Final version of the accepted paper. Cite as: *Ashraf Kamal and Muhammad Abulaish, CAT-BiGRU: Convolution and Attention with Bi-directional Gated Recurrent Unit for Self-Deprecating Sarcasm Detection, Cognitive Computation, Topical Collection: A Decade on Sentic Computing, Springer, pp. 1-20, 2021.*

Cognitive Computation manuscript No.
(will be inserted by the editor)

CAT-BiGRU: Convolution and Attention with Bi-directional Gated Recurrent Unit for Self-Deprecating Sarcasm Detection

Ashraf Kamal[1] · Muhammad Abulaish[2] (✉), *SMIEEE*

Abstract ***Background:*** Sarcasm detection has been a well-studied problem for the computational linguistic researchers. However, research related to different categories of sarcasm has still not gained much attention. Self-Deprecating Sarcasm (SDS) is a special category of sarcasm in which users apply sarcasm over themselves, and it is extensively used in social media platforms, mainly as an advertising tool for the brand endorsement, product campaign, and digital marketing with an aim to increase the sales volume.

Methods: In this paper, we present a deep learning approach for detecting SDS on `Twitter`. We propose a novel *Convolution and Attention with Bi-directional Gated Recurrent Unit* (`CAT-BiGRU`) model, which consists of an input, embedding, convolutional, Bi-directional Gated Recurrent Unit (BiGRU), and two attention layers. The convolutional layer extracts SDS-based syntactic and semantic features from the embedding layer, BiGRU layer retrieves contextual information from the extracted features in both preceding and succeeding directions, and attention layers are used to retrieve SDS-based comprehensive context representation from the input texts. Finally, *sigmoid* function is employed to classify the input texts as a self-deprecating or non-self-deprecating sarcasm.

Results and Conclusions: Experiments are conducted over seven `Twitter` datasets to evaluate the proposed (`CAT-BiGRU`) model using standard evaluation metrics. The experimental results are impressive and significantly better than many neural network-based baselines and state-of-the-art methods. In this paper, we have highlighted biologically-inspired and psychologically-motivated basis of the proposed approach to examine its affective capabilities with respect to `SenticNet`. The efficacy of the proposed model is evaluated on two `SenticNet`-based sentic computing resources – `Amazon word embedding` and `AffectiveSpace`. Based on the experimental results, we conclude that deep learning-based approaches have potential to detect SDS in social media texts accurately.

Keywords Self-deprecating sarcasm · Deep learning · Sentic computing · CNN · BiGRU · Attention mechanism

1 Introduction

`Twitter` is a prevalent micro-blogging service and provides a platform to express views, ideas, emotions, and sentiments about the events that are happening in the real-world. A registered user on `Twitter` can post messages (*aka* tweets) up to a maximum of 280 characters. Since the beginning, `Twitter`'s user-base is increasing exponentially, and it has become a substantial fact-finding source due to the presence of the huge amount of user-generated contents. Thus, tweets have become beneficial for many purposes, such as product endorsement, e-governance, open-source intelligence, election result prediction, opinion mining, sentiment analysis, Web surveillance, and cyber-security.

✉ Muhammad Abulaish
abulaish@ieee.org

· Ashraf Kamal
ashrafkamal.mca@gmail.com

[1]Department of Computer Science, Jamia Millia Islamia (A Central University), New Delhi, India

[2]Department of Computer Science, South Asian University, New Delhi, India

Tweets are generally precise, short, and informal, and they contain non-literal expressive words, bashes, grammatically incorrect words, unstructured phrases, and slangs. However, these syntactically imprecise and informal tweets are morphologically rich, and their computational analysis is advantageous to meet the aforementioned purposes. On the other hand, the informal and non-literal contents available on `Twitter` in the form of tweets include several categories of figurative language, such as sarcasm, irony, humor, etc. and their detection is crucial for many real-life applications like opinion mining and sentiment analysis [1].

1.1 Twitter and the Sarcasm Detection Problem

Since beginning, it has been found that user-generated contents on the Web are easily understandable by humans, but difficult for the machines to process it automatically [57]. Over the years, the enormous growth of unstructured and varied data has taken the form of *big data*. It is generated at an unprecedented rate over Online Social Networks (OSNs), and the accurate distillation of knowledge from such data has become an extremely challenging task [45]. Due to these reasons, researchers are giving tremendous emphasis on community detection problem [58]. Besides that, because of the rapid increase in human-computer interaction over OSNs, research problems of various interdisciplinary sciences are shifted towards computer science as well [2]. Figurative language is one of such problem, which is derived from the field of linguistics, psychology, and cognitive sciences, and now found in OSNs on a very large-scale [3]. Moreover, sarcasm is one of the most prominent categories of figurative language that is found over OSNs, especially on `Twitter`. According to the Macmillan English dictionary[1] *sarcasm* is defined as "*the activity of saying or writing the opposite of what you mean, or of speaking in a way intended to make someone else feel stupid or show them that you are angry*". Users post sarcastic tweets through scornful, ridicule, harsh, and tease associated words or phrases. The sentiment is always linked with sarcasm where profound emotion is articulated [46]. Due to this reason, existing sentiment analysis and polarity recognition systems are highly affected due to the presence of non-literal expressions and implicit meanings in sarcastic texts [4]. Self-Deprecating Sarcasm (SDS) is a special category of sarcasm[2] in which users completely refer themselves, and execute sarcasm using deprecated, undervalued, disparaged, or criticizing words and phrases. Formally, SDS[3] is defined as a "sarcasm that plays off of an exaggerated sense of worthlessness and inferiority". Figure 1 presents an exemplar tweet representing SDS in which a user has referred and deprecated to herself.

1.2 Why Self-Deprecating Sarcasm?

Over the last few years, due to growing interest of social media marketing, various commercial tools have been developed [49], and automatically capturing the users' sentiments through marketing campaigns and product preferences have raised interest in both the scientific community and the business world. In this line, affective computing and sentiment analysis areas play a crucial role [44]. Self-deprecating is one of the marketing campaign strategies to capture users' sentiments. Self-deprecating contents are mainly used for brand endorsement and product campaigning so that the sales volume can be scaled up [6]. Such contents enhance self-promotion marketing, and they improve the product-consumer relationship and create a place among the consumers. Moreover, self-deprecating contents are structured in such a way that a brand accepts flaws without affecting its brand value. Besides OSNs, self-deprecating contents are also seen in celebrities' interviews as well as in politicians' speeches [6]. Stieger et al. [5] considered sarcasm as aggressive humor. Interestingly, self-deprecating contents are composed using sarcasm and humor to express oneself down. The main purpose behind the use of SDS is to intensify self-deprecating marketing strategies for various purposes, such as brand endorsement, product campaign, digital and content marketing, and e-advertising to excel the sales volume. Figure 2 presents an exemplar SDS-based advertisement, wherein `Converse`, an American shoe company, expresses SDS advertisement using the phrase *shoes are boring* to promote their new *sneakers* shoes.

Fig. 1: An exemplar tweet representing self-deprecating sarcasm

[1] `https://bit.ly/2WsUkUk` (last accessed on Dec. 03, 20)
[2] `https://bit.ly/34n06rx` (last accessed on Dec. 03, 20)
[3] `https://bit.ly/3qmxJF9` (last accessed on Dec. 03, 20)

Fig. 2: A self-deprecating sarcasm-based advertisement

1.3 Our Contribution

As stated earlier, SDS is a special category of sarcasm where users apply sarcasm to themselves. On analyzing the tweets of multiple datasets used in this study, we have the following observations:

- There are several tweets in which users refer themselves. Such tweets are called self-referential tweets.
- All sarcastic or non-sarcastic tweets need not be self-referential.
- A self-referential tweet can be considered as a SDS, if it is sarcastic in nature.

This paper is an extension of one of our previously published conference paper [7] by conducting experiments over six benchmark datasets, additional pattern-based regular expression queries, inclusion of a new model called Convolution and Attention with Bi-directional Gated Recurrent Unit (`CAT-BiGRU`) based on deep learning techniques, addition of a detailed analysis of the experimental findings and comparison with many neural network baselines and state-of-the-art methods. The main idea behind the `CAT-BiGRU` model is to retrieve contextual representations from the candidate self-referential tweets to detect SDS. Incongruity using contextual representation plays an important role in sarcastic texts [8]. Since, a self-referential tweet can be considered as SDS only if it has sarcastic attributes, extracting such contextual representation helps to detect SDS accurately.

Each candidate self-referential tweet is converted into a self-referential input vector and passed to the pre-trained word embedding layer. Thereafter, output generated from the embedding layer is provided to a convolution, followed by a Bi-directional Gated Recurrent Unit (BiGRU), and two attention layers. The convolutional layer extracts SDS-based syntactic and semantic features at different positions of the input embedding vector using one-dimensional convolutional filters. As a result, low-level features from the high dimensional pre-trained embedding vector are extracted. These features are semantically robust and abstract, and they also reduce the overall dimensions of the candidate self-referential tweets. The BiGRU layer extracts contextual information-based sequences from the extracted features of the convolutional layer. BiGRU represents actual semantics using the contextual information which is significantly better than a simple Gated Recurrent Unit (GRU). It consists of both forward and backward directions, where preceding and succeeding contextual information sequences are extracted from the forward and backward directions, respectively. There are two attention layers used in our proposed `CAT-BiGRU` model, which provide distinct attention in terms of the distribution of weights to the contextual information-based variable-length sequences retrieved from the forward and backward directions of BiGRU for context representation. Finally, a comprehensive context representation is obtained by concatenating the outcomes of two attention layers, and it is forwarded to a *sigmoid* activation function to classify a candidate self-referential tweet as either SDS or Non-Self-Deprecating Sarcasm (NSDS).

In short, the main contributions of this paper can be summarized as follows.

- Exploring a novel SDS detection technique for textual data (tweets) with an aim to enhance SDS-based marketing strategy.
- Implementation of a filtration technique to identify candidate self-referential tweets from the datasets. The main intent behind this filtration is to remove all those tweets from the datasets that can never represent a SDS.
- Development of a deep learning-based `CAT-BiGRU` model for detecting SDS on `Twitter`.
- Generating a new `Twitter` hashtag-based annotated dataset for SDS detection tasks.

The rest of the paper is organized as follows. Section 2 presents a brief review of the state-of-the-art computational techniques for sarcasm detection. Section 3 presents an overview of the proposed approach. It also presents a description of the filtration module to identify candidate self-referential tweets and functional aspects of our proposed `CAT-BiGRU` model. Section 4 presents the datasets, experimental settings, evaluation metrics, and experimental results. It also presents a comparative analysis of the proposed approach with many neural network-based baselines and state-of-the-art methods. Section 5 presents an important discussion to analyze the effect of the `CAT-BiGRU` model on different embedding dimensions, parameters, and sentic computing resources. Finally,

section 6 concludes the paper with future directions of research.

2 Related Work

This section presents a detailed survey of various state-of-the-art techniques based on machine learning, deep learning, and other (i.e., rule- and linguistic-based) approaches for sarcasm detection. Besides that, we have also highlighted the current status and limitations of the existing methods.

