Sarcasm Discernment on Social Media Platform

Namasani Sagarika, *Bommadi* Sreenija Reddy, *Vanka* Varshitha, *Kodavati* Geetanjali, *N V* Ganapathi Raju*, *Latha* Kunaparaju
*Department of Information and Technology, Gokaraju Rangaraju Institution of Engineering and Technology, Telangana, India

Abstract. Past studies in Sarcasm Detection mostly make use of Twitter datasets collected using hashtag-based supervision but such datasets are noisy in terms of labels and language. To overcome the limitations related to noise in Twitter datasets, this News Headlines dataset for Sarcasm Detection is collected from two news website. TheOnion aims at producing sarcastic versions of current events and we collected all the headlines from News in Brief and News in Photos categories (which are sarcastic). We collect real (and non-sarcastic) news headlines from Huff Post. Sarcasm Detection on social media platform. The dataset is collected from two news websites, theonion.com and huffingtonpost.com. Since news headlines are written by professionals in a formal manner, there are no spelling mistakes and informal usage. This reduces the sparsity and also increases the chance of finding pre-trained embeddings. Furthermore, since the sole purpose of TheOnion is to publish sarcastic news, we get high-quality labels with much less noise as compared to Twitter datasets. Unlike tweets that reply to other tweets, the news headlines obtained are self-contained.

1. Introduction

The modern world can be described as a data-driven environment. The amount of data generated by a single network device has increased exponentially. The amount of data transmitted is vast due to the large number of devices linked to the internet. The information that is shared and distributed has a significant impact on people's lives [1] .

The Research is about caustic remarks, assertions, and pronouncements found in articles/posts on social media sites, as the title suggests. Sarcasm is a common linguistic phenomenon in online publications that expresses personal and highly felt feelings.

People have been known to utilize social media to propagate false information, spread rumors, and make meaningless interpretations of events. People start this problem by posting material that can mislead people or lead to misunderstandings regarding specific topics. Many NLP applications, such as attitude analysis and opiate detection, benefit from detecting sarcasm [2].

Automatic sarcasm detection is currently viewed as a straightforward text classification problem in current research. They ignore the imbalance between sarcastic and non-sarcastic samples in real applications and do not employ explicit features to detect sarcasm.

Based on data sets, we assess our proposed model. Our ensemble technique beats state-of-the-art sarcasm detection methods and popular unbalanced classification methods, according to experimental results. We start by looking at the properties of sarcastic sentences and then present a collection of features that can be used to detect sarcasm in social media. This sarcasm detection on social media platforms is collected from two news websites: THEONION and HUFFPOST, to overcome the constraints of noise in Twitter datasets. TheOnion's goal is to create satirical renditions of current events, therefore we gathered all of the headlines from the News in Brief and News in Photos sections (which are sarcastic). HuffPost provides us with real (non-sarcastic) news headlines.
The final output is obtained once all the detecting steps have been completed. Whether or if the information is ironic. Each record has three characteristics: is sarcastic: 1 if the record is sardonic, 0 if not. article link: a link to the original news article, headline: the title of the news article. It's useful for gathering further information. With the help of the following initiative, we can be more cautious about what information we find online and whether to believe it. There might be a lot of confusion between the facts and rumors if people believe erroneous information.

2. Literature Survey.

* Corresponding author: nvgraju@griet.ac.in

Many researchers have explored and tested the use of many sorts of features in text datasets, as well as various approaches for detecting sarcasm. The two types of sarcasm detection algorithms are supervised and semi-supervised, respectively. Other approaches for detecting sarcasm, such as deep learning and bootstrapping, are also becoming more popular.

To identify sarcasm in Twitter and Amazon product evaluations, Davidov et al. used the semi-supervised sarcasm identification algorithm (SASI). This is the first algorithm for detecting sarcasm that has been proven to be reliable. In reviews and tweets, they employed pattern- and punctuation-based features [9] .

Lukin and Walker, on the other hand, used the Bootstrapping approach to detect sarcasm and unpleasantness in online conversations using pattern-based characteristics [11,13].

Bouazizi and Ohtsuki proposed a pattern-based technique for detecting sarcasm, which included four features: sentiment-related, punctuation-related, pattern-related, syntactic, and semantic [12] .

