ELMO Embedding for Sarcasm Detection

Joon Jyoti Deka and Achyuth Sarkar

Abstract Sarcasm is the new form of message or text widely used on social media and micro-blogging sites and commonly used to convey messages or information in a hidden way. Sarcasm can be used for a variety of purposes, such as criticism or ridicule. However, this is difficult even for human recognition as the meaning conveyed is not simple. Hence, sarcasm recognition is helpful for analyzing sentiment of data extracted from various sites. Sentiment analysis is identifying and summarizing the attitudes and opinions of Internet users toward a particular topic. In this chapter, we emphasize the use of deep contextualized word representation through pre-trained ELMo models that can model complex characteristics of word use (e.g., Syntax and Semantic). Our model achieves an accuracy of 74.15%. In general, we give our preference to ELMo embeddings for the task of sarcasm detection.

Keywords Sentiment analysis · ELMo embeddings · Deep learning

1 Introduction

The popularity of social media is growing. With increasing popularity, social media is emerging as a platform for people to share their views and feelings. People react on a post or picture shared by others, write reviews on products, and express their opinions on different topics on these public platforms. Corporate and government organizations analyze this data available in the public domain to understand the sentiment of their customers or people. Sarcasm, in recent days, has emerged as a way to express their views and share information. It may convey a negative sentiment hidden under some positive utterance or sentences. Sarcastic comments contain hidden meanings that are very hard to understand and analyze for machines as well as for humans also. As a result of this, sarcasm detection has emerged as a popular topic in sentiment analysis. Sarcasm detection is a type classification problem in natural language processing. Traditionally, sarcasm prediction research was predominantly

J. J. Deka (✉) · A. Sarkar
National Institute of Technology, Nirjuli, Arunachal Pradesh, India

V. Goar et al. (eds.), *Advances in Information Communication Technology and Computing*, Lecture Notes in Networks and Systems 392,
https://doi.org/10.1007/978-981-19-0619-0_32

based on rules and statistical data [1]. Lately, with the increasing popularity of neural networks, studies have shifted toward deep learning approaches due to it their ability to learn automatically [2, 3].

In this paper, our work is based on embeddings from language models (ELMo) word embeddings that consider complex characteristics of words in a contextualized manner. The ELMo embedding is used on top of deep learning model. We build a BiLSTM model with ELMo embedding as the embedding layer. We also build a RNN model with Glove embedding layer on top. These two models are compared to each other.

The remainder of the paper is organized in the following way: We presented our methodology in Sect. 3 before discussing the experimental setup in Sect. 4. In Sect. 5, we discussed the results. Finally, a summary concludes the paper.

2 Related Work

As part of the natural language processing field, sarcasm detection has evolved as a new research area and has quickly gained popularity. The detection of sarcasm has traditionally been accomplished through a wide range of methods. Bamman [4] in his work has analyzed the relationship of author and the audience. Characteristics derived from the message context are also taken into consideration.

Bouazizi [5] developed an approach based on patterns to detect sarcasm. Sentiment, punctuation, syntactic semantic, and pattern-based features are used. Kreuz and Caucci [6] in their work study the linguistic features and came to conclusion interjections and punctuation plays a key role in identifying sarcasm. Joshi et al. [7] used four different features, including lexical, pragmatic, implicit, and explicit congruity for detecting sarcasm in a text corpus. Eisterhold [8] from his research found that it is possible to predict sarcasm by the previous and following sentences. Muresan [9] in his work analyzed the impact of lexical and pragmatic features on detecting sarcastic utterances in a sentence.

Recent years have seen an increase in popularity of deep neural network-based approaches in natural language processing, and it has been frequently used for sarcasm detection. A significant reason due to which deep neural networks have immense success over time is their capability to learn and gather knowledge about features automatically. Hazarika et al. [10] proposed a method that considers both contextual and content information for classification. They extract non-textual information from an open discussion thread along with stylistic and personality aspects encoded from user embeddings. Kumar [11] proposed a multi-head attention-based BiLSTM network that focuses on different areas of the comment to analyze the details of semantics in sentences. Zhang [12] developed a bidirectional synchronized recurrent neural network, which captures syntactic and semantic characteristics and a network of pooling neurons that extracts con. contextual features from the tweets. Poria et al. [13] have developed a CNN-based approach that utilizes pre-trained CNN to extract feelings, emotions, and personality traits to detect sarcasm. Gosh et al. [14]