2.1 Machine Learning-Based Approaches

Sarcasm detection can be considered as a binary classification task [9]. González-Ibáñez et al. [10] considered lexical (interjections and punctuations) and pragmatics (smiley, frowning faces, etc.) factors to identify sarcasm in tweets. They considered unigram, dictionary-based lexical and pragmatic features, and applied Logistic Regression (LR), Sequential Minimal Optimization (SMO), and Support Vector Machine (SVM) techniques for sarcasm detection. They noticed that SMO provides better results in comparison to LR. Lukin and Walker [11] considered forum post from "Internet argument corpus". They applied a bootstrapping technique to identify sarcasm and nastiness, and trained high precision classifiers based on both sarcasm and non-sarcasm posts. As a result, a large labeled dataset is generated to train different classifiers. Rajadesingan et al. [12] proposed a behavior modeling-based approach and diagnosed historical tweets for sarcasm detection. They considered text expression-, emotion-, contrast-, familiarity-, and complexity-based features and applied Decision Tree (DT), SVM, and LR classifiers. Bouazizi and Ohtsuki [1] considered a pattern-based approach for tweets. They extracted sentiment-, punctuation-, syntactic-, sematic-, and pattern-based features and used Naïve Bayes (NB), SVM, and maximum entropy classifiers for sarcasm detection.

Apart from supervised techniques, semi-supervised-based techniques are also considered in few studies. Tsur et al. [13] considered a semi-supervised technique for sarcasm identification. They considered three feature sets involving syntactic, patterns, and punctuations. They identified a large set of patterns from frequent words available on `Amazon` dataset. Davidov et al. [14] followed the same approach as [13] for analysing `Twitter` and `Amazon` product reviews. Recently, we proposed the first computational study on SDS detection task in [15], and highlighted different categories of sarcasm. In [15], we applied a rule-based approach to detect candidate self-around tweets, and identified various self-deprecating and hyperbolic features. Finally, we applied DT, NB, and bagging classifiers for detecting SDS.

2.2 Deep Learning-Based Approaches

In the last few years, deep learning-based approaches are broadly applied in numerous Natural Language Processing (NLP) problems [48], including the sarcasm detection task. Schifanella et al. [16] proposed sarcasm detection task in multimodal platforms, including `Twitter`, `Instagram`, and `Tumblr` on visual and textual components. They applied deep Convolutional Neural Network (CNN) and SVM, and considered lexical, subjectivity, n-grams, and visual-semantics features. Amir et al. [17] considered CNN to learn user- and utterance-based embeddings. They extracted contextual features by user embedding learning for sarcasm detection. In addition, they also highlighted content embedding learning using lexical representation in the convolutional layer. Zhang et al. [9] applied a bi-directional gated Recurrent Neural Network (RNN) for sarcasm detection. They used syntactic and semantic information to obtain contextual features in historical tweets via a pooling neural network. Poria et al. [18] considered CNN and SVM, and applied features, such as sentiment, emotion, and personality on balanced and unbalanced datasets. Avvaru et al. [19] considered a transformer-based model for sarcasm detection in conversation sentences over `Twitter` and `Reddit` datasets. Authors highlighted that consideration of larger corpus increases context and perform better in terms of accuracy. Dubey et al. [20] converted the sarcastic expressions into their literal expressions. Apart from the rule-, and statistical machine learning translation-based approaches, they considered deep learning-based techniques, such as encoder decoder-, attention-, and pointer generator-networks. Hazarika et al. [51] proposed `CASCADE`, a hybrid approach containing both content- and context-driven modeling to detect sarcasm on discussions post available on social media. Dubey et al. [21] detected sarcasm in the numerical portion of the texts using CNN and applied attention network-based deep learning models. Interestingly, they emphasized that sarcasm can also be involved in numbers.

2.3 Other Approaches

Besides machine learning and deep learning-based approaches, rule-based and linguistic-based approaches are also used for sarcasm detection. Riloff et al. [22] proposed to identify tweets with "positive sentiment words contradicting with negative situation phrases" and considered them as sarcasm. Khattri et al. [23] considered user historical tweets and proposed a "contrast-based predictor" which reported the sentiment contradictions within the target tweets. Further, Bharti et al. [24] proposed two algorithms namely, "parsing-based lexical generation algorithm" and "interjection word start". They considered lexical and hyperbolic (e.g., intensifier) features as an indicator of sarcasm. Liebrecht et al. [25] applied a linguistic approach based on balanced Winnow [26] technique for sarcasm detection. Likewise, Mishra et al. [27] considered lexical- and contextual-based features. They considered the gaze behavior of the readers to understand sarcasm and highlighted cognition cognizant techniques involving eye-tracking as a promising approach for sarcasm detection. Justo et al. [36] proposed `SOFOCO` "*Spanish Online Forums Corpus*", wherein authors extracted dialogic debates from online sources, and further annotated by crowdsourcing platform to perform automatic analysis of sarcasm and nastiness. Mehta et al. [47] discussed that personality trait can be used as an input for sarcasm detection task.

2.4 Current Status and Limitations

All of the aforementioned approaches confirm the richness and potential of the data available on OSNs, especially in the form of tweets for effective sarcasm detection. All approaches discussed above have considered only sarcasm detection, and research on detecting different categories of sarcasm has still not received much attention. Considering the fact that sarcasm occurs in different forms, and SDS is an important sarcasm category, development of SDS detection techniques deserves greater attention, because it is useful for numerous brand endorsement and product campaign purposes to boost and excel the sales volume [2]. Hence, the proposed SDS detection is a significant, non-trivial, and worth investigation task.

3 Proposed Approach

In this section, we present the functional details of our proposed `CAT-BiGRU` model for detecting SDS. Figure 3 presents a visualization of the work-flow of our proposed approach. Starting with a detailed description of the *data crawling*, *ethical aspects*, *data pre-processing*, and *self-referential tweets identification* modules, the functionality of `CAT-BiGRU` model is presented in the following sub-sections.

3.1 Data Crawling

In this study, we have considered a total of seven datasets, including six benchmark datasets. Since authors of the benchmark datasets are allowed to provide only tweet ids due to the `Twitter` policy, we have developed a data crawler using `Python 2.7` to curate tweets using the `Twitter REST` API and store them in a local repository. Since some of the tweets have been deleted or not available due to the protection criteria set by the `Twitter` users, our crawler was unable to curate such protected or deleted tweets at the time of crawling. In addition to six benchmark datasets, we have created a new dataset, namely `Twitter`-280 containing both sarcasm and non-sarcasm tweets through crawler from 1st June 2019 to 31st July 2019 using the hashtag-based annotation technique. The sarcasm tweets are collected using #sarcasm hashtag, whereas non-sarcasm tweets are collected using the #not, #education, #politics, #love, and #hate hashtags. Statistical details of all datasets are presented in sub-section 4.1. The newly created `Twitter`-280 dataset is publicly accessible, but as per the `Twitter` rules and guidelines and in light of the ethical aspects, we are restricted to provide only tweet ids for both sarcasm and non-sarcasm categories. The dataset and source code of the proposed approach is publicly accessible on the `GitHub`.[4]

3.2 Ethical Aspects

In OSNs, consideration of ethical aspects and following proper guidelines for data redistribution of online platforms have become a crucial task. In this work, we tried our best to make sure about the privacy and protection of accumulated tweets. The proposed work was carried out for academic research purposes to investigate the effectiveness and detection of SDS on tweets using both 140 and 280 character limit criteria set by `Twitter`, and we have crawled tweets as per `Twitter`'s rules and guidelines, accordingly.

Elovici et al. [37] presented several ethical aspects, which are taken into consideration during and after the

[4] `https://github.com/Ashraf-Kamal/Self-Deprecating-Sarcasm-Detection`

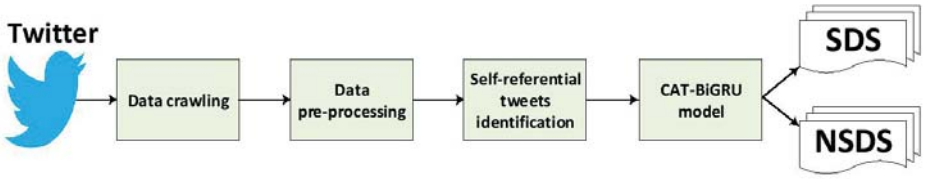

Fig. 3: Work-flow of the proposed `CAT-BiGRU` model

data crawling tasks. The experiment was performed after receiving clearance from the research ethics committee of the department, including my doctoral supervisor. We ensured that crawled tweets are not shared or will not be shared in the future with any organization or third party/person. During the experiment of this paper, we tried our level best to not violate `Twitter`'s terms of service[5] and privacy policy.[6] Further, as per `Twitter`'s content redistribution policy,[7] academic researchers are allowed to share an unlimited number of tweet ids only for peer-review or validation of research works. Considering this information, we will only distribute tweet ids of our dataset using the `GitHub` repository link given in sub-section 3.1.

3.3 Data Pre-processing

Since raw tweets contain various noise and unwanted information like numbers, punctuations, acronyms, etc., elimination of such undesirable information is important for better accuracy and efficiency of the system. To this end, we have applied data cleaning steps, such as removal of symbols, punctuation marks, URL's, retweets, mentions, ampersands, dots, white spaces, double quotes, emoticons, and numbers. Thereafter, we converted all tweets into lower-case letters and removed stop-words to reduce their length and retain only the significant information. While filtering stop-words, we have retained all *self-referential*-specific stop-words like *i*, *my*, *me*, *mine*, *myself*, *we*, *us*, *our*, and *are* in the tweets.

After data cleaning, we applied tokenization and Parts-of-Speech (POS) tagging over each tweet using the `spaCy`[8] tagger, wherein POS tagging is based on the `penn tree bank` English POS tagset.[9] Finally, all tweets containing less than three tokens/words were removed from the dataset, because for defining context of a word, say w, we need at least one word left and one word right to w.

3.4 Self-Referential Tweets Identification

On analyzing both sarcasm and non-sarcasm related tweets of the aforementioned datasets, we found that all tweets are not self-referential. Moreover, we found that if a tweet is not self-referntial then it can never be a SDS. Therefore, in line to the work of Zhao et al. [28], we applied a filtering mechanism to consider only self-referential tweets of the datasets for further processing. In brief, the main steps of this module can be summarized as follows:

(i) *Identification of explicit self-referential tweets:* This step aims to identify self-referential tweets based on the presence of some explicit patterns in the tweets. Table 1 presents a list of regular expressions-based patterns that are applied to identify self-referential tweets. These patterns are categorized as – *specific patterns* and *generic patterns*.

