# COMP9444 Project Summary

## BLUFF: Sarcasm Detection for News Headlines

Aravind Venkateswaran z5208102 | Fatemeh Karbaschi z5335968 | Leonie Dickson z5215454 | Terry Luo z5308257 | Veeraj Sharma z5113751

### I. Introduction

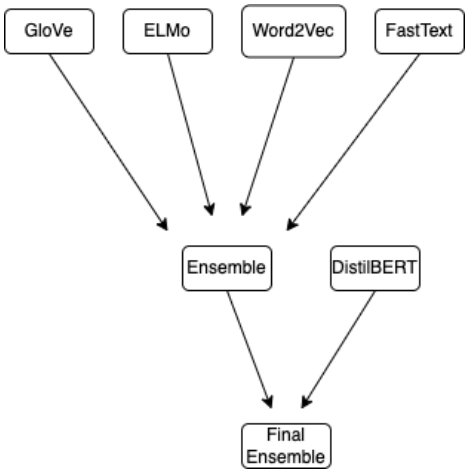
Sarcasm is a figure of speech and detecting its presence is an important part of natural language understanding. Depending on the context, sarcasm has the potential to flip the sentiment of a piece of text, having implications for key areas of NLP such as sentiment analysis. Literature shows sarcasm detection exercises have been conducted on social media content, for example detecting sarcasm in tweets (K.Ranganath et al, 2020). The aim of our project is to detect sarcasm in news headlines.

There are several existing NLP methods that can be used for a sequence classification problem like this, including traditional word embedding methods like word2vec, GloVe, or fastText, contextual language embedding like ELMo, and transformer-based methods like BERT (what is considered state of the art). Our proposed method is to combine several of these into an ensemble model which improves on the performance of each individual model.

### II. Methods

5 individual models were constructed and trained on the dataset. These all used pre-trained models and used fine-tuning to classify the output from the pretrained model as sarcastic or not sarcastic. These models were:

- DistilBERT
  - DistilBERT is a lightweight form of BERT, a transformer-based method. A linear layer was added on top to fine tune the DistilBERT output for classification.
- GloVe
  - Pretrained embeddings are downloaded and embedding vectors are produced for classification. These vectors are used in a neural network with Keras sequential API with a bidirectional LSTM and sigmoid dense layer (Pennington, J. et al., 2014).
- ELMo
  - ELMo word vectors are computed on top of a two-layer bidirectional language model (biLM) that uses LSTMs. A logistic regression was used for our binary classification problem with the vectors produced by the embeddings
- FastText
  - fastText is a library for efficient learning of word representations and sentence classification that uses pre-trained word vectors.
- Word2vec
  - Word2Vec vectors for the dataset were produced by the gensim.models.Word2Vec library. These vectors are used in a neural network with Keras sequential API consisting of bi-directional LSTM and GRU layers. The final layer has a sigmoid transfer function for binary classification.




Following the training of each of these 5 models individually, which are existing methods for sequence classification, an ensemble model was created by producing a weighted average of their prediction probabilities.

The ultimate aim for training the separate models was to combine them into an ensemble model. Ensemble modeling reduces variance and is principally the same as the bagging method in the random forest algorithm which trains different models parallel to each other. The rationale behind ensemble modeling is to make better predictions and achieve better performance in the combined model than any single contributing model. Moreover, an ensemble reduces the spread or dispersion of the predictions and model performance.

## III. Experimental Setup

The dataset used (News Headlines for Sarcasm Detection (Kaggle, 2019)) contains 28619 samples with headlines labeled as 1 (sarcastic) or 0 (not sarcastic). Data exploration showed this dataset to be mostly balanced. As a binary classification task, our evaluation strategy was to judge model performance based on common metrics including accuracy, precision and recall. The F-measure and Matthews' Correlation Coefficient were also computed as scores combining these metrics.

