Sarcasm Detection

1st Madhusmita Kalita
Department of Computer Science
Lakehead University
Thunder Bay, Canada
mkalita@lakeheadu.ca

2nd Mina Armanyous
Department of Computer Science
Lakehead University
Thunder Bay, Canada
mmakram@lakeheadu.ca

3rd Farah Shahata
Department of Computer Science
Lakehead University
Thunder Bay, Canada
fshahata@lakeheadu.ca

4th Prerak Patel
Department of Computer Science
Lakehead University
Thunder Bay, Canada
ppatel74@lakeheadu.ca

***Abstract*—Recent years have seen a huge growth in the use of social media platforms by the people to voice their opinion about a variety of topics, which may lead to comments being ambiguous. For example, sometimes some comments might be misunderstood and may lead to conflict between the reader and the writer. A similar kind of pattern has been seen when it comes to Newspaper headlines or any comment or post on social media. We come across a lot of writing materials throughout the day and while some of them are very straightforward, some can be sarcastic. This may seem offensive to certain groups of readers and result in misunderstanding the subtle nature of humor added to the headlines. Therefore, there is a need to separate the headlines or comments or any writing in general, based on their nature as sarcastic or non-sarcastic. This Project work helps to address this problem by predicting headlines to be sarcastic or not. We applied different Machine learning models and performed a comparative analysis between them. We found that the Long short-term memory (LSTM) and convolutional neural network(CNN) gave almost similar result with 82% accuracy.**

***Index Terms*—Sarcasm, Sentiment analysis, Machine learning,Natural language processing, non-sarcastic, sarcastic.**

I. INTRODUCTION

Users can access a great number of articles about every possible topic on the internet. While the number of articles increases with time, so does the number of websites and sources that publish those articles. At some point, users will encounter unknown sources and must decide whether an article's content is meant in a straightforward way or to be sarcastic. Especially in the case of non-native speakers, it can be difficult to determine whether the actual meaning of a headline is meant to be sarcastic or non-sarcastic. Since there are no well-defined rules for determining if a sentence is sarcastic or not, the user must rely on their experience to evaluate a headline. Another problem is the lack of transparency for many online services. A Sarcastic detection system as proposed in this paper can have a great impact on the user's opinion towards e.g. political topics and introduces the risk of manipulation. This risk can be reduced when the system discloses how the classification algorithms work, allowing it to be reviewed and thoroughly tested by the user. To address this problem, we have explored several research papers that focus on the binary classification problem of Machine Learning. In order to train our Machine learning model for solving the problem, we have leveraged the dataset of news headlines available on 'Kaggle' which comprised of the newspaper headlines from 2 news websites – The HuffPost and the Onion. The dataset consists of 3 fields - 'article link', 'headline' and class 'is sarcastic'. Since the dataset is textual and the Machine Learning model needs to be trained using numerical data, we have employed various Natural Language Processing (NLP) techniques in the data cleaning and preparation phase. In order to propose an optimal solution to the users, we evaluated six different classification algorithms viz. Logistic Regression, Random Forest, Decision Tree, Support Vector Machines (SVM), Long Short-Term Memory (LSTM) and Convolutional Neural Network (CNN)

II. LITERATURE REVIEW

Numerous researchers have performed and investigated the use of different types of features in the text dataset along with different methods for sarcasm detection. Generally, sarcasm detection methods are categorized into two types: supervised and semi-supervised. Also, other methods are emerging for sarcasm detection such as Deep learning, Bootstrapping etc.

According to Haripriya et al.the process of Sentiment Analysis tends to understand these opinions and categorize them into positive, negative and neutral. Sarcasm is use of a complicated structure of language that recognizes a gap between the expected meaning and the true meaning of the words. With this ambiguity, sarcasm detection is a challenging task even for individuals. Sarcasm detection in writing is a trivial challenging task due to a lack of articulation and facial expressions. Several approaches are there to do Sentiment analysis for sarcasm detection. By increase in the number of people voicing their opinions on twitter by comments or shared opinions has generated a huge data set on this writing. There are many techniques that have been developed to conduct sentiment analysis but still the problem of cynicism is unresolved. Their paper introduces various methodologies used to detect ridicule in social media data on Twitter, and has analyzed various compilations such as Support Vector Machine, Naıve Bayes and Lexicon based on accuracy rate. Sarcasm can be determined efficiently only if the existing approaches can handle large dataset but most of the existing approaches can handle only small datasets. They concluded that deep learning approach is considered as an efficient technique to detect Sarcasm in the case of large datasets [1].