González-Ibáez et al. looked at lexical and pragmatic characteristics in tweets that were retrieved using unigrams and dictionary-based methods for categorizing sarcastic, positive, and negative tweets using two classifiers: Support vector machine (SVM) and Logistic regression (LogR) [14].

Fersini et al. proposed Bayesian Model Averaging (BMA) as an ensemble technique for detecting sarcasm and irony in Microblogs, taking into consideration features such as pragmatic particles and PoS tags [15]. Barbieri et al. used a set of criteria to determine whether tweets were sarcastic or not, including Frequency, Written-spoken, Intensity, Structure, Sentiments, Synonyms, and Ambiguity [3-5].

They utilized a cutting-edge computational method based on supervised machine learning techniques. Novel multi-strategy ensemble learning method (MSELA) was also suggested by Liu et al. to identify sarcasm in both English and Chinese social media. They retrieved several feature sets for both English and Chinese texts, with English sarcasm features consisting of punctuation symbols, lexical and syntactic characteristics, and Chinese sarcasm features consisting of rhetorical, homophony, and construction aspects [7,8].

Sarcasm as a contrast between a good feeling and a terrible circumstance was studied by Riloffet al. They introduced a new bootstrapping technique that uses SVM to generate lists of positive and negative scenario words from sarcastic tweets [9-10].

Zafarani et al. developed a behavioral modelling framework for detecting sarcasm based on a set of characteristics that differed depending on the type of sarcasm [6]

Bharti et al. developed the parsing-based lexicon generation algorithm (PBLGA) in 2015 to identify sarcasm in tweets using a hyperbole feature and a natural language processing technique: PoS Tagging [13].

Ghosh et al. reframed the sarcasm recognition problem as a Literal/Sarcastic sense disambiguation (LSSD). They used unsupervised techniques and an SVM classifier with a modified kernel utilizing word embedding to analyse twitter data. Bharti et al. created a Hadoop-based system, a big data strategy, to identify sarcasm in real time, based on parsing and PoS tagging [5,15-19] .

3. Methodology.

In this paper, we will try to detect sarcasm using NLP and machine learning, taking the support of the deep learning. Being a software code, it's base coding language is python because of the efficiency and advantages that it provides to the programmers. Since this program focuses on detecting the sarcasm, the headlines of two magazines are taken as datasets, that is, TheOnion and HuffPost. Since the TheOnion aims for only sarcastic news, it is the best choice while HuffPost focuses on wide-ranging news.
In the past, many people tried to detect sarcasm using twitter tweets using hashtag-based, but these datasets are usually noisy, because most of the tweets are the reply to the other tweets. These makes many things difficult in terms of labels and language. Since the news headlines of the magazines are written by professional, we cannot expect any use of informal language and spelling or grammatical errors, resulting in reduction of pre-trained sparsity and increase of pre-trained embeddings.
Steps for detecting sarcasm is as follows:

1. Set-up the environment
   In this step, all the necessary python libraries are loaded to give us the perfect atmosphere for coding. These include pandas, NumPy, scikit-learn, matplotlib that are famous for dealing with datasets in python; keras which helps to introduce deep learning in python; and lastly, TensorFlow, a machine learning library that provides interface to the deep learning algorithms.
2. Load the Data
   In this step, the required data sets are loaded into the code, on which the code runs in order to yield the desired output. In this, the loaded data acts as the input to be processed by the code to detect sarcasm, to give us the required answer.
   - Load the json file: the json file is loaded into the code
   - Convert the dictionary to data frame: the extracted data from the json file is converted into data frames, i.e., in form of

the table, for the convenience of code so that it can move smoothly.

- Print few sample rows: few rows are printed to see whether we succeeded in making the dataframes. The output will contain three columns: article_link, headlines and is_sarcastic. Headlines and article_link are features whereas is_sarcastic is the label. Article_link will contain the link of the website from where the headline is taken, headline will contain the desired input while is_sarcastic will give us the output about whether the particular headline is sarcastic or not. If it is '1' then that means the headline is sarcastic and if '0' then it means it is not sarcastic.

