Robust Sarcasm Detection using Artificial Rabbits Optimizer with Multilayer Convolutional Encoder-Decoder Neural Network on Social Media

A. Palaniammal1, P. Anandababu2

1, 2Department of Computer and Information Science, Annamalai University.

1Corresponding Author : revathy20k@gmail.com

Received: 01 March 2023 Revised: 08 April 2023 Accepted: 02 May 2023 Published: 31 May 2023

Abstract - Nowadays, posting sarcastic comments on media platforms developed a general trend. People to pester or taunt others frequently utilize sarcasm. It is regularly stated that with tonal stress, inflexion from the speech or in the procedure of hyperbolic, lexical, and pragmatic aspects occur from the textual data. Sarcasm Detection (SD) utilizing Deep Learning (DL) on media platforms is an active study field in Natural Language Processing (NLP). Sarcasm is a figurative language method frequently exploited on social networks like Reddit, Twitter, and Facebook. Detecting sarcasm is essential to various applications like Sentiment Analysis (SA), opinion mining, and social network monitoring. DL techniques are demonstrated that effectual at sarcasm detection on media platforms. This study presents a robust sarcasm detection using Artificial Rabbits Optimizer with Multilayer Convolutional Encoder-Decoder Neural Network (ARO-MCEDNN) technique on social media—the presented ARO-MCEDNN technique concentrations on detecting sarcasm in social networking sites. Primarily, the ARO-MCEDNN technique follows a series of pre-processing data levels for transforming the input data into a compatible format. Followed by, Glove approach is applied for word embedding purposes. Moreover, the MCEDNN model is applied as a classification model to identify and categorize distinct kinds of sarcasm. Furthermore, the ARO algorithm is chosen as a hyperparameter optimizer of the MCEDNN model, enhancing the sarcasm detection performance. To highlight the advanced performance of the ARO-MCEDNN system, a sequence of simulations was performed.

Keywords - Social media, Natural language processing, Deep learning, Glove approach, Artificial rabbit optimizer.

1. Introduction

Sarcasm is widely adopted in social media platforms and microblogging websites in which individuals reprimand or mock, making it hard even for humans to differentiate whether what is told is what is intended [1]. The metaphorical way of sarcasm makes it a frequently cited threat for sentiment investigation. This sarcasm has an indirect pessimist sentiment but an optimistic exterior sentiment [2]. The key challenge and the merits of SD to SA have led to attention to Automatic SD (ASD) as a research issue [3]. ASD denotes estimative methods that foresee whether a specified text is in a sarcastic representation. In the text classification, SD is a critical tool with several suggestions for many fields encompassing health [4], safety, and sales. With the assistance of SD procedures, institutions can examine their customers’ emotional state toward their products. This offers significant help for those companies/institutions to enrich their product worth [5]. In SA, the classification of sarcasm is a crucial sub-work, particularly in categorizing tweets, for assigning implicit data within the note that a person conveys or shares among others [6]. Additionally, the system of the tweet may also be implemented in anticipating sarcasm, for example, converting the conflicts of a pessimistic/optimistic remark to its contradictory form. In Twitter, various problems make SA a challenging process.

SD signifies the implementation of DL/ML, Natural Language Processing (NLP), and statistical procedures for distinguishing satire or cynicism positioning for various impurities of a communication, text message, or document layers [7-9]. This is also often dignified as the binary classification issue. The previous research in predicting an ironic statement primarily intended on regulation-based and arithmetic procedures (i) the existence of sentiment alterations, interjections, punctuations, etc., and (ii) pragmatic and lexical structures [10]. A Deep Neural Network provides a process for fundamental crucial learning methods, alternatively implementing hand-engineered methods [11, 12]. The DL procedure has performed a progressive routine in diverse NLP structures like text
message narration, responding to queries, and machine translation [13]. This DL procedure has been researched in SD and appears to achieve great motivation.

This study presents a robust sarcasm detection using Artificial Rabbits Optimizer with Multilayer Convolutional Encoder-Decoder Neural Network (ARO-MCEDNN) technique on social media. Primarily, the ARO-MCEDNN technique follows a series of pre-processing data levels to alter the input information into a compatible format. Succeeded by the Glove approach is applied for word embedding purposes. Moreover, the MCEDNN model is applied as a classification model to identify and categorize distinct kinds of sarcasm. Furthermore, the ARO algorithm is chosen as a hyperparameter optimizer of the MCEDNN model, enhancing the sarcasm detection performance. To highlight the improved outcomes of the ARO-MCEDNN system, a sequence of simulations is executed.

