IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON ADVANCED DATA MINING METHODS FOR SOCIAL COMPUTING

Received December 12, 2019, accepted December 24, 2019, date of publication January 3, 2020, date of current version January 10, 2020.

Digital Object Identifier 10.1109/ACCESS.2019.2963630

Sarcasm Detection Using Multi-Head Attention Based Bidirectional LSTM

AVINASH KUMAR, VISHNU TEJA NARAPAREDDY, VEERUBHOTLA ADITYA SRIKANTH, ARUNA MALAPATI, AND LALITA BHANU MURTHY NETI

Birla Institute of Technology and Science at Pilani, Hyderabad 500078, India

Corresponding author: Avinash Kumar (p20150507@hyderabad.bits-pilani.ac.in)

ABSTRACT Sarcasm is often used to express a negative opinion using positive or intensified positive words in social media. This intentional ambiguity makes sarcasm detection, an important task of sentiment analysis. Sarcasm detection is considered a binary classification problem wherein both feature-rich traditional models and deep learning models have been successfully built to predict sarcastic comments. In previous research works, models have been built using lexical, semantic and pragmatic features. We extract the most significant features and build a feature-rich SVM that outperforms these models. In this paper, we introduce a multi-head attention-based bidirectional long-short memory (MHA-BiLSTM) network to detect sarcastic comments in a given corpus. The experiment results reveal that a multi-head attention mechanism enhances the performance of BiLSTM, and it performs better than feature-rich SVM models.

INDEX TERMS Sarcasm detection, deep learning, self-attention, machine learning, social data.

I. INTRODUCTION

Social media is one of the biggest platforms for people to express their opinion and share information. Many governments and corporate organizations use this data to understand the sentiment of the people towards products, movies, and political events. Sarcasm is a way to convey negative opinions using positive or intensified positive words. On social media people often use sarcasm to express their views, and is inherently difficult to analyze not only for a machine but even for a human. The presence of sarcastic comments has an important effect on sentiment analysis tasks. For example, *"It is a wonderful feeling to carry an expensive phone with short battery-life."* is a sarcastic sentence expressing negative sentiment about battery life using positive opinion words like *"wonderful feeling"*. Therefore, sarcasm detection is important to improve the performance of sentiment analysis tasks. Sarcasm detection can be modelled as a binary classification task to predict the given sentence(s) as sarcastic or non-sarcastic. The previous research works in predicting the sarcastic sentences have mainly focused on rule-based and statistical approaches using (a) lexical and pragmatic features (b) presence of punctuations, interjections, sentiment shifts, etc., [3]. A deep neural network (DNN) provides an approach to learn necessary features automatically instead of using handcrafted features. Deep learning models have achieved state-of-the-art performance in various natural language processing (NLP) tasks such as text summarization [5], machine translation [6] and question answering [7]. Deep learning models have been explored in sarcasm detection as well and seem to achieve interesting results. Attention mechanism helps to enhance the performance of deep learning models as shown in machine translation [8], sentence summarization [5] and improving reading comprehension [9]. Furthermore, the attention mechanism has been explored for text classification [19] as well.

In this paper, we present a multi-head attention based deep neural network for sarcasm detection.

The main contribution of this paper can be summarized as follows:

- We consider various handcrafted features and build a support vector machine (SVM) model for sarcasm detection.
- We present a Multi-Head self-Attention based Bidirectional Long Short-Term Memory (MHA-BiLSTM) network, which uses the above mentioned handcrafted features as well. This model helps to identify significant parts of the sentence to enhance the performance of sarcasm detection.
- We reproduce the results of state-of-the-art models on our datasets and perform a comparative study.

The associate editor coordinating the review of this manuscript and approving it for publication was Yongqiang Zhao.

Our proposed MHA-BiLSTM model outperforms all other models.

The rest of our paper is organized as follows:

After discussing motivation and related work in Section II and Section III, we present a detailed description of our attention-based model and feature-rich SVM in Section IV. In Section V, we discuss, details of our extensive experiments. In Section VI, we analyze our results and quantify the effectiveness of our model. In Section VII, we discuss how the attention - mechanism helps our deep neural network in sarcasm detection. Finally, we summarize our work in section VII.

II. MOTIVATION

As explained above, the sarcasm detection helps in enhancing the performance of sentiment analysis tasks. However, sarcastic comments are commonly used in the social media platform. Previously, several statistical machine learning and neural network approaches have been proposed to detect sarcasm but they seem to have limitations in capturing implicit patterns and context that is used to express sarcasm.

The attention mechanism plays an important role in deep learning networks to capture the explicit and latent context. Vaswani *et al.* [26] introduce a **multi-head attention mechanism** that uses multiple individual attention functions for capturing different contexts. Multi-head attention can jointly pay attention to information from different representation subspaces at different positions.

Attention function takes input as, a sequence of query $Q = \{Q_1, \ldots, Q_N\}$ and a set of key-value pairs $\{K, V\} = \{(K_1, V_1), .., (K_R, V_R)\}$. Multi-head attention model first transforms Q, K, and V into C sub-spaces, with different, learnable linear projections.

$$Q^c, K^c, V^c = QW_c{}^Q, KW_c{}^K, VW_c{}^V \tag{1}$$

Here, Q^c, K^c, and V^c represents the c-th head of query, key, and value, respectively. $\{W_c{}^Q, W_c{}^K, W_c{}^V\} \in \mathbb{R}^{d \times d_k}$ are parameter matrices, d and d_k represent the dimensionality of the model and its subspace. Furthermore, C attention functions are applied simultaneously to get the output states $O^1, \ldots, O^C$

$$O^c = A^c V^c \tag{2}$$

$$A^c = softmax(\frac{Q^c K^{c^T}}{\sqrt{d_k}}) \tag{3}$$

A^c represents the attention distribution produced by the c-th attention head. These output states are concatenated to produce the final state. Figure 1 describes the computation graph of the multi-head attention mechanism.

