International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

An Ensemble Framework with Optimal Features for Sarcasm Detection in Social Media Data

Haripriya V.[1*], Dr. Poornima G. Patil [2]

Submitted: 23/06/2023 **Revised**: 06/08/2023 **Accepted**: 23/08/2023

Abstract*:* The main objective of the proposed work is to identify sarcasm in social media data. Sarcasm is being used to degrade or convey contempt in writing and speaking. Sarcasm has been studied extensively in NLP. Feature selection and the nature of classifiers plays a major role in detecting sarcasm. Effective feature selection increases the accuracy of the sarcasm detection. The work has developed an Ensemble framework with optimal features for sarcasm detection in social media data .The framework consists of four important components i.e. Pre-processing, Feature extraction, Optimal Feature selection and Sarcasm detection. From the extracted features, the most optimal features has been selected and thereafter the training as well as testing is performed using a new optimized CNN model by adding projected weight function of CNN. The developed CNN is fine-tuned by standard AptenodytesForsteri Optimization Algorithm (AFO).The sarcasm detection is carried out using a new ensemble technique that has been constructed with optimized Recurrent Neural Network (RNN) deep learning classifier. To enhance the sarcasm detection accuracy the activation function of RNN is optimized via Firebug Swarm Optimization (FSO). An ensemble framework has been developed and the outcome shows the comparison result of proposed FSOCNN with existing algorithms such as Recurrent neural network, Convolutional neural network and Artificial neural network is 97.72% and the proposed method's performance has been compared to other existing models and represented the outcomes in accuracy, specificity, and F measure.

Keywords: *Activation function, Feature Selection, Neural Networks, Optimization, Sarcasm Detection..*

1 Introduction

The world's usage of social networks is increasing dramatically as the web evolves from a static social Network. Communication on social networks allows the user to create and receive information, encouraging people with diverse interests to share and communicate [1]. This large dataset has encouraged data analysts to use it in many applications. However, microblogging (140 characters per tweet) and informal language make it difficult to grasp user sentiment and do sentiment analysis [2]. Sarcasm also affects sentiment analysis and misclassifies people's ideas. Thus, it reduces sentiment analysis accuracy. Sarcasm is being used to degrade or convey contempt in writing and speaking. Positive phrases are used to convey negative emotions. Sarcasm or irony is becoming quite frequent on social media, yet difficult to detect [3].

Opinion mining uses sarcasm detection for applications in health, security, and sales [4]. Sarcasm detection is the task of finding irony in sentimental writing. However, sarcasm's figurative and creative nature presents a significant obstacle to Affective Computing Systems. So sarcasm detection is critical for sentiment analysis tasks. Sarcasm detection is a binary classification problem that predicts whether a text is sarcastic or not. Before, researchers employed rule-based and statistical techniques to predict sarcastic statements, such as (a) lexical and pragmatic features and (b) the presence of punctuations, interjections, sentiment shifts, etc., [5]. Recent research has switched to using deep learning to detect discriminating features. Using a deep neural network (DNN) instead of handcrafted features allows for faster learning. Deep learning models have surpassed human performance in tasks including text summarization [6], machine translation [7], and question answering [8]. Deep learning methods have also been investigated for sarcasm detection with impressive results [9]. The Attention Network mechanism improves deep learning model performance in machine translation, phrase summarization, and reading comprehension. The attention mechanism has also been analysed for text classification [10].

1Research Scholar, Visvesvaraya Technological University, Belagavi, INDIA
ORCID ID : 000-0003-2035-2452
[2] Department of MCA, Visvesvaraya Technological University, Belagavi, INDIA
ORCID ID : 0000-0003-4873-0435
[1]haripriyav07@gmail.com*
[2]poornima_g_patil@yahoo.com
** Corresponding Author Email: haripriyav07@gmail.com*

2 Related Work

Though Sarcasm Detection is a difficult and challenging task it has a huge scope in opinion mining and decision making. Researchers have attempted to solve the problems by introducing many new methods. Various works done on Sarcasm detection has been mentioned.

In 2022, Wen et al.[11] have developed a sememe and auxiliary enhanced attention neural model, SAAG. They introduced sememe knowledge to improve word representation learning in Chinese. A sememe is a fine-grained depiction of a word. Researchers used news headline in order to understand the context and background of sarcastic expressions at the sentence level. Then they generated the text expression representation gradually and dynamically. The proposed technique outperformed existing algorithms on a sarcastic dataset consisting of news text comments. In 2022, Govindan et al. [12] have considered tweets with negative sentiments to identify the occurrence of hyperboles for sarcasm detection. The most important hyperbole was Intensifier ($p<.0001$). The outcome illustrates that the hyperboles occur in an impartial tweets, which enhances the process of sarcasm detection.

In 2021, Meriem et al. [13] have projected a fuzzy sarcasm detection technique employing social information like responses, past tweets, likes, etc. The result demonstrated that using fuzzy logic improved the approach's precision metric and accuracy. Using degrees of significance the best recall, precision, and accuracy values were provided. In 2021, Hiremath et al. [14] developed a multiclass neural network framework for detecting sarcasm in cloud resources. Cognitive features such as voice signals and eye movements have been considered for the work that assist to detect sarcasm. They are highly fascinating and can build a solid basis in NLP for subsequent study.

A multi-level memory network has been projected by Ren et al. [15] that captures sarcastic features. This discrepancy between sentiment semantics and circumstance is captured by the second-level memory network in the proposed model. Authors have implemented a convolutional neural network to increase the memory network when no local information exists.

Deep learning approach has been used by Jain et al. [16] to detect sarcasm in code-switch tweets, combining English and Hindi. An attention layer with softmax and a CNN for sarcasm detection were combined in the proposed model. The SentiHindi feature vector was generated by handcrafting features using the subjective lexicon Hindi-SentiWordNet along with auxiliary pragmatic feature vector to obtain the count of pragmatic indicators in the tweet. The proposed softAttBiLSTM-feature-richCNN models were tested and validated. The model outperformed baseline deep learning methods with 92.71 % classification accuracy and 89.05 % F-measure.

Ren et al. [17] have detected sarcasm on tweets using various context-augmented neural models. The dataset results revealed that neural models outperformed discrete models in terms of performance. Sarcastic clues can be efficiently interpreted by Context-augmented neural networks from contextual information.

In 2021, Pandey et al. [18] introduced a hybrid attention-based model named Long Short Term Memory (HA-LSTM) network to recognize sarcastic statements. The proposed work involved sixteen distinct language features in its hidden layers, unlike the current LSTM model. Three benchmark datasets verify the proposed network. The combination of the linguistics features improves the model's performance compared to other models by up to 2% on three separate gold standard datasets. In 2021, authors in [19] presented a multimodal, multi-interactive, and multi-hierarchical neural network (M_3N_2). Because the brain perceived sarcasm in numerous ways, they used various data sources such as tweets, text present in an image and image description as M_3N_2 inputs. To address multimodalities, authors introduced a gate mechanism and guide attention (GA) to replicate the activities and cooperation of implicated brain regions while seeing different modes. For every modality interaction, a multihop method was employed to obtain modal information several times using GA.

In 2020, Pawar et al. [20] presented a system that detects sarcasm in both English and Hindi tweets. Because sarcasm was highly dependent and contextual, sentiment and other contextual signals may assist to detect it. The strategy worked well and the Random Forest classifier outperformed other classifiers. However, the derived patterns did not cover all sarcastic detection patterns.

