Performance Evaluation of Word Embeddings for Sarcasm Detection- A Deep Learning Approach

Annie Johnson[1], Karthik R[2]

[1] School of Electronics Engineering, Vellore Institute of Technology, Chennai.
[2]Centre for Cyber Physical Systems, Vellore Institute of Technology, Chennai.
annie.johnson2016@vitstudent.ac.in, r.karthik@vit.ac.in

Abstract. Sarcasm detection is a critical step to sentiment analysis, with the aim of understanding, and exploiting the available information from platforms such as social media. The accuracy of sarcasm detection depends on the nature of word embeddings. This research performs a comprehensive analysis of different word embedding techniques for effective sarcasm detection. A hybrid combination of optimizers including; the Adaptive Moment Estimation (Adam), the Adaptive Gradient Algorithm (AdaGrad) and Adadelta functions and activation functions like Rectified Linear Unit (ReLU) and Leaky ReLU have been experimented with.Different word embedding techniques including; Bag of Words (BoW), BoW with Term Frequency–Inverse Document Frequency (TF-IDF), Word2vec and Global Vectors for word representation (GloVe) are evaluated. The highest accuracy of 61.68% was achieved with theGloVeembeddings and an AdaGrad optimizer and a ReLU activation function, used in the deep learning model.

Keywords: Sarcasm, Deep Learning, Word Embedding, Sentiment Analysis

1 Introduction

Sarcasm is the expression of criticism and mockery in a particular context, by employing words that mean the opposite of what is intended. This verbal irony in sarcastic text challenges the domain of sentiment analysis.

With the emergence of smart mobile devices and faster internet speeds, there has been an exponential growth in the use of social media websites. There are about 5.112 billion unique mobile users around the world. Social media websites are recognized as global platforms for registering opinions and reviews. This data is invaluable to companies that use targeted product advertising and opinion mining. Hence,designing a reliable sarcasm detector is advantageous to numerous distinct domains.

Over 80% of the data generated from social media channels is unstructured [1]. Sarcasm detection from this unstructured text data is a challenge. As and when events occur, social media users post their opinions and views on these websites. Yet, determining the veracity and legitimacy of the data can be a very challenging task [1]. Another issue is that the data lacks visual or vocal context that assists humans in

ICASISET 2020, May 16-17, Chennai, India

DOI 10.4108/eai.16-5-2020.2303935

identifying sarcasm [2]. In this work,different techniquesare compared for sarcasm detection.

2 Related Works

Various researches related to sarcasm detection in social media blogs have been conducted over the last two decades.The text classification for the sarcasm detector can be done by using two approaches. These methods are predominantly classified as (1) Machine learning methods (2) Deep learning methods.

2.1 Machine Learning Based Methods

Classification related problems in sarcasm detection are handled with multiple machine learning algorithms including; Logistic Regression, Naïve Bayes classifier, Random Forestand Support Vector Machine (SVM). Chandra Pandey et al. performed classification using k-means clustering and cuckoo search. Unlike the traditional feature sets, the work proposed by Chandra Pandey et al. uses BoW representation for text classification [3].Chandra Pandey et al. employed the use of Gaussian Naive Bayes algorithm on Parts of Speech (PoS) tags for sarcasm detection [4]. The work proposed by Mukherjee et al. consisted of employing a similar Naïve Bayes classifier on the Twitter dataset that was collected. A hybrid combination of PoS tags, content words and function words were used in their work [5].

Sulis et al. used a number of classification algorithms witha novel set of structural and affective features. The features used in this work included tweet length and Linguistic Inquiry and Word Counts dictionary (LIWC) which analyses not only the polarity of the tweet, but also the psycholinguistic features in the text [6]. This set of features combined with the random forest classifier, boosted their accuracy as compared to that of Mukherjee et al. The performance of several algorithms, with different feature sets have been compared in the research works of Bouazizi et al. [7], Kumar et al. [8] and Bharti et al. [1],[9] .An improvement in accuracy is seen in the work proposed by Kumar et al. as the dataset that is used has over 25,000 records unlike the 6000 tweets used by Bouazizi et al.

The use of Graphical User Interfaces such as Auto- Waikato Environment for Knowledge Analysis (Auto-WEKA) [10] and SENTA [11] have proved to improve sarcasm detection. Methods involving the use of hand engineered featuresextracted from the text dataset are primarily used for sarcasm detection. A deep learning model provides for a more efficient method of classifying huge volumes of data.

