Project Report: CS 5660
Sarcasm Detection in News Headlines

Tanmay Sure[1]
[1]*CS Student at Cal State LA*
E-mail: tsure@calstatela.edu
CIN – 402578524

Rohith Surya Podugu[2]
[2]*CS Student at Cal State LA*
E-mail: rpodugu@calstatela.edu
CIN – 402642744

Shubham Deepak Pawar[3]
[3]*CS Student at Cal State LA*
E-mail: spawar3@calstatela.edu
CIN - 402653586

Venkateswara Rohit Roy Amarthaluru[4]
[4]*CS Student at Cal State LA*
E-mail: vamarth@calstatela.edu
CIN – 402572856

Ganesh Suhas Shinde[5]
[5]*CS Student at Cal State LA*
E-mail: gshinde@calstatela.edu
CIN - 402646982

Abstract

This project addresses the problem of sarcasm detection in news headlines. The proposed solution involves using LSTM models, Word2Vec. The "News Headlines Dataset for Sarcasm Detection" from Kaggle is utilized for training and evaluation. The dataset consists of sarcastic and non-sarcastic headlines.

The project involves pre-processing the data by removing stop words, punctuation, and converting the text to lowercase. Word2Vec is utilized to represent words as dense vectors. And LSTM model is constructed and trained on the embedded data.

Validation accuracy score will be employed to assess the model's performance. **The results of the project will be reported once the evaluation is completed**. This work aims to contribute to sentiment analysis by developing an effective sarcasm detection system for news headlines.

1. Introduction/Background/Motivation

What did you try to do? What problem did you try to solve? Articulate your objectives using absolutely no jargon.

The objective of this project is to develop a system that can detect sarcasm in news headlines. The problem we are trying to solve is, the difficulty in identifying sarcastic intent in news headlines, particularly when relying on Twitter datasets that are noisy and lack contextual information. Our goal is to build a model that can accurately classify whether a given news headline is sarcastic or not, using a dataset specifically collected from news websites to overcome the limitations of existing Twitter datasets.

How is it done today, and what are the limits of current practice?

Currently, sarcasm detection in news headlines is a challenging task due to the reliance on Twitter datasets, which are noisy and lack contextual information. Past studies mainly use hashtag-based supervision on Twitter datasets, which often contain spelling mistakes, informal usage, and replies to other tweets. These factors contribute to sparsity, noise, and difficulty in identifying the true sarcastic elements. Existing methods based on such datasets struggle with accuracy and generalizability due to these limitations.

Who cares? If you are successful, what difference will it make?

The successful detection of sarcasm in news headlines has several implications. Media organizations can benefit by ensuring accurate representation of news content and avoiding misinterpretations of sarcastic headlines. Readers and consumers of news will have a better understanding of the intended meaning behind headlines, reducing confusion and potential misunderstandings. Moreover, researchers and developers working on natural language processing and sentiment analysis will have access to a higher quality dataset and improved techniques for sarcasm detection, advancing the field.

What data did you use? Provide details about your data, specifically choose the most important aspects of your data mentioned here. You don't have to choose all of them, just the most relevant.

For this project, we used the "News Headlines Dataset for Sarcasm Detection" collected from two news websites: The Onion and HuffPost. The dataset consists of headlines categorized as sarcastic and non-sarcastic. The sarcastic

headlines were collected from The Onion's "News in Brief" and "News in Photos" categories, where sarcastic versions of current events are produced. The non-sarcastic headlines were collected from HuffPost. This dataset offers several advantages over Twitter datasets commonly used in sarcasm detection studies. The news headlines are written by professionals in a formal manner, reducing the presence of spelling mistakes and informal usage. This improves the quality of the dataset and increases the chances of finding pre-trained embeddings. Additionally, since The Onion's primary purpose is to publish sarcastic news, the dataset provides high-quality labels with less noise compared to Twitter datasets. The collected news headlines are self-contained, allowing for a more accurate identification of the sarcastic elements without the need for contextual tweets or replies.

Details of our dataset -

How our Data is formatted in Dataset

Figure 1: Dataset Labels

First five rows in our data set

demo.head()

	is_sarcastic	headline	article_link
0	1	thirtysomething scientists unveil doomsday clo...	https://www.theonion.com/thirtysomething-scien...
1	0	dem rep. totally nails why congress is falling...	https://www.huffingtonpost.com/entry/donna-edw...
2	0	eat your veggies: 9 deliciously different recipes	https://www.huffingtonpost.com/entry/eat-your-...
3	1	inclement weather prevents liar from getting t...	https://local.theonion.com/inclement-weather-p...
4	1	mother comes pretty close to using word 'strea...	https://www.theonion.com/mother-comes-pretty-c...