3. Problem Statement

The proposed work aims to address the underutilization of the vast amount of feedback generated in social media platforms to identify genuine customer sentiments. The proposed research work aims to eliminate the existing problems such as (1) Deficient feature selection techniques which can hinder the accurate identification of sarcastic expressions, impacting the overall accuracy of sentiment analysis. (2). Inadequate classifier selection which can lead to suboptimal results in identifying sarcasm and affect the accuracy of sentiment analysis. The proposed system aims to detect sarcasm and perform sentiment analysis more accurately using effective feature selection techniques and an ensemble framework.

4. Research Methodology

In this proposed work, a novel model for sarcasm detection has been developed by incorporating four major phases such as pre-processing followed by feature extraction, optimal feature selection and finally the sarcasm detection. The framework for the proposed approach is given in Fig.1 and the following are detailed procedures.

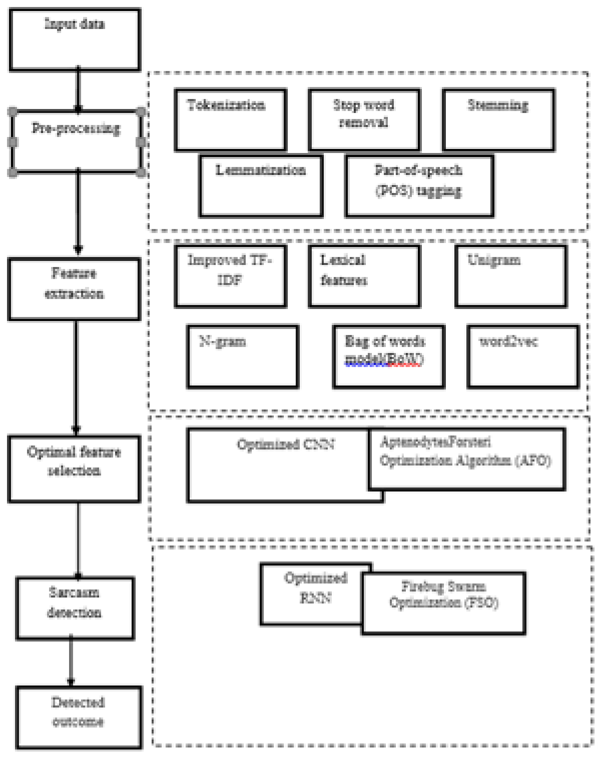


Fig 1: Architecture of the Proposed Model

Steps:

1) Read dataset collected from News Headlines

2) Preprocessing: URL removal, punctuation removal, stop works removal, stemming, Part-of-speech (POS) tagging, lemmatization and Tokenization.

3) Extract features: Features have been extracted from the dataset using various feature extraction techniques namely Improved Term frequency -Inverse document frequency (ITF-IDF), Lexical features, Unigram, N-gram, bag of words model (BoW), word2vec based features.

4) Optimal feature selection: Apply optimized CNN (weight function of CNN will be fine-tuned by standard AptenodytesForster Optimization Algorithm (AFO))

5) Classification: Apply Optimized Recurrent Neural Network (RNN) (the activation function of RNN will be fine-tuned by Firebug Swarm Optimization (FSO))

4.1 Pre-processing

Preprocessing has been performed on the collected raw data from News Headlines through URL removal, punctuation removal, stop works removal, stemming, Part-of-speech (POS) tagging, Lemmatization and Tokenization. Then, features are taken from pre-processed data.

(i) Tokenization: Tokenization is the process of breaking down a sequence of text into individual units, which are usually referred to as tokens. Tokens are the basic building blocks used in various NLP tasks, such as language modelling, machine translation, text classification, and more. Tokenization eliminates blank whitespace characters from text documents. In the context of natural language processing (NLP), tokenization typically involves dividing a sentence or a paragraph into words, phrases, symbols, or other meaningful linguistic units.

(ii) Stop word removal: These are common words like articles and prepositions which do not affect the context of the sentence and no role to analyze the text. The Natural language toolkit (NLTK) lexicon is often used to remove stop-words from datasets. Stop words are deleted to enhance the classification output.

(iii) Stemming: In order to create a root word, prefixes and suffixes are removed from the phrase. This method reduces the number of keyword spaces. Derivation of keywords from dissimilar keywords improves classification performance.

(iv) Lemmatization is a linguistic process used in natural language processing (NLP) and computational linguistics in order to reduce words to their base or root form, known as the lemma. Lemmatization allows to normalize words so that different forms of a word, such as plurals, verb conjugations, and various inflections, are all converted into a common base form. This makes it easier to analyze and compare words in text analysis and NLP tasks. The main aim of lemmatization is to group together words that share the same core meaning or concept, even if they appear in different forms. For example, the lemma of the words "running," "ran," and "runs" would all be "run."

(v) Part-of-speech (POS) tagging: Part-of-speech (POS) refers to the grammatical category or syntactic role that a word plays within a sentence. In natural language, words are classified into different parts of speech based on their grammatical and semantic functions. To analyze the sentence structure, and to determine the relationship between words and to extract the meaning from the text, part-of-speech of each word is essential. The part of speech of a word can be determined by its meaning and its context.

4.2 Feature extraction

The quality of classification relies on the selected features. The news social media data aspects may be classified into lexical, hyperbolic, pragmatic, sentiment, and contradiction. The proposed system extracts features like Improved Term Frequency–Inverse Document Frequency (TF-IDF), Lexical features, Unigram, N-gram, Bag of Words model(BoW), word2vec based features.

A numeric statistics which represents word's (period's) relevance for a text in a database for the feature extraction is Improved TF-IDF. Word frequency in one text must be compared to word frequency in other texts. TF-IDF is often used to avoid word filtering in text summarizing and classification purposes. This increases the proportionate frequency of a given text. Therefore, an improved TF-IDF based feature will be introduced in this research work. Text mining typically uses lexical features. Various preprocessing approaches are lexical features that shows the range of polarity. These features may be used in emotion mining to determine text emotion levels.

4.3 Optimal feature selection

A new optimized convolutional network model has been used in the proposed work for the selection of the most optimal features extraction. The projected weight function of CNN is fine-tuned by standard AptenodytesForsteri Optimization Algorithm (AFO) [24]. This algorithm executes on the basis of the emperor penguin's warm-hugging behavior inspired AFO. Penguins need to observe temperature changes, search for other penguins, move closer to the center of the colony, save energy and use memory while looking for a suitable location. These five techniques become five variable updating modes. Adaptive adjustment techniques are meant to combine the five strategies in the exploration and exploitation phases.

The Emperor penguin's scientific name is Aptenodytesforsteri. They are the biggest penguins of all the species. Males and females are roughly comparable in size. They inhabit Antarctica and have wings, a blackhead, a tail, and a white belly. They snuggle together for warmth and wind protection in order to survive the ferociously cold Antarctic winters. The four fundamental steps of the penguins' huddling mechanism include setting up a huddling border, evaluating the temperature profile, determining how far apart emperor penguins are from one another, and adjusting the mover.

Emperor penguins go through the four phases listed above, which are quantitatively modeled. Finding the most efficient mover and updating the emperor penguin's location are the main objectives of this method.

Penguins form a polygonal huddle barrier. In the process of creating huddle boundaries, the wind flow is also taken into account. To demonstrate the behavior of huddle boundary construction, the complex variable is employed. Consider that this wind's gradient and speed are represented by the letters ϕ and γ .

$$\gamma = \nabla\phi \tag{1}$$

The arithmetical formulation of probable rationale is given below based on (1).

$$q = \phi + j\mu \tag{2}$$

In (2), j described the fantasy invariable and ‘ μ ‘ is an arbitrary vector. Penguins' positions are arbitrarily adjusted based on where the position of the best-fitting penguin originated. At the center of the L-shaped polygon is the penguin that fits the situation the best.