– *Specific patterns*: Specific patterns are mainly based on either tokens or the sequential order of the tags and tokens, and vice-versa. In some of the patterns, tokens like *love* and *still* are fixed, because that frequently occur in self-referential tweets [38]. Moreover, these tokens are frequently occur in sarcastic tweets as well [3,22,24]. Similarly, tokens based on interjections, such as `oh`, `wow`, or `yeah` are also found as strong indicators. Effron [39] mentioned interjection as an overtly self-referential, and it is explicitly present in most of the self-referential instances. Interjection is an important linguistic marker in sarcastic tweets as well [3,6,24,41]. If a pre-processed tweet matches with any one of the *specific patterns* that are given

[5] `https://twitter.com/en/tos` (last accessed on Dec. 03, 20)

[6] `https://twitter.com/en/privacy` (last accessed on Dec. 03, 20)

[7] `https://developer.twitter.com/en/developer-terms/agreement-and-policy` (last accessed on Dec. 03, 20)

[8] `https://spacy.io/` (last accessed on Dec. 03, 20)

[9] `https://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html` (last accessed on Dec. 03, 20)

in table 1, then it is added into the list of explicit self-referential tweets, Exp_s.

– *Generic patterns*: Generic patterns are mainly first person singular/plural personal pronoun based tokens like '*i*' and '*we*'. Besides these, tokens based on their other grammatical variants, such as *my*', '*me*', '*mine*', '*myself*', '*are*', '*our*', '*us*', and '*ourselves*' are also referred as *generic patterns*. The personal pronouns based tokens are strong indicator to categorize a tweet as a self-referential tweet [40]. If a pre-processed tweet matches with any one of the *generic patterns* that are given in table 1, then it is added into the list of explicit self-referential tweets, Exp_s.

Table 1: Patterns to identify explicit self-referential tweets

Patterns	Category
UH $(i \mid my \mid we \mid are)$	Specific
$(we \mid i)$ $love$ $(it \mid when)$	Specific
$when$ $(my \mid our)$	Specific
$(am \mid are)$ $still$	Specific
$(myself \mid ourself)$ $(JJ \mid RB)$	Specific
$(oh \mid wow \mid yeah)$ $(i \mid we)$ $(real^*y \mid great^*)$	Specific
$(i \mid we)$ MD RB	Specific
$(i \mid am \mid my \mid me \mid mine \mid myself)$	Generic
$(we \mid are \mid us \mid our \mid ourselves)$	Generic

(ii) *Identification of clusters from explicit self-referential tweets:*
This step clusters all tweets of Exp_s in the form of connected components. Initially, Exp_s tweets are modeled as an undirected graph, wherein each explicit tweet represents a node, and similarity values between each pair of nodes are used to create edges. The similarity between two nodes (explicit self-referential tweets), say t_i and t_j, is calculated using the `Jaccard coefficient`, which is defined in equation 1. In this equation, T_i and T_j represent the set of tri-grams of the tweets t_i and t_j, respectively. In case of tweets, unigrams, bigrams, and trigrams are the most adopted n-grams [42,43]. We have taken tri-grams with sliding window of size 1 instead of larger n-grams (4-grams or 5-grams) in our experiment. An edge between a pair of self-referential tweets (nodes) is created only if the Jaccard similarity between their tri-grams is greater than a threshold of 0.6, as given in [28]. Finally, all explicit self-referential tweets of a connected component form a cluster.

$$J(t_i, t_j) = \frac{|T_i \cap T_j|}{|T_i \cup T_j|} \quad (1)$$

(iii) *Pattern-mining from clusters:*
After identifying the clusters of the explicit self-referential tweets, frequent patterns (tri-grams) from the clusters are mined in this step. To this end, the occurrence probability of all patterns for each cluster is calculated, and the patterns having the probability value greater than 0.5 are considered as frequent patterns. For example, if a cluster consists of 20 tweets and a tri-gram pattern "love being ignored" is available in 10 out of 20 tweets, then such pattern is considered as a frequent tri-gram pattern. All identified frequent patterns are used to create a list of frequent patterns, F_P, and a list of unique frequent patterns, P, is created by removing the duplicate patterns.

(iv) *Identification of implicit self-referential tweets:*
This steps considers all those tweets that do not have any match in step (i) mentioned above, and termed as implicit tweets. The main purpose of this step is to recover from the low *recall* value. The identified patterns in the previous step are used to identify implicit self-referential. To this end, each implicit tweet is tokenized into tri-grams and matched with the set of frequent patterns, P. If an implicit tweet contains any frequent pattern of P, then it is considered as an implicit self-referential tweet, and added to the list of implicit self-referential tweets, Imp_s. Table 2 presents a partial list of three implicit self-referential tweets.

Table 2: Few sampler self-referential tweets identified using frequent patterns

S. No.	Implicit self-referential tweet
1	failed physics exams great.
2	absolutely love being left hang.
3	wow nothing like bit happiness.

(v) *Merging with explicit self-referential tweets:*
In this step, both sets of explicit self-referential tweets and implicit self-referential tweets are merged together to generate a list, S_t, of candidate self-referential tweets, i.e., $S_t = Exp_s \cup Imp_s$. In the remaining part of this paper, the list (S_t) is used as an input for SDS detection.

3.5 CAT-BiGRU Model

This section presents a new `CAT-BiGRU` model used in our proposed approach for detecting SDS. The sequence of words in a candidate self-referential tweet represents

important characteristics to determine whether the tweet represents a SDS or not. As stated in section 1.3, contextual representation contributes significantly to self-deprecating sarcastic texts. In recent years, various neural network models, like RNN, have shown better performance in many NLP applications, and obtained remarkable outcomes with less number of features. The architecture of RNN is sequential, and it can process arbitrary length sequences, mainly to perform sequence modeling tasks. GRU belongs to the RNN family, and it overcomes the complicated word modeling task associated with unstructured texts. Although, GRU extracts contextual information from the texts, it does not retrieve important information from the identified contextual data.

`CAT-BiGRU` model aims to improve the aforementioned drawbacks by integrating a convolutional, BiGRU, and two attention layers together. Figure 4 presents the architecture of our proposed `CAT-BiGRU` model. Motivated by Liu and Guo [29] architecture, the convolutional layer is used to extract SDS-based syntactic and semantic features from the candidate self-referential tweets, enabling BiGRU to extract contextual information as sequences from the features extracted by the convolutional layer in both forward and backward hidden layers. Two attention layers are applied to extract SDS-based context representation using the weights of the important words. These representations are retrieved from the preceding and succeeding contextual information sequences of BiGRU. Further, the contextual information retrieved from the attention layers is concatenated for a comprehensive context representation. Finally, it is forwarded to the *sigmoid* activation function to classify a candidate self-referential tweet as either SDS or NSDS. In brief, the overall functioning of the `CAT-BiGRU` model can be summarized as follows:

Input layer:
Each candidate self-referential tweet is tokenized, converted into sequences, and replaced with its dictionary index value, i.e., $S \epsilon R^{1 \times N}$, where N represents words (tokens) count in the entire training dataset containing candidate self-referential tweets. To make each candidate self-referential tweet of the same length, a fixed value of padding is used, i.e., $S \epsilon R^{1 \times K}$, where K is the maximum length of a candidate self-referential tweet, and it is same for all candidate self-referential tweets in the dataset. In this study, the value of K is set as 20. Thereafter, it is transformed into a matrix form, where each row represents a self-referential tweet vector, and passed to the word embedding layer.

Embedding layer:
Embedding layer works as a hidden layer in neural network architectures. It shows distributed representations of the words as low-dimensional real-valued dense vectors learned from a large corpus in a continuous embedding space. In word embedding, words that are semantically related to each other have similar vector representation. Furthermore, semantic and syntax relations of words depend on the context factor, and it is useful for many NLP applications involving text classification, machine translation, and speech recognition. In this paper, Global Vectors (`GloVe`), a pre-trained word embedding of 200-dimensions based on `Twitter`-specific data that consists of 27 billion tokens is used. It is one of the popular pre-trained word embedding, which directly takes the global statistics of the large corpus. It considers co-occurrence matrix dataset for training, where word pairs based on *target* and *context* are taken to encode the semantic information. In this paper, the self-referential input vector generated from the input layer is feed to the pre-trained `GloVe` word embedding layer, which converts each token into a distributional vector of dimension D. As a result, the input self-referential matrix is converted into $S \epsilon R^{K \times D}$.

Convolutional layer:
The convolutional layer is employed for dimension reduction task, and it also captures the sequence information from the input embedding vector. In `CAT-BiGRU`, a one-dimensional convolutional operation takes place in the convolutional layer. We have considered a total of 256 filters and a window size of 3, which moves on the embedding vector for extracting features. As a result, various sequences are generated that grasp the SDS-based syntactic and semantic features. Equation 2 presents an n^{th} feature sequence f_n, which is generated from a window of words x_t, where W_t and B represent the filter weight and bias term, respectively; and r$(\cdot)$ represents the non-linear activation function as *rectified linear unit* (*aka* ReLU). All 256 filters perform the convolutional operation from top to bottom on an input candidate self-referential tweet. Finally, the feature sequence is obtained as $L_f = [f_1, f_2, ..., f_{256}]$.

$$f_n = r(W_t \cdot x_t + B) \tag{2}$$

BiGRU layer:
BiGRU seems the right fit for sequential modeling tasks, and it represents word-vector representation. It works in both forward and backward directions,

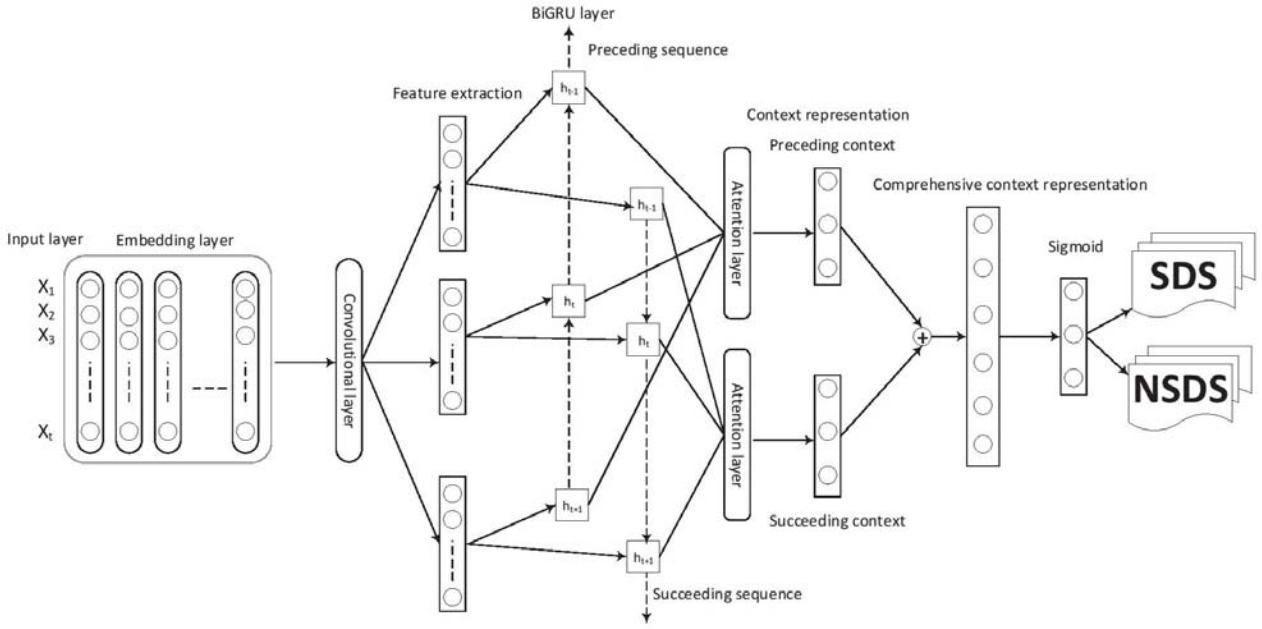

Fig. 4: Architecture of the proposed CAT-BiGRU model

and obtains contextual information-based sequences from the features generated by the convolutional layer. It contains a forward GRU ($\overrightarrow{GRU}$) which represents the succeeding feature sequences (i.e., f_1 to f_{256}) and a backward GRU ($\overleftarrow{GRU}$) which represents the preceding feature sequences (i.e., f_{256} to f_1). Formally, equations 3 and 4 present BiGRU outputs in forward and backward directions, respectively. It obtains annotations for SDS-based words (tokens) by summarizing both forward and backward directions. These annotated words (tokens) contain contextual information in a candidate self-referential tweet. Annotation for a given feature sequence L_f of a candidate self-referential tweet S_t is obtained by the $\overrightarrow{g_f}$ and $\overleftarrow{g_b}$ for the forward and backward hidden states, respectively. Both states compile the information which is collected around L_{f_n} to retrieve contextual information-based sequences related to the SDS.