2. Related Works

Elkamchouchi et al. [14] present a state-of-the-art (HCOA-SACDC) approach called Hosted Cuckoo Optimizer Algorithm with SAE-Enabled SD and Classification method. This HCOA-SACDC technique mainly focused on the classification and SD in the OSN atmosphere. Likewise, for practical feature extraction, the TF-IDF technique was utilized. Besides, for categorizing and recognizing sarcasm, the SAE model was utilized. Pandey and Singh [15] projected a hybrid BERT-LSTM technique.

A pretrained BERT method was utilized for creating embedding for the code-mixed database. Sharma et al. [16] devised a new hybrid method using an AE called an embedding-based approach. The structure devises using sentence embedding from LSTM-AE, universal sentence encoder and bidirectional encoder representation transformers. The text over images was considered for managing multimedia content like videos and images. The last structure was modelled after an ablation study of different hybrid fusions of methods.

Ren et al. [17] offer multilayer memory networking by employing sentiment semantics to capture sarcastic terminologies’ attributes. The author utilised the first-level memory network in this technique for capturing sentiment semantics and the contrast between the situation and sentiment semantics in all sentences. The second-level memory networking is employed. Zhang et al. [18] devised a novel SLSD (stance-level sarcasm detection) task. In contrast, the aim is to expose the latent stances of the author and dependent upon them to find the sarcastic polarity from the textual data. The authors present an innovative stance-centred graph attention network (SCGAT) and an integral structure, which entails BERT.

Zhao et al. [19] offer a Coupled-Attention Network (CAN), which can successfully integrate text information and images as a unified structure, therefore realizing the fusion of various resources.

3. The Proposed Model

In this research, we have presented a new ARO-MCEDNN model for SD in social media. The presented ARO-MCEDNN system emphasizes sarcasm detection and classification in social networking sites. It follows a four-stage process: data pre-processing, Glove based word embedding, MCEDNN-based sarcasm detection, and ARO-based hyperparameter tuning. Fig. 1 expresses the workflow of the ARO-MCEDNN model.

3.1. Data Pre-processing

A primary step, the ARO-MCEDNN technique follows a series of pre-processing data levels to convert the input dataset into a compatible form. Text pre-processing cleans the actual textual data [20]. A textual pre-processed model was crucial for application on NLP tasks. Each text module reached and then pre-processed serves as the vital input elements provided as textual information application. Pre-processed contains many approaches for translating the unique text as a well-determined method: lexical analysis (ignoring case sensitivity, removing punctuations, symbols or special characters, and word tokenization), lemmatization and removal of stop words.

3.2. Glove Word Embedding

In this work, the Glove approach is applied for word embedding purposes. The gloVe is a Stanford-established un-supervised learning approach which generates word vector depictions [21]. GloVe is a word-embedded system which utilizes co-occurrence matrices for computing word resemblances from the corpus and trains a regression approach for learning word vectors which will capture the connections amongst wordings dependent upon frequently they can appear together in the corpus—gloVe mechanism on two essential elements like nearest neighbours and linear sub-structures.

The cosine similarity between two words is utilized for measuring the neighbourhood distance. The cosine similarity defines that the exact two wordings are semantically or linguistically. The resemblance metrics generated in this neighbourhood estimation comprise a single scalar which gauges the connection among two wordings. While two sentences generally have further complex connections than a single number is defined, this simplicity can require that revision. The model needs to connect one or more numbers with the word pair to statistically capture the data desired to distinguish between two exact words.
3.3. Sarcasm Detection by Implementing MCEDNN Model

At this stage, the MCEDNN model is applied as a classification model to identify and categorize different types of sarcasm. The MCEDNN approach depended on an encoding-decoding structure with several layers of convolutional and attention mechanisms [22]. Fig. 2 represents the structure of CEDNN. These methods are given training on wordings with rare wording segmentation as subwords. Let us input source sentence $S$ provide as an order of $m$ source tokens, $s_1, ..., s_m$ and $s_i \in V_s$, whereas $V_s$ implies the source vocabulary. The source token can be embedded from continuous space as $s_1, ..., s_m$. The embedded $s_i \in \mathbb{R}^d$ is provided as $s_i = w(s_i) + p(i)$, whereas $w(S)$ refers to the word embedded, and $p(i)$ signifies the position embedded equivalent to the position $i$ of token $s_i$ from the source sentence. Both embedded can be acquired in embedding matrices, which are given training and other networking parameters. The encoding and decoding are composed of $L$ layers each. The source token embeddings, $s_1, ..., s_m$, are linearly mapped for obtaining input vectors of the first encoded layer, $h_1^0, ..., h_m^0$, in which $h_i^0 \in \mathbb{R}^h$. Moreover, $h$ refer to the input and output dimensional of every encoded and decoded layer. By adding the biases $b \in \mathbb{R}^h$ and multiplying a vector with weighted $W \in \mathbb{R}^{h \times d}$ the linear mapping was completed:

$$h_i^0 = Ws_i + b \quad (1)$$

During the 1st encoded layer, $2h$ convolution filters of size $3 \times h$ mapping all the sequences of 3 consecutive input vectors to feature vector $f_i^1 \in \mathbb{R}^{2h}$. Paddings (represented by $<pad>$) are supplemented at the start and end of the source sentence to retain a similar count of resultant vectors as the source tokens.

