions, ICTUS 2017*, Dec. 2018, vol. 2018-Janua, pp. 157–162, doi: 10.1109/ICTUS.2017.8285996.

[7] J. Reitan, J. Faret, B. Gambäck, and L. Bungum, "Negation scope detection for twitter sentiment analysis," in *6th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis, WASSA 2015 at the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015 - Proceedings*, 2015, pp. 99–108, doi: 10.18653/v1/w15-2914.

[8] R. S. Patil and S. R. Kolhe, "Supervised classifiers with TF-IDF features for sentiment analysis of Marathi tweets," *Social Network Analysis and Mining*, vol. 12, no. 1, p. 51, Dec. 2022, doi: 10.1007/s13278-022-00877-w.

[9] R. Keeling *et al.*, "Empirical comparisons of CNN with other learning algorithms for text classification in legal document review," in *Proceedings - 2019 IEEE International Conference on Big Data, Big Data 2019*, Dec. 2019, pp. 2038–2042, doi: 10.1109/BigData47090.2019.9006248.

[10] S. Rosenthal, N. Farra, and P. Nakov, "SemEval-2017 Task 4: Sentiment analysis in Twitter," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 2017, pp. 502–518, doi: 10.18653/v1/s17-2088.

[11] A. Sharma and S. Dey, "A document-level sentiment analysis approach using artificial neural network and sentiment lexicons," *ACM SIGAPP Applied Computing Review*, vol. 12, no. 4, pp. 67–75, Dec. 2012, doi: 10.1145/2432546.2432552.

[12] F. Strohm, "The impact of intensifiers , diminishers and negations on emotion expressions," University of Stuttgart, 2017.

[13] E. H. Ali, H. A. Jaber, and N. N. Kadhim, "New algorithm for localization of iris recognition using deep learning neural networks," *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, vol. 29, no. 1, pp. 110–119, Jan. 2023, doi: 10.11591/ijeecs.v29.i1.pp110-119.

[14] T. Thomas, N. Pradhan, and V. S. Dhaka, "Comparative analysis to predict breast cancer using machine learning algorithms: a survey," in *Proceedings of the 5th International Conference on Inventive Computation Technologies, ICICT 2020*, Feb. 2020, pp. 192–196, doi: 10.1109/ICICT48043.2020.9112464.

[15] E. H. Ahmed, M. R. M. Alsemawi, M. H. Mutar, H. O. Hanoosh, and A. H. Abbas, "Convolutional neural network for the detection of coronavirus based on X-ray images," *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, vol. 26, no. 1, pp. 37–45, Apr. 2022, doi: 10.11591/ijeecs.v26.i1.pp37-45.

[16] P. Mukherjee, Y. Badr, S. Doppalapudi, S. M. Srinivasan, R. S. Sangwan, and R. Sharma, "Effect of negation in sentences on sentiment analysis and polarity detection," *Procedia Computer Science*, vol. 185, pp. 370–379, 2021, doi: 10.1016/j.procs.2021.05.038.

[17] W. Sharif, N. A. Samsudin, M. M. Deris, and R. Naseem, "Effect of negation in sentiment analysis," in *2016 6th International Conference on Innovative Computing Technology, INTECH 2016*, Aug. 2017, pp. 718–723, doi: 10.1109/INTECH.2016.7845119.

[18] M. Sobhana, S. C. Chaparala, D. N. V. S. L. S. Indira, and K. K. Kumar, "A disaster classification application using convolutional neural network by performing data augmentation," *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, vol. 27, no. 3, pp. 1712–1720, Sep. 2022, doi: 10.11591/ijeecs.v27.i3.pp1712-1720.

[19] M. M. Hossain, D. Chinnappa, and E. Blanco, "An analysis of negation in natural language understanding corpora," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 2022, vol. 2, pp. 716–723, doi: 10.18653/v1/2022.acl-short.81.