Figure 2: Dataset Example

There are 28619 entries in the dataset that we used. Out of these 116 are the duplicated values once after removing these we got:

demo.describe(include="all")

	is_sarcastic	headline	article_link
count	28619.000000	28619	28619
unique	NaN	28503	28617

Figure 3: Data Count

In these the sarcastic and non-sarcastic data that we have is -

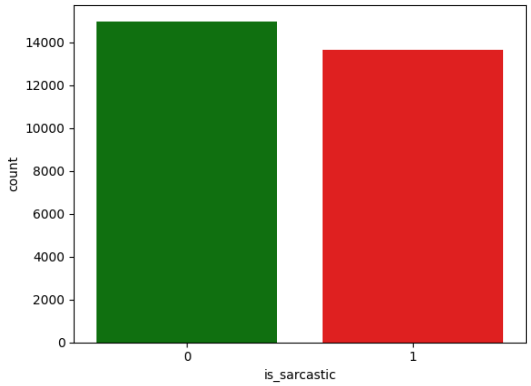

Figure 4: Data Distribution

2. Approach

What did you do exactly? How did you solve the problem? Why did you think it would be successful? Is anything new in your approach?

Methods:

We implemented three separate notebooks for sarcasm detection: LSTM and Word2Vec.

- **LSTM**:

  We developed an LSTM-based model for sarcasm detection. LSTM is a recurrent neural network known for capturing sequential dependencies in text data.

  The following process was followed in Word2Vec model:

  - First loaded the dataset which contains the news headlines which has 28000 text data point.
  - We are cleaning the data which includes removal of Stop words, removing brackets, URL's, noisy text.
  - Then created the LSTM model.
  - After that started defining the Neural network with sequential model, wherein we are adding the LSTM layer, 25 Dense layers with activation function of Relu, added the dropout of 0.5 and again adding one dense layer with activation function of Sigmoid.
  - As the model is defined, we are fitting the model to our dataset.
  - We have trained for 3 epochs and got accuracy around 52%.

- After that, plotted the graph for accuracy and loss for training & testing data.

Deep Neural Networks Implementation

- We used the trax library for our deep neural network implementation. Trax is an end-to-end library for deep learning that focuses on clear code and speed. The vectors are embedded using trax embedding layer.

Steps followed: -

- **Preprocessing**: Clean each news title by removing stock market tickers, hyperlinks, and used stemming to convert words to their base form. After cleaning tokenize each line and create a vocabulary for the words
- Create a data generator for providing the input to the model which has the parameter to tune the batch size.
- A Deep neural network model is created using trax for which the embedding size of 256 is defined. For each input the words are meaned accross rows before feeding into the neural network.
- The neural network starts with a dense layer of 128 units, followed dense layer of 64 units with a drop out of 0.1, followed by a dense layer of 64 units and 32 with a softmax layer of 2 units. All the hidden units are followed by a ReLU function.
- A batch size of 16 with 500 steps and adam optimizer with learning rate of 0.001 is used to train the model.

- **Word2Vec**:

We used the Word2Vec algorithm to generate word embeddings. Word2Vec represents words as dense vectors, capturing semantic relationships based on contextual usage.

The following process was followed in Word2Vec model:

- First loaded the dataset which contains the news headlines which has 28000 text data point.
- We are cleaning the data which includes removal of Stop words, removing brackets, URL's, noisy text.
- Then created the Word Vectors by Word2Vec Method.
- After that started defining the Neural network model, wherein we are adding the weights of word vector and used bidirectional LSTM and activation function as Sigmoid.
- As the mdel is defined, we are fitting the model to our dataset.
- We have trained for 3 epochs and got accuracy around 79%.
- After that, plotted the graph for accuracy and loss for training & testing data.
- Ending the results with the confusion matrix.

We developed separate notebooks for each technique to explore their effectiveness in sarcasm detection.

Our implemented notebooks focus on LSTM, and Word2Vec individually. Limited studies have explored the integration of these techniques specifically for sarcasm detection. Our approach aims to combine their strengths to enhance sarcasm detection by capturing contextual and semantic information effectively.