$$f_i^1 = \text{Conv}(h_{i-1}^1, h_i^1, h_{i+1}^1) \quad (2)$$

In Eq. (2), Conv(.) is the convolutional function. This remains a non-linearity utilizing gated linear units (GLU):

$$\text{GLU}(f_i^1) = f_i^{1,u} \circ \sigma(f_i^{1,v}) \quad (3)$$

In Eq. (3), $\text{GLU}(f_i^1) \in \mathbb{R}^h$, $\circ$ and $\sigma$ define the component-wise multiplication and sigmoid activation functions correspondingly, and $f_i^{1,u,v}$ signifies the elements of $f_i^1$ in indices $u$ to $v$. The input vector to the encoded layer is supplemented as the residual connection. The resultant vector of the $l^{th}$ the encoded layer is provided as follows:

$$h_i^l = \text{GLU}(f_i^l) + h_{i-1}^l \quad i = 1, ..., m \quad (4)$$

All the resultant vectors of the last encoded layer, $h_i^l \in \mathbb{R}^h$ was linearly mapped to obtain the encoded resultant vector, $e_l \in \mathbb{R}^d$, utilizing weighted $W_e \in \mathbb{R}^{d \times h}$ and biases $b_e \in \mathbb{R}^d$.
Now, assume the generation of target words $t_n$ at the $n^{th}$ time step in decoded, with $n - 1$ target wordings priorly created. All the embedded $t_j \in \mathbb{R}^d$ was linearly mapped to $g^0_j \in \mathbb{R}^d$ and passed as input to the first decoded layer. During all the decoded layers, convolutional functions followed by non-linearity can be executed on the preceding decoded layer resultant vectors. $g^{l-1}_j$, whereas $j = 1, ..., n$:

$$y^l_j = GLU(Conv(g^{l-1}_{j-3}, g^{l-1}_{j-2}, g^{l-1}_{j-1}))$$  \hspace{1cm} (6)

In Eq. (6), $Conv(\cdot)$ and $GLU$ correspondingly denote the convolutional and non-linearity, and $y^l_j$ develops the decoded layer at the $j^{th}$ time step from the $l^{th}$ decoded layer. The complex filtering’s number and dimensions are much similar to the individuals from the encoded. All the decoded layer has their attention component. For computing attention at layer $l$ before forecasting the target token at the $n^{th}$ time step, the decoded state $y^l_n \in \mathbb{R}^d$ is linearly mapped to the $d$-dimension vector with weight $W_x \in \mathbb{R}^{d \times h}$ and bias $b_x \in \mathbb{R}^d$, adding the preceding target tokens embedded:

$$z^l_n = W_x y^l_n + b_x + t_{n-1}$$ \hspace{1cm} (7)

The attention weighted $a^l_{n,i}$ can be evaluated by the dot product of encoded output vectors $e_1, ..., e_m$ with $z^l_n$ and normalizing with softmax:

$$a^l_{n,i} = \frac{\exp(e^l_i z^l_n)}{\Sigma_{k=1}^m \exp(e^l_k z^l_n)} \hspace{1cm} i = 1, ..., m$$ \hspace{1cm} (8)

The source context vector $x^l_n$ was evaluated by executing the attention weighted to the encoded output vector summary and the source embedded.

$$x^l_n = \Sigma_{i=1}^m a^l_{n,i} (e_i + s_i)$$ \hspace{1cm} (9)

Then, the context vector $x^l_n$ is mapped linearly to $c^l_n \in \mathbb{R}^d$. The resultant vector of the $l$-th decoder layer, $g^l_n$, denotes the sum of $c^l_n$, $y^l_n$, and the preceding layer’s resultant vector $g^{l-1}_n$.

$$g^l_n = y^l_n + c^l_n + g^{l-1}_n$$ \hspace{1cm} (10)

The last decoder layer output vector $g^l_n$ is mapped linearly to $d_n \in \mathbb{R}^d$. Dropout is employed at the decoding output, embedding, and prior to each encoding and decoding layer. Then, the vector of the decoding output is mapped to the vocabulary size ($V$) of the target, and softmax can be evaluated for obtaining the probability of the target word.

$$0_n = W_0 d_n + b_0 \in \mathbb{R}^{V \times d}, b_0 \in \mathbb{R}^V$$ \hspace{1cm} (11)

$$p(t_n = w_1 | t_1, ..., t_{n-1}, S) = \frac{\exp(o_{n,i})}{\Sigma_{k=1}^V \exp(o_{n,k})}$$ \hspace{1cm} (12)

Where $w_i$ denotes the $i^{th}$ word in the target vocabulary $V_t$.