$$\overrightarrow{g_f} = \overrightarrow{GRU}(L_{f_n}), n \in [1, 256] \tag{3}$$

$$\overleftarrow{g_b} = \overleftarrow{GRU}(L_{f_n}), n \in [256, 1] \tag{4}$$

Attention layer:
In neural networks, an attention mechanism highlights the important keywords and minimizes the effect of non-keywords by specifying distinct weights to each word of a text. In this paper, two attention layers are used to allow different weights for the words in a candidate self-referential tweet to strengthen the understanding of the SDS-based words/tokens. Attention mechanism focuses on self-deprecating sarcastic keyword-based features and minimizes the effect of non-keywords in a candidate self-referential tweet. The word annotation $\overrightarrow{g_f}$ is first provided to retrieve a hidden representation $\overrightarrow{\alpha_f}$ by using one layer perceptron. It is formally presented in equation 5, where w and b represent weight and bias, respectively, and tanh is the hyperbolic tangent function.

$$\overrightarrow{\alpha_f} = \tanh(w\overrightarrow{g_f} + b) \tag{5}$$

The CAT-BiGRU model highlights the importance of each word, and it is done by calculating the similarity between $\overrightarrow{\alpha_f}$ and $\overrightarrow{\beta_f}$, where $\overrightarrow{\beta_f}$ represents word-level context vector, which is initialized randomly and fully learned at the time of training the CAT-BiGRU model. It is considered as a high-level representation of the self-deprecating sarcastic words from the input candidate self-referential tweets. Further, it obtains a normalized weight $\overrightarrow{z_f}$ for each word using the *softmax* activation function, as given in equation 6, where $*$ and exp(.) represent multiplication and exponential function, respectively.

$$\overrightarrow{z_f} = \frac{\exp(\overrightarrow{\alpha_f} * \overrightarrow{\beta_f})}{\sum_{i=1}^{N}(\exp(\overrightarrow{\alpha_f} * \overrightarrow{\beta_f}))} \tag{6}$$

Thereafter, equation 7 presents the forward context representation F_c, and it is computed using $\overrightarrow{g_f}$ and $\overrightarrow{z_f}$. Similarly, equation 8 presents the backward context representation B_c, and it is computed using

weighted sum of the word annotation $\overleftarrow{g_b}$ and the normalized weight $\overleftarrow{z_b}$ in the backward direction. Hence, annotation of a particular feature sequence L_f is determined by concatenating the forward and backward context representations F_c and B_c, respectively. A comprehensive context representation $S_c = [F_c, B_c]$ is obtained by concatenating F_c and B_c which represents a set of comprehensive features. Finally, it is feed to the *sigmoid* activation function, which is a two-class logistic regression function, to classify a candidate self-referential tweet as either SDS or NSDS.

$$F_c = \sum(\overrightarrow{z_f} * \overrightarrow{g_f}) \tag{7}$$

$$B_c = \sum(\overleftarrow{z_b} * \overleftarrow{g_b}) \tag{8}$$

In our proposed `CAT-BiGRU` model, drop out is used to reduce over-fitting and improve generalization error by dropping a random sample of neurons during the training process. A binary cross-entropy loss function is used for classifier training which interprets *self-referential* tweets label as SDS or NSDS. Further, all datasets are divided into a training set and a testing set, where 80% data is used for training and 20% is used for testing. The batch size and verbose values are taken as 256 and 2, respectively. We have considered *Adam* optimization algorithm and a total of 100 epochs to train the model which classifies candidate self-referential tweets as either SDS or NSDS.

4 Experimental Setup and Results

This section presents the experimental details of our proposed approach. It includes the description of the datasets, experimental settings, evaluation metrics and results, and comparative analysis with neural network-based baselines and state-of-the-art methods, as discussed in the following sub-sections.

4.1 Datasets

The proposed approach is evaluated over seven datasets, including six benchmark datasets including `Twitter` data. All these datasets are based on the old 140 characters limit. Besides these, we have created a new `Twitter` dataset, namely `Twitter`-280, which is based on the new 280 characters limit. Table 3 presents a brief statistics of the datasets.

Out of the seven datasets given in table 3, tweets of three datasets viz. Ptácek et al. [30], SemEval 2015 [4], and Riloff et al. [22] are manually annotated as either SDS or NSDS. These datasets are directly passed to the `CAT-BiGRU` model for SDS detection.

Table 3: Statistics of the datasets

Datasets	#Sarcasm	#Non-sarcasm	Total (#tweets)
Ptácek et al. [30]	53088	98195	151283
SemEval 2015 [4]	1526	2366	3892
Riloff et al. [22]	370	1431	1801
Bamman and Smith [32]	7702	7358	15060
Ling and Klinger [31]	26776	25800	52576
Ghosh and Veale [33]	19452	22251	41703
`Twitter`-280	17488	25134	42622

Table 4: Class-distributions of the manually annotated datasets

Datasets	#SDS	#NSDS	Total (#tweets)
Ptácek et al. [30]	10793	11207	22000
SemEval 2015 [4]	1189	1310	2499
Riloff et al. [22]	483	417	900

Class-distributions of these three manually annotated datasets are given in table 4.

Another three benchmark datasets viz. Ling and Klinger [31], Bamman and Smith [32], and Ghosh and Veale [33], and the newly created `Twitter`-280 dataset of table 3 are used for the identification of self-referential tweets, as discussed in sub-section 3.4. Table 5 presents the statistics of the datasets after filtering out non-self-referential tweets using the self-referential tweets identification module.

Table 5: Statistics of the datasets after filtering out non-self-referential tweets

Datasets	#Sarcasm	#Non-sarcasm	Total (#tweets)
Bamman and Smith [32]	3366	3548	6914
Ling and Klinger [31]	22591	18864	41455
Ghosh and Veale [33]	12767	13991	26758
`Twitter`-280	14492	19389	33881

4.2 Experimental Settings

In this paper, *data crawling*, *data pre-processing*, and *self-referential tweets identification* modules are implemented in `Python 2.7`. The `CAT-BiGRU` model for SDS detection task is implemented in `Python 3.5` and executed using a neutral network API, `Keras`,[10] which is a high-level neural network library in `Python`, and it is mainly used for experimental evaluation of

[10] `https://keras.io/`(last accessed on Dec. 03, 20)

the deep learning models. Table 6 presents the values of different hyperparameters used to implement the `CAT-BiGRU` model.

Table 6: Hyperparameters and their values used to implement the `CAT-BiGRU` model

Hyperparameter	Value
Embedding dimension	200
Padding sequences	20
Number of filters	256
Filter width	3
Dropout	0.4
Number of neurons (GRU)	256

4.3 Evaluation Metrics

The proposed approach is evaluated using four standard evaluation metrics – *precision*, *recall*, *f-score*, and *accuracy*. These metrics are defined formally in equations 9, 10, 11, and 12 in terms of *True positive* (TP), *False Positive* (FP), *True Negative* (TN), and *False Negative* (FN). *TP* is defined as the number of correctly identified SDS tweets. *FP* is defined as the number of incorrectly identified SDS tweets. *TN* is defined as the number of correctly identified NSDS tweets. Finally, *FN* is defined as the number of incorrectly identified NSDS tweets.

$$Precision\,(P) = \frac{TP}{TP + FP} \tag{9}$$

$$Recall\,(R) = \frac{TP}{TP + FN} \tag{10}$$

$$F\text{-}score = \frac{2 \times P \times R}{P + R} \tag{11}$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{12}$$

4.4 Evaluation Results and Comparative Analysis

This section presents the evaluation results of our proposed approach and comparison with neural network-based baselines and state-of-the-art methods over all datasets in terms of *precision*, *recall*, *f-score*, and *accuracy*. Table 7 presents the evaluation results, where the bold entries highlight the best results across manually annotated and hashtag labeled datasets. This section also presents a comparative analysis over all datasets in terms of training and validation accuracy values, as given in figure 5.

4.4.1 Comparison with Neural Network-Based Baselines

Starting with a brief description of the neural network-based baseline methods and their different combinations, this section presents a comparison of our proposed approach with the baseline methods.

- CNN: Kim [34] introduced CNN. It is used as a baseline for comparison with our proposed model. In our experiment, the filter width and number of filters are set as 3 and 100, respectively.
- LSTM: Long-Short Term Memory (LSTM) [35] is a kind of RNN which does not suffer with the vanishing gradient problem. It consists of three digital gates (input, output, and forget) and a cell memory state. In our experiment, a total of 256 neurons are considered.
- BiLSTM: Likewise BiGRU, Bi-directional Long-Short Term Memory (BiLSTM) consists of a forward LSTM and a backward LSTM. In our experiment, a total of 256 neurons are considered.
- CNN-LSTM: It is a combination of the CNN and LSTM model. In our experiment for CNN-LSTM combination, filter width, number of filters, and number of neurons are set as 3, 256, and 256, respectively.
- CNN-BiLSTM: It is a combination of the CNN and BiLSTM model. In our experiment for CNN-BiLSTM combination, filter width, number of filters, and number of neurons are set as 3, 256, and 256, respectively.

The results presented in table 7 show that our proposed approach using the `CAT-BiGRU` model outperforms the neural network-based baseline methods over all datasets. Overall, SemEval 2015 [4] and Riloff et al. [22] achieved better results in terms of all evaluation metrics over all datasets. SemEval 2015 et al. [22] achieved highest *precision* and *accuracy* values, whereas Riloff et al. [22] achieved highest *recall* and *f-score* values. On the other hand, hashtags labeled datasets also perform better in terms of all evaluation metrics for the proposed approach. Bamman and Smith [32] performs better in terms of *precision* and *accuracy* values, Ling and Klinger [31] achieved highest *recall* value, and highest *f-score* value is same for Bamman and Smith [32] and Ling and Klinger [31] over hashtag labeled datasets. The newly created `Twitter-280` dataset also receives better results for the proposed approach in terms of the aforementioned evaluation metrics.