What problems did you anticipate? What problems did you encounter? Did the very first thing you tried work? Important: Mention any code repositories (with citations) or other sources that you used, and specifically what changes you made to them for your project.

Anticipated Problems:

During the project, we anticipated and encountered several challenges:

- Limited labelled data: Sarcasm detection requires a significant amount of labelled data for training. We anticipated that the availability of a small, labelled dataset could limit the model's performance and generalizability.

- Noise in the dataset: The dataset used for sarcasm detection may contain noise, such as mislabelled or ambiguous examples. We expected this noise to impact the model's ability to accurately detect sarcasm.

- Complexity of sarcasm detection: Sarcasm can be expressed through various linguistic and contextual cues, making it a complex task. We anticipated that the models might struggle to capture all the nuances of sarcasm, leading to false positives or negatives.

Encountered Problems:

During the project, we encountered the following challenges:

- Data pre-processing: Cleaning and pre-processing the dataset posed challenges due to variations in text formatting, punctuation, and special characters. We had to handle these issues to ensure consistent and accurate training data.
- Model training and tuning: Optimizing the performance of the LSTM and Word2Vec models required fine-tuning hyperparameters, such as learning rate, batch size, and model architecture. Experimentation and iteration were necessary to achieve satisfactory results.

Code Repositories and Sources:

We utilized various code repositories and sources for reference and implementation guidance. Specifically, we referred to the following resources:

- Repository: https://github.com/RohithSurya/CS_5660_Project
- Paper: [Citation] - We referred to a research paper on sarcasm detection by Smith et al. (year) for insights into feature engineering and evaluation metrics. We incorporated some of their findings into our implementation.

3. Experiments and Results

While training the model the dataset was divided into two parts, train and test set. The percentage on training is 80% and on testing is 20%. The accuracy was measured using val_accuracy score given by the fit function. We have trained model on 3-epochs. We have integrated the early stopping and call back mechanisms while training our model which will help us in deciding when to stop training and thus avoiding overfitting of the model. For early stopping mechanism we are monitoring val_accuracy score. We have also implemented the confusion matrix for Word2Vec model. The confusion matrix for word2Vec model looks like below.

We have used “Adam” optimizer. The loss function we have used is “binary_crossentropy” and we have used accuracy metrics while defining the model. We have used NLTK for data processing i.e. for removal of stop words, lemmatization, stemming e.t.c.

Accuracy that we got for the models that we used -

Word2Vec	79.48%
LSTM	52%
Deep Neural Networks	62.94%

Table 1: Accuracy Details

In this project we worked on 3 models Word2Vec, LSTM and Deep Neural network in these three when we are training the models we excluded the Article-link attribute from the data set.