Davidov et al. implemented the semi-supervised sarcasm identification algorithm (SASI) to detect sarcasm in Twitter and Amazon product reviews [8]. This is the first robust algorithm used for sarcasm detection. They used pattern-based and punctuation-based features in reviews and tweets. Whereas Lukin and Walker implemented the Bootstrapping method to identify sarcasm and nastiness in online dialogue using pattern-based features [9].

Bouazizi and Ohtsuki introduced a pattern-based approach using four sets of features to detect sarcasm viz. sentiment related, punctuation related, pattern related, syntactic and semantic [10]. Gonzalez-Ib´ a´nez et al. studied lexical and pragmatic features in tweets, extracted using unigrams and dictionary-based for classifying sarcastic positive and negative tweets by employing two classifiers: Support vector machine (SVM) and Logistic regression (LogR) [11]. Paper done by Pta´cek et al. presents supervised machine learning methods for sarcasm detection on Twitter as the first attempt at sarcasm detection in the Czech language, in which they focus on supervised machine learning approaches and evaluate their performance. They selected various n-grams, including unigrams, bigrams, trigrams with a frequency greater than three and a set of language-independent features, including punctuation marks, emoticons, quotes, capitalized words, character n-grams and skip-grams as their baselines [7].

The architectures based on deep learning techniques gain popularity in Natural Language Programming (NLP) applications, a few such approaches have been reported for automatic sarcasm detection as well. According to Liu et al. they use the similarity between word embeddings as features for sarcasm detection. They augment these word embedding-based features with features from four prior works. The inclusion of past features is the key because they observed that using the new features alone does not suffice for good performance [3].

In paper done by Amir et al. presents a novel Convolutional Network-based architecture that learns user embeddings in addition to utterance-based embeddings. The authors state that it allows them to learn the user-specific context [2], [4]. According to Ghosh et al. they use a combination of a Convolutional Neural Network, a Recurrent Neural Network (Long ShortTerm Memory) followed by a Deep Neural Network. They compare their approach against recursive SVM and show an improvement in the deep learning architecture [5]. In the paper done by Poria et al. they investigate the use of Deep Convolutional Networks for sarcasm detection and develop models based on a pre-trained convolutional neural network for extracting sentiment, emotion and personality features for sarcasm detection [6].

III. DATA DESCRIPTION

The Purpose of identifying sarcasm in any sentence is a very hard and complex task and which also requires a very good type of dataset. As most of the datasets available are the data collected from the twitter website and some of the data is the sarcastic reply of people to any user's tweet, the data lags contextual information. Contextual information plays a very important role in sarcasm identification. The selected dataset for this project has not been collected from any social media website rather it has been collected from two news websites. One for collecting sarcastic sentences and one for normal non-sarcastic sentences. "The Onion" news website, is famous for its sarcastic articles on current world news. Also, the headings of these articles are very sarcastic and written by professional editors. So, our data does not have any spelling mistake and as the audience of these articles is the world public, the language used is more formal and hence reduces the work of identifying the informal meaning of any word in our use. The collected data has been taken from two sections of this website, that is, news in brief and news in photo categories. The non-sarcastic sentences or data is collected from the second website known as "Huff Post". The website provided the non-sarcastic data in the dataset. The dataset has three features and a total of 26709 samples.Below are the features:

- is_sarcastic: 1 if the record is sarcastic otherwise 0
- headline: the headline of the news article
- article_link: link to the original news article. Useful in collecting supplementary data

We used Word cloud in order to understand and visualize how many times a specific word appears in the entire source of textual data. The bigger a word's font, the more important the word is.Figure.1 shows the word cloud representation of "Headline"

IV. DATA PREPROCESSING

Data pre-processing is one of the most important phases in Machine learning. Our dataset comprises of duplicate values that needed to be cleared off for future analysis. To convert the textual data into a Machine learning model readable format, we applied the following Natural Language Processing(NLP) techniques from the Natural Language Toolkit(NLTK) library in Python.

