# Fake News Challenge

Prajwal Rao (5176504) & Julian Blair (3463793)
COMP9417 2018s1 – Assignment 2, Topic 1.5

## Background

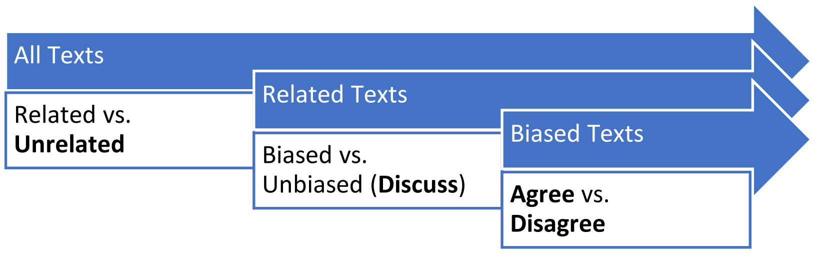
The Fake News Challenge was hosted in 2017 by a group of academic and industry volunteers in the interest of developing a robust and efficient method to identify misleading or false news media – a problem which has gained interest since the 2016 U.S. presidential election. This process, known as *stance detection*, could then be used as part of a larger solution for a fact-checking data pipeline.

Provided by the competition organisers is a set of labelled training data, and unlabelled competition data for scoring. Both sets of data contained rows of article headlines and related bodies. The labelled data comes with one of four possible classes: agree, disagree, discuss, and unrelated. These labels were selected by the challenge organisers from the past work of Ferreira and Vlachos (2016) on *stance detection*. The aim of the competition is to predict these classes for the unlabelled competition data.

The competition also provides a baseline implementation for challenge participants, which creates a train-test split with the provided training data, builds some basic natural language processing features such as headline-body comparisons and contrived word frequencies, cross-validates several iterations of a gradient-boosting classifier for all four classes, then scores the best classifier iteration.

### Problem Structure Hypothesis

Our approach for this challenge was to logically restructure the four-class problem provided into three two-class problems: Related/Unrelated, Biased/Unbiased, and Agree/Disagree.


All Texts
Related vs. **Unrelated**
Related Texts
Biased vs. Unbiased **(Discuss)**
Biased Texts
**Agree** vs. **Disagree**

Taking a divide-and-conquer approach to this problem not only increases the amount of data in each class and potentially increase the quality of predictions, but also allows for the accentuation of different features of the data in different contexts. For instance, we can derive from an article headline that agrees with the topic that it (a) must be related to the topic in the first place, (b) takes a biased stance on the topic, and (c) agrees with the topic.

In addition, breaking the classification problem up into stages allows for easier diagnosis of classification error, as will be discussed in the Results section of this report.

## Implementation

Some aspects of our implementation remained the same relative to the baseline, such as the 10-fold cross-validation on a gradient boosting classifier, as we determined upon testing alternative classifiers and parameters that the existing implementation performed the best on the hold-out test dataset. Most of our implementation changes outside of restructuring the classification problem were with the features of the data.

### 1. Common Word Filter

The first major modification was the addition of a common word filter to the existing headline-body comparison features. This was implemented without the use of libraries, and simply aggregates all the words used per article ($W$), calculates the coefficient of variance across all articles ($A$) for each word ($w$), then returns the bottom 0.5% of results:

$$cv = \sigma_{I(W=w)} / \mu_{I(W=w)}$$

The coefficient of variance is used since it produces a general approach for finding words that are not common across all articles. This is a more general and flexible approach than using a stop word lexicon, while being a more efficient and simple approach than using a natural language toolkit module.

### 2. Paraphrasing

One way to reduce dimensionality of the lexicon of articles in the training data is to identify words in the same lexical category with similar meaning, and collapse them into grouped sets. The WordNet package in the Natural Language Toolkit performs this task, and thus was added as a word filter before performing the binary co-occurrence feature generation functions provided in the baseline.

### 3. Multinomial Naive Bayes Stacking, Formatting Features

The last two key additions to our solution were a Multinomial Naive Bayes classifier meta-feature, and an 'all-caps' text format feature.

The Naive Bayes classifier, provided in the scikit-learn package, produces a $2 \times A$ matrix of class log probabilities per article, which is then used as a set of features for the Gradient Boosting classifier. This effectively partially converts the Gradient Boosting classifier into a multi-level (stacked) meta-classifier, but since the Naive Bayes classifier reduces the dimensionality of features down to one feature per class, the probability of overfitting the competition dataset is reduced.

The 'all-caps' text format feature is a frequency of capitalised words, normalised by text length and the number of capitalised words in the total corpus of articles.

### Overfitting Issues

Several high-dimensionality (and similar aggregated low-dimensionality) features were also tested, such as punctuation frequencies and opinion lexicons. However, all of these features were found to either overfit the training data, or otherwise mislead the classifier in predictions. While the contrived list of words used in some data features by the baseline may not have been a robust solution, its simplicity produced the best performance on the hold-out test dataset.

## Results and Discussion

Figure 1 illustrates the relative score improvement at each stage of classification compared to the baseline implementation.