### 3.4. Hyperparameter Tuning by Implementing ARO Algorithm

At last, the ARO model is chosen as a tuning optimizer of the MCEDNN model, enhancing the sarcasm detection performance. The ARO model stimulates the natural behaviour of rabbits during random hiding and detour foraging [23]. Where $rand$ denotes a random integer within [0,1]. using Eq. (13), the initial population can be randomly generated.

$$X = rand \times (UB - LB) + LB$$ \hspace{1cm} (13)

In all the iterations, the optimization method is performed using the three fundamental strategies: energy shrinking, detour foraging, and random hiding. The detour foraging method concentrates on discovering novel solutions and, through using (14) to (18), is mathematically modelled. If $f(X_i^{t+1})$ is lesser than $f(X_i^t)$, the solution position $X_i^t$ is replaced with $X_i^{t+1}$. $f(X_i)$ characterizes the fitness value corresponding to the solution $X_i$.

$$X_i^{t+1} = X_i^t + R. (X_i^t - X_f^t) + round(0.5(0.05 + r_i)) \hspace{1cm} n_i$$ \hspace{1cm} (14)

$$R = (e - e^{(c-1)/2}) \times \sin(2\pi r_2)$$ \hspace{1cm} (15)

$$c = \begin{cases} 1, & \text{if } k == g(k) \ k = 1, ..., d \text{ and } t = 1, ..., \lfloor r_3, d \rfloor \end{cases}$$ \hspace{1cm} (16)

$$g = randperm(d)$$ \hspace{1cm} (17)

$$n_i, N(0,1)$$ \hspace{1cm} (18)

$$f_i^t = \begin{cases} f_i^{t+1}, & \text{iff } f_i^{t+1} < f_i^t \\ f_i^t, & \text{if } f_i^{t+1} \geq f_i^t \end{cases}$$ \hspace{1cm} (19)

$$X_i^t = \begin{cases} X_i^{t+1}, & \text{if } f_i^{t+1} < f_i^t \\ X_i^t, & \text{if } f_i^{t+1} \geq f_i^t \end{cases}$$ \hspace{1cm} (20)

Where $X_i^{t+1}$ denotes the next location of the present solution $X_i^t$, $r_1$, $r_2$ and $r_3$ indicate random numbers within [0,1]. $T$ denotes the maximal amount of iterations, and $d$ and $n$ denote the problem dimension and the population size, correspondingly.
Note that the ceiling function, round point to rounding towards the nearest integer, and \textit{randperm} specifies an integer permutation from 1 to \( d \) arbitrarily. The next phase is expressed by Eq. (21) to (23). If \((X_i^{t+1})\) is lesser than \((X_i^t)\), the solution position \(X_i^{t+1}\) is replaced with \(X_i^t\)

\[
X_i^{t+1} = X_i^t + R \times (r_4 \times b_{ir}^t - X_i^t) \quad (21)
\]

\[
b_{ir} = \frac{X_i^t + \frac{T-t+1}{T} \cdot r_4 \cdot g_r \cdot X_i^r}{0,\text{else}} \quad (22)
\]

\[
gr = \begin{cases} 
1, & \text{if } k = [r_5, d], l = 1, \ldots, d \\
0, & \text{else}
\end{cases} \quad (23)
\]

Where \( b_{ir}^t \) signifies a burrow chosen randomly wise from \( d \) burrows for hiding. \( r_4 \) and \( r_5 \) indicate random numbers between zero and one.

Lastly, the energy shrink phase balances the exploration and exploitation features. \( r \) denotes the random integer within \([0, 1]\). When \( A > 1 \), the exploration stage is carried out. Alternatively, else, the exploitation phase will be applied.

\[
A = 4 \left( 1 - \frac{t}{T} \right) \ln \left( \frac{1}{t} \right) \quad (24)
\]

When the exploration and exploitation phases are applied, the location of the better solution is upgraded when \( f(X_i^{t+1}) \) is greater than \((X_i^{t})\), as follows.

\[
f_{best}^t = \begin{cases} 
 f_i^{t+1}, & \text{if } f_i^{t+1} < f_{best}^t \\
 f_{best}^t, & \text{if } f_i^{t+1} \geq f_{best}^t
\end{cases}
\]

\[
X_{best}^t = \begin{cases} 
 X_i^{t+1}, & \text{if } f_i^{t+1} < f_{best}^t \\
 X_{best}^t, & \text{if } f_i^{t+1} \geq f_{best}^t
\end{cases}
\]