To survive in the winter, the penguins gather together and keep the fitness high. It is expected that the fitness μ is set to zero when the polygon's radius S is greater than one. Fitness t_e is set to one if not. The equation below calculates the fitness difference between T_e the actual fitness inside the huddle and the global fitness outside the huddle barrier.

$$T_e = \left(t_e - \frac{Maximum_{iter}}{y - \max imum_{iter}} \right) \tag{3}$$

$$t_e = \begin{cases} 0, & S > 1 \\ 1, & S < 1 \end{cases} \tag{4}$$

Equation (3) is to represents the t_e warmth outline and y is present iteration. ‘$Maximum_{iter}$’ is the highest boundary of iteration and mentioned in (3).

The best-fitting emperor penguin ‘ h ’ is most important in figuring out where other emperor penguins are. All other emperor penguins' locations are changed correspondingly. The emperor penguins' distance from one another is determined when the huddle barrier is created. Equation (5) gives the distance calculation's mathematical formulation.

$$(\vec{E}) = \left(Abs\left(C(\vec{Z}).\vec{R}(t) - \vec{A}.\vec{R}_{fq}(t)\right)\right) \tag{5}$$

$(\vec{Z})$and$(\vec{A})$illustrates search coefficient, $\vec{R}(t)$ and $\vec{R}_{fq}(t)$ denotes the current and predicted local and global values. $(\vec{Z})$and$(\vec{A})$ are estimated as in (6-8),

$$(\vec{Z}) = \left(O \times \left(U + R_{grid}(accuracy)\right) \times Rand()\right) - U \tag{6}$$

$$R_{grid}(accuracy) = Abs\left(\vec{R} - \vec{R}_{ep}\right) \tag{7}$$

$$(\vec{A}) = Rand() \tag{8}$$

The arithmetic formulation of $\overrightarrow{C(Z)}$ is described in (9),

$$\overrightarrow{C(Z)} = \sqrt{\left(h.e^{-\frac{t}{m}} - e^{-t} \right)^2} \tag{9}$$

The altered position of the penguin after iteration is shown in (10).

$$\vec{R}_{ep}(t+1) = \vec{R}(t) - \vec{Z}.\vec{E} \quad (10)$$

During the huddling habit of penguins, the location of the best-fit penguin is recalculated.

The following pseudo code contains various steps required to implement the Emperor Penguin optimizer algorithm.

Pseudo Code:

Input: Initialize randomly emperor penguins population $\vec{R}_{ep}(t = 1,2,...n)$

Output: greatest finest explanation

```
Initialize the variables $U, \vec{Z}, \vec{A}, C(), R, and\ \ Max_{iteration}$
Calculate robustness assessment of every explore mediator
        **While** $(t < \max_{iteration})$ do    # iteration to search the global best solution
        **Fitness** $(R_{ep})$
        $S \leftarrow Rand()$
        $if\ (S > 1)$ **then**
                $u \leftarrow 0$
        **else**
                $u \leftarrow 1$
                **end if**
                $U \leftarrow \left(u - \frac{\max_{iteration}}{t - \max_{iteration}}\right)$
        **For** $j \leftarrow 1$ **to n do**
                **For** $k \leftarrow 1$ **to n do**
                Calculate the vectors $(\vec{Z})$ and $(\vec{A})$
                Calculate the function $C(\vec{Z})$
                Renew locations of the present agent
                **end for**
        **end for**
update the variables $U, \vec{Z}, \vec{A}, C()$
Restructure the investigate mediators that go away from the periphery
        **Fitness** $(R_{ep})$
Modernize $\vec{R}$ if it is enhanced than the prior assessment
        $t \leftarrow t+1$
        **end while**
return $\vec{R}$
**end procedure**
**Procedure** FITNESS ( $\vec{R}_{ep}$ )
        **For** $j \leftarrow 1$ **to n do**
                $FIT[j] \leftarrow FITNESS\_FUNCTION(\vec{R}_{ep})$
        **End for**
                $FIT_{best} \leftarrow BEST(FIT[])$
Return $FIT_{best}$
**End procedure**
**Procedure** $BEST(FIT[])$
        $best \leftarrow FIT[0]$
                $for\ j \leftarrow 1\ \ to\ \ n\ \ \ do$
                        $if\ FIT[j] < best\ \ then$
                        $best \leftarrow FIT[j]$
                        **End if**
                **End for**
Return best
**End procedure**
```

4.4 Sarcasm detection

The sarcasm detection is carried out using a new ensemble technique that will be constructed with optimized Recurrent Neural Network (RNN) deep learning classifier. The activation function of RNN is optimized via Firebug Swarm Optimization (FSO) [25] has been optimized in order to increase the accuracy. Firebug Swarm Optimization (FSO) is based on the behaviour of Firebug reproductive swarming behaviour. Individual bugs in a swarm of Firebugs are spontaneously looking for optimal solutions in a search space. The classifier has been trained via the optimal features acquired with the optimized CNN model. The final outcome has been acquired from optimized RNN.

An improved Recurrent Neural Network (RNN) deep learning classifier is used to build a novel ensemble strategy for sarcasm detection. RNNs can be useful for sarcasm detection because of their ability to capture sequential information and context in text. Sarcasm often relies on subtle cues and context, making it challenging to detect using traditional NLP methods. RNNs address this challenge by processing text sequentially, considering the order of words and their relationships in a sentence.

Here's how RNNs help to improve sarcasm detection:

1. Sequential Processing: RNNs process text data word by word, maintaining a hidden state that carries information from previous words. This allows them to consider the context and dependencies between words, which is crucial for understanding sarcasm.

2. Contextual Understanding: Sarcasm frequently entails a discrepancy between the intended meaning and the literal interpretation of words. RNNs can model this context and understand the underlying sentiment or tone of the text, helping them recognize the presence of sarcasm.

3. Long-term Dependencies: RNNs have the ability to retain information from earlier parts of a sentence, even when it becomes relevant later. This is essential for understanding sarcasm, as the sarcastic element may not be apparent until the end or middle of a sentence.

4. Handling Variability: Sarcasm can take various forms and can be expressed in different ways. RNNs can adapt to different sentence structures and word arrangements, making them more flexible in detecting sarcasm across various contexts.

RNNs are effective in certain cases, sarcasm detection remains a challenging task, and there might be instances where it is still difficult for any model to accurately identify sarcasm. In order to achieve that the work has developed Optimized RNN using Firebug Swarm Optimization (FSO).The activation function of RNN is optimized using Firebug Swarm Optimization (FSO) to increase the sarcasm detection accuracy. Based on the reproductive swarming behavior of firebugs, Firebug Swarm Optimization (FSO) was developed as shown in figure 2. In a search space, individual Firebugs in a swarm are haphazardly looking for the best answers. The best characteristics obtained with an improved CNN model will be used to train the classifier. The optimized RNN will provide the final result.

The impact of an Optimized RNN using Firefly Swarm Optimization (FSO) for sarcasm detection can be significant and beneficial in several ways:

1. Improved Sarcasm Detection Accuracy: The primary objective of using FSO to optimize the RNN is to enhance the model's sarcasm detection accuracy. By fine-tuning the activation function and possibly other parameters, the optimized RNN can better capture the subtle contextual cues and dependencies that indicate sarcasm. This can result in a more accurate and reliable sarcasm detection system.

2. Enhanced Natural Language Understanding: Sarcasm is a complex linguistic phenomenon that involves context, tone, and nuances. By optimizing the RNN using FSO, the model can develop a better understanding of the sequential nature of language and the subtle linguistic patterns associated with sarcasm. This can improve the overall natural language understanding capabilities of the RNN beyond just sarcasm detection.