A 75:25 split into training and testing datasets was taken, with the same training and testing data used to train all 5 models. When training DistilBERT, the dataset was again split into 90:10 for training and validation.

Key model hyperparameters include number of epochs and batch size, which differed for each model used. GloVe and word2vec additionally had the embedding dimensions as hyperparameters. A logistic regression was applied for classification on top of the ELMo embeddings, and the maximum number of iterations for this regression was adjusted for this model. The fastText model required words to be split into N-grams, which was another hyperparameter to be adjusted. Refer to codebase for details on what hyperparameters were chosen for each model.

## IV. Results

We have used different evaluation metrics including Accuracy, Precision, Recall, F1 score and MCC to compare the performance of the models. An explanation for each of these evaluation methods has been added to the notebook. These evaluations show the DistillBERT model has the highest performance and ELMo has the lowest performance among the individual models. Further graphs describing model performance including confusion matrices are available in the Jupyter Notebook.

| Model | Accuracy | Precision | Recall | MCC | F1 |
|---|---|---|---|---|---|
| DistilBERT | 0.92 | 0.93 | 0.90 | 0.84 | 0.92 |
| word2vec | 0.88 | 0.86 | 0.88 | 0.75 | 0.87 |
| GloVe | 0.82 | 0.83 | 0.79 | 0.64 | 0.81 |
| ELMo | 0.79 | 0.79 | 0.76 | 0.58 | 0.78 |
| fastText | 0.81 | 0.82 | 0.78 | 0.62 | 0.80 |
| Ensemble | 0.93 | 0.94 | 0.91 | 0.86 | 0.92 |

The ensemble model was successful in boosting model performance, with this method improving on each of the existing methods used individually. With 93% accuracy, we do not believe that this model is sufficient to replace human judgment, however it could act as a guide providing a suggested classification to users.

## V. Conclusions

The key strength of our ensemble model is in its boosted accuracy. A key limitation is that it is computationally expensive, requiring 5 models to be trained, and further loses explainability. Our model had more false negatives than false positives (better precision than recall), which interestingly, mirrors how humans misinterpret sarcasm. People can often miss sarcastic comments and interpret them to be serious, more frequently than they interpret a serious comment to be sarcastic.

This task is similar to fake news detection and has similar limitations - to truly detect sarcasm, much more outside context would be needed, and frequently humans themselves cannot understand sarcasm when they lack the necessary contextual knowledge. This means that the model may not be effective in classifying sarcastic headlines in other contexts (eg. headlines from Australia, headlines in the future, or headlines from other news outlets) as the topics may vary from those the model was trained on, and as such the model would lack the data representing this knowledge. Bias may also come from the headline sources, with the models potentially differentiating between HuffPost and The Onion, rather than sarcasm itself.

The worst performing individual model was ELMo - this might be attributed to only a simple logistic regression model being applied on top of the embeddings produced by ELMo. Future work would include applying neural network layers on top of ELMo to investigate if this could improve its performance in this classification task.

From the word clouds produced from the model's classifications, it appears that the word 'new' is commonly found in all 4 quadrants (true positives and negatives, false positives and negatives). It could be that in this task, the word 'new' does not add any useful information, so future work to improve on this task could include treating 'new' as a stopword.

# References


1. K.Ranganath, MD.Sallauddin and Shabana (2017). Recent Trends In Sarcasm Detection on Online Social Networks. International Journal of Computer Sciences and Engineering, 5(10), pp.235–239. doi:10.26438/ijcse/v5i10.235239.
2. Kaggle. 2019. News Headlines Dataset For Sarcasm Detection. [online] Available at: <https://www.kaggle.com/datasets/rmisra/news-headlines-dataset-for-sarcasm-detection> [Accessed 2 August 2022].
3. Pennington, J. (2014). GloVe: Global Vectors for Word Representation. [online] Stanford.edu. Available at: https://nlp.stanford.edu/projects/glove/.