It can be observed that among baseline methods, CNN achieved the highest *precision*, *f-score*, and

Table 7: Performance evaluation results over Ptácek et al. [30] (DS-1), SemEval 2015 [4] (DS-2), Riloff et al. [22] (DS-3), Bamman and Smith [32] (DS-4), Ghosh and Veale [33] (DS-5), Ling and Klinger [31] (DS-6), and `Twitter`-280 (DS-7) datasets in terms of *precision*, *recall*, *f-score*, and *accuracy*. The bold entries in the table highlight the best results across the manually-annotated and hashtag labeled datasets

Evaluation metrics ↓	Datasets →	Manually annotated			Hashtag labeled			
		DS-1	DS-2	DS-3	DS-4	DS-5	DS-6	DS-7
Precision	Proposed approach	**0.90**	**0.92**	**0.91**	**0.86**	**0.84**	**0.84**	**0.77**
	Abulaish and Kamal [15]	0.67	0.72	0.66	0.54	0.52	0.76	0.57
	Ghosh and Veale [33]	0.73	0.62	0.53	0.51	0.61	0.73	0.61
	CNN	0.80	0.80	0.77	0.73	0.76	0.81	0.75
	LSTM	0.75	0.70	0.58	0.60	0.62	0.72	0.57
	BiLSTM	0.75	0.69	0.56	0.60	0.61	0.72	0.57
	CNN-LSTM	0.73	0.67	0.53	0.60	0.57	0.70	0.33
	CNN-BiLSTM	0.72	0.66	0.54	0.52	0.56	0.69	0.7
Recall	Proposed approach	**0.89**	**0.93**	**0.98**	**0.85**	**0.83**	**0.87**	**0.75**
	Abulaish and Kamal [15]	0.36	0.42	0.48	0.37	0.34	0.48	0.16
	Ghosh and Veale [33]	0.72	0.33	0.97	0.24	0.60	0.74	0.65
	CNN	0.81	0.79	0.87	0.69	0.71	0.85	0.62
	LSTM	0.77	0.68	0.95	0.54	0.52	0.76	0.33
	BiLSTM	0.76	0.71	0.97	0.51	0.54	0.75	0.29
	CNN-LSTM	0.72	0.45	0.96	0.35	0.33	0.73	0.37
	CNN-BiLSTM	0.72	0.35	0.95	0.22	0.28	0.73	0.72
F-score	Proposed approach	**0.89**	**0.92**	**0.94**	**0.85**	**0.84**	**0.85**	**0.76**
	Abulaish and Kamal [15]	0.46	0.53	0.55	0.43	0.41	0.58	0.25
	Ghosh and Veale [33]	0.72	0.43	0.68	0.32	0.60	0.74	0.63
	CNN	0.80	0.79	0.81	0.70	0.73	0.83	0.68
	LSTM	0.75	0.68	0.72	0.57	0.56	0.74	0.41
	BiLSTM	0.75	0.69	0.71	0.55	0.57	0.73	0.38
	CNN-LSTM	0.72	0.53	0.68	0.40	0.41	0.71	0.35
	CNN-BiLSTM	0.72	0.45	0.69	0.30	0.37	0.70	0.71
Accuracy	Proposed approach	**0.90**	**0.93**	**0.92**	**0.86**	**0.84**	**0.84**	**0.80**
	Abulaish and Kamal [15]	0.57	0.61	0.60	0.53	0.54	0.63	0.57
	Ghosh and Veale [33]	0.73	0.58	0.53	0.51	0.53	0.71	0.57
	CNN	0.80	0.82	0.80	0.72	0.75	0.81	0.75
	LSTM	0.75	0.71	0.62	0.61	0.61	0.71	0.61
	BiLSTM	0.76	0.71	0.57	0.61	0.61	0.70	0.61
	CNN-LSTM	0.73	0.62	0.54	0.56	0.56	0.67	0.57
	CNN-BiLSTM	0.72	0.60	0.53	0.52	0.55	0.67	0.57

accuracy values, except in Riloff et al. [22], wherein BiLSTM achieved highest *recall* value. Similarly, for hashtag labeled datasets, CNN reports the highest *precision*, *recall*, *f-score*, and *accuracy* values. However, the proposed approach performs 15.00% better in terms of *precision*, 16.04% better in terms of *f-score*, and 13.41% better in terms of *accuracy* values in comparison to CNN, and 1.03% better in terms of *recall* value in comparison to BiLSTM for manually annotated datasets. Similarly, the proposed approach performs 17.80% better in terms of *precision*, 2.35% better in terms of *recall*, 2.40% better in terms of *f-score*, and 19.44% better in terms of *accuracy* values in comparison to CNN for hashtag labeled datasets.

Figure 5 presents a visualization-based comparative analysis in terms of training and validation accuracy values for all datasets. It can be observed from this figure that the proposed approach outperforms neural network-based baseline methods. Overall, CNN reports significantly better results among all neural network-based baseline methods in terms of training and validation accuracy values. Some interesting observations can be inferred from the aforementioned results that the manually annotated datasets perform better in comparison to the hashtags labeled datasets for our proposed approach and all baseline methods, because such datasets are more fine-grained in comparison to the hashtags labeled datasets in which tweets have naturally annotated labels by the registered users on `Twitter`. CNN performs significantly better in comparison to other baseline methods because it extracts local contextual features from the input dataset, helping to generate global feature vectors that can be useful for the classification task. In addition,

the newly created Twitter-280 dataset contains tweets of up to 280 characters. The increase in tweet-length also increases overall explicit context and incongruity, and it affects the performance of the applied methods in comparison to other datasets that contain tweets of maximum 140 characters.

4.4.2 Comparison with State-of-the-Art Methods

This section presents a comparative analysis of our proposed approach with the following state-of-the-art methods.

- Abulaish and Kamal [15]: In this paper, a rule-based and machine learning techniques are applied for detecting SDS over Ptácek et al. [30] dataset. They reported it as the first work towards automatic detection of SDS.
- Ghosh and Veale [33]: In this paper, authors proposed a neural network model for document-level sarcasm detection. They considered a stacking approach which consists of CNN, LSTM, and DNN layers. Their work has outperformed various neural and non-neural baselines.

Table 7 presents the comparison results of our proposed method with the aforementioned state-of-the-art methods. It can be observed from this table that our proposed approach using the CAT-BiGRU model outperforms both state-of-the-art methods. Similar to the neural network-based baseline methods, previous works also perform better on manually annotated datasets in comparison to the hashtag-labeled datasets. Our proposed approach performs 27.77% better in terms of *precision*, 1.03% better in terms of *recall*, 38.23% better in terms of *f-score*, and 52.45% better in terms of *accuracy* values in comparison to the state-of-the-art methods over manually annotated datasets. Likewise, our proposed approach performs 59.25% better in terms of *precision*, 17.56% better in terms of *recall*, 14.86% better in terms of *f-score*, and 62.26% better in terms of *accuracy* values in comparison to the state-of-the-art methods over hashtag-labeled datasets. Figure 5 presents a visualization-based comparative analysis in terms of training and validation accuracy values over all datasets. It can be observed from this figure that the proposed approach outperforms state-of-the-art methods, and again manually annotated datasets show better performance in comparison to the hashtag-labeled datasets. Based on the aforementioned results, it can be inferred that inclusion of two attention layers in our proposed model which function in both preceding and succeeding directions provides better contextual representations in comparison to the Ghosh and Veale [33] method, wherein stacking approach is adopted without any attention layer mechanism. Ghosh and Veale [33] method functions in one direction only and lacks contextual representations for sarcasm detection in both directions.

5 Discussion

This section presents an analysis to show the effects of different GloVe embedding dimensions, GRU parameters, and sentic computing resources on CAT-BiGRU model over all aforementioned datasets.

5.1 Effect of GloVe Embedding Dimensions

Choosing the right embedding dimension is a challenging task. Embedding dimension refers to the total number of features that it encodes. Lower dimensions provide fewer features and lower accuracy values, whereas higher dimensions provide large number of features and higher accuracy, but a chance of over-fitting. If the corpus is not large and training time is not a constraint, then a higher dimension is a good choice. GloVe provides different pre-trained word vector embedding dimensions, such as 25, 50, 100, and 200, especially for the Twitter corpus. Although, considering the above facts, we have trained our CAT-BiGRU model on 200-dimension Twitter specific GloVe word embedding, here we analyze its performance by varying the number of dimensions as 25, 50, and 100. Figures 6 and 7 present the classification results of CAT-BiGRU model for different GloVe embedding dimensions – 25, 50, 100, and 200 in terms of *f-score* and *accuracy* values, respectively. It can be observed that the CAT-BiGRU model performs better on GloVe 200 dimensions in comparison to 25, 50, and 100 dimensions across all datasets.

Overall, manually annotated datasets provides better results in comparison to the hashtag-labeled datasets. Riloff et al. [22] and SemEval 2015 [4] provide highest *f-score* and *accuracy* values, respectively among both (manually annotated and hashtag-labeled) datasets. Bamman and Smith [32] and Ling and Klinger [31] provides highest *f-score* value for hashtag-labeled datasets. However, Bamman and Smith [32] also provides the highest accuracy value for hashtag-labeled datasets. Moreover, it can also be observed that the performance over Twitter-280 dataset is low in comparison to other datasets in terms of *f-score* and *accuracy* values. Based on these results, it can be inferred that the higher pre-trained embedding