The **self-attention mechanism** tries to learn the internal relationships among each semantic space. Lin *et al.* [19] proposes self-attention to extract an interpretable sentence embedding using self-attention. This approach does not require extra inputs except the sentence and provides a sentence representation that abstracts sentence-level meanings.

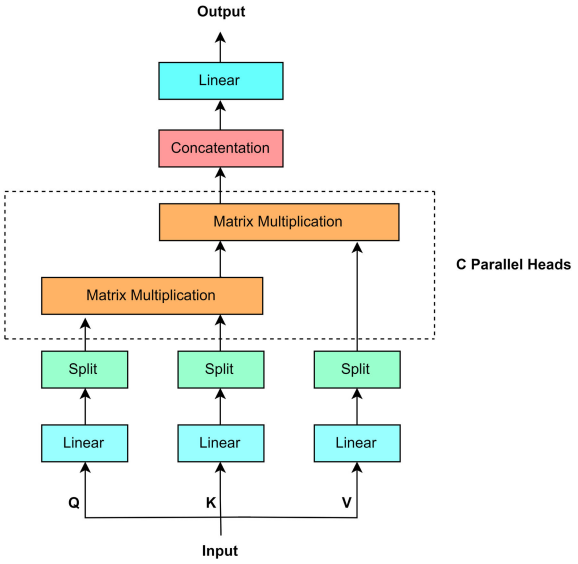


FIGURE 1. The computation graph of multi-head self-attention mechanism.

Motivated by this approach, we present a multi head attention based deep neural network for sarcasm detection.

III. RELATED WORK

The sarcasm detection task is a relatively new area of research in natural language processing and it has become a popular area of research in recent years. The main approaches for sarcasm detection have focused on finding lexical and pragmatic features to detect sarcasm in a given sentence. Several approaches have been implemented in literature which exhibit promising ways to discover interesting signals to identify sarcasm. Tepperman *et al.* [28] perform experiments to detect sarcasm in spoken language, specifically in the expression ''yeah right'', using spectral, contextual and prosodic cues. Eisterhold *et al.* [27] find that sarcasm can be recognized based on the statement preceding and following the sarcastic statement. Kreuz and Caucci [1] study lexical features and find that the presence of interjections and punctuation play an effective role in identifying sarcasm in the given corpus. Amir *et al.* [4] and Gonzalez-Ibanez *et al.* [11] show that oral or gestural expressions represented by emoticons and special keyboard characters are useful indicators of sarcasm. Davidov *et al.* [2] use syntactic and pattern-based features to train a sarcasm classifier. Liebrecht *et al.* [12] consider n-grams along with other signals like intensifiers, exclamations as features and Buschmeier *et al.* [13] propose features such as hyperbole, quotation marks and ellipsis as features for this task. Riloff *et al.* [14] demonstrate that the presence of positive sentiment in a negative situation provides an effective clue about sarcasm. Joshi *et al.* [15] use, multiple features, including lexical, pragmatic, implicit and explicit incongruity. Bouazizi and Ohtsuki [30] come up with a pattern-based approach to identify sarcasm on Twitter. Rajadesingan *et al.* [29] investigate the psychology

aspect behind sarcasm. They present behavioral modelling for sarcasm detection using Twitter data. They recognize different forms of sarcasm and demonstrate the importance of historical tweets provides contextual information and helps in sarcasm detection.

In the recent past, the deep neural network-based approach has gained popularity for the sarcasm detection task. One important reason behind the success of neural networks is the ability to learn latent features automatically. This ability makes neural networks very powerful and enables to explore implicit semantic patterns that are difficult to capture using manually extracted features, such latent features help in the detection of sarcasm. Ghosh and Veale [16] proposes a deep neural network-based sarcasm detection system by stacking a convolutional neural network on top of Long Short-Term Memory (LSTM). Amir *et al.* [4] build a deep neural network to focus on contextual features rather than lexical and syntactic patterns. Their network learns user embeddings and helps in improving the context-based sarcasm detection. Zhang *et al.* [21] develop a network by combining a bi-directional Gated Recurrent Unit (GRU) with a pooling neural network to detect sarcasm. Poria *et al.* [17] come up with a CNN based approach for sarcasm detection. They use a pre-trained CNN for extracting sentiment, emotion and personality features for sarcasm detection. Sulis *et al.* [22] explore tweets to understand the difference between sarcasm and irony. They come up with a combination of sentiment, structural and psycholinguistic features to differentiate between irony and sarcasm. Hazarika *et al.* [18] present a fusion approach, they extract contextual information from the discourse section of a discussion thread, also they use user embeddings to encode stylometric and personality features of users. Their sarcasm detection model shows promising results on a large Reddit[1] corpus.

IV. METHODS

Sarcasm detection tasks can be considered as a binary classification task, where every sentence(s) is classified as either sarcastic or non-sarcastic. In this section, we introduce our deep learning model using a multi-head attention based Bidirectional Long Short-Term Memory (MHA-BiLSTM) network. We also develop a support vector machine (SVM) model that shows the importance of handcrafted features in classification.