2.2 Deep Learning Based Methods

Various deep learning architectures have been utilized in the recent research in sarcasm detection. Dutta et al. presented an approach that involved the stacking of Elman-type RNNs on top of each other to construct a deep RNN for text classification [12]. This work used GloVe embeddings which performed better than the BoW

embeddings used by Porwal et al. [13]. The work proposed by Mehndiratta et al. [14] also concluded that GloVe achieved a higher accuracy.

The work proposed by Ren et al. focuses on utilizing the contextual information with respect to the target tweet, rather than feature engineering. This work uses Word2vec embeddings as their input features to capture contextual information from the dataset [15]. Naza et al. proposed a Convolutional Neural Network (CNN) architecture for sarcasm detection that makes use of softmax activation for classifying the tweets. This work has not only used Word2vec word embeddings but also incorporated the tweet length to improve their model [16]. Ghosh et al. proposed an Artificial Neural Network (ANN) model for sarcasm detection. Unlike, Ren et al., Ghosh et al. make use of GloVe instead of Word2vec embeddings [17][18].

By adding attention and LSTM to these basic CNN architectures, the metrics obtained during classification can be improved by a great deal. An Attention-based LSTM network was employed in the work presented by Ghosh et al. This work uses LIWC features to train their model [19]. PoS tags are passed as the input features to the bidirectional Long Short-Term Memory (BiLSTM) network in the work proposed by Patro et al. [20]. Son et al. presentedthe sAtt-BLSTM convNet deep learning model. It is a hybrid of a soft attention-based bidirectional long short term memory (sAtt-BLSTM) and a convolution neural network (convNet) model for sarcasm classification. This model performs significantly better than Ghosh et al in terms of accuracy, F1 score, precision and recall. Son et al. use GloVe embeddings unlike Ghosh et al. [21]. The work proposed by Subramanian et al not only utilizes GloVe embeddings, but also stylometric features of the text. Therefore, this work performs better than the models that use only word embeddings [22][26].It also performs better than the Embeddings from Language Model (ELMo) embeddings used in the work presented by Ilié et al. [23][27].

3 Materials and Methods

3.1 Dataset

In this work, the Self-Annotated Reddit Corpus (SARC) constructed by Khodak et al. is used [24]. Social media platforms make large-scale, user-generated content available for analysis. However, due to restrictions in text length in such platforms, authors use acronyms and abbreviations in their statements. To avoid the number abbreviations used in the text, the Reddit platform is preferred to other social media platforms [25]. The available SARC dataset consists of 1008799 records. There are 505208 sarcastic and 503591 non-sarcastic posts in the dataset. In this work, a balanced dataset was extracted from SARC.

3.2 Proposed System

To logically mine the Reddit posts present in the SARC dataset, pre-processing is carried out on the text data. The SARC dataset consists of 1008799 records. A

balanced dataset is constructed by randomly selecting, 50,000 sarcastic and 50,000 non-sarcastic data records from SARC. Preprocessing is performed on this balanced dataset. The different kinds of preprocessing steps that are carried out are listed below.The posts from Reddit are converted to type string and target labels are converted to type integer.Duplicate posts and stop words are removed. Punctuation, URLs and special characters such as @, # are removed.Non-ASCII English characters are removed. The words are converted to lower case.Natural Language tool-Kit (NLTK) is used for tokenization. WordNetLemmatizer is employed for stemming to the root word.

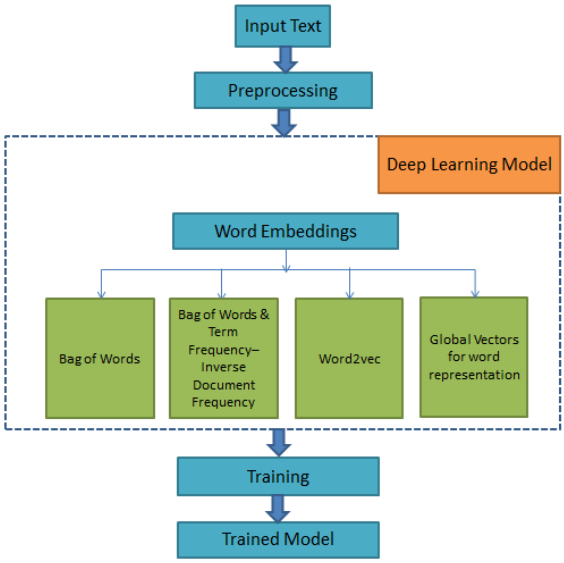

Fig. 1. Block diagram depicting the systematic flow of the deep learning model.