Algorithm 1: Pseudocode of ARO Algorithm

Define population size (\( N_p \)), problem dimensions (\( D \)), upper and lower boundaries, and stop criterion (\( T \)).
Set ARO parameter.
Population initialization (\( X \)).
Calculate cost function (1).
Allocate a solution that violates the constraint with a higher penalty value.
Describe better and worse solutions.
For \( r=2 : T \)
For \( i=1 : N_p \)
Compute fitness values of novel solutions (\( f_i^{t+1} \)).
Assure the constraint and include a cost of penalty if the novel solution interrupts the constraint.
Upgrade individual locations and the better solution location.
Else
Describe the novel location of every individual (\( X_i^{t+1} \)) in \( X \) based on exploring ARO’s Eqs (21) to (23).
Compute fitness values of new solutions (\( f_i^{t+1} \)).
Ensure problem constraint and include the penalty cost if the novel solution disrupts the constraint.
Upgrade the individual location and the better solution location.
End if
End for
End for

The selection of fitness function is a critical feature in the ARO model. Exploiting is done for solution encoding to evaluate the aptitude (goodness) of the candidate solution. Later, the value of accuracy is the primary condition used for the FF design.

\[
Fitness = \max (P) \quad (27)
\]

\[
P = \frac{TP}{TP+FP} \quad (28)
\]

Whereas TP and FP signify the true and false positive values.

4. Experimental Validation
In this study, the SD outputs of the ARO-MCEDNN approach can be examined on two datasets: Twitter 2013 dataset [24, 25] with 1956 samples and the headlines-2019 dataset [26, 27] with 28619 samples, as denoted in Table 1. The confusion matrix of the ARO-MCEDNN method on the sarcasm detection process under the Twitter 2013 dataset is given in Fig. 3. The results indicate that the ARO-MCEDNN technique correctly identifies sarcastic and non-sarcastic sweets. For instance, with 70% of TRP, the ARO-MCEDNN technique recognizes 212 sarcastic and 1134 non-sarcastic tweets. Additionally, with 30% of TSP, the ARO-MCEDNN approach recognizes 83 sarcastic and 496 non-sarcastic tweets. At last, with 80% of TRP, the ARO-MCEDNN system recognizes 242 tweets with sarcasm and 1305 without sarcasm. Table 2 gives a comprehensive sarcasm recognition outcome of the ARO-MCEDNN technique under the Twitter 2013 dataset. The results imply the ARO-MCEDNN technique’s effectual ability to recognise sarcastic and non-sarcastic samples. For instance, with 70% of TRP, the ARO-MCEDNN approach obtains average \(acc\_y\) of 97.53%, \(prec\_n\) of 96.35%, \(rec\_t\) of 97.53%, \(F\_score\) of 96.93%, and MCC of 93.87%. Meanwhile, with 30% of TSS, the ARO-MCEDNN approach gains average \(acc\_y\) of 96.86%, \(prec\_n\) of 97.76%, \(rec\_t\) of 96.86%, \(F\_score\) of 97.30%, and MCC of 94.61%. Eventually, with 80% of TRP, the ARO-MCEDNN method achieves average \(acc\_y\) of 97.75%, \(prec\_n\) of 98.21%, \(rec\_t\) of 97.75%, \(F\_score\) of 97.98%, and MCC of 95.96%. Finally, with 20% of TSP, the ARO-MCEDNN method attains average \(acc\_y\) of 98.96%, \(prec\_n\) of 98.96%, \(rec\_t\) of 98.96%, \(F\_score\) of 98.96%, and MCC of 97.92%.

The TACY and VACY of the ARO-MCEDNN system on the Twitter 2013 dataset are illustrated in Fig. 4. The figure demonstrated that the ARO-MCEDNN algorithm had depicted more significant results with enhanced TACY and VACY values. Note that the ARO-MCEDNN technique has gained maximal TACY outputs. The TLOS and VLOS of the ARO-MCEDNN system on the Twitter 2013 dataset are represented in Fig. 5. The figure illustrates that the ARO-MCEDNN algorithm has demonstrated greater results with lesser TLOS and VLOS values. Note that the ARO-MCEDNN system has gained minimum VLOS outputs.

Table 3 gives an overall sarcasm recognition outcome of the ARO-MCEDNN algorithm on the Headline 2019 dataset. The outcomes inferred the ARO-MCEDNN system’s effectual ability in recognising sarcastic and non-sarcastic samples. For instance, with 70% of TRP, the ARO-MCEDNN approach gains average \(acc\_y\) of 98.90%, \(prec\_n\) of 98.95%, \(rec\_t\) of 98.90%, \(F\_score\) of 98.92%, and MCC of 97.85%. In the meantime, with 30% of TSS, the ARO-MCEDNN approach gains average \(acc\_y\) of 98.89%, \(prec\_n\) of 98.91%, \(rec\_t\) of 98.89%, \(F\_score\) of 98.90%, and MCC of 97.81%. Afterwards, with 80% of TRP, the ARO-MCEDNN approach gains average \(acc\_y\) of 99.45%, \(prec\_n\) of 99.45%, \(rec\_t\) of 99.45%, \(F\_score\) of 99.45%, and MCC of 98.90%. Finally, with 20% of TSP, the ARO-MCEDNN algorithm gains average \(acc\_y\) of 99.36%, \(prec\_n\) of 99.35%, \(rec\_t\) of 99.36%, \(F\_score\) of 99.35%, and MCC of 98.70%.