A. DEEP LEARNING-BASED APPROACH

We build a multi-head self-attention based deep neural network. We use Long Short-term Memory Network (LSTM) to build a deep neural model. LSTM is a type of Recurrent Neural Network (RNN) that is capable of learning and remembering long term dependencies without encountering vanishing gradient descent or exploding problems [31]. There are three gates and a cell memory state in LSTM. Each cell of LSTM is computed as follows:

$$X = \begin{bmatrix} h_{t-1} \\ x_t \end{bmatrix} \tag{4}$$

$$f_t = \sigma(W_f \cdot X + b_f) \tag{5}$$

$$i_t = \sigma(W_i \cdot X + b_i) \tag{6}$$

$$o_t = \sigma(W_o \cdot X + b_o) \tag{7}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot tanh(W_c \cdot X + b_c) \tag{8}$$

$$h_t = o_t \odot tanh(c_t) \tag{9}$$

where i_t, f_t, o_t and c_t represents the input gate, forget gate, output gate and cell state. W_i, W_f, $W_o \in \mathbb{R}^{d \times 2d}$ are the weighted matrices that are used for mapping the hidden layer input to three gates and the input cell state. b_i, b_f, $b_o \in \mathbb{R}^d$ are biases. σ is the sigmoid function and $\odot$ stands for element-wise multiplication. x_t and h_t denote the input and hidden vector of LSTM cell unit. The bidirectional LSTM consists of a forward LSTM layer and a backward LSTM layer. The forward layer captures the historical information of the sequence; the backward layer captures the future information of the sequence. Both layers are connected to the same output layer. Our network uses Bidirectional LSTM with the multi-head mechanism. Multi-head attention allows the model to jointly attend to information from different representation subspaces at different positions. Figure 2 shows the architecture of our network. This Multi-Head Attention-based Bidirectional LSTM (MHA-BiLSTM), consists of five main parts.

1) WORD EMBEDDING LAYER

Given an input comment S that consists of N words x_i, where $i \in [1, N]$. For each word in S, we first look up the embedding matrix $E \in \mathbb{R}^{V \times d}$, where V is a fixed-sized vocabulary, and d is size of word embedding. E is initialized by a pre-trained word embedding vector. Every word x_i is converted into a its vector representation w_i. Thus, comment S can be represented as a sequence of words in the form of a 2-D matrix of shape N-by-d.

$$S = (w_1, w_2, w_3, \ldots, w_N)^T \tag{10}$$

2) WORD ENCODER LAYER

Each word in the comment S is independent of other words, when the words are represented by making use of word embedding E. In this layer, a new representation for each word is achieved by summarizing contextual information from both the directions in a comment. The bidirectional LSTM is a combination of forward LSTM $\overrightarrow{h}$,which reads the comment from x_1 to x_N and a backward LSTM $\overleftarrow{h}$, which reads the comment from x_N to x_1:

$$\overrightarrow{h}_t = \overrightarrow{LSTM}(w_t, \overrightarrow{h_{t-1}}) \tag{11}$$

$$\overleftarrow{h}_t = \overleftarrow{LSTM}(w_t, \overleftarrow{h_{t+1}}) \tag{12}$$

We concatenate forward hidden state $\overrightarrow{h}$ and backward hidden state $\overleftarrow{h}$ i.e. $h_t = [\overrightarrow{h_t}, \overleftarrow{h_t}]$, to obtain hidden state

[1] https://www.reddit.com/

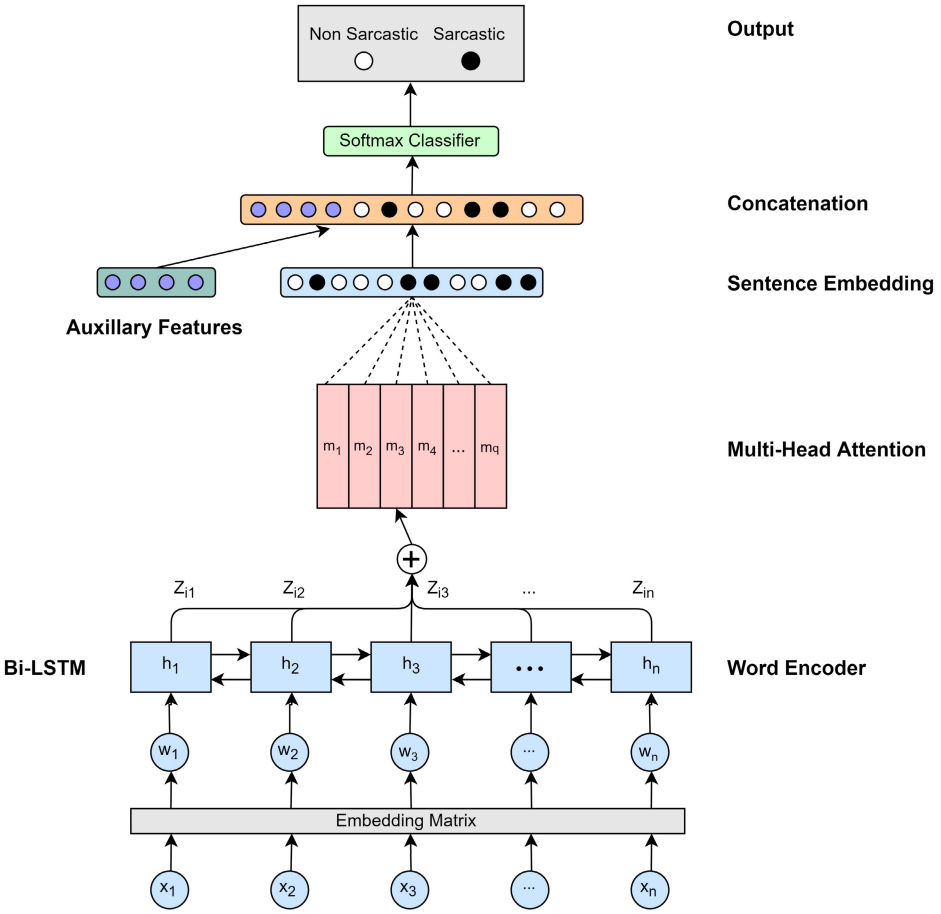


FIGURE 2. Multi-Head Self Attention Network showing the sentence embedding model combined with a fully connected and softmax layer for sarcasm detection.