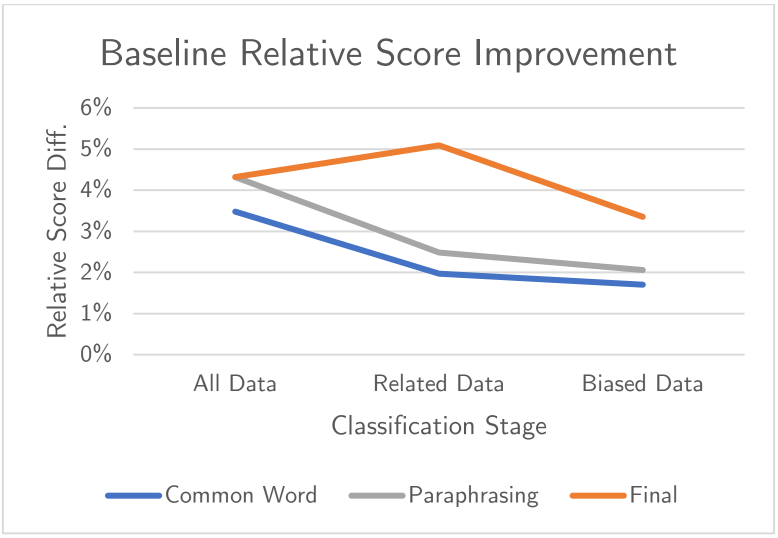


Figure 1

Each feature modification was implemented only at classification stages where a gain in score for the hold-out test dataset was observed for that stage, which is reflected in the improvement of each implementation iteration when predicting the classes of the competition dataset. Significant gains that were experienced when implementing the common word filter for related/unrelated classification, and the Naive Bayes classifier for biased/unbiased classification.

Figure 2 compares the relative scores for predictions on the competition dataset by the baseline and final implementations.

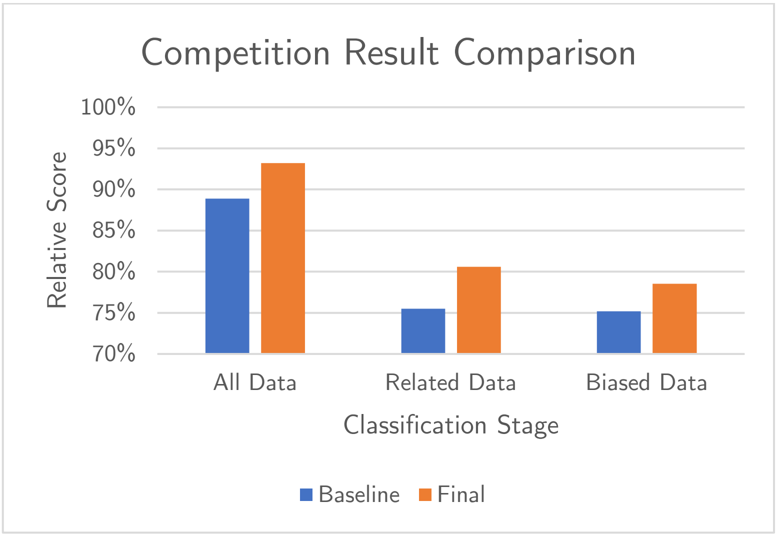


Figure 2

A relative score improvement of 3.4% from the baseline was achieved at the final stage of classification, with the largest classification error being produced at the biased/unbiased classification stage, as illustrated by the significant drop in relative score for both the baseline and our final implementation.

Table 1 is a confusion matrix of the classifications predicted by our final implementation on the competition dataset.

| Actual\Predict | Agree | Disagree | Discuss | Unrelated | % Correct |
|---|---|---|---|---|---|
| **Agree** | 971 | 26 | 748 | 158 | **51.0%** |
| **Disagree** | 293 | 17 | 262 | 125 | **2.4%** |
| **Discuss** | 924 | 22 | 3149 | 369 | **70.5%** |
| **Unrelated** | 157 | 6 | 400 | 17786 | **96.9%** |
| **% Correct** | **41.4%** | **23.9%** | **69.1%** | **96.5%** | **86.3%** |

Table 1

Our implementation achieved a final score of **9152.25/11651.25** (**78.55%**), which would have ranked at 13th place out of the 80 groups who participated (CodaLab, 2017). The lowest performing classifications were for articles where the headline disagreed with the topic (2.4% correct), with the largest number of such headlines being misclassified 'agree' at the agree/disagree classification stage. The relative lack of disagreeing headlines (1.7% of total) relative to agreeing headlines (7.4% of total) in the training data meant that an insufficient number of contrary examples was provided during feature generation, which increased the likelihood of misclassification at the final stage.

## Conclusion

Our implementation has demonstrated that, given the high dimensionality problem inherent in text-based classification, simpler features that make correct assumptions about the underlying textual structure tend to lead to the highest predictive gains.

Many of the submissions that achieved a higher competition score used deep learning techniques to classify the article corpus, as opposed to using more explainable, structured machine learning techniques. That being said, imposing some structure by making reasonable assumptions about the data can still produce good predictive accuracy for multi-class classification problems.

## Acknowledgement

Thanks to the Fake News Challenge team for hosting the competition, as well as for providing a baseline implementation, which we used as a starting point for our project.

## References

CodaLab (2017) *Fake News Challenge Stage 1 (FNC-I) - Stance Detection*, June, [Online], Available: https://competitions.codalab.org/competitions/16843#results [29 May 2018].

Ferreira, W. and Vlachos, A. (2016) 'Emergent: a novel data-set for stance classification', Association for Computational Linguistics, San Diego, California, 1163–1168.