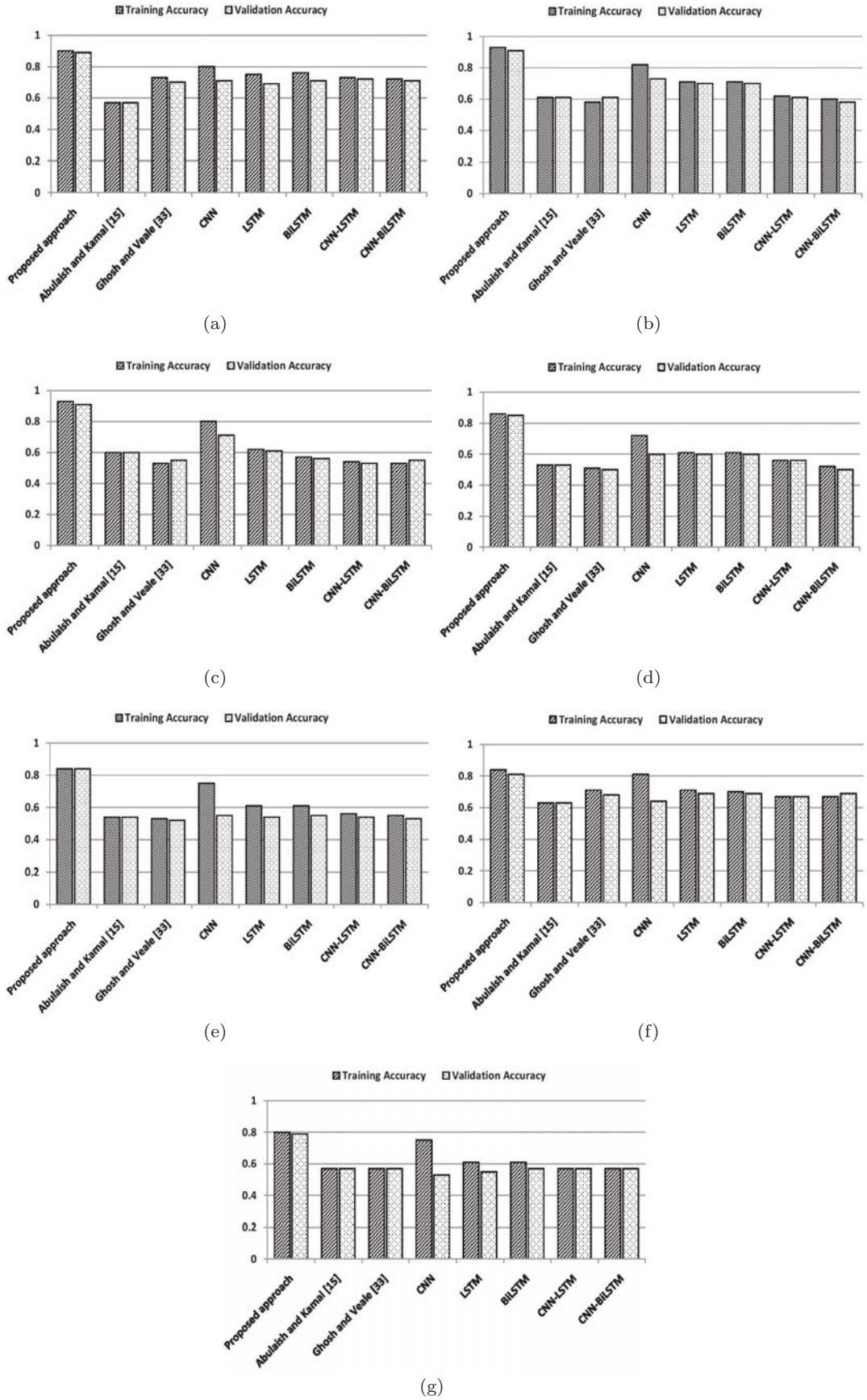

Fig. 5: Training and validation accuracy values over (a) Ptácek et al. [30], (b) SemEval 2015 [4], (c) Riloff et al. [22], (d) Bamman and Smith [32], (e) Ghosh and Veale [33], (f) Ling and Klinger [31], and (g) `Twitter`-280 datasets

dimension is better for the feature extraction process, and it is also beneficial for the CAT-BiGRU model for detecting SDS.

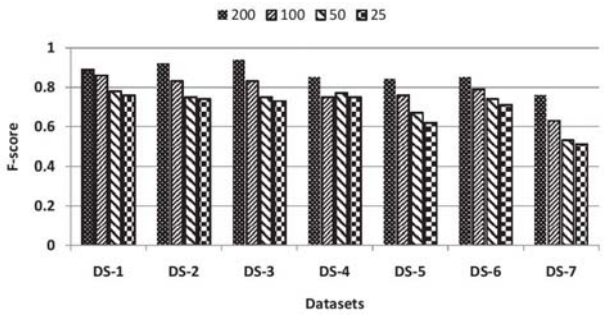

Fig. 6: Effect of different GloVe embeddings dimensions (200, 100, 50, and 25) on the classification results of CAT-BiGRU model over Ptácek et al. [30] (DS-1), SemEval 2015 [4] (DS-2), Riloff et al. [22] (DS-3), Bamman and Smith [32] (DS-4), Ghosh and Veale [33] (DS-5), Ling and Klinger [31] (DS-6), and Twitter-280 (DS-7) datasets in terms of *f-score*

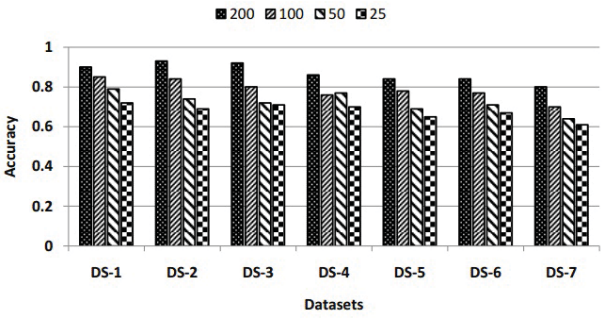

Fig. 7: Effect of different GloVe embeddings dimensions (200, 100, 50, and 25) on the classification results of CAT-BiGRU model over Ptácek et al. [30] (DS-1), SemEval 2015 [4] (DS-2), Riloff et al. [22] (DS-3), Bamman and Smith [32] (DS-4), Ghosh and Veale [33] (DS-5), Ling and Klinger [31] (DS-6), and Twitter-280 (DS-7) datasets in terms of *accuracy*

5.2 Effect of Parameters

Parameter tuning plays an important role in deep learning models. This section presents an analysis of the effect of different parameters viz. number of GRU hidden units, optimization algorithms, and activation functions on the CAT-BiGRU model.

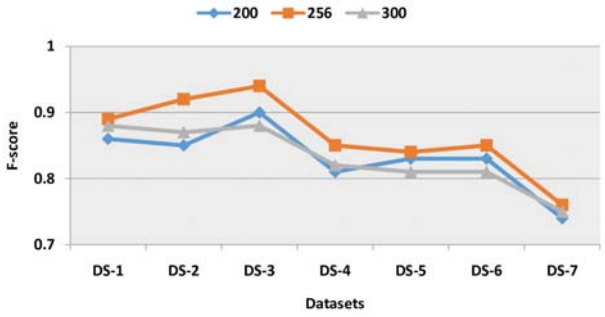

Fig. 8: Effect of different GRU *hidden units* (200, 256, and 300) on the classification results of CAT-BiGRU model over Ptácek et al. [30] (DS-1), SemEval 2015 [4] (DS-2), Riloff et al. [22] (DS-3), Bamman and Smith [32] (DS-4), Ghosh and Veale [33] (DS-5), Ling and Klinger [31] (DS-6), and Twitter-280 (DS-7) datasets in terms of *f-score*

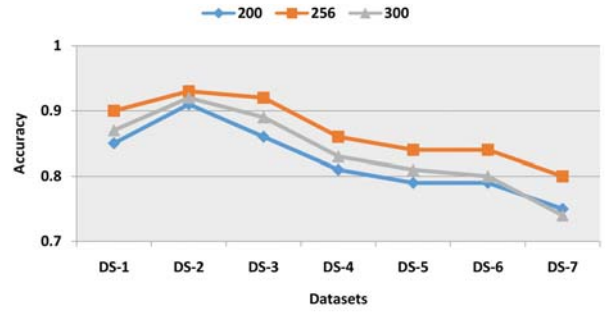

Fig. 9: Effect of different GRU *hidden units* (200, 256, and 300) on the classification results of CAT-BiGRU model over Ptácek et al. [30] (DS-1), SemEval 2015 [4] (DS-2), Riloff et al. [22] (DS-3), Bamman and Smith [32] (DS-4), Ghosh and Veale [33] (DS-5), Ling and Klinger [31] (DS-6), and Twitter-280 (DS-7) datasets in terms of *accuracy*

5.2.1 Number of GRU Hidden Units

The number of hidden units is an important parameter for the performance of any neural network-based model. Although, we have considered a total number of 256 hidden units in our proposed CAT-BiGRU model, here we analyze its performance by varying the number of GRU hidden units. Figures 8 and 9 present the classification results of CAT-BiGRU model for different GRU hidden units (200, 256, and 300) in terms of *f-score* and *accuracy* values, respectively across all datasets. It can be observed that Riloff et al. [22] and SemEval 2015 [4] provide significantly better results in comparison to other datasets in terms of *f-score* and *accuracy* values, respectively. On the other hand, Twitter-280 provides the lowest performance. These

results show that GRU with 256-hidden units perform better across all datasets. These results also indicate that the number of hidden units has a significant impact on the performance of the `CAT-BiGRU` model.

5.2.2 Optimization Algorithms

Optimization algorithms can affect the performance of a classification model. This section presents an analysis of the performance of `CAT-BiGRU` model using two different optimization algorithms – *Adam* and *RMSprop* in terms of *f-score* and *accuracy* values over all datasets.

Both *RMSprop* and *Adam* are popular adaptive stochastic algorithms to train neural network models. *RMSprop* maintains per-parameter adaptive learning rates, depending on the mean of the recent magnitudes of the gradients in terms of weight. It is mainly suitable for online and non-stationary problems. However, it suffers with the sparse gradient problem and lacks the bias-correction factor in the second-order moment estimation. On the other hand, *Adam* does not suffer with the sparse gradient problem, and it also solves the bias-correction problem which helps *Adam* to outperform *RMSprop* towards the end of the optimization process where the gradients become sparser. Moreover, *Adam* optimizes each parameter individually with different and adaptive learning rates (*aka* `alpha`) parameter. It includes other parameters like `beta1` and `beta2` that measure the exponential decay rate for the first-moment and second-moment estimates, respectively to change the learning rate for each weight of the neural network.

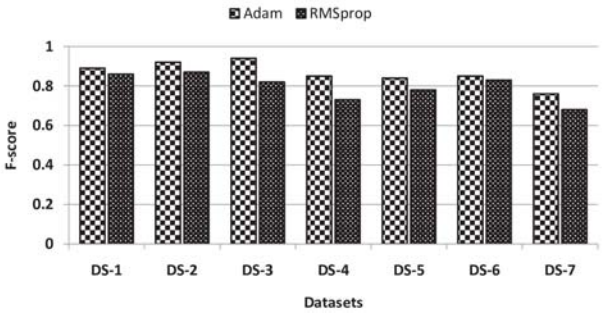

Fig. 10: Effect of different optimization algorithms (*Adam* and *RMSprop*) on the classification results of `CAT-BiGRU` model over Ptácek et al. [30] (DS-1), SemEval 2015 [4] (DS-2), Riloff et al. [22] (DS-3), Bamman and Smith [32] (DS-4), Ghosh and Veale [33] (DS-5), Ling and Klinger [31] (DS-6), and `Twitter`-280 (DS-7) datasets in terms of *f-score*

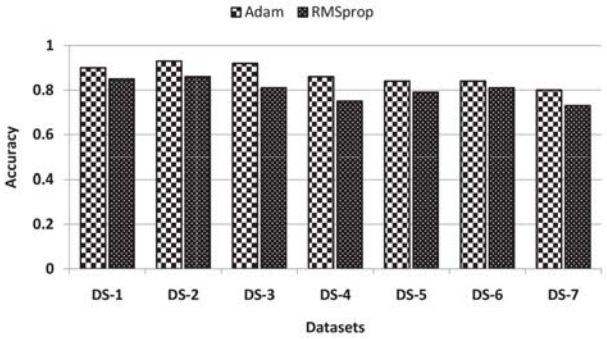

Fig. 11: Effect of different optimization algorithms (*Adam* and *RMSprop*) on the classification results of `CAT-BiGRU` model over Ptácek et al. [30] (DS-1), SemEval 2015 [4] (DS-2), Riloff et al. [22] (DS-3), Bamman and Smith [32] (DS-4), Ghosh and Veale [33] (DS-5), Ling and Klinger [31] (DS-6), and `Twitter`-280 (DS-7) datasets in terms of *accuracy*

Figures 10 and 11 present the effect of *Adam* and *RMSprop* optimization algorithms on the classification results of `CAT-BiGRU` model in terms of *f-score* and *accuracy* values over all datasets. It can be observed that Riloff et al. [22] and SemEval 2015 [4] provide highest *f-score* and *accuracy* values, respectively for *Adam* optimization algorithm. On the other hand, `Twitter`-280 provides lowest *f-score* and *accuracy* values for the *Adam* optimization algorithm. Overall, it can be observed from these figures that the results obtained using the *Adam* optimizer over all datasets are comparatively better than the results obtained using the *RMSprop* optimizer.