3.2.1 Approach

Existing approaches are unable to detect the subtleties of sarcasm in text as lexicon based features do not represent context incongruity. The context incongruity parameter can be measured with the help of word embeddings. Word Embeddings capture semantic similarity or discordance scores which can enhance sarcasm detection. Hence, by incorporating word embeddings in the feature extraction process, an improvement in the sarcasm detection methods can be observed.

3.2.2 Word Embeddings

This work experiments with four types of word embeddings:

1. Bag of Words (BoW): This technique is used to convert text into word count vectors that can be used in the deep learning model. The frequency with which each word appears in the document out of all the words in the document is considered while constructing the corresponding vector representation.

2. BoW and TF-IDF: The TF-IDF score for each word is calculated and used to further enhance the word embedding representation. TF-IDF is a numerical statistic that is intended to reflect the significanceof a word in a document. The wordcounts are replaced with the TF-IDF scores in the entire corpus.

3. Word2Vec: Here, the unique words in the corpus are assigned corresponding vectors in the vector space. Words sharing common contexts are grouped in close proximity to each other. Word2Vec learns by streaming sentences. Each sentence is then represented by a weighted average of the word embeddings of each word present in the sentence having a dimension of 100. These vectors represent the word with respect to a local context.

4. GloVe: Each sentence is represented by a weighted average of the GloVe word embeddings of each word in the sentence. Unlike Word2Vec, GloVe aims at creating word vectors in a vector space by making use of global count statistics of each word rather than only the local information. GloVe learns based on a co-occurrence matrix. The learned word vectors are then trained such that their differences predict co-occurrence ratios. These ratios are used to create 100 dimensional word vectors.

3.2.3Training the Deep Learning Model

A simple deep learning architecture is used to compare various word embedding techniques. The embedding layer has a vocabulary of 2115 unique words and an input length of 100. An embedding space of 100 dimensions is used in the proposed work. A dense layer having 128 neurons is used on the embedding vectors. This layer is followed by a dropout layer, which is connected to the output layer.

3.2.4 Classification

The final layer of the deep learning architecture performs binary classification. This output layer consists of a dense layer with 1 neuron, thus presenting a single column vector output.

4Results and Discussion

The proposed experiment was conductedon the balanced dataset extracted from the SARC benchmark dataset. The source code was executed in Tensorflow Deep Learning programming framework with the Tesla K80 GPU. Various optimizer functions are used with the word embeddings in the output layer. These optimizers

include; Adam, AdaGrad and Adadelta functions. Both ReLU and Leaky ReLU activation functions were used. The model was trained for 100 epochs. The accuracy, precision, recall and F1 scores are analyzed for every experiment that is carried out.The results of the conducted experiments have been recorded in table 1.

Table 1. Performance comparison of various optimizer and activation functions on the deep learning model, using four different word embedding techniques.

Word Embedding	Optimizer	Activation	Train accuracy	Validation accuracy	Test accuracy	Precision	Recall	F1 score
BoW	Adam	ReLU	53.83	52.23	52.23	0.52	0.52	0.52
	Adam	Leaky ReLU	53.14	52.55	52.55	0.52	0.52	0.52
	Adagrad	ReLU	52.53	52.01	52.02	0.53	0.52	0.52
	Adagrad	Leaky ReLU	52.01	52.22	52.22	0.52	0.52	0.52
	Adadelta	ReLU	52.00	52.03	52.03	0.52	0.52	0.52
	Adadelta	Leaky ReLU	52.12	52.39	52.39	0.52	0.52	0.52
BoW and TF-IDF	Adam	ReLU	56.57	52.01	52.01	0.52	0.52	0.52
	Adam	Leaky ReLU	55.62	51.29	51.33	0.52	0.52	0.52
	Adagrad	ReLU	53.36	51.63	51.63	0.53	0.52	0.52
	Adagrad	Leaky ReLU	53.19	52.00	52.07	0.52	0.52	0.52
	Adadelta	ReLU	54.72	51.12	51.12	0.52	0.52	0.52
	Adadelta	Leaky ReLU	54.20	51.04	51.04	0.52	0.52	0.52
Word2vec	Adam	ReLU	58.97	52.29	52.28	0.52	0.52	0.52
	Adam	Leaky ReLU	58.85	51.85	51.85	0.52	0.52	0.52
	Adagrad	ReLU	55.46	52.54	52.54	0.53	0.53	0.53
	Adagrad	Leaky ReLU	55.21	52.41	52.41	0.53	0.52	0.52
	Adadelta	ReLU	59.52	52.03	52.02	0.53	0.52	0.52
	Adadelta	Leaky ReLU	57.75	52.39	52.39	0.53	0.52	0.52
GloVe	Adam	ReLU	59.01	52.00	52.00	0.52	0.52	0.52
	Adam	Leaky ReLU	56.64	51.23	51.20	0.52	0.52	0.52
	Adagrad	ReLU	63.41	61.68	60.14	0.57	0.56	0.56
	Adagrad	Leaky ReLU	60.15	53.18	53.18	0.53	0.53	0.53
	Adadelta	ReLU	55.71	50.12	50.12	0.52	0.52	0.52
	Adadelta	Leaky ReLU	56.20	51.80	51.81	0.52	0.52	0.52