3. Better Generalization: An optimized RNN can lead to improved generalization on unseen data. By leveraging FSO to fine-tune the model's parameters, it may become more robust and capable of handling diverse sarcasm instances from different contexts and writing styles.

4. Reduced False Positives/Negatives: Sarcasm detection systems often face challenges in distinguishing sarcasm from genuine expressions, leading to false positives or false negatives. An optimized RNN can potentially reduce such errors, leading to a more reliable and trustworthy sarcasm detection system.

5. Scalability and Efficiency: Depending on the dataset size and complexity, traditional RNN training can be computationally intensive. However, if FSO helps in identifying the optimal parameters more efficiently, it can reduce training time and resource requirements, making the optimized RNN more scalable for real-world applications.

6. Novel Optimization Techniques: The use of Firefly Swarm Optimization (FSO) showcases the exploration of new optimization techniques for improving neural network models. This research could contribute to the development and understanding of optimization methods for various other NLP tasks and neural network architectures.

It's important to note that assessing the effectiveness of an optimized RNN using FSO for sarcasm detection would necessitate thorough testing across diverse datasets and benchmarking against contemporary models. The success of this approach will hinge on factors like data accuracy, representation quality, fine-tuning of hyper parameters, and other variables that can impact the overall performance of the sarcasm detection system.

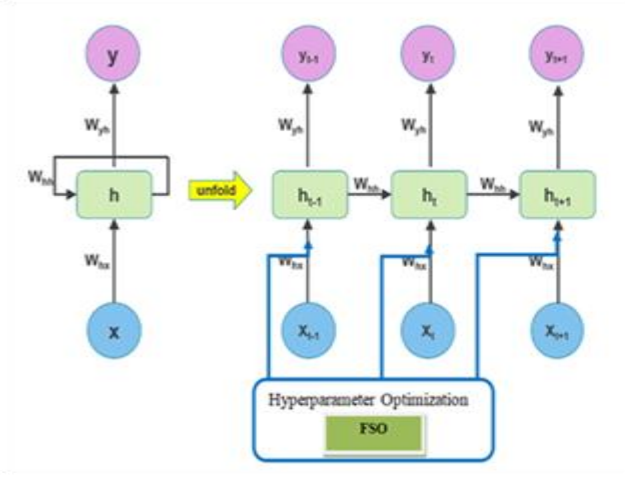


Fig 2: Optimized RNN

Overall, an optimized RNN represented in Fig. 2, using FSO has the potential to advance the state of the art in sarcasm detection and contribute to more accurate and sophisticated natural language understanding models.

Let's dive into the equations that define an RNN:

1. Notation:

- $.x_u$ represents the input at time step u

- h_u represents the hidden state at time step u

- Y_u is the output at time step u

2. Hidden State Update:

- The hidden state h_u at time step u has been computed based on the given input $.x_u$ and the previous hidden state h_{u-1} and same has been given in (11):

$$H = act\left[h_{u-1}W_{hh} + W_{xh}.x_u + B_h\right] \quad (11)$$

In this equation, W_{hh} is the weight matrix for the recurrent connections, W_{xh} is the weight matrix for the input connections, and B_y is the bias term for the hidden state.

3. Output Computation:

- The output Y_u at time step u is calculated using the current hidden state h_u and mentioned in (12):

$$Y_u = act\left[h_u W_{hy} + B_y\right] \quad (12)$$

Here, W_{hy} is the weight matrix connecting the hidden state to the output, and B_y is the bias term for the output.

4. Training:

- To train the RNN, you need a labelled dataset of inputs and corresponding targets.

- Define a loss function like cross-entropy loss to quantify the gap between the predicted output that is expected and the label that is really present.

Y_u and the true label.

- The model's parameters (weights and biases) are updated using back propagation through time (BPTT), which calculates gradients for each time step and propagates them back to update the parameters.

5. Backpropagatione through Time (BPTT):

- BPTT extends back propagation to handle sequential data.

- The gradients at each time step are calculated by accumulating the gradients from the current time step and the subsequent time steps.

- The gradients are then backpropagated through time, applying the chain rule to update the model's parameters.

The basic RNN suffers from the "vanishing gradient" problem, where gradients diminish as they propagate through time, making it challenging to capture long-term dependencies. By using FSO to update weights will improve the training of RNN

The updated weight W_{u+1} for performing fractional-order calculus on FSO given in (13):

$$W_{u+1} + W_u = D_1 \times rand(q - Y_U) + D_2 \times rand(H - Y_U) \quad (13)$$

Where D_1 and D_2 illustrates the linear coefficients, q and H illustrates hadamard multiplication, W_u current weight.

The L.H.S of the aforementioned formula provides the following description of the discrete formulation of the derivative of order number is given as follows:

$$E^{\beta}[W_{u+1}] = D_1 \times rand(q - Y_U) + D_2 \times rand(H - Y_U) \quad (14)$$

Equation. (14) states that the particle speed has a partial order number with a range of $\beta = 0$ to $\beta = 1$ ($\Delta\beta = 0.1$) and r = 4:

$$W_{j+1} = \beta W_u + \frac{1}{2}\beta(1-\beta)W_{u-1} + \frac{1}{6}\beta(1-\beta)(2-\beta)W_{u-2} + \frac{1}{24}\beta(1-\beta)(2-\beta)(3-\beta)W_{u-3}D_1 \times rand(q - Y_U) + D_2 \times rand(H - Y_U) \quad (15)$$

The additional degrees of freedom are the minor calculus derivative's main benefit. The probability regulating the speed of elements with regard to the copied order is shown in (15). The primary benefit of the basic FSO approach was this. Additional fractional calculus derivative probability levels took into account the detailed description of the progression of multiple cycles via the tailored enhancement of architectural control, display, and planning.

5. Results

The final outcome of the proposed work is an ensemble model which helps to classify the text into sarcastic or non-sarcastic. The results are represented in the following section.

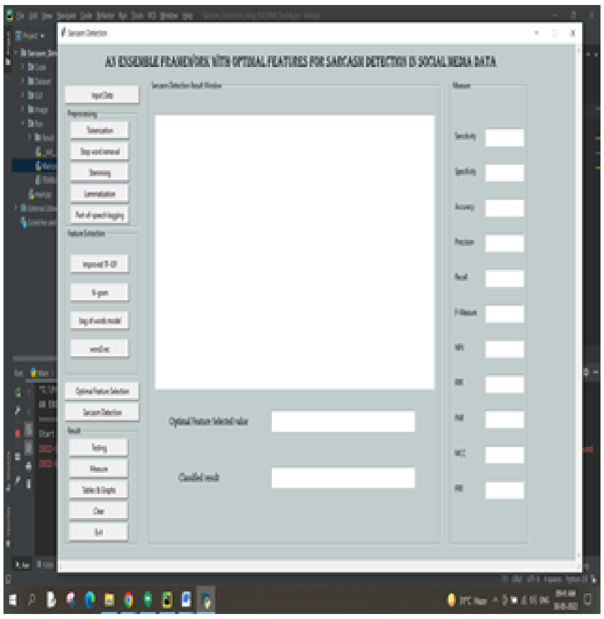


Fig 3. Ensemble Framework

Fig. 3 represents the Ensemble framework with all the possible menus including Input data, various steps involved in the Preprocessing steps, Feature Engineering, Optimization and the Result. On the right side of the figure Measure columns have been added. Optimal Feature selection and the execution has been represented in Fig. 4. The result shows the optimal feature selected value. The sarcasm detection process is mentioned in Fig. 5(a) and Fig. 5(b) represents the optimal selected value and the classified output. To test the sarcasm detection classification, testing has been performed and the same is represented in Fig. 6.