5.2.3 Effect of Activation Functions

Like optimization algorithms, activation functions also play a key role on the performance of the classification model. In this section, we analyze the effect of different activation functions (*sigmoid* and *softmax*) on the performance of `CAT-BiGRU` model. Both functions are generally used in logistic regression and neural networks. However, *sigmoid* is suitable for two-class logistic regression, whereas *softmax* is suitable for multi-class logistic regression.

Figures 12 and 13 visualize the effect of *sigmoid* and *softmax* activation functions on classification results of `CAT-BiGRU` model in terms of *f-score* and *accuracy* values, respectively over all datasets. It can be observed that Riloff et al. [22] and SemEval 2015 [4] provide highest *f-score* and *accuracy* values, respectively for *sigmoid* activation function over all datasets, whereas `Twitter`-280 provides lowest *f-score* and *accuracy* values for *sigmoid* activation function. It can be

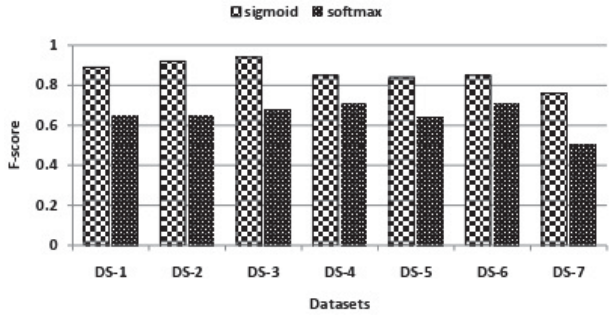

Fig. 12: Effect of different *activation functions* (*sigmoid* and *softmax*) on the classification results of `CAT-BiGRU` model over Ptácek et al. [30] (DS-1), SemEval 2015 [4] (DS-2), Riloff et al. [22] (DS-3), Bamman and Smith [32] (DS-4), Ghosh and Veale [33] (DS-5), Ling and Klinger [31] (DS-6), and `Twitter`-280 (DS-7) datasets in terms of *f-score*

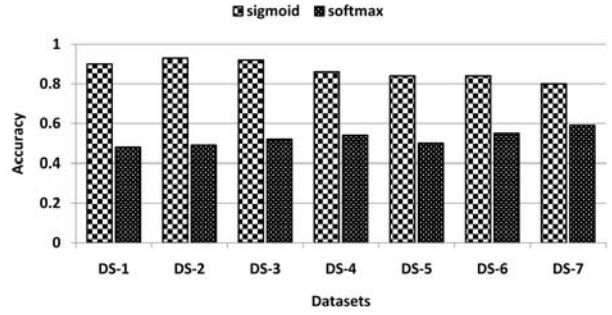

Fig. 13: Effect of different *activation functions* (*sigmoid* and *softmax*) on the classification results of `CAT-BiGRU` model over Ptácek et al. [30] (DS-1), SemEval 2015 [4] (DS-2), Riloff et al. [22] (DS-3), Bamman and Smith [32] (DS-4), Ghosh and Veale [33] (DS-5), Ling and Klinger [31] (DS-6), and `Twitter`-280 (DS-7) datasets in terms of *accuracy*

observed from these figures that the classification results of `CAT-BiGRU` model obtained using *sigmoid* function is better than the results obtained using *softmax* function over all datasets.

5.3 Effect of Sentic Computing Resources

`SenticNet` is a popular and common sense knowledge base for concept-level sentiment analysis [52]. Apart from common sense knowledge, it also considers affective knowledge via biologically-inspired and psychologically-motivated emotional categorization model, wherein emotions are analyzed into independent, but connected via affective dimensions [54,56,59]. In this section, we present the effects of two `SenticNet`-based sentic computing resources based on vector space model (word embedding) – `Amazon word embedding` (`Amazon WE`) [53] and `AffectiveSpace` [50] on the proposed `CAT-BiGRU` model.

`Amazon WE` is a `sentic computing` resource of `SenticNet` which is based on `word2vec` model and provides a 300-dimensional sentiment embeddings generated from the `Amazon` product reviews, and that also includes affective information. On the other hand, `AffectiveSpace` is a 100-dimensional vector space representation of `AffectNet`, which is a matrix of affective commonsense knowledge and `SenticNet` is built on it. In this section, both sentic computing resources are used to evaluate the classification results of our proposed `CAT-BiGRU` model, and also compared with `Twitter`-specific `GloVe` embedding over all datasets.

Table 8: Effect of `GloVe` and sentic computing resources (`Amazon WE` and `AffectiveSpace`) on the classification results of `CAT-BiGRU` model over Ptácek et al. [30] (DS-1), SemEval 2015 [4] (DS-2), Riloff et al. [22] (DS-3), Bamman and Smith [32] (DS-4), Ghosh and Veale [33] (DS-5), Ling and Klinger [31] (DS-6), and `Twitter`-280 (DS-7) datasets in terms of *f-score*

Datasets	GloVe	Amazon WE	Affective Space
DS-1	0.89	0.85	0.79
DS-2	0.92	0.86	0.81
DS-3	0.94	0.84	0.69
DS-4	0.85	0.80	0.75
DS-5	0.84	0.81	0.58
DS-6	0.85	0.82	0.81
DS-7	0.76	0.75	0.73

Table 9: Effect of `GloVe` and sentic computing resources (`Amazon WE` and `AffectiveSpace`) on the classification results of `CAT-BiGRU` model over Ptácek et al. [30] (DS-1), SemEval 2015 [4] (DS-2), Riloff et al. [22] (DS-3), Bamman and Smith [32] (DS-4), Ghosh and Veale [33] (DS-5), Ling and Klinger [31] (DS-6), and `Twitter`-280 (DS-7) datasets in terms of *accuracy*

Datasets	GloVe	Amazon WE	Affective Space
DS-1	0.90	0.82	0.77
DS-2	0.93	0.87	0.79
DS-3	0.92	0.85	0.63
DS-4	0.86	0.83	0.76
DS-5	0.84	0.80	0.67
DS-6	0.84	0.81	0.75
DS-7	0.80	0.78	0.72

Table 8 and 9 present the classification results of `CAT-BiGRU` model using `GloVe`, `Amazon WE`, and `AffectiveSpace` in terms of *f-score* and *accuracy* values, respectively over all datasets. Interestingly, the proposed `CAT-BiGRU` model using both sentic computing resources provides good results in terms of *f-score* and *accuracy* values over all datasets. However, results obtained using `Amazon WE` is better in comparison to `AffectiveSpace`. Based on these results it can be inferred that inclusion of sentic computing resources in `CAT-BiGRU` model can boost its accuracy for detecting SDS.

6 Conclusion and Future Work

SDS is a special category of sarcasm which is mainly used as an effective tool for product campaign and marketing. In this paper, we have proposed a novel `CAT-BiGRU` model for SDS detection. The proposed model consists of an input, embedding, convolutional, BiGRU, and two attention layers, and it is evaluated over seven datasets from different perspectives. Experimental results of `CAT-BiGRU` are promising and significantly better in comparison to various neural network-based baselines and state-of-the-art methods. One of the main aims of this novel SDS detection technique is to enhance the SDS-based marketing strategy. We plan to develop a full-fledged web-based tool to read user-supplied inputs and provide SDS score, polarity value, different forms of visualization, and various levels of emotion using biologically-inspired and psychologically-motivated `SenticNet`-based sentic computing resources like `The Hourglass of Emotions` [55] as output. The tool could be useful for both marketing management team and end-users for analysis, recommendation, and extraction of information about the latest trend in SDS-based advertisements of a product or brand. In addition, extending the proposed approach of SDS detection in multilingual data that can be operational on multimodal platforms seems one of the interesting future directions of research.

Acknowledgment

Supported by Visvesvaraya PhD Scheme, MeitY, Govt. of India. 'MEITY-PHD-555'.

Compliance with Ethical Standards

Conflict of interest

The authors declare that they have no conflict of interest.

Ethical Approval

This article does not contain any studies with human participants or animals performed by any of the authors.

References

1. Bouazizi M, Ohtsuki T. Opinion mining in twitter how to make use of sarcasm to enhance sentiment analysis. In: Proceedings of the IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), Paris, France, August 25–28, 2015; pp. 1594–1597.
2. Abulaish M, Kamal A, Zaki MJ. A survey of figurative language and its computational detection in online social networks. ACM Transactions on the Web. 2020;14(1): pp. 1–52.
3. Bouazizi M, Ohtsuki T. A pattern-based approach for sarcasm detection on twitter. IEEE Access. 2016;4(1): pp. 5477–88.
4. Ghosh A, Li G, Veale T, Rosso P, Shutova E, Barnden J, Reyes A. Semeval-2015 task 11: sentiment analysis of figurative language in twitter. In: Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval), Denver, Colorado, June 4–5, 2015; pp. 470–478.
5. Stieger S, Formann AK, Burger C. Humor styles and their relationship to explicit and implicit self-esteem. Personality and Individual Differences. 2011;50(5): pp. 747–50.
6. Kamal A, Abulaish M. Self-deprecating humor detection: a machine learning approach. In: Proceedings of the 16th International Conference of the Pacific Association for Computational Linguistics (PACLING), Hanoi, Vietnam, October 11–13, 2019; pp. 1–13.
7. Kamal A, Abulaish M. An LSTM-based deep learning approach for detecting self-deprecating sarcasm in textual data. In: Proceedings of the 16th International Conference on Natural Language Processing (ICON), Hyderabad, India, December 18–21, 2019; pp. 1–10.
8. Joshi A, Bhattacharyya P, Carman MJ. Automatic sarcasm detection: a survey. ACM Computing Surveys. 2017;50(5): pp. 1–22.
9. Zhang M, Zhang Y, Fu G. Tweet sarcasm detection using deep neural network. In: Proceedings of the 26th International Conference on Computational Linguistics (COLING), Osaka, Japan, December 11–17, 2016; pp. 2449–2460.
10. González-Ibáñez R, Muresan S, Wacholder N. Identifying sarcasm in Twitter: a closer look. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL), Portland, Oregon, June 19–24, 2011; pp. 581–586.