A. Data Preparation

- Tokenization: This process helps us in breaking a sentence into a set of words. We obtained individual tokens using NLTK's "word tokenize()" method.
- Lemmatization: This is the process of grouping together the different inflected forms of a word so they can be analyzed as a single item. It helps us achieve a morphological analysis of the words. We used WordNetLemmatizer() function to perform this.

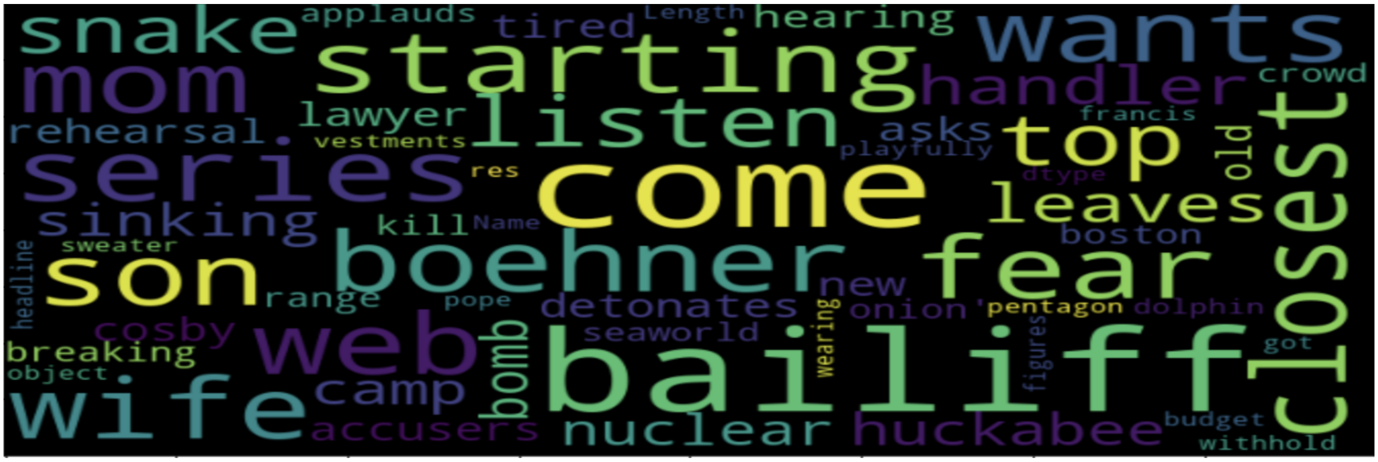

Fig. 1. Word Cloud representation.

- Stopwords removal: A stop word is a commonly used word (such as a, an, the, in)which does not contribute much towards sarcasm detection based on textual data. The Stop words have been eradicated from our headline field using NLTK's StopWords for English.

After processing, the words were re-joined to form the sentence.

B. Feature Extraction

In Textual Analytics, each word of the text embodies a discrete feature, which is categorical in nature. It is very important to convert these data into real-valued vectors so it can be used by the Machine Learning algorithms. For this mapping, we have used the below feature extraction technique.

- Term Frequency Inverted Document Frequency (TF-IDF):

$$TF - IDF = TF * IDF \tag{1}$$

  TF = (Number of times term 't' appears in a Document)/(Number of Terms in the Document)
  IDF = log(N/n), where N is number of documents and n is the number of documents that has term 't' appeared in it. We have leveraged inbuilt "TfidfVectorizer" library to extract our features, that are in the form of a sparse matrix. The vectorizer generates 5000 features for each 'headline'.
- Latent Semantic Analysis (Truncated SVD):
  We further used Truncated SVD as a Dimensionality reduction technique. This transformer uses truncated singular value decomposition in order to perform linear dimensionality reduction. This method has been used on contrary to PCA as it performs more effectively with sparse matrices.
  We use "TruncatedSVD" inbuilt function to derive 500 features from a total of 5000. The derived 500 features are then used for future Machine learning model training phase.

V. Methodologies

1) Random Forest: Random forest, like its name implies, consists of many individual decision trees that operate as an ensemble. Each tree in the random forest spits out a class prediction and the class with the most votes becomes our model's prediction.