The TACY and VACY of the ARO-MCEDNN methodology on the Headline 2019 dataset are illustrated in Fig. 7. The figure represented that the ARO-MCEDNN model has illustrated higher outputs with enhanced TACY value and VACY value. It can be noticeable that the ARO-MCEDNN model has obtained increased TACY outputs. The TLOS value and VLOS value of the ARO-MCEDNN methodology on the Headline 2019 dataset are represented in Fig. 8. The figure pointed out that the ARO-MCEDNN model has illustrated higher results with reduced TLOS and VLOS values. It can be visible that the ARO-MCEDNN model has obtained decreased VLOS outputs.
Finally, the sarcasm detection outputs of the ARO-MCEDNN approach are related to existing methods in Table 4. Fig. 9 represents relative research of the ARO-MCEDNN approach in terms of accuracy. The outputs indicate that the ARO-MCEDNN technique reaches effectual results over compared methods. Based on accuracy, the ARO-MCEDNN technique gains improved accuracy of 99.45% while the Vanilla-CNN, Fracking sarcasm, ELMo-BiLSTM, ELMo-BiLSTM FULL, and A2Text-Net models accomplish reducing accuracy of 72.63%, 90.15%, 77.06%, 78.49%, and 93.39% respectively.

Fig. 10 shows relative research of the ARO-MCEDNN method in terms of precision, recall, and F-score. The outcomes stated that the ARO-MCEDNN technique reaches effectual results over compared techniques. Based on precision, the ARO-MCEDNN method gains improved precision of 99.45%, while the Vanilla-CNN, Fracking sarcasm, ELMo-BiLSTM, ELMo-BiLSTM FULL, and A2Text-Net approaches accomplish reducing precision of 71.48%, 88.74%, 76.34%, 78.02%, and 92.41% correspondingly. As well, concerning recall, the ARO-MCEDNN approach obtains an improved recall of 99.45% while the Vanilla-CNN, Fracking sarcasm, ELMo-BiLSTM, ELMo-BiLSTM FULL, and A2Text-Net methods accomplish reducing recall of 67.76%, 88.63%, 75.77%, 74.22%, and 91.47% correspondingly. Finally, based on F-score, the ARO-MCEDNN system acquires improved F-score of 99.45%, while the Vanilla-CNN, Fracking sarcasm, ELMo-BiLSTM, ELMo-BiLSTM FULL, and A2Text-Net methods accomplish reducing F-score of 68.88%, 88.54%, 76.13%, 75.99%, and 90.53% correspondingly. These outputs reassured the supremacy of the ARO-MCEDNN method over other existing techniques.

Table 1. Dataset details

<table>
<thead>
<tr>
<th>Class</th>
<th>Data Set</th>
<th>Twitter-2013</th>
<th>Headlines-2019</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sarcastic</td>
<td></td>
<td>308</td>
<td>13634</td>
</tr>
<tr>
<td>Non Sarcastic</td>
<td></td>
<td>1648</td>
<td>14985</td>
</tr>
<tr>
<td>Total No. of Samples</td>
<td></td>
<td>1956</td>
<td>28619</td>
</tr>
</tbody>
</table>