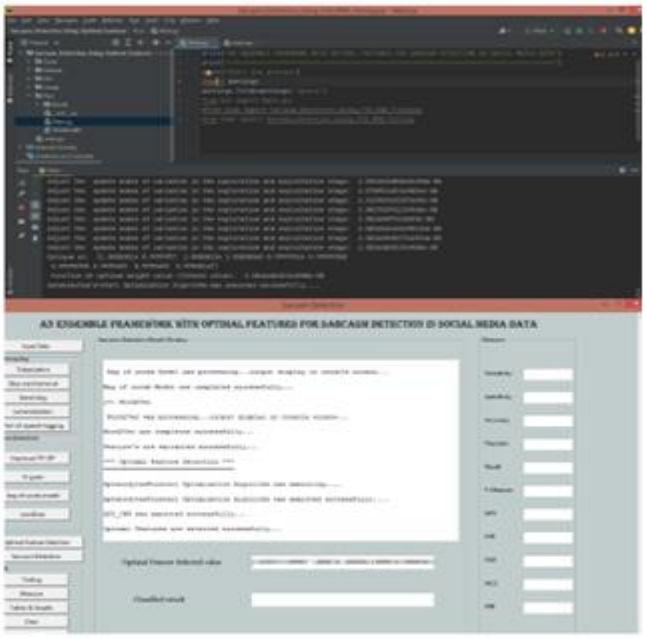

AN ENSEMBLE FRAMEWORK WITH OPTIMAL FEATURES FOR SARCASM DETECTION IN SOCIAL MEDIA DATA
Optimal Feature Selected value
Classified result

Fig 4. Optimal Feature Selection

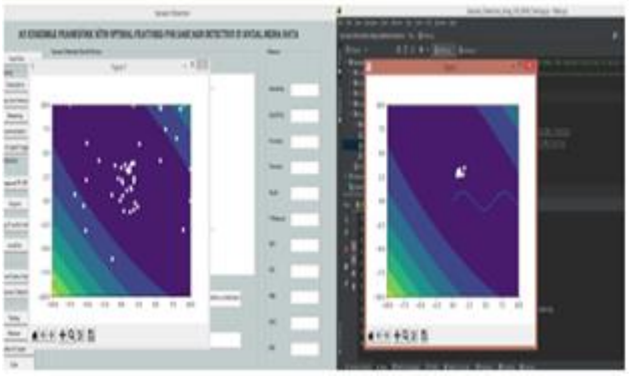
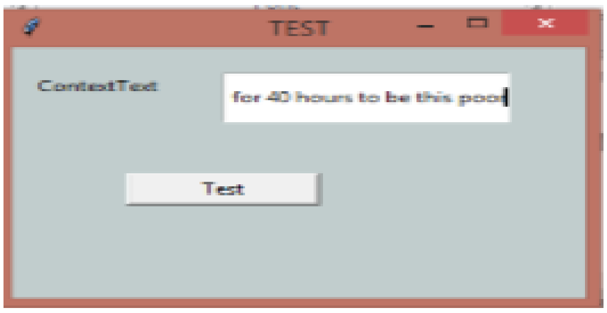

AN ENSEMBLE FRAMEWORK WITH OPTIMAL FEATURES FOR SARCASM DETECTION IN SOCIAL MEDIA DATA

Fig 5.(a) Sarcasm Detection

AN ENSEMBLE FRAMEWORK WITH OPTIMAL FEATURES FOR SARCASM DETECTION IN SOCIAL MEDIA DATA
Optimal Feature Selected value
Classified result

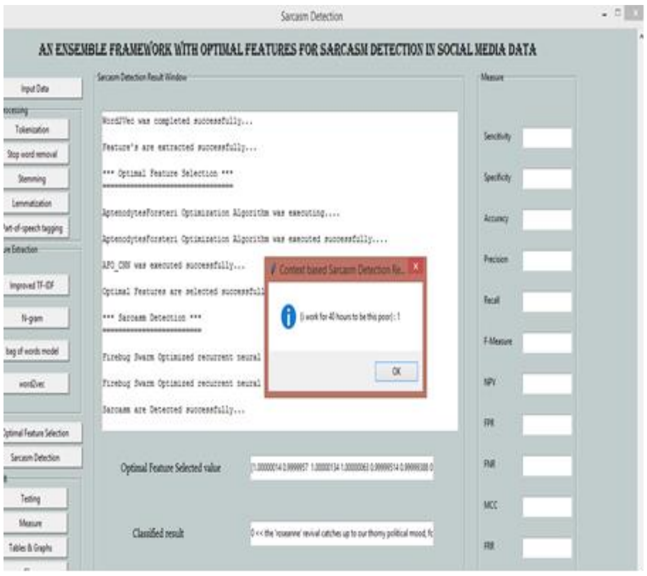
Fig5. (b) Sarcasm Detection


TEST

ContextText for 40 hours to be this poor

Test

Fig 6: Testing



```
Word2Vec was completed successfully...

Feature's are extracted successfully...

*** Optimal Feature Selection ***
*********************************

AptenodytesForsteri Optimization Algorithm was executing....

AptenodytesForsteri Optimization Algorithm was executed successfully....

AFO_CNN was executed successfully...

Optimal Features are selected successfull

*** Sarcasm Detection ***
*************************

Firebug Swarm Optimized recurrent neural

Firebug Swarm Optimized recurrent neural

Sarcasm are Detected successfully...
```

(i work for 40 hours to be this poor) : 1

Optimal Feature Selected value: 1.00000014 0.9999957 1.0000134 1.0000063 0.99999514 0.99999388 0

Classified result: 0 << the 'roseanne' revival catches up to our thorny political mood, fc

Fig 7: Classified Output

The classified output in the form of binary value is represented in Figure 7. The value 1 represents sarcastic comment and the value 0 represents non-sarcastic comment. Figure 8(a) and Figure 8(b) represents the tables consisting of the comparison results of algorithms such as Existing CNN, RNN and ANN. The comparison has been done with the proposed algorithm named proposed_FSOCNN. The table consists of Accuracy, Specificity, Sensitivity, Computation Time and Fitness vs Iteration. The following Table 1 represents the comparison results of ANN with proposed FSO_CNN.

Table 1: Comparison result of ANN with proposed FSO_CNN-Measures

Measures	**ANN**	**Proposed FSO_CNN**
Accuracy	91.8	97.49
Precision	90.61	97.96
Recall	93.55	98.63
FScore	93.06	98.29
Sensitivity	90	94.34
Specificity	93.55	98.63
MCC	83.64	93.54
NPV	93.10	96.15
FPR	0.5	0.057
FRR	0.0645	0.014
FNR	0.0645	0.014

Table 2: Comparison result ofProposed_FSOCNN,Existing_RNN ,Existing_CNN and Existing_ANN

Method	**Proposed_FSOCNN**	**Existing_RNN**	**Existing_CNN**	**Existing_ANN**
Accuracy	97.727	94.623	93.421	91.803
Precision	98.214	96.105	94.534	93.666
Recall	98.8023	94.594	91.304	90.692
F-Measure	98.507	93.333	89.361	87.815
Sensitivity	98.802	94.594	91.304	90.692
Specificity	98.802	94.594	91.304	90.101
MCC	93.753	88.596	84.643	82.833
NPV	96.153	95.683	94.175	93.548
FPR	0.0446	0.0622	0.0736	0.0829
FNR	0.0119	0.054	0.0869	0.093
FRR	0.0119	0.054	0.0869	0.093
Computation Time	54345	61481	74234	84746