representation h_t for each word x_t. This process helps in capturing information of whole sentence around every word x_t. We denote all the hidden state of the words x_t as $H \in \mathbb{R}^{N\times 2p}$, where size of $\overrightarrow{h}$ and $\overleftarrow{h}$ be p.

$$H = (h_1, h_2, \ldots, h_N) \tag{13}$$

3) SENTENCE LEVEL MULTI-HEAD ATTENTION LAYER

In a given comment, a specific part of the comment plays an important role in detecting sarcasm. However, there can be multiple factors by which a word can be paid attention to, hence we need multiple heads of attention wherein each word is given appropriate importance from multiple factors to represent the overall semantics of the comment.

$$Y = tanh(W_{k1}H^T) \tag{14}$$
$$Z = softmax(W_{k2}Y) \tag{15}$$

The attention layer takes whole hidden states H as input and multiply it with $W_{k1} \in \mathbb{R}^{g\times 2p}$, the output is then passed to tanh function to get Y. To extract attentions of each component from different factors Y is multiplied $W_{k2} \in \mathbb{R}^{q\times g}$ and that is passed to *softmax* to calculate the normalized importance weight along different heads (q) resulting a vector of weights Z.

A combination of equation (14) and (15) looks like a 2-layer multilayer perceptron (MLP), with the hidden unit number g and parameters as W_{k1} and W_{k2}.

We multiply hidden states of word H with weight vector Z and compute the q weighted sum resulting in a matrix M, which represents the sentence embedding.

$$M = ZH \tag{16}$$

4) AUXILIARY FEATURES CONCATENATION

We extract semantic, sentiment and punctuation based handcrafted features (refer section IV-B for more details) for the given comment. Using these features we create an auxiliary feature vector $F \in \mathbb{R}^d$ of d dimensions and then combine it with self-attentive sentence embedding M to generate new sentence representation $M' = M \oplus F$.

5) SOFTMAX LAYER

We pass new sentence representation M' to *softmax* layer for sarcasm detection as shown below :

$$\hat{y} = softmax(W_f M' + b_f) \tag{17}$$

TABLE 1. Semantic features based on LIWC dictionary.

LP	I, them, walk, and, few, many, never, second, thousand, above, to, etc.
PP	Mate, husband, buddy, adult, baby, happy, love, hurt, worried, hate, always, would, listen, view, cheek, clinic, eat, arrive, end, until, blood, feel, touch,etc.
PC	Job, majors, xerox, earn, hero, cook, chat, movies, apartment, kitchen, family, cash, church, temple, coffin, kill, etc.

where W_f and b_f are the weight matrix and bais of the final linear layer. The binary cross-entropy loss is used to train this model.

$$loss = -\sum_{i=1}^{e} y_i log(\hat{y_i}) \tag{18}$$

Here, e denotes the number of class labels. y_i is class labels for the i-th class and $\hat{y_i}$ represents predicted probability for the i-th class.

B. STATISTICAL MACHINE LEARNING-BASED APPROACH

We develop a statistical machine learning-based model using SVM. The following handcrafted features are used in this model.

1) SEMANTIC FEATURES

Lexical features of sentence can provide insightful features about sarcasm, we use the LIWC dictionary [25] to get patterns based on semantic information. The LIWC dictionary includes 64 different lexical categories that are classified under the following 3 classes.

- *Linguistic Processes (LP)*: Words that are classified as pronouns, articles, verbs, adverbs, conjunctions, negations, quantifiers, etc. are grouped under this class.
- *Psychological Processes (PP)*: Words that are tagged as social, affective, cognitive biological processes are grouped under this class.
- *Personal Concerns (PC)*: Words that relate to work, achievement, leisure, home, religion, death, etc are grouped under this class.

Table 1 provides the summarized information about 3 classes of LIWC dictionary. For a given sentence, we take each word and do a lookup using the LIWC dictionary. Count of words that belong to each class of LIWC is considered as semantic features for sarcasm detection.

2) SENTIMENT FEATURES

Sarcasm is used to express irritation or anger about a negative situation. Therefore, people use exaggerated and very positive expressions to describe their adverse situation. We consider the following sentiment related features for sarcasm detection

- Hyperbole: The presence of three consecutive positive or negative words in a sentence is called hyperbole [32]. For example *well just so long as the defense pulls money away from women's health care, it's* ***win win right****?. Win win right* is hyperbolic. We consider the count of such hyperbole occurrences in a given sentence as a feature.
- Positive/Negative Punctuation: This feature indicates whether there exists at least one positive with no negative words and vice versa, ending with an exclamation or question mark amongst four consecutive words in a sentence. For example *don't you* ***love it when reddit*** *gives you negative points just for stating your own subjective opinion on beer ?*. In this sentence *"love"* is the positive word and no negative word exists within the span of four words, and it ends with a question mark.
- Positive/Negative Ellipsis: This feature indicates whether there exists at least one positive with no negative words and vice versa, followed by an ellipsis amongst four consecutive words in a sentence. For example: to me,*food is a basic* ***right, so I have*** *the right to the food in your pantry. . ..* In this sentence *"right"* is a positive word within the span of four words, and it ends with an ellipsis.
- Maximum Length Positive/Negative phrase: Maximum length of contiguous positive or negative words in a sentence is also considered as one of the features. For example *yeah, he was an evil man because he wrote* ***brilliant, well-reasoned*** *legal opinions I disagreed with!*. This sentence has a maximum length as three for contiguous positive words.

3) PUNCTUATION FEATURES

Semantic and sentiment related features are not enough to detect the sarcasm in the given sentence. Sarcasm is a sophisticated form of speech that not only plays with words and meanings but also employs behavioral aspects like low tones, facial gestures or exaggeration. These facets are expressed using punctuation while writing a sentence. To capture such facets we extract following punctuation-based features in a given sentence.