in his work have analyzed sarcasm detection based on a particular type of context, i.e., conversation context. He has addressed two issues, which are whether modeling a conversation context helps sarcasm detection and if we can figure out which part of the conversation triggered sarcasm. Riloff et al. [15] introduced a bootstrapped learning method that can collect the positive sentiment phrases and negative activities or states from tweets. Based on these collected lists, we can recognize sarcastic tweets. Amir et al. [16] have developed a deep neural network that automatically detects sarcastic utterances by learning and exploiting user embeddings for both content and user. Cai et al. [17] focused on multi-modal sarcasm detection for tweets. Image attributes and text, image features are taken as the three modalities, and a multi-modal hierarchical fusion model is employed for the sarcasm detection task. Mishra et al. [18] introduced a method based on cognitive features extracted from the patterns of eye movement of the readers. Their work implemented a cognitive feature-based model for sarcasm detection.

3 Methodology of Our Model

The following sections describe the procedures that were followed in our work. Figure 1 outlines our general approach for detecting sarcasm. It starts with data preparation. At first, we divide the dataset into a train and a test sets. After that, we perform data preprocessing on the training set. After removing all unnecessary characters and words, we lemmatize the training corpus that removes inflectional endings in words and return words to its dictionary form. At last, we change the four different class of sarcasm into numerical form where each sarcasm class label gets its numerical value. Then we, import the ELMo embeddings and passed it as an embedding layer on top of bidirectional LSTM deep learning model.

3.1 Word Embeddings

For our experiment purpose, we take two word embedding techniques (1) ELMo embedding and (2) Glove embedding. A deep bidirectional language model is used by ELMo, and it is trained on large text data to produce vector form. For every token t_x, a single layer in biLM calculates a set of 2L + 1 representations

$$R_x = \left\{ h_{k,j}^{LM} | j = 0, \ldots, L \right\}$$

R-layers are collapsed by ELMo into one vector,

$$ELMo_x = E(R_x : \Theta_e)$$

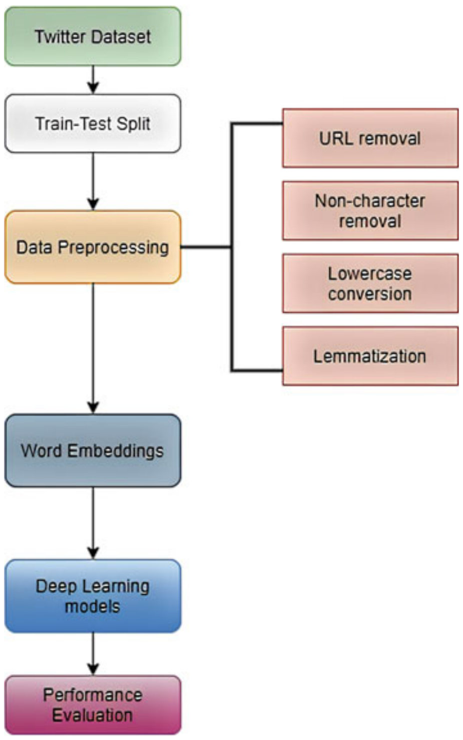


Fig.1 Diagram of work-flow of the experiment

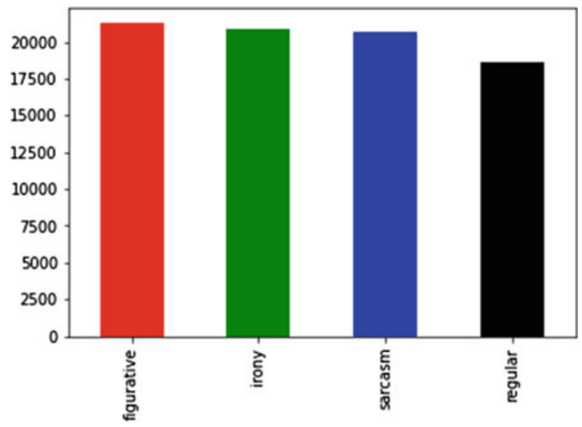
Given a sequence of tokens, it forms a context-independent token representation using pre-trained word embeddings, then the model formed a context-dependent representation.