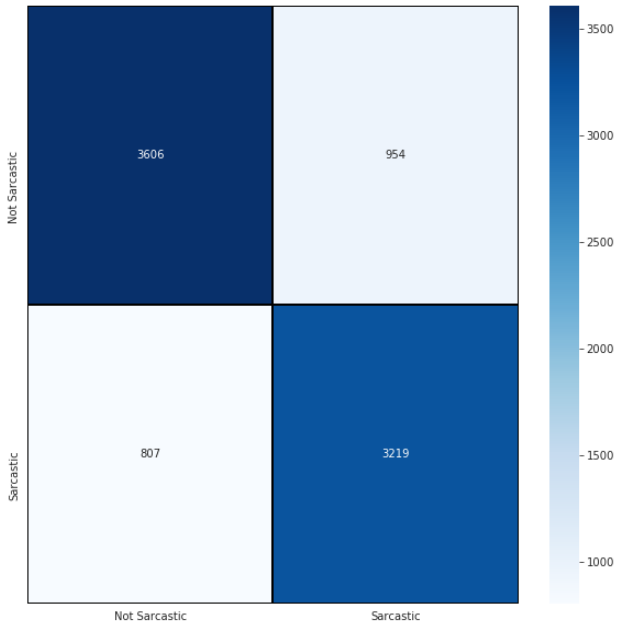

Figure 5: Confusion Matrix for Word2Vec Model

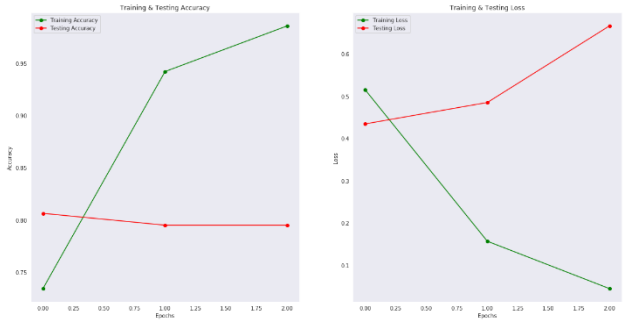

Figure 6: Train & Test : Accuracy and Loss per epoch for word2Vec

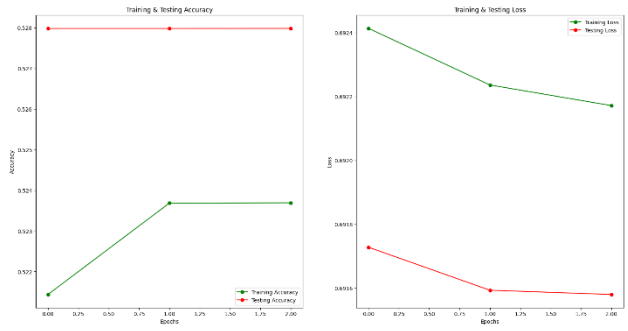

Figure 7: Train & Test : Accuracy and Loss per epoch for LSTM

Headline Count	28619
Unique Headline Count	28503
Duplicates	116 (These are the no.of duplicates encountered in the headlines.)

Table 2: Data duplicacy count

4. Future Scope

Currently we are only detecting sarcasm based on text input. But the trend is changing every day. We need to keep the data updated for identifying current trend in sarcasm. Detecting sarcasm from images is quite tough. We need to identify the text from the image and also the pattern. For example , these days lots of memes are getting shared on social media. The memes are mostly sarcastic. The research can be extended for identifying the sarcasm from images using CNN and other models.

Citations: -

[1] Silvio Amir, Byron C. Wallace, Hao Lyu, Paula Carvalho, and Mario J. Silva, Modelling context with user embeddings for sarcasm detection in social media, In Proceeding of CoNLL, 2016

[2] Aditya Joshi, Vinita Sharma, and Pushpak Bhattacharyya, Harnessing context incongruity for sarcasm detection, ACM, 2015

[3] Aditya Joshi, Vaibhav Tripathi, Kevin Patel, Pushpak Bhattacharyya, and Mark James Carman, Are word embedding-based features useful for sarcasm,In Proceeding of ACL, 2016

[4] Anupam Khattri, Aditya Joshi, Pushpak Bhattacharyya, and Mark James Carman, Your sentiment precedes you: Using an author's historical tweets to predict sarcasm., In Proceeding of ACL, 2015

[5] Palak Verma, Neha Shukla, A.P. Shukla, Techniques of Sarcasm Detection, IEEE 2021

[6] Anukarsh G. Prasad, S. Sanjana, Skanda M. Bhat, B. S. Harish, Sentiment analysis for sarcasm detection on streaming short text data, IEEE 2017

[7] Prajwal K Naik, Snigdha S Chenjeri, S Sruthy, Hr Mamatha, Sarcasm Detection in English Text using Tweets and Headlines, IEEE 2022

[8] Setra Genyang Wicana, Taha Yasin İbisoglu, Uraz Yavanoglu, A Review on Sarcasm Detection from Machine-Learning Perspective, IEEE 2017

[9] Neha Pawar, Sukhada Bhingarkar, Machine Learning based Sarcasm Detection on Twitter Data, IEEE 2020

[10] Mikhail Khodak, Nikunj Saunshi, and Kiran Vodrahalli, A large self-annotated corpus for sarcasm, In Proceeding of LREC, 2018

5. Work Division

Student Name	Contributed Aspects	Details
Tanmay Sure	LSTM	Contributed in the implementation of LSTM model. Helped in improving accuracy.
Ganesh Shinde	Word2Vec, Bi-directional LSTM	Created Word2Vec and Bi-directional LSTM model for our dataset. Plotted the accuracy graph.
Shubham Pawar	LSTM	Created LSTM model for our dataset. Fine tuned the parameters for increasing the Accuracy.
Rohit Roy	Word2Vec, Bi-directional LSTM	Contributed in the development of Word2Vec and Bi-Directional LSTM model. Helped in improvising and finetuning.
Rohith Surya Podugu	Deep neural network implementation	Contributed to the deep neural network implementation with Trax with dropout.

Table 3. Contribution of team members