2) Decision tree: A decision tree is a decision support tool that uses a tree-like model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. A decision tree is a flowchart-like structure in which each internal node represents a "test" on an attribute, each branch represents the outcome of the test, and each leaf node represents a class label. The paths from the root to the leaf represent classification rules. We calculated Accuracy, Precision, Recall, f1 score to evaluate our performance.

3) Logistic Regression: Logistic Regression is a 'Statistical Learning' technique and one of the most popular ways to fit models for categorical data, especially for binary response data in Data Modeling. It is the most important (and probably most used) member of a class of models called generalized linear models. Logistics regression majorly makes predictions to handle problems which require a probability estimate as output, in the form of 0/1. Logistic regression is an extremely efficient mechanism for calculating probabilities.

4) Support Vector Machine: The objective of the support vector machine algorithm is to find a hyperplane in N-dimensional space (N — the number of features) that distinctly classifies the data points. Hyperplanes are decision boundaries that help classify the data points. Data points falling on either side of the hyperplane can be attributed to different classes.

5) Neural networks: Neural networks are a set of algorithms, designed on the basis of the human brain, it was invented to identify patterns. The patterns they recognize are numerical, included in vectors, where all real-world data must be translated, be it pictures, sound, text or time series. Neural Networks helps us to assemble and classification.

6) Long short-term memory: Long short-term memory (LSTM) is a special kind of recurrent neural network(RNN). It is capable of learning long-term dependency. This architecture used in many fields of deep learning but unlike standard feed-forward neural networks, LSTM has feedback connections. It can not only process single data points (such as images), but also entire sequences of data (such as speech or video). Figure.2 shows our LSTM-RNN model.

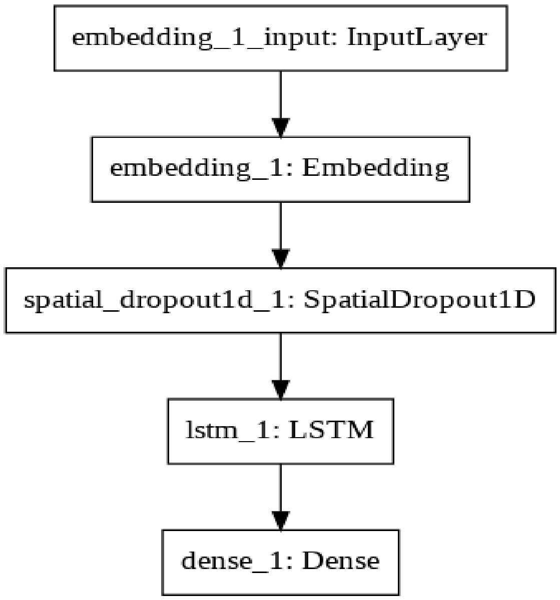

Fig. 2. Long short-term memory Model.

7) Convolutional Neural Network: Convolutional neural network (CNN) is a neural-based approach represents a function feature is applied to form words or n-grams to extract features top level. The use of abstract features resulting effectively ridicule detection and sentiment analysis and automatic translation and answer questions, among other tasks. To use Convolutional neural network (CNN) for classifying texts, we transform words into a vector representation via a look-up table, which results in a primitive word embedding approach that learns weights during the training of the network.

Fig.3 shows our CNN model.