Glove embedding creates an embedding matrix. It calculates the ratio of co-occurrence probability of a probe word in relation with two words in the corpus. If w_i, w_j are two words and w_k is the probe word, then a function F calculates the probability

$$F\left(w_i, w_j, w_k\right) = \frac{P_{ik}}{P_{jk}}$$

The ratio tells us about the interrelation between the words w_k, $w_{i,}$ and w_j. Glove calculates the elements that are hidden between words to predict their probability of occurring together and produce an embedding matrix of the vector representations.

Fig. 2 Distribution of tweets in each label



4 Experimental Setup

4.1 Dataset

In this experiment, a public Twitter Sarcasm dataset is used. This dataset contains more than a million comments or tweets that are of four types: sarcastic, irony, regular, and figurative. Our dataset is made up of two fields: (1) user's comments/tweets and (2) the label of the comments (sarcastic/irony/figurative/regular). Figure 2 depicts the number of tweets in each label. We can see that the maximum number of tweets is in the figurative category and the regular class contains the minimum number of tweets. After that, we lemmatize the text corpus remove

4.2 Model Configuration

This segment illustrates the preprocessing and hyper-parameter settings that we have applied.

(A) **Preprocessing and Word Embeddings**

At first, we sliced the dataset into two separate training and testing sets in the ratio of 70:30. After splitting the dataset, we cleanse the data on the training set to get rid of irrelevant aspects from the tweets. The test set is used for validation purpose and kept as unknown raw data. We clean the data by deleting URLs, different punctuation symbols, white spaces, and numbers. We bring all the text corpus to lowercase alphabets. After that, we lemmatize the text corpus to remove inflectional endings in words and return words to its dictionary form. We then changed labels of the tweets to a numeric set using label binarizer so that each label has its numeric value. Finally, we apply pre-trained word embeddings, ELMo [19] that generates vectors bypassing text through the deep learning model. It analyzes the terms within the situation that they are applied. For our RNN model, we use Glove embeddings that produce an embedding matrix of vectors representation of words.

(B) Hyperparameters and Training Details

For our deep learning models, we apply a pre-trained ELMo model that consists of 93.6 million parameters, 2 highway layers, and LSTM hidden layer size of 4096 and output size of 512. In our ELMo-BiLSTM model, we have an input layer with input shape of 1, i.e., one sentence at a turn. We pass the ELMo embeddings with the help of lambda layer. Return from the embedding layer is transferred to a BiLSTM layer with weight of 1024. We use dense layers with hidden feature of 512 and 256 and with an activation function as 'ReLU.' We specify the dropout of 0.5. In the output layer, we used an activation function 'softmax.' For our glove-RNN model, we use an embedding layer on top with embedding matrix as weights. The hidden units of the dense layers are set to 100 and 32 with activation function as 'ReLU.' Activation function of the output layer is set as softmax.

Models are trained by using Adam optimizer [20], loss function equals to categorical cross-entropy and 'accuracy' as metrics. Batch size of 128 and the number of epoch are set to 6 to train our models. We did the training of the models on the training dataset, and behavior of the models is assessed on the test set after each epoch. Interpretation of the models is made with the help of precision, recall, and F-score values of the test data.

(C) Evaluation Metrics

In our analyses, precision, recall, and F-score are used to analyze the behavior of our model. Ratio between sentences that are correctly predicted and the total number of sentences is termed as precision. Ratio of the sentence that is predicted correctly to the actual number of correct sentences is called recall, and F-scores represent the means of precision and recall. These are calculated as follows

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

$$\text{F-score} = \frac{2(\text{Precision} \times \text{Recall})}{\text{Precision} + \text{Recall}}$$

where TP is the number of true prediction, FP is the number of sentences that are falsely predicted, FN is the amount of sentences that are predicted as false but actually true.