The graphical representation of the above table is represented in the below figures:

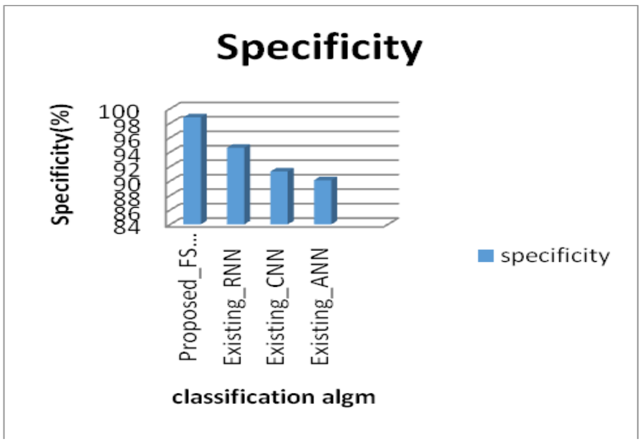


Fig 8(a): Specificity result

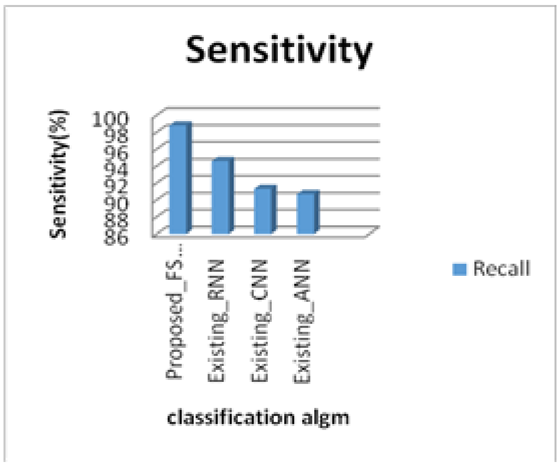


Fig8(b): Sensitivity Comparison

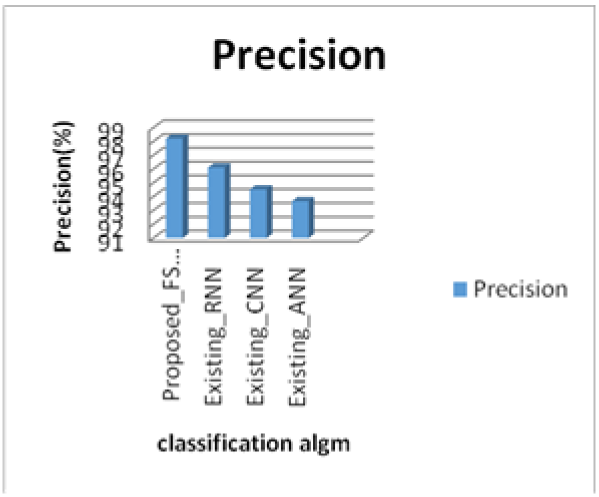


Fig 8(c): Recall rate

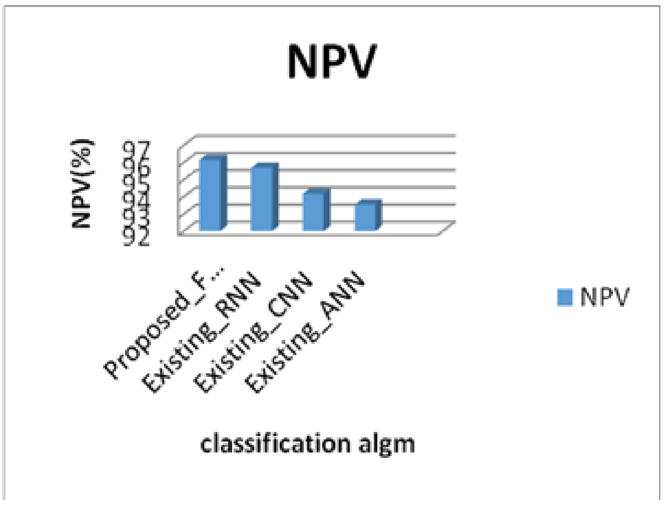


Fig8(d): Precision rate

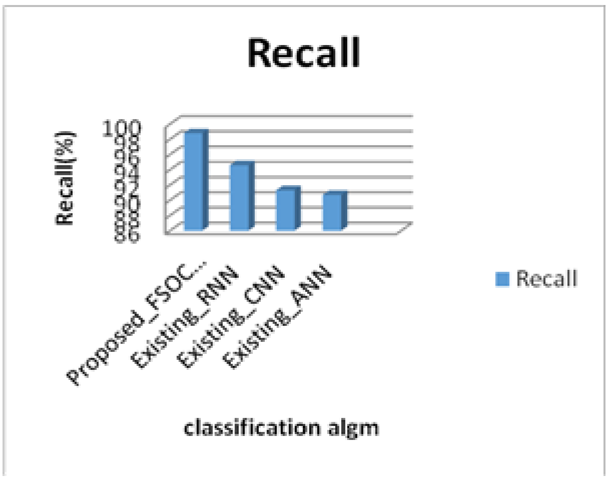


Fig 8(e): NPV comparison result

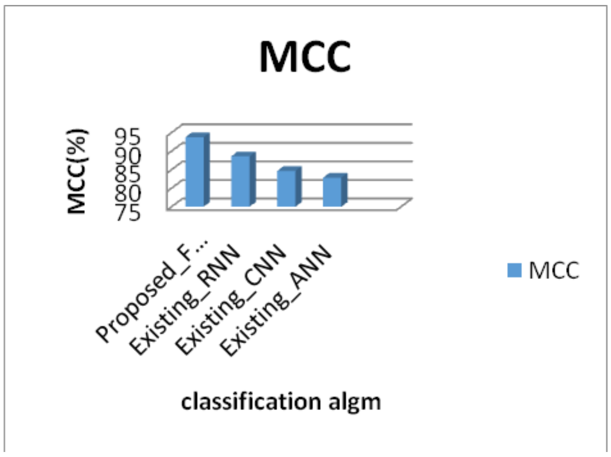


Fig 8(f): MCC Comparison result

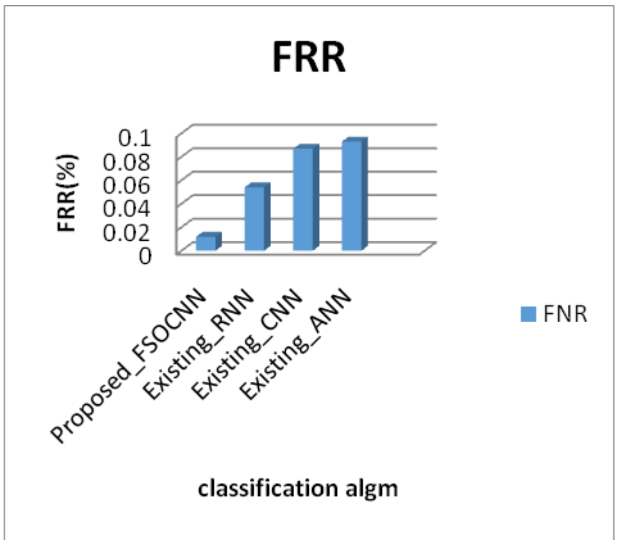


Fig 8(g):FRR comparison result

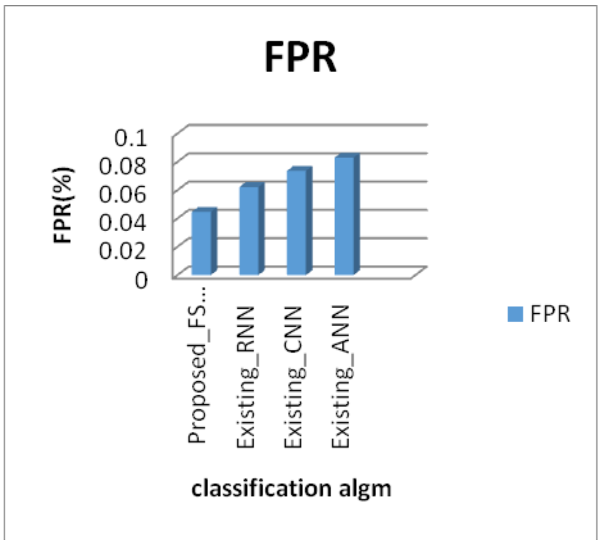


Fig 8(h): FPR Result

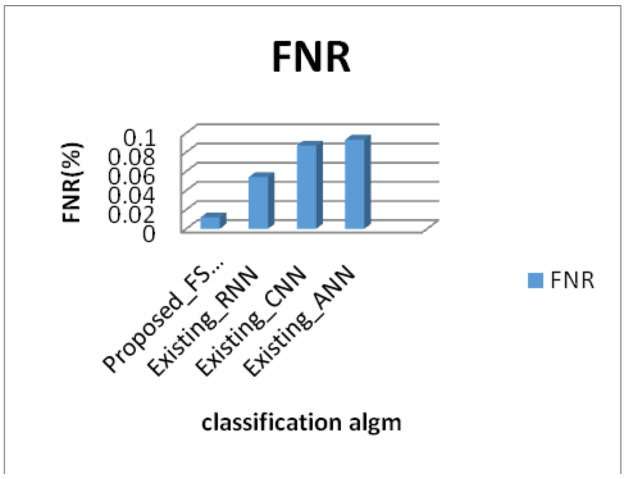


Fig 8(i): FNR Result

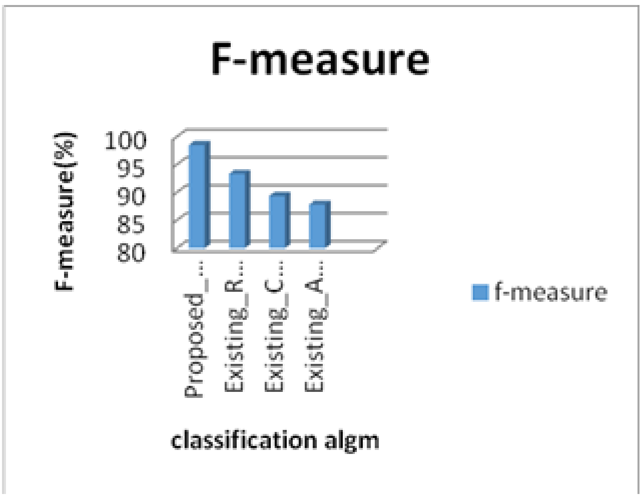


Fig 8(j):F_measures comparison

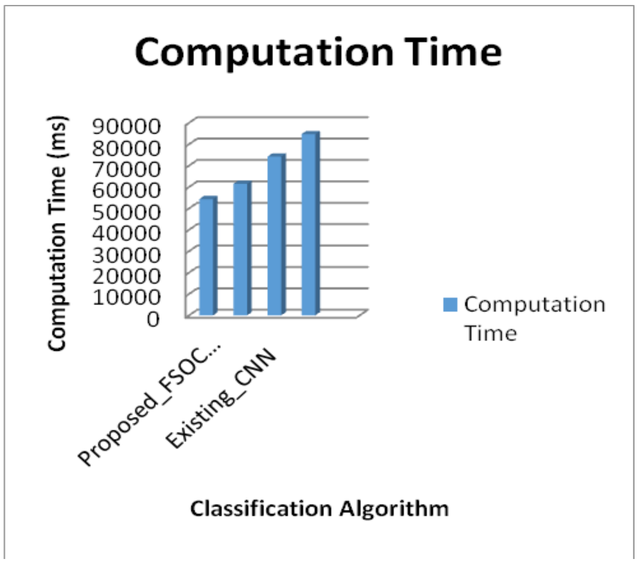


Fig 8(k): Computation time

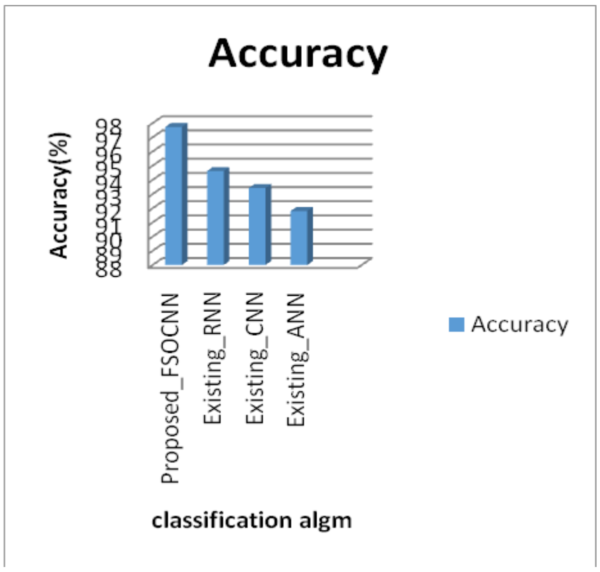


Fig 8(l): Accuracy Comparison result

Table 3. represents the comparison results of Fitness vs Iteration of existing and proposed algorithms.

Table3 : Comparison result of Fitness vs Iteration

Iteration	Existing CSA	Existing MBO	Existing HHO	Proposed AFO_CNN
10	32	56	94	114
20	53	78	117	137