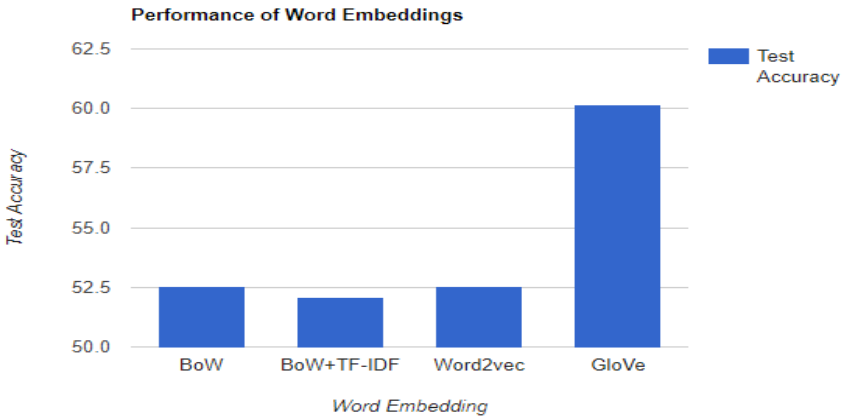

Fig. 2. Bar chart depicting the best test accuracies observed for the word embeddings.

The BoW embeddings performed best with the Adam optimizer and Leaky ReLU activation function. To further enhance the embedding, TF-IDF weights were added to the BoW representation. This hybrid performed best with Adagrad and Leaky ReLU. The Word2vec model performed well with Adagrad and ReLU. However, with the GloVe embeddings, a significant increase in accuracy can be observed.

5 Conclusion

This paper aims to depict the relevance of word embeddings in sarcasm detection. 24 different combinations of word embedding techniques, optimizers and activation functions were used to compare the performance of word embeddings. Both the BoW embeddings and the BoW embeddings incorporated with TF-IDF scores, performed best with the Adam and Adagrad optimizers and the Leaky ReLU activation function, with accuracies of52.55% and 52.07% respectively. The Word2vec model gave an accuracy of 52.54% with Adagrad and ReLU. However, the precision, recall and F1 score are observed to remain constant among all the combinations experimented with except for the GloVe embeddings.The AdaGrad optimizer with the ReLU activation function gave the highest accuracy of 60.14% with the GloVe word embeddings. The GloVe embeddings achieved a precision of 0.57, a recall of 0.56 and a F1 score of 0.56. Through this work, the challenges in detecting sarcasm from social media platforms are discussed. This work emphasizes the significance of global context based word embeddings in sarcasm detection based on the experimental results obtained.

References

[1] Bharti S, Vachha B, Pradhan R, Babu K, Jena S. Sarcastic sentiment detection in tweets streamed in real time: a big data approach. Digital Communications and Networks. 2016;2(3):108-121.

[2] Salas-Zárate M, Paredes-Valverde M, Rodriguez-García M, Valencia-García R, Alor-Hernández G. Automatic detection of satire in Twitter: A psycholinguistic-based approach. Knowl Based Syst. 2017;128:20-33.

[3] Chandra Pandey A, Singh Rajpoot D, Saraswat M. Twitter sentiment analysis using hybrid cuckoo search method. Inf Process Manag. 2017;53(4):764-779.