- Number of quotes: Quotes in a sentence emphasizes that the meaning of the word is intended to be different from its literal meaning. For example *As you can see, this 'premium' product is, in fact, a piece of garbage.*, quotes used with word *'premium'* is used to inverse the meaning of the word.
- Number of exclamation marks: Usage of exclamation marks increases the intensity of the emotions conveyed without changing the sentiment orientation. "The book is great!!!!" is more intense than "The book is great."
- Number of question marks: Combination of question marks and exclamation marks express intensified emotions. For example: *Really?!! I thought this was the circus*

TABLE 2. Details of the sarcastic (sarc) and non-sarcastic (non-sarc) comments in the training datasets.

	No. of comments		avg. no. of words per comment	
	non-sarc	sarc	non-sarc	sarc
Balanced	57000	57000	30.36	27.58
Imbalanced	57000	19000	30.49	27.64

TABLE 3. Details of the sarcastic (sarc) and non-sarcastic (non-sarc) comments in the testing datasets.

	No. of comments		avg. no. of words per comment	
	non-sarc	sarc	non-sarc	sarc
Balanced	24160	24160	24.91	20.02
Imbalanced	34843	10754	24.95	22.32

- Number of ellipsis: Ellipsis represents a pause in conversations that are sometimes used to convey sarcasm. *Sure.... That sounds great!!*
- Number of interjections: Words like ''aah'', ''eh'', ''hmm'', etc, Aah! is used to express emotion in an abrupt and exclamatory way. The occurrence of such words gives a clue about sarcasm, For example, *Aah! I love cleaning my room mom.....* Here, *''Aah!''* is used in a sarcastic way to express annoyance.

V. EXPERIMENTAL SETUP

A. DATASETS

Table 2 and Table 3 presents the details of the training and testing datasets respectively.

We use a large self-annotated corpus for sarcasm, SARC [1] [20] to create our datasets. This dataset contains more than a million sarcastic and non-sarcastic comments written on social media site Reddit, which is a topic-specific forum. In each forum, users write their comments in the context of the titled topic or other's comments. SARC dataset contains the user's current comment, author details, and parent comments (if any). This dataset contains comments from 1246058 users (118940 in training and 56118 in the testing set) distributed across 6534 forums (3868 in training and 2666 in the testing set). We use training and testing datasets of the SARC corpus to create balanced and imbalanced datasets for our experiments. Our dataset comprises two fields 1). User's comment 2). The label of the comment (sarcastic/non-sarcastic). The balanced dataset contains an equal proportion of sarcastic and non-sarcastic comments in training and testing datasets. Our imbalanced dataset represents a real-world scenario, where sarcastic comments are ideally lesser than non-sarcastic comments. We maintain a 25:75 ratio (approx.) between the sarcastic and non-sarcastic comments in both training/testing datasets.

[1] http://nlp.cs.princeton.edu/SARC/

B. EVALUATION METRICS

We use *Precision*, *recall* and *F-score* for evaluating the performance of sarcasm detection model. Precision is the ratio of correctly predicted sarcastic sentences to total predicted sarcastic sentences. A recall is the ratio of correctly predicted sarcastic sentences to all actual sarcastic sentences. F-score is the harmonic mean of precision and recall. These metrics are calculated as

$$Precision = \frac{tp}{tp + fp} \tag{19}$$

$$Recall = \frac{tp}{tp + fn} \tag{20}$$

$$F - score = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{21}$$

where tp is the number of correctly predicted sarcastic sentences. fp is the number of sentences that predicted as sarcastic, but in actuality those are non-sarcastic. fn is the number of sentences that are Predicted as non-sarcastic, but in actual they are sarcastic.

C. MODEL CONFIGURATION

This subsection describes the pre-processing and hyperparameter setting, we have used.

1) PRE-PROCESSING AND WORD EMBEDDINGS

We cleanse the data by removing stop words, URLs, hashtags, non-ASCII-English characters and perform case folding. We split each training set randomly into two training (90%) and validation (10%). Sentences are then tokenized into words and using these words, we create a vocabulary. The tokens appearing less than 5 times in the vocabulary are replaced with a special UNK token. Finally, we use pre-trained word embeddings, Glove [24] to convert each word in a sentence into a vector.

2) HYPERPARAMETERS AND TRAINING DETAILS

We train all our statistical-based machine learning models using SVM with RBF kernel with default parameters and a maximum number of iterations 1000.

For baseline model BiLSTM, we use 100-dimension word embedding, 100 hidden units and fix the dropout of 0.5. Word encoder layer of MHA-BiLSTM also uses the same settings as BiLSTM. For self-attentive sentence embeddings, we use the MLP layer of the hidden unit of 200 (the g in section III). We use the 4 attention heads and 11-dimensional auxiliary feature vector.

We train our deep neural network models using standard Adam optimizer [23] with a learning rate of 0.005, a mini-batch size of 128 and the number of epochs as 50. Models get trained on the training dataset and its performance gets evaluated on the validation dataset after every epoch using the F - score. If after consecutive 5 epochs performance of a model doesn't increase, then we do early stopping and save the best model. We evaluate the performance of this model

TABLE 4. Precision, Recall and F-score of previous approaches on datasets.

Approach	Balanced			Imbalanced		
	Precision	Recall	F-score	Precision	Recall	F-score
Liebrecht et al. [12]	67.94%	75.02%	71.31%	58.38%	35.03%	43.79%
Gonz´alez-Ib´anez et al. [11]	72.71%	73.17%	72.94%	64.56%	39.23%	48.81%
Buschmeier et al. [13]	72.12%	73.70%	72.90%	64.09%	38.37%	48.00%
Joshi et al. [15]	72.58%	73.49%	***73.03%***	64.65%	39.33%	***48.91%***

TABLE 5. Precision, Recall and F-score of our approaches on datasets.