30	75	93	134	157
40	96	117	152	179
50	113	134	171	197

+----------+-----------+--------------+--------------+--------------
+----------------

Fig. 9 represents the comparison results of Fitness vs Iteration of existing and proposed algorithms.

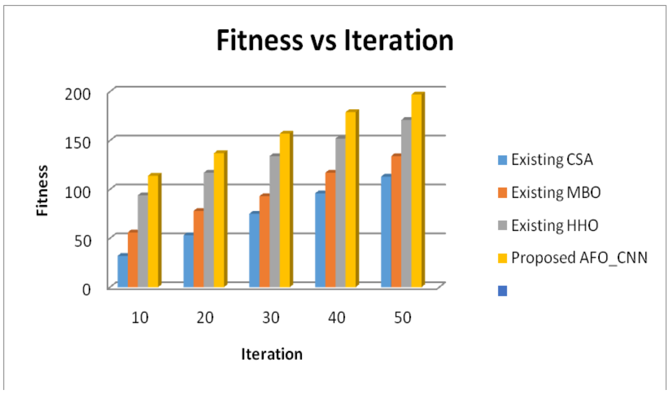


Fig 9: Fitness vs Iteration result

6. Conclusion

In this work, an EnsembleFramework with Optimal Features has been developed for Sarcasm Detection. Proposed work includes four phases: Data preprocessing, Feature Extraction, Optimal feature selection and Sarcasm detection. Most optimal features has been chosen using optimized Convolutional Neural Network model. The projected weight function of CNN has been fine-tuned by standard AptenodytesForsteri Optimization Algorithm (AFO), which was inspired from the emperor penguin's warm-hugging behavior. Sarcasm detection was carried out using optimized recurrent neural network in which activation function of Recurrent Neural Network is optimized via Firebug Swarm Optimization. FSO is based on Firebug reproductive swarming behaviour. The classifier has been trained via the optimal features acquired with optimized CNN model and the final outcome was acquired from optimized RNN.