3. Drop article_link from dataset
   Since our main focus will be headlines, article_link is unnecessary, so it is better to drop this column. This will help in increase in time and space complexity of the code.
   - Print few rows: it is better to check the output to see whether the required column is dropped or not. In this step, output should only show headline and is_sarcastic column.
4. Get length of each headline and add that to dataset
   In this we get the length of each headline, in other word, the size of each input, and add it as a new column in the dataset. It will help the code to understand the size of the data, thus increasing its efficiency.
   - Print few rows: in this step, the output is checked. The output must contain three columns, headline, is_sarcastic and headline_len.
5. Initialize parameters
   This step initializes three parameters. They are: max_feature, maxlen and embedding_size. Max_feature is the number of the most frequent words taken from tokenizer which is limited to only 10000 words. Maxlen is the length of each sentence which is restricted to 25. And, embedding_size is the size of embedding vector which is limited to 200.
6. Apply TensorFlow. Keras tokenizer and get indices of words.
   The tokenizer is declared in this step. The tokenizer is a TensorFlow. Keras tokenizer with word limit of 10000. Fit this tokenized into the headline column so that is can convert the headlines into the required tokens. Then, convert the text to sequence through padding.
   - Padding sequence: it is the process of padding each example to maximum length. The targeted column is then converted to NumPy array.

   There is no need for oversampling or under sampling because the data is reasonably balanced in terms of sarcastic and non-sarcastic labelling.
7. Vocab mapping
   In this, each token broken from the dataset is assigned with a value in ascending order. There is no word at the 0^{th} index.
8. Create embedding matrix:
   A collection of all words and their associated embeddings is called an embedding matrix. An embedding matrix is created based on tokens that is used in creating a model.
9. Define model
   Add an Embedding layer, a Bi-directional (LSTM) layer, flatten it, and then dense and dropout layers as needed using a Sequential model instance. Finally, for binary classification, create a final thick layer with sigmoid activation. We did not train the embeddings created from glove data because we were able to achieve decent results without it.
10. Compile the model
    Compile the model using the adam optimizer in order to get its accuracy. The binary_crossentropy is declared in order to compares the predicted output and actual output in the form of '0' and '1'.
11. Fit the model
    The model is fit into the selected batch size, which is 64.
12. Plot the model graph and the loss cross epochs
    A model graph, which is the accuracy model is printed to give us the output in the form of the graphs. The accuracy model is a machine learning model is the metric used to evaluate which model is the most effective in identifying correlations and patterns between variables in a dataset based on the input, or training, data. Epoch concludes when it has finished training all the observations in dataset while its loss is a scalar value that we try to minimize during model training. The lesser the loss, the more accurate our projections are.
13. Confusion matrix
    A confusion matrix is a table that shows how well a classification model (or "classifier") performs on a set of test data for which the real values are known. It has four basic terms, which actually gives whole number:

- True positive (TP): these are the cases in which we predicted a sentence to be sarcastic and is actually sarcastic.
- True negative (TN): when we predict a sentence to be not sarcastic and it is actually not sarcastic.
- False positive (FP): the situation in which we predict a sentence to be sarcastic when it is not a sarcastic sentence.
- False negative (FN): these represent those circumstances when we predicted a non-sarcastic as a sarcastic sentence.

4. Architecture.

A system architecture is a conceptual model that specifies a system's structure, behavior, and perspectives. An architecture description is a description and representation of a system that is organized in such a way that it allows for reasoning about the system's structure and behavior.
The datasets for detecting sarcasm are collected from the news headlines of two magazines, namely, TheOnion, and HuffPost that has general news. Data extraction is the process of converting the json dataset files into data frames. In this process feature extraction occurs. Later the headlines are broken into tokens by the process of tokenization. The sequence model is created on the embedding layer and bidirectional layer. For the binary classification, dense layer with sigmoid activation is used. As the output, the Accuracy model is created and the test accuracy is shown.

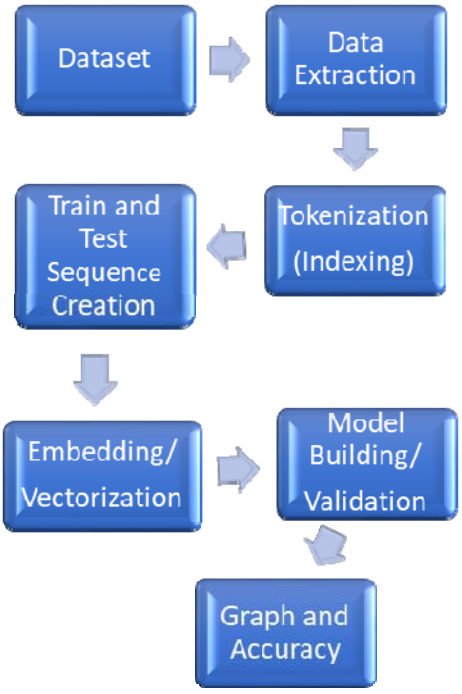

Fig. 1. Architecture of Sarcasm Discernment

5. Implementation and Result Analysis.

In this section, we'll go over a deep analysis of the modules that are involved in the implementation of this research.