Approach	Balanced			Imbalanced		
	Precision	Recall	F-score	Precision	Recall	F-score
SVM	72.30%	75.97%	74.09%	52.14%	53.70%	52.91%
BiLSTM	69.41%	77.75%	73.34%	59.94%	43.18%	50.20%
MHA-BiLSTM-w/o-auxiliary-features	71.92%	80.02%	75.76%	59.61%	50.00%	54.38%
MHA-BiLSTM	72.63%	83.03%	***77.48%***	60.26%	53.71%	***56.79%***

using precision, recall, and F-score on the test dataset. The results reported are an average of five such runs.

VI. RESULTS

In this section, we discuss the results of our experiments and related findings.

A. STATISTICAL MACHINE LEARNING MODEL Vs DEEP NEURAL NETWORK MODEL

As discussed in section IV, we build feature-rich SVM, BiLSTM and MHA-BiLSTM. We also build MHA-BiLSTM-w/o-auxiliary-feature, which is a variation of MHA-BiLSTM. This model does not use auxiliary-features in its network architecture. We compare the performance of our models using the details shown in Table 5. It is observed that BiLSTM without attention shows the least F-score on both the datasets. Feature-rich SVM performs better than BiLSTM but its performance is inferior to MHA-BiLSTM by a significant margin on both the datasets. F-score comparisons between BiLSTM and MHA-BiLSTM-w/o-auxiliary-features show that MHA-BiLSTM-w/o-auxiliary-features performs better than BiLSTM by 2.42% and 4.18% on the balanced and imbalanced dataset, respectively. It depicts that the multi-head self-attention mechanism improves the performance of the deep neural network. Performance comparison between MHA-BiLSTM-w/o-auxiliary-features and MHA-BiLSTM is also very interesting, MHA-BiLSTM outperforms MHA-BiLSTM-w/o-auxiliary-features on both the datasets. It shows manually designed auxiliary features play an important role in boosting the performance of MHA-BiLSTM. Figure. 6 and Figure. 7 graphically illustrates the comparative results of our models on both the datasets. Further analysis of the results shows that MHA-BiLSTM, classifies comments as sarcastic with better recall compare to SVM and MHA-BiLSTM-w/o-auxiliary-features. Recall of MHA-BiLSTM is better than BiLSTM, MHA-BiLSTM-w/o-auxiliary-features, and SVM by an absolute margin of 5.28%, 2.99% and 7.06% on the balanced dataset. Similarly, on the imbalanced dataset, recall of MHA-BiLSTM is better than BiLSTM, MHA-BiLSTM-w/o-auxiliary-features and SVM by 10.53%, 3.71% and 0.01% respectively.

Further, we also conduct an extensive experiment to understand the **importance of the number of attention-heads** in MHA-BiLSTM. We build different versions of MHA-BiLSTM by varying the number of heads and evaluated their F-score on both the datasets. Experimental results of table 6 show that by increasing the number of attention heads from 1 to 4 enhances the F-score of MHA-BiLSTM on both the datasets, but a further increase of attention-head degrades the performance of MHA-BiLSTM. Figure. 4 and Figure. 5 shows variation in the F-score of MHA-BiLSTM corresponding to the number of attention-heads on both the datasets.

B. BASELINE Vs. OUR MODELS

We reproduce results of previous research work reported by Liebrecht *et al.* [12], Gonzalez-Ibanez *et al.* [11], Buschmeier *et al.* [13] and Joshi *et al.* [15]. Table 4 shows the precision, recall, and F-score of four previous approaches on the balanced and imbalanced dataset. We notice that among these approaches, features from Joshi *et al.* [15] perform the best and achieve F-score of 73.03% and 48.91% on the balanced and imbalanced dataset, respectively.

We use Table 4 and 5 to compare the performance of state-of-the-art models with the models proposed in our study. It is evident that all our proposed models perform better than the state-of-the-art models. The F-score comparison shows that our MHA-BiLSTM outperforms the model built by Joshi *et al.* [15] by a significant margin of 4.45% and 7.88% on the balanced dataset and imbalanced dataset, respectively.

FIGURE 3. Distribution of attention weights over some input comments that is classified as making the sarcastic. The darker the color and larger the font, the higher is the weight.

FIGURE 4. F-Score of MHA-BiLSTM on balanced dataset corresponding to different attention-heads.

FIGURE 5. F-Score of MHA-BiLSTM on imbalanced dataset corresponding to different attention-heads.

VII. DISCUSSION

Analysis of Attention: The intuition behind multi-head self-attention is to extract different aspects of the comment. Single attention-head usually focuses on a particular segment of comment, like a set of related words or phrases. This mechanism helps to understand an aspect of semantics in the comment. Multiple attention-heads focus on different parts of the comment to understand various aspects and forms overall semantics of the comment. Figure.3 shows, distribution of

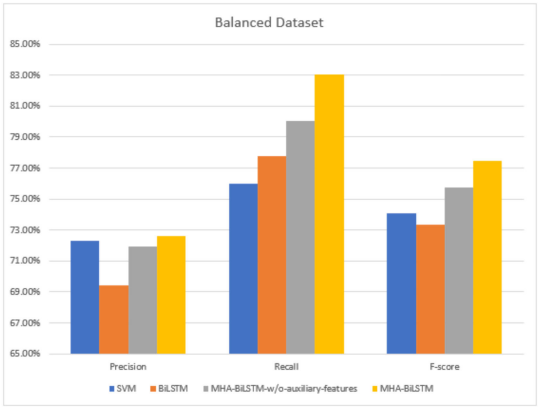


FIGURE 6. Precision, Recall and F-Score Comparison of our models in Balanced dataset.