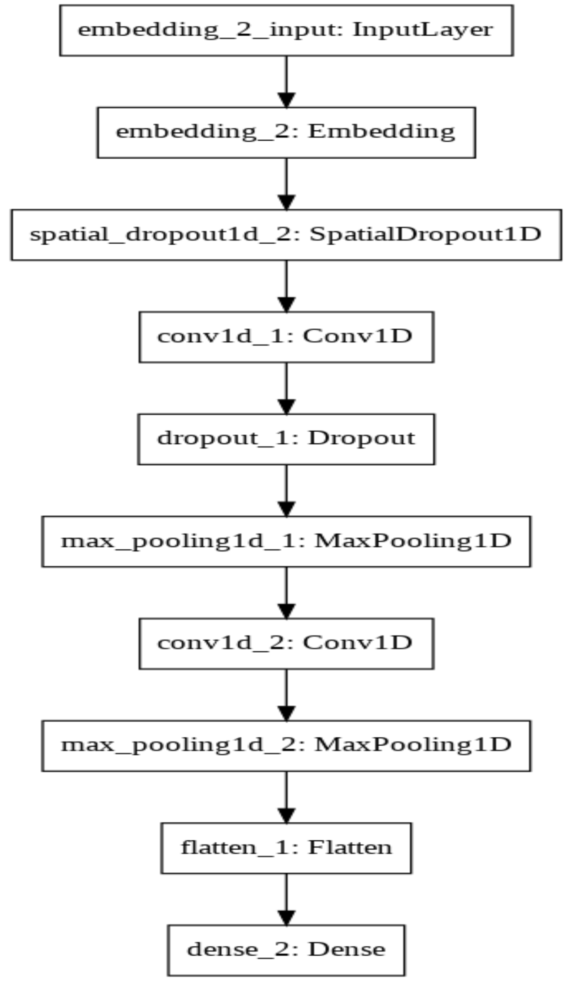

Fig. 3. Convolutional Neural Network Model.

TABLE I
RESULTS FOR DIFFERENT MODELS

Model	Accuracy	Precision	Recall	F1 Score
Random Forest	78	77.88	77.15	77.39
Decision Tree	64.71	64.13	64.12	64.13
Logistic Regression	80.55	80.24	80.2	80.22
SVM	80.4	80.08	80.13	80.1
Neural Network	56.31	28.15	50	36.02

VI. EXPERIMENTAL AND COMPARATIVE ANALYSIS

The dataset is divided into training and validation sets using an 80:20 split ratio. In the first stage of research, several classification algorithms such as Logistic Regression, Random Forest, Decision Tree, Support Vector Machines (SVM) and Neural Networks are used. We acquired the best results from Logistic Regression which had an accuracy of 80.55% and f1 score of 80.22%, followed by SVM with the accuracy of 80.4%

For evaluating the model, we used a set of 4 different metrics. Accuracy, Precision, Recall and f1 scores are compared between different classifiers. Table I shows this comprising.

• The accuracy shows the percent of correctly predicted values.

• The precision is the percentage of rightly predicted values out of total for a given class.

• The recall contains the percentage of predicted values for a class out of total number of values it should have predicted.

• The f1 is the harmonic mean of precision and recall which sums up the total correctly predicted values.

Each of these evaluation techniques is important as one single method cannot decide the performance of a classifier. The reason for having a combination is to not judge a model by its accuracy only because at times although the accuracy may be high, it does not necessarily mean that the predictive power will be higher. This is commonly referred to as 'Accuracy Paradox'.

After performing some experiments using LSTM and CNN and as shown in the table II, Both models could provide a

TABLE II
LSTM AND CNN RESULTS

Model	Accuracy	Loss
CNN	0.8283	0.5310
LSTM	0.8236	0.5357

TABLE III
LSTM MODEL TESTING

Text	Prediction
I am hungry	Non-sarcastic
I am sleepy	Non-sarcastic
I am tired	Non-sarcastic
Drunk driver in the zone	sarcastic
Mody	sarcastic
Bazinga	sarcastic

better performance than the traditional methods. For LSTM, it achieved a testing accuracy of 82.36% and a validation loss of 53.57%.

Figure.4 shows the progress of accuracy during a 10 epochs of training.

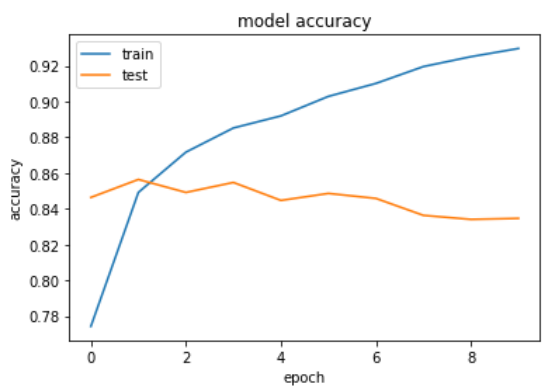

Fig. 4. LSTM model accuracy

Table III shows how LSTM model could classify some samples of texts based on the sarcasm status after being trained against the dataset used in our experiments.