[4] Pandey AC, Seth SR, Varshney M. Sarcasm Detection of Amazon Alexa Sample Set. Lecture Notes in Electrical Engineering Advances in Signal Processing and Communication. 2019:559-564.

[5] Mukherjee S, Bala P. Sarcasm detection in microblogs using Naïve Bayes and fuzzy clustering. Technol Soc. 2017;48:19-27.

[6] Sulis E, Irazú Hernández Farías D, Rosso P, Patti V, Ruffo G. Figurative messages and affect in Twitter: Differences between #irony, #sarcasm and #not. Knowl Based Syst. 2016;108:132-143.

[7] Bouazizi M, Otsuki Ohtsuki T. A Pattern-Based Approach for Sarcasm Detection on Twitter. IEEE Access. 2016;4:5477-5488.

[8] Kumar R, Kaur J. Random Forest-Based Sarcastic Tweet Classification Using Multiple Feature Collection. Intelligent Systems Reference Library Multimedia Big Data Computing for IoT Applications. 2020:131-160.

[9] Bharti SK, Pradhan R, Babu KS, Jena SK. Sarcasm Analysis on Twitter Data Using Machine Learning Approaches. Lecture Notes in Social Networks Trends in Social Network Analysis. 2017:51-76.

[10] Ravi K, Ravi V. A novel automatic satire and irony detection using ensembled feature selection and data mining. Knowl Based Syst. 2017;120:15-33.

[11] Bouazizi M, Ohtsuki T. A Pattern-Based Approach for Multi-Class Sentiment Analysis in Twitter. IEEE Access. 2017;5:20617-20639.

[12] Dutta S, Chakraborty A. A Deep Learning-Inspired Method for Social Media Satire Detection. Advances in Intelligent Systems and Computing Soft Computing and Signal Processing. 2019:243-251.

[13] Porwal S, Ostwal G, Phadtare A, Pandey M, Marathe MV. Sarcasm Detection Using Recurrent Neural Network. 2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS). 2018:746-748.

[14] Mehndiratta P, Soni D. Identification of sarcasm using word embeddings and hyperparameters tuning. Journal of Discrete Mathematical Sciences and Cryptography. 2019;22(4):465-489.

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

[16] Naz F, Kamran M, Mehmood W, et al. Automatic identification of sarcasm in tweets and customer reviews. Journal of Intelligent & Fuzzy Systems. 2019;37(5):6815-6828.

[17] Ghosh A, Veale T. Magnets for Sarcasm: Making Sarcasm Detection Timely, Contextual and Very Personal. Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing. 2017.

[18] Ghosh A, Veale DT. Fracking Sarcasm using Neural Network. Proceedings of the 7th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis. 2016.

[19] Ghosh D, Fabbri AR, Muresan S. Sarcasm Analysis Using Conversation Context. Computational Linguistics. 2018;44(4):755-792.

[20] Patro J, Bansal S, Mukherjee A. A deep-learning framework to detect sarcasm targets. Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP). 2019: 6337-6343.
[21] Son LH, Kumar A, Sangwan SR, Arora A, Nayyar A, Abdel-Basset M. Sarcasm Detection Using Soft Attention-Based Bidirectional Long Short-Term Memory Model With Convolution Network. IEEE Access. 2019;7:23319-23328.
[22] Subramanian J, Sridharan V, Shu K, Liu H. Exploiting Emojis for Sarcasm Detection. Social, Cultural, and Behavioral Modeling Lecture Notes in Computer Science. 2019;11549:70-80.
[23] Ilić S, Marrese-Taylor E, Balazs J, Matsuo Y. Deep contextualized word representations for detecting sarcasm and irony. Proceedings of the 9th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis. 2018.
[24] Khodak M, Saunshi N, Vodrahalli K. A Large Self-Annotated Corpus for Sarcasm. 2017.
[25] Joshi A, Bhattacharyya P, Carman MJ. Automatic Sarcasm Detection. ACM Computing Surveys. 2017;50(5):1-22.
[26] R. Joseph Manoj, M. D. Anto Praveena, K. Vijayakumar, “An ACO–ANN based feature selection algorithm for big data”, Cluster computing, springer,march 2018.
[27] Dafni Rose J, Vijayakumar K, “Data Transmission Using Multiple Medium Concurrently”, IJET,2018.