Fake News Challenge

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

Background

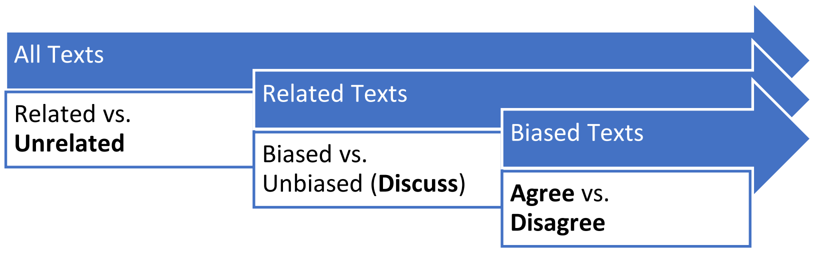
The Fake News Challenge (*FNC*) was hosted in 2017 by a group of academic and industry volunteers in the interest of developing a robust and efficient process to identify misleading or false news media – a problem which has gained the interest of the public since the 2016 U.S. presidential election. The challenge posed to competition participants is to implement a module of this process that accurately identifies the position an article takes on a given topic, a process known as *stance detection*. Such a module could be included as part of a larger solution for a news fact-checking pipeline.

Provided by the *FNC* competition organisers is a set of labelled training data, and unlabelled competition data for scoring. Both sets of data contain rows of article headlines and related article bodies. The training data comes with one of four possible classes, each associated with the stance of headlines relative to the body text: **agree**, **disagree**, **discuss**, and **unrelated**. These labels were selected by the challenge organisers based on the past work of Ferreira and Vlachos (2016) on *stance detection*. Competitors are scored based on the ability of their algorithm to predict these classes for the unlabelled competition data.

Also provided by the competition organisers to the *FNC* participants is a baseline implementation, which generates a train-test split with the provided training data, builds some features using basic natural language processing methods such as *n-gram* headline-body comparisons and contrived word frequencies, cross-validates several iterations of a gradient-boosting multi-class classifier, then tests the best performing classifier iteration on the competition dataset.

Problem Structure Hypothesis

Our approach to this challenge was to logically restructure the four-class problem provided into three smaller 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 by aggregating classes in stages not only increases the amount of training data in each class in the training stages, but also allows for the accentuation of different features of the data in different contexts. For instance, we can derive from an article headline labelled **agree** 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 unchanged relative to the baseline, such as the 10-fold cross-validation method performed on a gradient boosting classifier, as we determined upon testing alternative classifiers and parameters that the existing implementation performed the best on our hold-out test dataset. Most of our implementation changes outside of restructuring the classification problem were related to the features of the data.

1. Common Word Filter

The first major modification to the baseline was the addition of a common word filter to the existing *n-gram* headline-body comparison features. This was implemented without the use of libraries, and simply aggregates all word frequencies per article W, calculates the coefficient of variance cv across all articles for each word:

$$cv = \sigma_W / \mu_W,$$

then returns the bottom 0.5% of results. The coefficient of variance is used since it provides a general approach for finding words that are heterogeneously distributed across all articles. This approach is more general and flexible than using a fixed stop word lexicon, while being more efficient and simple than using a natural language toolkit module.

2. Paraphrasing

One way to reduce dimensionality of the lexicon of the training data article corpus is to identify words in the same lexical category with similar meaning, then collapse them into grouped sets of synonyms. The *WordNet* package in the *Natural Language Toolkit* performs this task, and was thus added as a word filter before performing the *n-gram* 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 $A \times C$ matrix of class C log-probabilities for each article A, 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, and 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 these features were found to have either overfit the training data, or otherwise mislead the classifier in predictions. While the contrived list of words used in some 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.

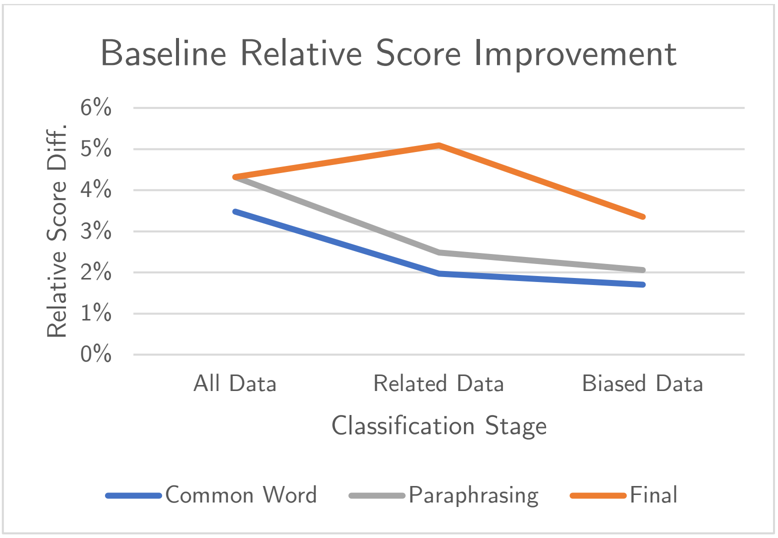


Figure 1

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.

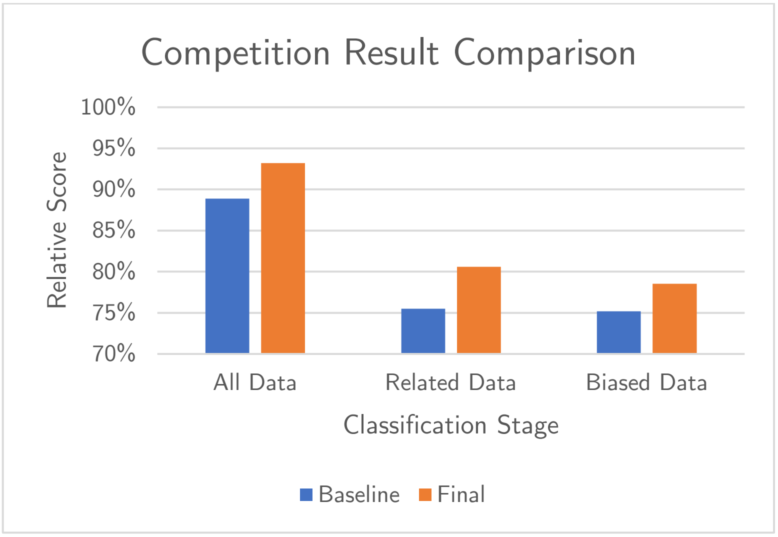


Figure 2

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 'Related Data' classification stage, as illustrated by the significant drop in relative score for both the baseline and final implementations.

Table 1 is a confusion matrix for 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 which participated in the *FNC* (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 as **agree** at the 'Biased Data' classification stage. The relative lack of **disagree** training data (1.7% of total) relative to **agree** training data (7.4% of total) meant that an insufficient number of contrary examples were provided during feature generation, which increased the likelihood of misclassification on the test datasets.

Conclusion

Our implementation has demonstrated that given the high-dimensionality problem of text-based classification, simpler data features that make accurate 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 perform *stance detection* on the article corpus, as opposed to using more explainable, structured machine learning techniques. Imposing some structure by making reasonable assumptions about the data, however, 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.