Table 2. Sarcasm recognition outcome of ARO-MCEDNN approach on twitter 2013 data

<table>
<thead>
<tr>
<th>Class</th>
<th>Accur_{y}</th>
<th>Prec_{n}</th>
<th>Recal_{t}</th>
<th>F score</th>
<th>MCC</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Training Phase (70%)</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sarcastic</td>
<td>96.36</td>
<td>93.39</td>
<td>96.36</td>
<td>94.85</td>
<td>93.87</td>
</tr>
<tr>
<td>Non Sarcastic</td>
<td>98.69</td>
<td>99.30</td>
<td>98.69</td>
<td>99.00</td>
<td>93.87</td>
</tr>
<tr>
<td>Average</td>
<td>97.53</td>
<td>96.35</td>
<td>97.53</td>
<td>96.93</td>
<td>93.87</td>
</tr>
<tr>
<td><strong>Testing Phase (30%)</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sarcastic</td>
<td>94.32</td>
<td>96.51</td>
<td>94.32</td>
<td>95.40</td>
<td>94.61</td>
</tr>
<tr>
<td>Non Sarcastic</td>
<td>99.40</td>
<td>99.00</td>
<td>99.40</td>
<td>99.20</td>
<td>94.61</td>
</tr>
<tr>
<td>Average</td>
<td>96.86</td>
<td>97.76</td>
<td>96.86</td>
<td>97.30</td>
<td>94.61</td>
</tr>
<tr>
<td><strong>Training Phase (80%)</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sarcastic</td>
<td>96.03</td>
<td>97.19</td>
<td>96.03</td>
<td>96.61</td>
<td>95.96</td>
</tr>
<tr>
<td>Non Sarcastic</td>
<td>99.47</td>
<td>99.24</td>
<td>99.47</td>
<td>99.35</td>
<td>95.96</td>
</tr>
<tr>
<td>Average</td>
<td>97.75</td>
<td>98.21</td>
<td>97.75</td>
<td>97.98</td>
<td>95.96</td>
</tr>
<tr>
<td><strong>Testing Phase (20%)</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sarcastic</td>
<td>98.21</td>
<td>98.21</td>
<td>98.21</td>
<td>98.21</td>
<td>97.92</td>
</tr>
<tr>
<td>Non Sarcastic</td>
<td>99.70</td>
<td>99.70</td>
<td>99.70</td>
<td>99.70</td>
<td>97.92</td>
</tr>
<tr>
<td>Average</td>
<td>98.96</td>
<td>98.96</td>
<td>98.96</td>
<td>98.96</td>
<td>97.92</td>
</tr>
</tbody>
</table>
Fig. 3 Confusion matrix of twitter 2013 dataset (a-b) TRP/TSP of 70:30 and (c-d) TRP/TSP of 80:20

Fig. 4 TACY and VACY outcome of ARO-MCEDNN method on twitter 2013 dataset

Fig. 5 TLOS and VLOS outcome of ARO-MCEDNN method on twitter 2013 dataset
Fig. 6 Confusion matrix of headline 2019 dataset (a-b) TRP/TSP of 70:30 and (c-d) TRP/TSP of 80:20

Fig. 7 TACY and VACY outcome of ARO-MCEDNN method on Headline 2019 dataset
Table 3. Sarcasm recognition outcome of ARO-MCEDNN method on headline 2019 dataset

<table>
<thead>
<tr>
<th>Class</th>
<th>Accu</th>
<th>Prec</th>
<th>Reca</th>
<th>F_score</th>
<th>MCC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training Phase (70%)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sarcastic</td>
<td>98.43</td>
<td>99.31</td>
<td>98.43</td>
<td>98.87</td>
<td>97.85</td>
</tr>
<tr>
<td>Non Sarcastic</td>
<td>99.38</td>
<td>98.58</td>
<td>99.38</td>
<td>98.98</td>
<td>97.85</td>
</tr>
<tr>
<td>Average</td>
<td>98.90</td>
<td>98.95</td>
<td>98.90</td>
<td>98.92</td>
<td>97.85</td>
</tr>
<tr>
<td>Testing Phase (30%)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sarcastic</td>
<td>98.61</td>
<td>99.09</td>
<td>98.61</td>
<td>98.85</td>
<td>97.81</td>
</tr>
<tr>
<td>Non Sarcastic</td>
<td>99.17</td>
<td>98.73</td>
<td>99.17</td>
<td>98.95</td>
<td>97.81</td>
</tr>
<tr>
<td>Average</td>
<td>98.89</td>
<td>98.91</td>
<td>98.89</td>
<td>98.90</td>
<td>97.81</td>
</tr>
<tr>
<td>Training Phase (80%)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sarcastic</td>
<td>99.40</td>
<td>99.45</td>
<td>99.40</td>
<td>99.42</td>
<td>98.90</td>
</tr>
<tr>
<td>Non Sarcastic</td>
<td>99.50</td>
<td>99.45</td>
<td>99.50</td>
<td>99.47</td>
<td>98.90</td>
</tr>
<tr>
<td>Average</td>
<td>99.45</td>
<td>99.45</td>
<td>99.45</td>
<td>99.45</td>
<td>98.90</td>
</tr>
<tr>
<td>Testing Phase (20%)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sarcastic</td>
<td>99.45</td>
<td>99.19</td>
<td>99.45</td>
<td>99.32</td>
<td>98.70</td>
</tr>
<tr>
<td>Non Sarcastic</td>
<td>99.27</td>
<td>99.50</td>
<td>99.27</td>
<td>99.38</td>
<td>98.70</td>
</tr>
<tr>
<td>Average</td>
<td>99.36</td>
<td>99.35</td>
<td>99.36</td>
<td>99.35</td>
<td>98.70</td>
</tr>
</tbody>
</table>