5 Results and Discussion

As discussed, we build deep learning models with deep contextualized word representation with the help of ELMo embeddings. These are already trained on a huge set of textual data. We add these embeddings to BiLSTM deep learning model. We have likewise built a RNN which uses Glove embeddings and is also a form of word embeddings, and the embedding matrix is passed into RNN through an embedding layer. We analyze our models on various metrics and found that performance of the BiLSTM model with ELMo embedding is quite satisfactory than that of the RNN model with Glove embedding. The BiLSTM model with ELMo embedding outperforms the RNN model with Glove embedding. We can see the data loss graph of BiLSTM model in Figure 3, as we can see that as the number of epoch is increasing, the data loss of both training and validation sets is decreasing and both ends at a value of 51.13% and 50.02%, respectively. As both the training and validation loss are decreasing and their difference is low, we can say that the model prevents overfitting. The graph in Figure 4 plots the accuracy score of ELMo-BiLSTM model for train and validation set. We observed that the accuracy of the train set increases and becomes constant after that at a value of 73.89%. The accuracy of the validation set, on the other hand, remains constant throughout all the epoch, and after the last epoch, the validation accuracy becomes 74.15%. An accuracy score of 73.68% was achieved for the Glove embedding-based RNN model on the train set, and a score of 73.71% was achieved on the validation set. Data loss in the training set is 50.60% and 50.11% in the validation set. From Table 1, we can say that on basis of the results obtained on the validation set, BiLSTM model with ELMo embedding on top exceeds the RNN model with Glove embedding.

The ELMo embedding models the complex characteristics of words and also how the uses of words can vary across different context. Vector produced are function of layers of bidirectional language model. Representations are character-based and use clues from robust representations for tokens that are not seen in training. Whereas, Glove produces global vectors. Using features that are hidden, the co-existence count

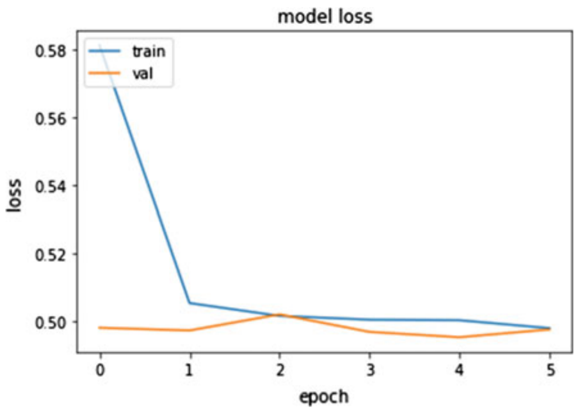


Fig. 3 Data loss graph of ELMo-BiLSTM model

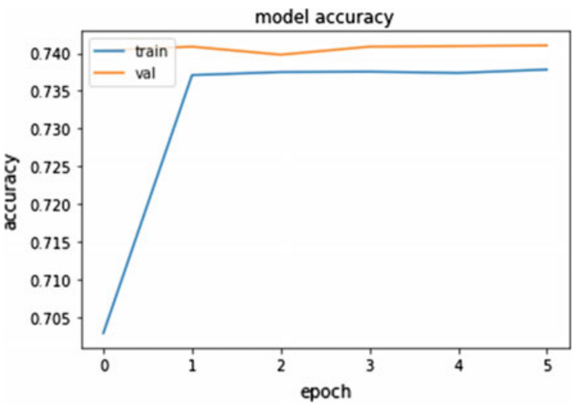


Fig. 4 Accuracy graph of ELMo-BiLSTM model

Table 1 Comparison table

Models	Accuracy (%)	Data loss (%)
RNN (Glove embedding)	73.71	50.11
ELMo-Bidirectional LSTM	74.15	50.02

is predicted by calculating the comparability between words. Glove embedding gives same vector representation for same word occurring in different sentences but ELMo word embedding takes the context in which the word is used in consideration and based on that gives different vector representation for different context. Glove cannot grasp the context of the word and fail to produce separate vector representation. This makes the ELMo embedding more effective than the Glove embedding.

6 Conclusion

Sarcasm is the unique form of satire. Using sarcastic, ironic messages, and post is increasing in social media platforms. Sarcasm is nowadays used to scrutinize or to express opinions about diverse issues related to politics, society, etc.