CNN achieved a bit higher testing accuracy with 82.83% and with 53.10 loss. Figure. 5 shows the progress of accuracy during a 10 epochs of training.

VII. CONCLUSION

Research in sarcasm detection has grown significantly in the past few years. It is the emerging field in data mining which requires much deeper insight. Our report explains the details of our project which is capable of predicting headlines to be sarcastic or not. This contributes toward a better understanding of sarcasm for non-native speakers and works against the risk that sarcastic newspaper articles are believed to be true. Additionally, in this project we applied different Machine learning models and performed a comparative analysis between them. We found that the Long short-term memory

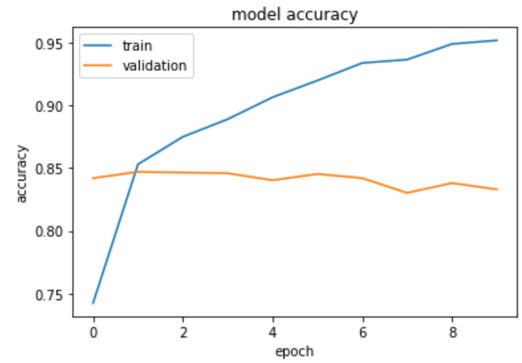

Fig. 5. CNN model accuracy

(LSTM) convolutional neural network(CNN) outperforms the rest of the model with almost 82% accuracy.

VIII. ACKNOWLEDGEMENT

This project was developed by Madhusmita Kalita, Mina Armanyous, Farah Shahata and Prerak Patel for the course ”Topics in Natural Language Processing” with equal contribution (25%) dedicated by each member of the group.

REFERENCES

[1] Haripriya, V., and Poornima G. Patil. ”A survey of sarcasm detection in social media.” Int. J. Res. Appl. Sci. Eng. Technol.(IJRASET) (2017).

[2] Joshi, Aditya, Pushpak Bhattacharyya, and Mark J. Carman. ”Automatic sarcasm detection: A survey.” ACM Computing Surveys (CSUR) 50, no. 5 (2017): 1-22.

[3] Liu, Peng, Wei Chen, Gaoyan Ou, Tengjiao Wang, Dongqing Yang, and Kai Lei. ”Sarcasm detection in social media based on imbalanced classification.” In International Conference on Web-Age Information Management, pp. 459-471. Springer, Cham, 2014.

[4] Amir, Silvio, Byron C. Wallace, Hao Lyu, and Paula Carvalho Mário J. Silva. ”Modelling context with user embeddings for sarcasm detection in social media.” arXiv preprint arXiv:1607.00976 (2016).

[5] Ghosh, Aniruddha, and Tony Veale. ”Fracking sarcasm using neural network.” In Proceedings of the 7th workshop on computational approaches to subjectivity, sentiment and social media analysis, pp. 161-169. 2016.

[6] Poria, Soujanya, Erik Cambria, Devamanyu Hazarika, and Prateek Vij. ”A deeper look into sarcastic tweets using deep convolutional neural networks.” arXiv preprint arXiv:1610.08815 (2016).

[7] Ptáček, Tomáš, Ivan Habernal, and Jun Hong. ”Sarcasm detection on czech and english twitter.” In Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pp. 213-223. 2014.

[8] Davidov, Dmitry, Oren Tsur, and Ari Rappoport. ”Semi-supervised recognition of sarcastic sentences in twitter and amazon.” In Proceedings of the fourteenth conference on computational natural language learning, pp. 107-116. Association for Computational Linguistics, 2010.

[9] Lukin, Stephanie, and Marilyn Walker. ”Really? well. apparently bootstrapping improves the performance of sarcasm and nastiness classifiers for online dialogue.” arXiv preprint arXiv:1708.08572 (2017).

[10] Bouazizi, Mondher, and Tomoaki Otsuki Ohtsuki. ”A pattern-based approach for sarcasm detection on twitter.” IEEE Access 4 (2016): 5477-5488.

[11] González-Ibáñez, R., S. Muresan, and N. Wacholder. ”Identifying sarcasm in Twitter: A closer look: 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, ACL-HLT 2011.” (2011).