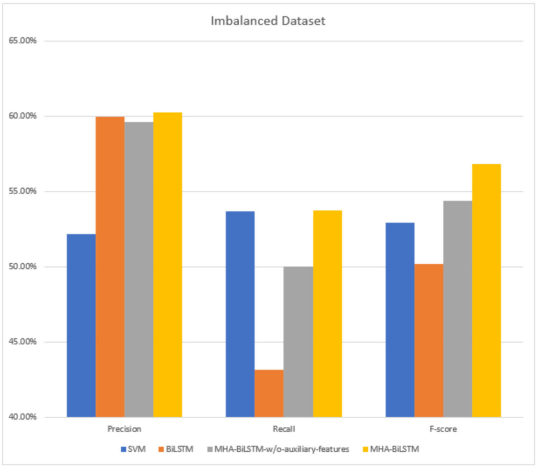


FIGURE 7. Precision, Recall and F-Score Comparison of our models in Imbalanced dataset.

TABLE 6. F-score values corresponding to the number of heads in MHA-BiLSTM.

No. of heads	Balanced	Imbalanced
1	73.86%	52.03%
2	75.08%	53.71%
3	76.41%	55.64%
4	***77.48%***	***56.79%***
5	75.93%	55.86%
6	75.28%	55.03%
7	74.46%	54.26%
8	73.69%	53.18%

attention weights over some Reddit comments while making the sarcastic decision. The darker shade of colour indicates the higher weight of attention and lighter shade indicates for smaller weight. This visualization has been created by taking a list of words from comment and corresponding attention weights. This figure shows the attention ability of the model.

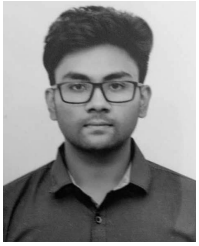
VIII. CONCLUSION

In this paper, we develop a feature-rich SVM model that uses the semantic, sentiment and punctuation based hand-crafted features for sarcasm detection. We compare our SVM model with four previous works and find that our feature-rich model has a better F-score than others. The major contribution of this work is to introduce multi-head attention based Bidirectional Long-Short Term Memory (MHA-BiLSTM) for sarcasm detection. Our proposed neural network consists of two main layers: word encoder and sentence level multi-head attention. The word encoder layer provides a new representation for each word by summarizing its contextual information from both the directions in a comment. The sentence-level multi-head attention layer simultaneously focuses on different parts of the comment to understand various aspects of the semantics of comment. We find that the inclusion of manually generated auxiliary features in the network further enhances the effectiveness of the BiLSTM model. Our experimental results show that MHA-BiLSTM outperforms all other models.