Cleaning the data:

- Although, it looks that our data is clean but the headline column has some special symbols that have to be eliminated.
- So, we are using Regular Expression to eliminate special symbols.

Feature and label extraction:

- It's time to take our data and extract characteristics and labels.
- It appears that article link and headline could be considered features.
- Article link, on the other hand, has no bearing on the label prediction.
- So, the headline column is the only feature we have. The only label is sarcastic.

Stemming of features:

- The process of reducing a word to its word stem, which affixes to suffixes and prefixes or the roots of words known as a lemma, is known as stemming.
- Natural language understanding (NLU) and natural language processing (NLP) both benefit from stemming (NLP).

Vectorization of features using TF-IDF Vectorizer:

- TF-IDF is an abbreviation for Term Frequency-Inverse Document Frequency and is a very common algorithm to transform the text into a meaningful representation of numbers.
- The technique is widely used to extract features across various NLP applications.

Model accuracy and loss across epochs

```
# summarize history for accuracy
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train', 'validation'], loc='upper left')
plt.show();
```

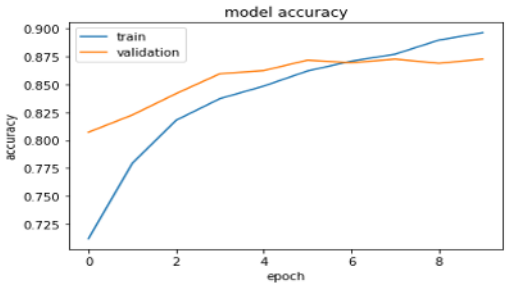

Fig. 2. Model accuracy across epochs

```
# summarize history for loss
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'validation'], loc='upper left')
plt.show();
```

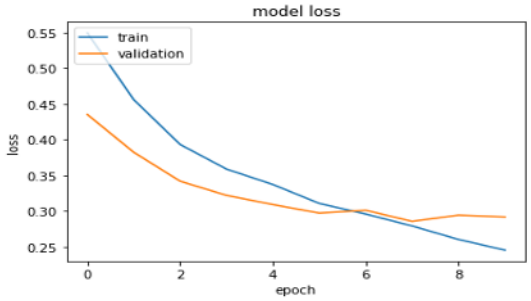

Fig. 3. Model loss across epochs

Accuracy**:**

```
In [28]: loss, accuracy = model.evaluate(X_test, Y_test)
         167/167 [==============================] - 15s 88ms/step - loss: 0.2864 - accuracy: 0.8779
In [30]: print(f'My test loss is {loss*100:.2f}% and test accuracy is {accuracy*100:.2f}%')
         My test loss is 28.64% and test accuracy is 87.79%
```

Fig. 4. Model accuracy across epochs

Prediction**:**

	Article_link	Headline	Is_sarcastic
0	https://www.huffingtonpost.com/entry/versace-b...	former versace store clerk sues over secret 'b...	0
1	https://www.huffingtonpost.com/entry/roseanne-...	the 'roseanne' revival catches up to our thorn...	0
2	https://local.theonion.com/mom-starting-to-fea...	mom starting to fear son's web series closest ...	1
3	https://politics.theonion.com/boehner-just-wan...	boehner just wants wife to listen, not come up...	1
4	https://www.huffingtonpost.com/entry/jk-rowlin...	j.k. rowling wishes snape happy birthday in th...	0

Fig. 5. Model prediction

6. Conclusion.

The detection of sarcasm has piqued the interest of many Natural Language Processing experts. In the previous ten years, it has made enormous strides. As a result of recent developments in deep learning and its open knowledge base, researchers are more swiftly adopting deep learning techniques and their usage in sarcasm detection. Feature augmentation using word embedding and its application in word embedding has enhanced sarcasm recognition. Researchers are also trying to figure out how to include context into the learning process. Even people can't always tell when someone is being sarcastic. As a result, it appears that the problem remains unsolvable. Its randomness is still challenging to replicate in machines.