In this study, we have focused on the problem of detecting sarcasm using ELMo embedding. We used a Glove embedding-based RNN model and an ELMo embedding-based bidirectional long-short term memory (ELMo-BiLSTM) for detecting sarcasm from sentences. A public Twitter Sarcasm dataset for this purpose. We split the dataset into training and testing sets. Performed data preprocessing to clean the data and applied pre-trained word embeddings. Our ELMo-BiLSTM model incorporates an embedding layer that uses the ELMo word embeddings to generate vector representations of words. The ELMo vector representation of a word depends on the entire sentence and therefore gives different vector representations for the same word, summing up the contextual information of the comment. Glove embedding represents the co-existence of the word and thus gives the same vector representation for a word in a different context. Our results show that the ELMo-BILSTM surpasses the Glove-RNN model. It is observed that due to different vector representations of words according to the context in which they were used, ELMo embedding-based model performs better, and data loss decreased by 0.09% and accuracy of the model increased by 0.44% compared to the Glove embedding model. The difference in the input sizes and hidden units has also affected the performance of the models.

In the future, we would like to experiment with the ELMo embedding with other deep learning models. We would experiment with the model configuration and introduce different feature extraction techniques that can extract features related to sarcasm.

References

1. Gupta, R., Kumar, J., Agrawal, H., Kunal.: A statistical approach for sarcasm detection using twitter data. In: 2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS), pp 633–638 (2020)
2. Razali, M.S., Halin, A.A., Ye, L., Doraisamy, S., Norowi, N.M.: Sarcasm detection using deep learning with contextual features. IEEE Access **9**, 68609–68618 (2021)
3. Liu, L., Priestley, J.L., Zhou, Y., Ray, H.E., Han, M.: A2text-net: a novel deep neural network for sarcasm detection. In: 2019 IEEE First International Conference on Cognitive Machine Intelligence (CogMI), pp 118–126 (2019)
4. Bamman, D., Smith, N.A.: Contextualized sarcasm detection on Twitter. In: Ninth International AAAI Conference on Web and Social Media (2015)
5. Bouazizi, M., Otsuki Ohtsuki, T.: A pattern-based approach for sarcasm detection on Twitter. IEEE Access **4**:5477–5488 (2016)
6. Kreuz, R., Caucci, G.: Lexical influences on the perception of sarcasm. In: Proceedings of the Workshop on Computational Approaches to Figurative Language, pp 1–4 (2007)
7. Joshi, A., Sharma, V., Bhattacharyya, P.: Harnessing context incongruity for sarcasm detection. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (volume 2: short papers), pp 757–762 (2015)
8. Eisterhold, J., Attardo, S., Boxer, D.: Reactions to irony in discourse: evidence for the least disruption principle. J. Pragmat. **38**(8), 1239–1256 (2006)
9. Gonzalez-Ibanez, 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: Human Language Technologies, pp 581–586 (2011)
10. Hazarika, D., Poria, S., Gorantla, S., Cambria, E., Zimmermann, R., Mihalcea, R.: Cascade: Contextual Sarcasm Detection in Online Discussion Forums. arXiv preprint arXiv:1805.06413 (2018)
11. Kumar, A., Narapareddy, V.T., Aditya Srikanth, V., Malapati, A., Neti, L.B.M.: Sarcasm detection using multi-head attention based bidirectional lstm. IEEE Access **8**:6388–6397 (2020)
12. Zhang, M., Zhang, Y., Fu, G.: Tweet sarcasm detection using deep neural network. In: Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, pp 2449–2460 (2016)
13. Poria, S., Cambria, E., Hazarika, D., Vij, P.: A deeper look into sarcastic tweets using deep convolutional neural networks. arXiv preprint arXiv:1610.08815 (2016)
14. Ghosh, D., Fabbri, A.R., Muresan, S.: The role of conversation context for sarcasm detection in online interactions. CoRR abs/1707.06226 (2017)

15. 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, pp 704–714 (2013)
16. Amir, S., Wallace, B.C., Lyu, H., Carvalho, P., Silva, M.J.: Modelling context with user embeddings for sarcasm detection in social media. CoRR abs/1607.00976 (2016)
17. Cai, Y., Cai, H., Wan, X.: Multi-modal sarcasm detection in twitter with hierarchical fusion model. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp 2506–2515 (2019)
18. Mishra, A., Kanojia, D., Nagar, S., Dey, K., Bhattacharyya, P.: Harnessing cognitive features for sarcasm detection. CoRR abs/1701.05574 (2017)
19. Peters, M.E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., Zettlemoyer, L.: Deep contextualized word representations. In: Proceedings of NAACL (2018)
20. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization, arXiv preprint arXiv:1412.6980 (2014)