Table 4. The relative outcome of the ARO-MCEDNN method with current systems

<table>
<thead>
<tr>
<th>Methods</th>
<th>Accu</th>
<th>Prec</th>
<th>Reca</th>
<th>F_score</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARO-MCEDNN</td>
<td>99.45</td>
<td>99.45</td>
<td>99.45</td>
<td>99.45</td>
</tr>
<tr>
<td>Vanilla-CNN</td>
<td>72.63</td>
<td>71.48</td>
<td>67.76</td>
<td>68.88</td>
</tr>
<tr>
<td>Fracking Sarcasm</td>
<td>90.15</td>
<td>88.74</td>
<td>88.63</td>
<td>88.54</td>
</tr>
<tr>
<td>ELMo-BiLSTM</td>
<td>77.06</td>
<td>76.34</td>
<td>75.77</td>
<td>76.13</td>
</tr>
<tr>
<td>ELMo-BiLSTM FULL</td>
<td>78.49</td>
<td>78.02</td>
<td>74.22</td>
<td>75.99</td>
</tr>
<tr>
<td>A2Text-Net</td>
<td>93.39</td>
<td>92.41</td>
<td>91.47</td>
<td>90.53</td>
</tr>
</tbody>
</table>

Fig. 8 TLOS and VLOS outcome of ARO-MCEDNN approach on Headline 2019 dataset
5. Conclusion

In this research, we have presented a new ARO-MCEDNN algorithm for SD in social-media platforms. The proposed ARO-MCEDNN algorithm emphasizes sarcasm detection and classification in social networking sites. It follows a four-stage process: data pre-processing, Glove based word embedding, MCEDNN-based sarcasm detection, and ARO-based hyperparameter tuning. Primarily, the ARO-MCEDNN technique follows a series of pre-processing data levels for converting the input dataset into a compatible format. Succeeded by the Glove approach is applied for word embedding purposes.

Moreover, the MCEDNN model is applied as a classification model to identify and categorize different types
of sarcasm. Furthermore, the ARO algorithm is chosen as a hyperparameter optimizer of the MCEDNN model, enhancing the sarcasm detection performance. A series of simulations were performed to accentuate the enhanced performance of the ARO-MCEDNN algorithm. The simulation outputs demonstrate the significant outcomes of the ARO-MCEDNN algorithm over current methods. In future, ensemble DL techniques can be derived to boost the detection performance of the ARO-MCEDNN algorithm.

References

7. Anuraj Mohan et al., “Sarcasm Detection using Bidirectional Encoder Representations from Transformers and Graph Convolutional Networks,” Procedia Computer Science, vol. 218, pp. 93-102, 2023. [CrossRef] [Google Scholar] [Publisher Link]
9. Uthkarsha Sagar, “A Broad Survey of Natural Language Processing,” SSRG International Journal of Computer Science and Engineering, vol. 6, no. 12, pp. 15-18, 2019. [CrossRef] [Publisher Link]
16. Dilip Kumar Sharma et al., “Sarcasm Detection Over Social Media Platforms using Hybrid Auto-Encoder-Based Model,” Electronics, vol. 11, no. 18, pp. 2844, 2022. [CrossRef] [Google Scholar] [Publisher Link]
17. Lu Ren et al., “Sarcasm Detection with Sentiment Semantics Enhanced Multi-Level Memory Network,” Neurocomputing, vol. 401, pp. 320-326, 2020. [CrossRef] [Google Scholar] [Publisher Link]
18. Yazhou Zhang et al., “Stance Level Sarcasm Detection with BERT and Stance-Centered Graph Attention Networks,” ACM Transactions on Internet Technology (TOIT), 2022. [CrossRef] [Google Scholar] [Publisher Link]
19. Xuan Zhao, Jimmy Huang, and Haitian Yang, “CANs: Coupled-Attention Networks for Sarcasm Detection on Social Media,” In 2021 International Joint Conference on Neural Networks (IJCNN), pp. 1-8, 2021. [CrossRef] [Google Scholar] [Publisher Link]

Dilip Kumar Sharma et al., “Sarcasm Detection over Social Media Platforms Using Hybrid Ensemble Model with Fuzzy Logic,” *Electronics*, vol. 12, no. 4, pp. 937, 2023. [CrossRef] [Google Scholar] [Publisher Link]

Shamil Chollampatt, and Hwee Tou Ng, “A Multilayer Convolutional Encoder-Decoder Neural Network for Grammatical Error Correction,” *In Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018. [CrossRef] [Google Scholar] [Publisher Link]


Ayush Jain et al., “Detection of Sarcasm Through Tone Analysis on Video and Audio Files: A Comparative Study on AI Models Performance,” *SSRG International Journal of Computer Science and Engineering*, vol. 8, no. 12, pp. 1-5, 2021. [CrossRef] [Google Scholar] [Publisher Link]