COMP4420 Project Report: Sarcasm Detection in Headlines

Bui, Nam (#01963609), Conners, Riley (#01943861), Zuk, Sam (#01642608)

1 Abstract

This project seeks to explore different sentiment analysis techniques in the task of sarcasm detection in newspaper headlines. We compare the performance of a Naive Bayes model and LSTM model, with and without pre-trained word2vec embeddings, on the task. Naive Bayes achieved roughly 80% accuracy and the LSTM approaches achieved roughly 90%. Different embeddings had a negligible effect on classification, but the switch from a Naive Bayes to an LSTM approach showed significant improvement.

2 Introduction

Sarcasm is a feature of natural language that is notoriously difficult to define and identify in both the spoken and written word. It is defined as "the use of words that convey the opposite meaning to cause or show irritation." [1] The assumption that this sarcastic intention will be recognized is typically contingent upon the listener/reader knowing some outside piece of contextual information beforehand. However, this external information isn't always known, and even when it is, the relationship between it and the statement at hand may not always be clear. When this happens, the meaning can be obscured as a result, often leading to avoidable scenarios involving miscommunication.

Recognizing sarcasm typically involves picking up on subtle cues and nuance that can be difficult to identify. This can often pose a challenge for populations who encounter greater difficulty when processing certain aspects of a language. For example, someone trying to interpret a language they don't speak natively will likely have to expend more mental effort to parse out meaning from words, which in turn makes it more difficult to pick up on nuance, including sarcasm. Being unfamiliar with the cultural norms, idioms, etc. that inform the established meaning of the locally spoken language can also be a source of confusion. In addition, many neurodivergent people, in particular those with autism, can struggle to recognize and/or communicate certain social cues in conversation due to differences between their cognitive experience of language and what is expected of them.

Finally, there are unique challenges faced in detecting sarcasm in the written word. It is often possible in practice to infer a statement is sarcastic, even without necessarily having the context to understand *why* by listening to changes in the tone of the speaker. However, when translated into the written word, some or all of this information is lost, making sarcasm even more difficult to detect when only text is given. With the Internet now being extremely important to modern infrastructure, and with text being the predominant medium for online communication, this problem has become increasingly apparent over the years. This project shall explore and contrast different approaches to disambiguating sarcasm by applying concepts from the fields of computational linguistics and machine learning.

3 Data

The dataset used for this project is a collection of 28,619 tagged newspaper headlines – 13,635 of which originating from the satirical publication *TheOnion* and the other 14,984 coming from the non-satirical publication *The Huffington Post* (*HuffPost*). The data was collected from TheOnion's "News in Brief" and "News in Photos" sections and HuffPost's news archive page in 2019 [3].

Each headline is represented as a JSON object with three attributes:

- `is_sarcastic` (integer): the headline's label – 1 if sarcastic, 0 if not.

- `headline` (string): the text of the headline, case-converted to be all lowercase.
- `article_link` (string): the URL of the referenced article.

Our models used the headline text to predict whether or not the article is sarcastic. Article links were not used by the models themselves, but were useful in validating the authenticity of the provided headlines. Further research might find these links useful for the purposes of obtaining and training on the article text instead of / in addition to the headline alone.

The main limitation of this data collection strategy is its limited scope. It would be unwise to assume findings on a set of headlines from only two outlets are representative of the task of sarcasm detection as a whole. Models might pick up on words and phrases that are less indicative of sarcasm and more indicative of writing style, formatting guidelines, and/or other details particular to one source or the other. In addition, since the sort of sarcasm employed by *TheOnion* tends to be more obvious and outlandish, it's possible this data may produce models that fail to identify statements that are sarcastic in subtler ways.

For example, the two bigrams that occur most frequently amongst *TheOnion* headlines are ['report', ':'] (427 occurrences) and ['area', 'man'] (231 occurrences). In a headline like "Report: God directly communicating with you through this headline," sarcasm is conveyed through the juxtaposition of the formal tone with something later in the text that is exaggerated, absurd, etc.. It is unlikely that the presence of phrases like "report:" on their own are indicative of sarcasm; models will need to be able to pick up on these subtleties in order to be successful.

However, despite the lack of broad generalizability, this data presents a useful example of a sarcasm detection problem from which meaningful insights can be gained. Additionally, data collected from reputable media outlets has advantages over data collected from public social media platforms, which is often used in sarcasm research.

Since headlines are typically proofread and written in a formal style, their text tends to contains less slang, fewer spelling and grammar mistakes, and a lower frequency of very uncommon words. The source-based approach to data labelling also helps reduce ambiguity about label accuracy, since the sarcastic intent of a writer at a satire publication is easy to identify.

4 Method

To begin, the dataset described in section 3 was partitioned 70 / 20 / 10 into training / validation / test sets respectively. All articles labeled as genuine in the test dataset were then manually reviewed to ensure there were no incorrect labels.