REFERENCES

[1] R. J. Kreuz and M. G. Caucci, "Lexical influences on the perception of sarcasm," in *Proc. Workshop Comput. Approaches Figurative Lang.*, 2007, pp. 1–4.
[2] D. Davidov, O. Tsur, and A. Rappoport, "Semi-supervised recognition of sarcastic sentences in Twitter and Amazon," in *Proc. 14th Conf. Comput. Natural Language Learn.*, 2010, pp. 107–116.
[3] A. Joshi, P. Bhattacharyya, and J. M. Carman, "Automatic sarcasm detection: A survey," *ACM Comput. Surv.* (CSUR), vol. 50, no. 5, p. 73, 2017.
[4] S. Amir, C. B. Wallace, H. Lyu, P. Carvalho, and J. M. Silva, "Modelling context with user embeddings for sarcasm detection in social media," in *Proc. CONLL* 2016.
[5] A. M. Rush, S. Chopra, and J. Weston, "A neural attention model for abstractive sentence summarization," 2015, *arXiv:1509.00685*. [Online]. Available: https://arxiv.org/abs/1509.00685
[6] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer, "Neural architectures for named entity recognition," 2016, *arXiv:1603.01360*. [Online]. Available: https://arxiv.org/abs/1603.01360
[7] D. Golub and X. He, "Character-level question answering with attention," 2016, *arXiv:1604.00727*. [Online]. Available: https://arxiv.org/abs/1604.00727
[8] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," 2014, *arXiv:1409.0473*. [Online]. Available: https://arxiv.org/abs/1409.0473
[9] K. M. Hermann, T. Kocisky, E. Grefenstette, L. Espeholt, W. Kay, M. Suleyman, and P. Blunsom, "Teaching machines to read and comprehend," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 1684–1692.
[10] P. Carvalho, L. Sarmento, M. J. Silva, and E. de Oliveira, "Clues for detecting irony in user-generated contents: Oh···!! it's 'so easy' ;-)," in *Proc. ACM 1st Int. CIKM Workshop Topic-Sentiment Anal. Mass Opinion*, 2009, pp. 53–56.
[11] R. Gonzalez-Ibanez, S. Muresan, and N. Wacholder, "Identifying sarcasm in Twitter: A closer look," in *Proc. 49th Annu. Meeting Assoc. Comput. Linguistics, Hum. Lang. Technol.*, vol. 2, 2011, pp. 581–586.
[12] C. Liebrecht, F. Kunneman, and A. van den Bosch, "The perfect solution for detecting sarcasm in tweets #not," in *Proc. 4th Workshop Comput. Approaches Subjectivity, Sentiment Social Media Anal.*, 2013, pp. 29–37.
[13] K. Buschmeier, P. Cimiano, and R. Klinger, "An impact analysis of features in a classification approach to irony detection in product reviews," in *Proc. 5th Workshop Comput. Approaches Subjectivity, Sentiment Social Media Anal.*, 2014, pp. 42–49.
[14] E. Riloff, A. Qadir, P. Surve, L. De Silva, N. Gilbert, and R. Huang, "Sarcasm as contrast between a positive sentiment and negative situation," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2013, pp. 704–714.
[15] A. Joshi, V. Sharma, and P. Bhattacharyya, "Harnessing context incongruity for sarcasm detection," in *Proc. 53rd Annu. Meeting Assoc. Comput. Linguistics, 7th Int. Joint Conf. Natural Lang. Process.*, vol. 2, 2015, pp. 757–762.
[16] A. Ghosh and D. T. Veale, "Fracking sarcasm using neural network," in *Proc. 7th Workshop Comput. Approaches Subjectivity, Sentiment Social Media Anal.*, 2016, pp. 161–169.
[17] S. Poria, E. Cambria, D. Hazarika, and P. Vij, "A deeper look into sarcastic tweets using deep convolutional neural networks," in *Proc. COLING 26th Int. Conf. Comput. Linguistics, Tech. Papers*, 2016, pp. 1601–1612.
[18] D. Hazarika, S. Poria, S. Gorantla, E. Cambria, R. Zimmermann, and R. Mihalcea, "Cascade: Contextual sarcasm detection in online discussion forums," in *Proc. 27th Int. Conf. Comput. Linguistics*, 2018, pp. 1837–1848.
[19] Z. Lin, M. Feng, C. N. dos Santos, M. Yu, B. Xiang, B. Zhou, and Y. Bengio, "A structured self-attentive sentence embedding," in *Proc. 5th Int. Conf. Learn. Represent. (ICLR)*, 2017.
[20] M. Khodak, N. Saunshi, and K. Vodrahalli, "A large self-annotated corpus for sarcasm," 2017, *arXiv:1704.05579*. [Online]. Available: https://arxiv.org/abs/1704.05579
[21] M. Zhang, Y. Zhang, and G. Fu, "Tweet sarcasm detection using deep neural network," in *Proc. COLING 26th Int. Conf. Comput. Linguistics, Tech. Papers*, 2016, pp. 2449–2460.
[22] E. Sulis, D. I. H. Faras, P. Rosso, V. Patti, and G. Ruffo, "Figurative messages and affect in twitter: Differences between irony, sarcasm and not," *Knowl.-Based Syst.*, vol. 108, pp. 132–143, Sep. 2016.
[23] D. P Kingma and J. L. Ba, "Adam: Amethod for stochastic optimization," in *Proc. 3rd Int. Conf. Learn. Represent.*, 2014.
[24] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2014.
[25] J. W. Pennebaker, M. E. Francis, and R. J. Booth, *Linguistic Inquiry and Word Count (LIWC): LIWC2001*. Mahwah, NJ, USA: Erlbaum Publishers, 2001.
[26] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. NIPS*, 2017.
[27] J. Eisterhold, S. Attardo, and D. Boxer, "Reactions to irony in discourse: Evidence for the least disruption principle," *J. Pragmatics*, vol. 38, no. 8, pp. 1239–1256, 2006.
[28] J. Tepperman, D. Traum, and S. S. Narayanan, "Yeah right: Sarcasm recognition for spoken dialogue systems," in *Proc. InterSpeech*, Pittsburgh, PA, USA, Sep. 2006, pp. 1838–1841.
[29] A. Rajadesingan, R. Zafarani, and H. Liu, "Sarcasm detection on Twitter: A behavioral modeling approach," in *Proc. 18th ACM Int. Conf. Web Search Data Mining*, Feb. 2015, pp. 79–106.
[30] M. Bouazizi and T. O. Ohtsuki, "A pattern-based approach for sarcasm detection on Twitter," *IEEE Access*, vol. 4, pp. 5477–5488, 2016.
[31] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
[32] R. W. Gibbs and H. L. Colston, Eds., *Irony in Language and Thought*. New York, NY, USA: Routledge, 2007.

AVINASH KUMAR received the M.S. degree in software systems from the Birla Institute of Technology and Science at Pilani, India, 2014, where he is currently pursuing the Ph.D. degree in computer science. He worked in the field of data science and data engineering in Oracle, Barclays, and HSBC. He is also working for Microsoft, Hyderabad, and India, as a Senior Data and an Applied Scientist. His current research interests include natural language processing, deep learning, and time series forecasting.



VISHNU TEJA NARAPAREDDY is currently pursuing the B.E. degree (Hons.) in computer science with the Birla Institute of Technology and Science at Pilani, Hyderabad Campus. He has worked as a Summer Intern with Amazon India and the Regional Remote Sensing Centre, Jodhpur. His research interests include natural language processing, information retrieval, and deep learning.

VEERUBHOTLA ADITYA SRIKANTH is currently pursuing the B.E. degree (Hons.) in computer science with the Birla Institute of Technology and Science at Pilani, Hyderabad Campus. He has worked with Microsoft India and the Srujana Center for Innovation, LVPEI. His research interests include Natural language processing, reinforcement learning, and multitask learning.

ARUNA MALAPATI received the bachelor's degree from Gulbarga University, the master's degree from BITS Pilani, and the Ph.D. degree from NIT, Karnataka. She has been working as an Associate Professor with the Department Computer Science and Information Systems, BITS Pilani, Hyderabad Campus, since 2010. She moved to BITS in 2010 after gaining 11 years of teaching and two years of industry experience. She worked as a Software Developer in U.K., before transiting into an Academic Career. She has published 14 articles in national and international journals and 25 papers in conferences. Her research interests are information retrieval, data mining, big data, and machine learning. Most of her work has been on improving the understanding of Natural Language documents and Music mainly through the application of data mining and Machine Learning.

LALITA BHANU MURTHY NETI received the Ph.D. degree from IIT Bombay, Mumbai, India. He worked in IT industry for more than ten years in the roles of an Enterprise Architect and the Program Manager. In June 2010, he joined the Birla Institute of Technology and Science at Pilani, Hyderabad Campus, as a faculty with the Computer Science and Information Systems Department. He is currently an Associate Professor and the Head of the Department. His current research interests are machine learning, deep learning, and empirical software engineering.