11. Lukin S, Walker M. Really? well. apparently bootstrapping improves the performance of sarcasm and nastiness classifiers for online dialogue. In: Proceedings of the Workshop on Language in Social Media (LASM), Atlanta, Georgia, June 13, 2013; pp. 30–40.
12. Rajadesingan A, Zafarani R, Liu H. Sarcasm detection on twitter: a behavioral modeling approach. In: Proceedings of the 8th Association for Computing Machinery International Conference on Web Search and Data Mining (WSDM), Shanghai, China, February 2–6, 2015; pp. 97–106.
13. Tsur O, Davidov D, Rappoport A. ICWSM—a great catchy name: semi-supervised recognition of sarcastic sentences in online product reviews. In: Proceedings of the 4th International Association for the Advancement of Artificial Intelligence Conference on Weblogs and Social Media (ICWSM), Washington, DC, USA, May 23–26, 2010; pp. 162–169.
14. Davidov D, Tsur O, Rappoport A. Semi-supervised recognition of sarcastic sentences in twitter and amazon. In: Proceedings of the 14th Conference on Computational Natural Language Learning (CoNLL), Uppsala, Sweden, July 15–16, 2010; pp. 107–116.
15. Abulaish M, Kamal A. Self-deprecating sarcasm detection: an amalgamation of rule-based and machine learning approach. In: Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence (WI), Santiago, Chile, December 3–6, 2018; pp. 574–579.
16. Schifanella R, de Juan P, Tetreault J, Cao L. Detecting sarcasm in multimodal social platforms. In: Proceedings of the ACM on Multimedia Conference (ACM MM), Amsterdam, Netherlands, October 15–19, 2016; pp. 1136–1145.
17. Amir S, Wallace BC, Lyu H, Silva PC. Modelling context with user embeddings for sarcasm detection in social media. In: Proceedings of the 20th Special Interest Group on Natural Language Learning Conference (SIGNLL) on Computational Natural Language Learning (CoNLL), Berlin, Germany, August 7–12, 2016; pp. 167–177.
18. Poria S, Cambria E, Hazarika D, Vij P. A deeper look into sarcastic tweets using deep convolutional neural networks. In: Proceedings of the 26th International Conference on Computational Linguistics: Technical Papers (COLING), Osaka, Japan, December 11-16, 2016; pp. 1601–1612.
19. Avvaru A, Vobilisetty S, Mamidi, R. Detecting sarcasm in conversation context using transformer-based models. In: Proceedings of the 2nd Workshop on Figurative Language Processing (ACL), July 9, 2020; pp. 98–103.
20. Dubey A, Joshi A, Bhattacharyya P. Deep models for converting sarcastic utterances into their non sarcastic interpretation. In: Proceedings of the ACM India Joint International Conference on Data Science and Management of Data (CoDS-COMAD), Koklata, India, January 3–5, 2019; pp. 289–292.
21. Dubey A, Kumar L, Somani A, Joshi A, Bhattacharyya P. "When numbers matter!": detecting sarcasm in numerical portions of text. In: Proceedings of the 10th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis (WASSA), June 6, 2019; pp. 72–80.
22. Riloff E, Qadir A, Surve P, De Silva L, Gilbert N, Huang R. Sarcasm as contrast between a positive sentiment and negative situation. In: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP), Seattle, Washington, USA, October 18-21, 2013; pp. 704–714.
23. Khattri A, Joshi A, Bhattacharyya P, Carman MJ. Your sentiment precedes you: Using an author's historical tweets to predict sarcasm. In: Proceedings of the 6th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis (WASSA), Lisboa, Portugal, September 17, 2015; pp. 25–30.
24. Bharti SK, Babu KS, Jena SK. Parsing-based sarcasm sentiment recognition in twitter data. In: Proceedings of the IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), Paris, France, August 25–28, 2015; pp. 1373–1380.
25. Liebrecht CC, Kunneman FA, Van Den Bosch AP. The perfect solution for detecting sarcasm in tweets #not. In: Proceedings of the 4th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis (WASSA), Atlanta, Georgia, June 13–14, 2013; pp. 29–37.
26. Littlestone N. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. Machine Learning. 1988;2(4): pp. 285–318.
27. Mishra A, Kanojia D, Bhattacharyya P. Predicting readers' sarcasm understandability by modeling gaze behavior. In: Proceedings of the 13th Association for the Advancement of Artificial Intelligence Conference on Artificial Intelligence (AAAI), Phoenix, Arizona, USA, February 12–17, 2016; pp. 3747–3753.
28. Zhao Z, Resnick P, Mei Q. Enquiring minds: Early detection of rumors in social media from enquiry posts. In: Proceedings of the 24th International Conference on World Wide Web (WWW), Florence, Italy, May 18–22, 2015; pp. 1395–1405.
29. Liu G, Guo J. Bidirectional LSTM with attention mechanism and convolutional layer for text classification. Neurocomputing. 2019;337: pp. 325–338.
30. Ptácek T, Habernal I, Hong J. Sarcasm detection on czech and english twitter. In: Proceedings of the 25th International Conference on Computational Linguistics (COLING), Dublin, Ireland, August 23–29, 2014; pp. 213–223.
31. Ling J, Klinger R. An empirical quantitative analysis of the differences between sarcasm and irony. In: Sack H, Rizzo G, Steinmetz N, Mladenic D, Auer S, Lange C. (eds). The Semantic Web. ESWC 2016. LNCS, Springer; 9989: pp. 203–216.
32. Bamman D, Smith NA. Contextualized sarcasm detection on twitter. In: Proceedings of the 9th International Association for the Advancement of Artificial Intelligence Conference on Web and Social Media (ICWSM), Oxford, UK, May 26–29, 2015; pp. 574–577.
33. Ghosh A, Veale T. Fracking sarcasm using neural network. In: Proceedings of the 15th North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT), San Diego, California, USA, June 12–17, 2016; pp. 161–169.
34. Kim Y. Convolutional neural networks for sentence classification. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, October 25–29, 2014; pp. 1746–1751.
35. Hochreiter S, Schmidhuber J. Long short-term memory. Neural Computation. 1997;9(8): pp. 1735–80.
36. Justo R, Alcaide JM, Torres MI, Walker, M. Detection of sarcasm and nastiness: new resources for spanish

language. Cognitive Computation. 2018;10(6): pp. 1135–1151.
37. Elovici Y, Fire M, Herzberg A, Shulman H. Ethical considerations when employing fake identities in online social networks for research. Science and Engineering Ethics. 2014;20(4): pp. 1027–1043.
38. Zinck A. Self-Referential Emotion. In Emotions Hold the Self Together 2011; pp. 105–133.
39. Effron M. If only this was a detective novel Self-Referentiality as Metafictionality (Doctoral dissertation, Newcastle University), 2010.
40. Boot AB, Sang ET, Dijkstra K, Zwaan RA. How character limit affects language usage in tweets. Palgrave Communications. 2019;5(1): pp. 1–13.
41. Kreuz R, Caucci G. Lexical influences on the perception of sarcasm. In: Proceedings of the Workshop on Computational Approaches to Figurative Language, Rochester (ACL), NY, USA, April 26, 2007; pp. 1–4.
42. Bengio Y, Ducharme R, Vincent P, Jauvin C. A neural probabilistic language model. Journal of Machine Learning Research. 2003;3(2): pp. 1137–1155.
43. Carvalho J, Plastino A. On the evaluation and combination of state-of-the-art features in Twitter sentiment analysis. Artificial Intelligence Review. 2020; pp. 1–50.
44. Cambria E. Affective computing and sentiment analysis. IEEE Intelligent Systems. 2016;31(2): pp. 102–107.
45. Hussain A, Cambria E. Semi-supervised learning for big social data analysis. Neurocomputing. 2018;275: pp. 1662-1673.
46. Majumder N, Poria S, Peng H, Chhaya N, Cambria E, Gelbukh A. Sentiment and sarcasm classification with multitask learning. IEEE Intelligent Systems. 2019;34(3): pp. 38–43.
47. Mehta Y, Majumder N, Gelbukh A, Cambria E. Recent trends in deep learning based personality detection. Artificial Intelligence Review. 2020;53(4): pp. 2313—2339.
48. Young T, Hazarika D, Poria S, Cambria E. Recent trends in deep learning based natural language processing. IEEE Computational Intelligence Magazine. 2018;13(3): pp. 55–75.
49. Cambria E, Grassi M, Hussain A, Havasi C. Sentic computing for social media marketing. Multimedia Tools and Applications. 2012;59(2): pp. 557–577.
50. Cambria E, Fu J, Bisio F, Poria S. Affectivespace 2: enabling affective intuition for concept-level sentiment analysis. In: Proceedings of the Association for the Advancement of Artificial Intelligence (AAAI), Austin, Texas, USA, January 25–30, 2015; pp. 508–514.
51. Hazarika D, Poria S, Gorantla S, Cambria E, Zimmermann R, Mihalcea R. CASCADE: contextual Sarcasm Detection in Online Discussion Forums. In: Proceedings of the 27th International Conference on Computational Linguistics (COLING), Santa Fe, New-Mexico, USA, August 20–26, 2018; pp. 1837–1848.
52. Cambria E, Li Y, Xing FZ, Poria S, Kwok K. Senticnet: SenticNet 6: ensemble application of symbolic and subsymbolic AI for sentiment analysis. In: Proceedings of the 29th ACM International Conference on Information & Knowledge Management (CIKM), October 19–23, 2020; pp. 105–114.
53. Poria S, Cambria E, Gelbukh A. Aspect extraction for opinion mining with a deep convolutional neural network. Knowledge-Based Systems. 2016;108: pp. 42–49.
54. Cambria E, Livingstone A, Hussain A. The hourglass of emotions. In: Cognitive behavioural systems 2012; pp. 144–157. Springer, Berlin, Heidelberg.
55. Susanto Y, Livingstone AG, Ng BC, Cambria E. The hourglass model revisited. IEEE Intelligent Systems. 2020;35(5): pp. 96–102.
56. Wang Z, Ho SB, Cambria E. A review of emotion sensing: categorization models and algorithms. Multimedia Tools and Applications. 2020: pp. 1–30
57. Camacho D, Panizo-LLedot A, Bello-Orgaz G, Gonzalez-Pardo A, Cambria E. The four dimensions of social network analysis: an overview of research methods, applications, and software tools. Information Fusion. 2020;63: pp. 88–120.
58. Cavallari S, Cambria E, Cai H, Chang KC, Zheng, VW. Embedding both finite and infinite communities on graphs. IEEE computational intelligence magazine. 2019;14(3): pp. 39–50.
59. Ma Y, Nguyen KL, Xing FZ, Cambria E. A survey on empathetic dialogue systems. Information Fusion. 2020;64: pp. 50–70.