The steps for tokenizing the dataset were:

1. Tokenize hyphens.
2. Tokenize single quotes.
3. Transform contractions to canonical form.
4. NLTK word tokenize.
5. Address edge cases.

The vocabulary included all tokens with $count > 5$.

A Naive Bayes model was used to get initial performance baselines. Along with the tokenized dataset, lemmatization was used to group words with the same meaning, like 'says' and 'said', together. Hyperparameter search was done on the smoothing parameter of the model. We found that smoothing factor $\alpha = 1.5$ performed the best, although other parameters performed closely.

Word2vec embeddings pre-trained on the Google News dataset were then fine-tuned over the sarcasm dataset to better fit the dataset. [2] Embeddings for words that were common and unique to the dataset were also added. Since word2vec does not have an unknown token, we mapped the unknown token to the 0 vector, which is what Rishabh and Prahal did in their research. [3] An LSTM model was then trained with and without the pre-trained embeddings. The architecture of the LSTM model consisted of embedding layer, LSTM layer, and a fully-connected feedforward network. Gradient clipping, early stoppage, batching, dropout, and learning rate scheduling were used during training. Results can be found in section 5.

5 Results

To compare our results with the results of Rishabh and Prahal, we measured the accuracy of our models on the test set. [3] Additionally, since this is a binary classification task, we also used precision, recall, and F1 metrics.

Model	Accuracy	Precision	Recall	F1
Rishabh and Prahal Hybrid NN	0.897	N/A	N/A	N/A
Naive Bayes	0.793	0.806	0.795	0.800
Plain LSTM				
LSTM w/ word2vec				

Table 1: Comparison of model performance.

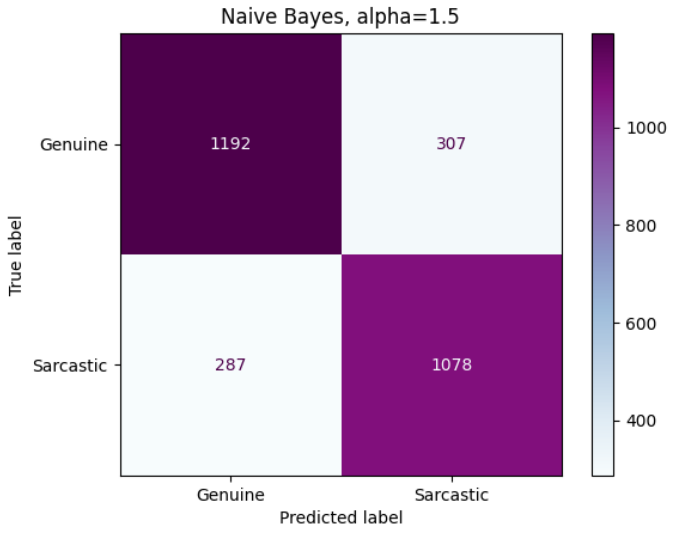

Figure 1: Confusion matrix of models.

The LSTM-based models do not show significant differences in performance. This implies that the largest factor to performance is the LSTM layer itself.

6 Conclusion

In conclusion, we found that an LSTM approach outperformed the Naive Bayes. Sarcasm is a complex concept to identify, especially with only text input. This project could be extended in the future to work on further customized dictionaries with word embeddings for new words like Trump. Customized embeddings for names and/or events could lead to even better recognition of the meaning behind headlines.

Additionally, it may be useful to include article text in the dataset so that the model can compare the article text and headline text because sarcasm comes from a semantic difference between the two. In terms of the model, an attention layer or stacked LSTM could also be added to see how it affects performance.

7 Contribution Chart:

Student Name & ID	Tasks/Subtasks	Commentary on Contribution
Bui, Nam (#01963609)	Tokenized Dictionary Created and Ran Bayes Model Created LSTM Model Debugged Models	Created the original files for both the Bayes and LSTM models, debugged issues with the models, and helped debug tokenization issues with the dataset.
Conners, Riley (#01943861)	Split Data and Validated Tests Created Data Loader Ran Initial Runs of LSTM	Split the data and manually reviewed test set, created the initial dataloader, and performed initial hyperparameter tuning tests with LSTM.
Zuk, Sam (#01642608)	Exploratory Data Analysis Tokenized Dictionary Created Custom Word Embeddings Ran Final Run of LSTM Model	Conducted the exploratory data analysis, helped with tokenizing the the dictionary and creating custom word embeddings, and created tables on the final data on the run of the LSTM.

8 References

[1] Merriam-Webster. Sarcasm. `https://www.merriam-webster.com/dictionary/sarcasm`.

[2] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26, 2013.

[3] Rishabh Misra and Prahal Arora. Sarcasm detection using news headlines dataset. *AI Open*, 4:13–18, 2023.