Following the findings, it is evident that context will play an important role in detecting sarcasm. Furthermore, much research has been done on the importance of context in detecting sarcasm. The speaker's and listener's knowledge base in the context of the subject is crucial for sarcasm recognition. Even for humans, sarcasm is impossible to discern without it. We could look into a number of things.

Assume we have a central repository that stores information on users, cultures, and the global context. In a variety of situations, these parameters are crucial. Consider the following sentence: "It's sunny outside, and I'm at my desk." It might be snarky if the user is American, but for an Indian user who isn't sarcastic, we'd know that an American would find it hysterical.

Furthermore, we could discern whether the speaker is an American or an Indian if we knew user-specific information. Then we'd know if the individual was being sarcastic or not. Let's imagine we have a statement where understanding the inferred emotion rather than the expressed one necessitates knowledge of a universal fact. As a result, having a global context would aid us in determining the implicit emotive content.

7. Bibliography.

1. McKinney, Wes. *Python for data analysis,* O'Reilly Media, Inc.", (2012).
2. Bird, Steven, Ewan Klein, and Edward Loper. *Natural language processing with Python*. " O'Reilly Media, Inc.", (2009).
3. Li Deng, Yang Liu, *Deep Learning in Natural Language Processing,* publisher: Springer
4. Aurélien Géron, *Hands-On Machine Learning with Scikit-Learn and TensorFlow*: *Concepts*, *Tools*, *and Techniques to Build Intelligent Systems* 1st Edition
5. Ghosh, Debanjan, Weiwei Guo, and Smaranda Muresan. "Sarcastic or not: Word embeddings to predict the literal or sarcastic meaning of words." *proceedings of the 2015 conference on empirical methods in natural language processing*. (2015).
6. Rajadesingan, Ashwin, Reza Zafarani, and Huan Liu. "Sarcasm detection on twitter: A behavioral modeling approach." *Proceedings of the eighth ACM international conference on web search and data mining*. (2015).
7.Liu, Peng, et al. "Sarcasm detection in social media based on imbalanced classification." *International Conference on Web-Age Information Management*. Springer, Cham, (2014).
8. Xue, Bai, Chen Fu, and Zhan Shaobin. "A study on sentiment computing and classification of sina weibo with word2vec." (2014).
9. Davidov, Dmitry, Oren Tsur, and Ari Rappoport. "SSRA." *Proceedings of the fourteenth conference on computational natural language learning*. (2010).
10. Berkowitz, Dan, and David Asa Schwartz. "Miley, CNN and The Onion"**10.1** (2016).
11. Lukin, Stephanie M., et al. "Argument strength is in the eye of the beholder: Audience effects in persuasion." *arXiv preprint arXiv:**1708.09085*** (2017).
12. Bouazizi, Mondher, and Tomoaki Otsuki Ohtsuki. "Pattern based approach." *IEEE Access* 4 (2016).
13. Bharti, Santosh Kumar, Korra Sathya Babu, and Sanjay Kumar Jena. Sentiment recognition in twitter data. *2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*. IEEE, (2015).
14. González-Ibánez, Roberto, Smaranda Muresan, and Nina Wacholder. "Identifying sarcasm in twitter: a closer look." Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies. (2011).
15. Pozzi, Federico Alberto, Elisabetta Fersini, and Enza Messina. "Bayesian model averaging and model selection for polarity classification." International Conference on Application of Natural Language to Information Systems. Springer, Berlin, Heidelberg, (2013).

16. Dhanalaxmi, B., Apparao Naidu, G., Anuradha, K. Adaptive PSO based association rule mining technique for software defect classification using ANN, Procedia Computer Science (2015)
17. Kora, P., Kalva, S.R. Hybrid Bacterial Foraging and Particle Swarm Optimization for detecting Bundle Branch, Block,SpringerPlus (2015)
18. Kumar, P., Singhal, A., Mehta, S., Mittal, A. Real-time moving object detection algorithm on high-resolution videos using GPUs, Journal of Real-Time Image Processing (2016)
19. Raju, NV Ganapathi, Machine learning based power saving mechanism for fridge: An experimental study using GISMO III board.*Materials Today: Proceedings* (2020)