[20] V. L. H. Josephine, A. P. Nirmala, and V. L. Alluri, "Impact of hidden dense layers in convolutional neural network to enhance performance of classification model," in *IOP Conference Series: Materials Science and Engineering*, Apr. 2021, vol. 1131, no. 1, p. 012007, doi: 10.1088/1757-899x/1131/1/012007.

[21] I. Gupta and N. Joshi, "Feature-based Twitter dentiment snalysis with improved negation handling," *IEEE Transactions on Computational Social Systems*, vol. 8, no. 4, pp. 917–927, Aug. 2021, doi: 10.1109/TCSS.2021.3069413.

[22] S. M. Mohammad, S. Kiritchenko, and X. Zhu, "NRC-Canada: building the state-of-the-art in sentiment analysis of tweets," *SEM 2013 - 2nd Joint Conference on Lexical and Computational Semantics*, vol. 2, pp. 321–327, 2013.

[23] U. Farooq, H. Mansoor, A. Nongaillard, Y. Ouzrout, and M. A. Qadir, "Negation handling in sentiment analysis at sentence level," *Journal of Computers*, vol. 12, no. 5, pp. 470–478, 2017, doi: 10.17706/jcp.12.5.470-478.

[24] U. Lal and P. Kamath, "Effective negation handling approach for sentiment classification using synsets in the WordNet lexical database," in *2022 1st International Conference on Electrical, Electronics, Information and Communication Technologies, ICEEICT 2022*, Feb. 2022, pp. 1–7, doi: 10.1109/ICEEICT53079.2022.9768641.

[25] X. Zhu, S. Kiritchenko, and S. M. Mohammad, "NRC-Canada-2014: recent improvements in the sentiment analysis of tweets," in *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, 2014, pp. 443–447, doi: 10.3115/v1/s14-2077.

BIOGRAPHIES OF AUTHORS

Mrs. Mamatha Mylarappa pursued Bachelor of Engineering from University Visveswaraya College of Engineering, Bangalore University, India in 2008 and Master of Technology from Dr. Ambedkar Institute of Technology, VTU in year 2011. She is currently pursuing Ph.D. Her main research work focuses on sentiment analysis, data mining and multimodal analysis. She has 4 years of teaching experience. She can be contacted at email: mmtha.s@gmail.com.

Mr. Shiva Kumar B.N has completed Bachelor of Engineering from T John Institute of Technology, Bangalore in 2020 and Masters of Engineering from University of Visvesvaraya College of Engineering, Bangalore in 2023. His research interests include data mining. He can be contacted at email: shivakumar15gowda@gmail.com.

Dr. Thriveni J. Gowda has completed Bachelor of Engineering, Masters of Engineering and Doctoral Degree in Computer Science and Engineering. She has 4 years of industrial experience and 28 years of teaching experience. Currently she is Professor in Department of CSE, University of Visvesvaraya College of Engineering, Bangalore. She has over 120 research papers to her credit. She has produced nine doctorate students and currently guiding 08 Ph.D. students. Her research interests include computer networks, web mining, IoT, cloud computing and security. She can be contacted at email: drthrivenij@gmail.com.

Dr. Venugopal K. Rajuk is the Former Vice Chancellor of Bangalore University. He was the Principal of University of Visvesvaraya College of Engineering (UVCE) for 15 years, Chairman, Department of Electronics and Communications, Computer Science Engineering and Information Science Engineering in UVCE. He has a distinguished and brilliant academic career with Eleven Degrees including two Ph.D.'s, one in Economics from Bangalore University and another Ph.D. in Computer Science and Engineering from IIT Madras. He has authored and edited 80 books on Computer Science and Economics, which include Petrodollar and the World Economy, C Aptitude, Mastering C, Microprocessor Programming, Mastering C++, and Digitial Circuits and Systems. Out of the 101 patents he has filed, 34 have been granted. During his four decades of service at UVCE has has over 1,000 research papers to his credit and has awarded Ph.D.'s to 30 students. His research interests include computer networks, wireless sensor networks, parallel and distributed systems, digital signal processing, and data mining. He is a Fellow of IEEE and an ACM Distinguished Educator. He can be contacted at email: venugopalkr@gmail.com.