The dataset for evaluation has been obtained from: https://www.kaggle.com/rmisra/news-headlines-dataset-for-sarcasm-detection.

Labels- 1: sarcastic and 0: Non- sarcastic.

References

[1] M. Bedi, S. Kumar, M. S. Akhtar and T. Chakraborty, "Multi-modal Sarcasm Detection and Humor Classification in Code-mixed Conversations," in IEEE Transactions on Affective Computing. doi: 10.1109/TAFFC.2021.3083522

[2] J. Li, H. Pan, Z. Lin, P. Fu and W. Wang, "Sarcasm Detection with Commonsense Knowledge," in IEEE/ACM Transactions on Audio, Speech, and Language Processing, vol. 29, pp. 3192-3201, 2021. doi: 10.1109/TASLP.2021.3120601

[3] Y. Zhang et al., "CFN: A Complex-Valued Fuzzy Network for Sarcasm Detection in Conversations," in IEEE Transactions on Fuzzy Systems, vol. 29, no. 12, pp. 3696-3710, Dec. 2021. doi: 10.1109/TFUZZ.2021.3072492

[4] L. H. Son, A. Kumar, S. R. Sangwan, A. Arora, A. Nayyar and M. Abdel-Basset, "Sarcasm Detection Using Soft Attention-Based Bidirectional Long Short-Term Memory Model With Convolution Network," in IEEE Access, vol. 7, pp. 23319-23328, 2019. doi: 10.1109/ACCESS.2019.2899260

[5] Y. Diao et al., "A Multi-Dimension Question Answering Network for Sarcasm Detection," in IEEE Access, vol. 8, pp. 135152-135161, 2020. doi: 10.1109/ACCESS.2020.2967095

[6] Kumar, V. T. Narapareddy, V. Aditya Srikanth, A. Malapati and L. B. M. Neti, "Sarcasm Detection Using Multi-Head Attention Based Bidirectional LSTM," in IEEE Access, vol. 8, pp. 6388-6397, 2020. doi: 10.1109/ACCESS.2019.2963630

[7] C. I. Eke, A. A. Norman and L. Shuib, "Context-Based Feature Technique for Sarcasm Identification in Benchmark Datasets Using Deep Learning and BERT Model," in IEEE Access, vol. 9, pp. 48501-48518, 2021. doi: 10.1109/ACCESS.2021.3068323

[8] S. Rendalkar and C. Chandankhede, "Sarcasm Detection of Online Comments Using Emotion Detection," 2018 International Conference on Inventive Research in Computing Applications (ICIRCA), Coimbatore, India, 2018, pp. 1244-1249. doi: 10.1109/ICIRCA.2018.8597368

[9] P. Verma, N. Shukla and A. P. Shukla, "Techniques of Sarcasm Detection: A Review," 2021 International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE), Greater Noida, India, 2021, pp. 968-972. doi: 10.1109/ICACITE51222.2021.9404585

[10] S. Porwal, G. Ostwal, A. Phadtare, M. Pandey and M. V. Marathe, "Sarcasm Detection Using Recurrent Neural Network," 2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS), Madurai, India, 2018, pp. 746-748. doi: 10.1109/ICCONS.2018.8663147

[11] [11] Wen, Z., Gui, L., Wang, Q., Guo, M., Yu, X., Du, J. and Xu, R., 2022. Sememe knowledge and auxiliary information enhanced approach for sarcasm detection. Information Processing & Management, 59(3), p.102883.

[12] [12] Govindan, V. and Balakrishnan, V., 2022. A machine learning approach in analysing the effect of hyperboles using negative sentiment tweets for sarcasm detection. Journal of King Saud University-Computer and Information Sciences.

[13] Meriem, A.B., Hlaoua, L. and Romdhane, L.B., 2021. A fuzzy approach for sarcasm detection in social networks. Procedia Computer Science, 192, pp.602-611.

[14] Hiremath, B.N. and Patil, M.M., 2021. Sarcasm Detection using Cognitive Features of Visual Data by Learning Model. Expert Systems with Applications, 184, p.115476.

[15] Ren, L., Xu, B., Lin, H., Liu, X. and Yang, L., 2020. Sarcasm detection with sentiment semantics enhanced multi-level memory network. Neurocomputing, 401, pp.320-326.

[16] Jain, D., Kumar, A. and Garg, G., 2020. Sarcasm detection in mash-up language using soft-attention based bi-directional LSTM and feature-rich CNN. Applied Soft Computing, 91, p.106198.

[17] Ren, Y., Ji, D. and Ren, H., 2018. Context-augmented convolutional neural networks for twitter sarcasm detection. Neurocomputing, 308, pp.1-7.

[18] Pandey, R., Kumar, A., Singh, J.P. and Tripathi, S., 2021. Hybrid attention-based long short-term memory network for sarcasm identification. Applied Soft Computing, 106, p.107348.

[19] F. Yao, X. Sun, H. Yu, W. Zhang, W. Liang and K. Fu, "Mimicking the Brain's Cognition of Sarcasm From Multidisciplines for Twitter Sarcasm Detection," in IEEE Transactions on Neural Networks and Learning Systems.

doi: 10.1109/TNNLS.2021.3093416

[20] N. Pawar and S. Bhingarkar, "Machine Learning based Sarcasm Detection on Twitter Data," 2020 5th International Conference on Communication and Electronics Systems (ICCES), Coimbatore, India, 2020, pp. 957-961.

[21] M. S. Razali, A. A. Halin, L. Ye, S. Doraisamy and N. M. Norowi, "Sarcasm Detection Using Deep Learning With Contextual Features," in IEEE Access, vol. 9, pp. 68609-68618, 2021. doi: 10.1109/ACCESS.2021.3076789

doi: 10.1109/ICIIS51140.2020.9342742

[22] P. Shrikhande, V. Setty and D. A. Sahani, "Sarcasm Detection in Newspaper Headlines," 2020 IEEE 15th International Conference on Industrial and Information Systems (ICIIS), RUPNAGAR, India, 2020, pp. 483-487.

[23] Z. Yin and F. You, "Multi-Modal Sarcasm Detection in Weibo," 2021 6th International Symposium on Computer and Information Processing Technology (ISCIPT), Changsha, China, 2021, pp. 740-743. doi: 10.1109/ISCIPT53667.2021.00156

[24] Zhe Yang, Li Bao Deng, Junfeng Liu, "Aptenodytes Forsteri Optimization: Algorithm and applications", Knowledge-Based Systems, 2021

[25] Mathew MithraNoel, VenkataramanMuthiah-Nakarajan,Advait Sanjay Trivedi,"A new biologically inspired global optimization algorithm based on firebug reproductive swarming behaviour", Expert Systems with Applications, 2021

[26] Ranga, K. K. ., Nagpal, C. K. ., & Vedpal, V. (2023). Trip Planner: A Big Data Analytics Based Recommendation System for Tourism Planning. International Journal on Recent and Innovation Trends in Computing and Communication, 11(3s), 159–174. https://doi.org/10.17762/ijritcc.v11i3s.6176

[27] Steven Martin, Kevin Hall, Ana Rodriguez, Ana Flores, Ana Silva. Machine Learning for Emotion Recognition in Educational Settings. Kuwait Journal of Machine Learning, 2(2). Retrieved from http://kuwaitjournals.com/index.php/kjml/article/view/186