Fake News Stance Detection: 2-step Approach

Junhan Liu
University of Waterloo,
j896liu@uwaterloo.ca
Github: https://github.com/jimjimliu/msci-text-analytics-s20.git

Abstract

We are hurtling toward a digital time where information becomes cheap and easy to spread. Fake news is arguably one of the most serious challenges facing the news industry today. With the help of artificial intelligence technologies, particularly machine learning and natural language processing, it might be a method to combat such problem. As one of the stages in the process of detecting fake news, automating stance detection could serve as a useful building block. Stance detection aims to determine the association between a body article and the news' declared headline. During this process, a model classifies the stance of the body text relative to the claim made in the headline into one of four categories: agree, disagree, discuss, and unrelated.

1 Introduction

Following the advent of the Internet, information is presented and is pushed into people's daily lives especially when electronic devices are easily possessed. More information emerges in different formats on various platforms such as videos, pictures, tweets, etc.; however, written texts remains its power on spreading information quickly and efficiently. As spreading information becomes easy, publishers use false and misleading information, i.e. click baits, to further their interests. Detection of misleading information is, therefore, needed.

Fake News Challenge (FNC-1) (Rao and Pomerleau, 2017) suggests the first step of the complete fake news detection process is called stance detection. It aims to determine how relevant a body article is to its headline. Many times, news articles may have attractive headlines to gain people's attention; whereas, the content of an article's body does not match the content described by the headline. The task is to classify a news instance into one of four categories which are agree, disagree, discuss, and unrelated. Classification of related pairs to agree, disagree, and discuss weighted more score than classifying a pair to unrelated/related. More specifically, a system should be developed to estimate the stance of the body towards the headline when given a news article headline and a news article body. More information on this fake news challenge, data, evaluation metrics can be found on the official website: fakenewschallenge.org.

The classification between agree, disagree, and discuss involves a more careful feature engineering than classifying an instance to related or unrelated. The raw input data set is imbalanced. After re-sampling the data set, the binary classification accuracy reaches above 90% if using simple tf-idf vector as a feature or using cosine similarity as a feature. The accuracy of classifying related pairs into agree, disagree, or discuss reaches an accuracy only about 66% after re-sampling. The goal of our model aims to beat the FNC-1 baseline's accuracy which is 75.20% and 73.9% after re-sampling. Similar tasks were done repeatedly after 2017 by many scholars and teams of students from different universities. Our feature selection adopts some ideas from a similar project (Akhtar and Jha, 2018).

Inspired by the FNC-1's winner team SOLAT in the SWEN's (Baird et al., 2017) combined approach, we are using a 2-step approach to build the model. As shown in Figure 1, a conventional classifier SVM aims to perform a binary classification which predicts the stance of an instance as related or unrelated. Meanwhile, a 3-

layer feed-forward neural network determines if an instance has a stance of agree, disagree, or discuss. The neural network is using SoftMax activation function as the final layer. The final step is to combine the results generated by each classification model by swapping stance "related" to one of the following three labels predicted by a neural network: agree, disagree, and discuss. The feature selection uses a subset whichever contributes the most to the final accuracy from the following possible choices:

1. Cosine similarity between headline and body
2. The Polarity of headline and body
3. Kl divergence
4. N-gram overlap
5. Jaccard similarity
6. Word vectorization using TF

The distribution of data is badly imbalanced and "unrelated" itself takes 73% of the overall data set. Re-sampling before model fitting needs to be done using either over sampling or under sampling. Before re-sampling, the baseline has a 75.2% testing accuracy and 73.9% after over sampling. Our model testing accuracy reaches 86.5% before re-sampling and 81% after.

The goal of this short paper is to present a description of our Fake News Stance Detection system. The following sections discuss the detailed approach of building the system, some relevant literature, and future improvements.

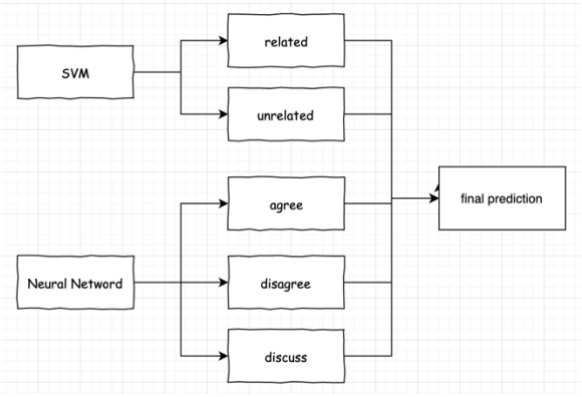

Figure 1: General flow chart

	Stance	Proportion
1	Unrelated	0.73131
2	Discuss	0.17828
3	Agree	0.0736012
4	Disagree	0.0168094

Table 1: Distribution of data

2 Related Work

Text analysis, a sub-branch of natural language processing area, itself has many applications and unsolved tasks including fake news detection, question answering, etc. Authors apply various approaches at different levels to attempt to solve these problems. In terms of fake news detection, many approaches have been proposed to employ syntactical (Aldwairi and Alwahedi, 2018) and textual (Zhang et al., 2020) structures as features. The top three winners of FNC-1 (Rao and Pomerleau, 2017) challenge incorporated different combinations of features, such as word vector embeddings, tf vectorizations, cosine similarities between tf-idf vectors, bag-of-words (BOW), and polarity from FNC-1 baseline (Baird et al., 2017; Hanselowski et al., 2018; Riedel et al., 2017). The third winner (Riedel et al., 2017) of the contest applies a simple features combination, tf vectorization of headline and body that each has a length of 5,000 and cosine similarity between headline and body, to achieve a fair accuracy of 81.72% and a leader board score of 9521.5.

3 Approach

In this sub-section, we introduce a detailed approach of our model building process and feature selection. At textual and statistical level, a combination of several kinds of features outperforms a simple tf or tf-idf vectorization as features. On top of that, we apply a combined approach to train and test data where two models train the same data set using different features to generate different class labels as Figure 1 illustrates.

3.1 Preprocessing

Before building the feature representation, we perform some basic text filtering to the input data set. Numbers, special characters, punctuations, and stop words are removed from news headlines and article bodies. Only English characters remain. These basic filtering procedures are also included and provided by many Python APIs such as nltk.tokenize[1] and sklearn.TfidfVectorizer[2].

3.2 Feature Selection

The final combination of features is selected based on their performance on the accuracy and leaderboard [3] score. Fitting iterations are done using the training set with different features selected in Table 2. Figure 2 shows our final feature combinations applied to the models. On average, cosine similarity and n-gram overlap together give the best results among all the features listed in Table 2.

In Table 3, it shows the testing accuracies' growth information when adding more and more features to a feature set. By applying a single feature, the cosine similarity between the word vectorizations of headline and body, the testing accuracy reaches 81.5% on average after re-sampling. Adding more features included in Table 2 repeatedly does not give us better or a noteworthy boost on the testing results.

- **Cosine Similarity**
  We employ the cosine similarity between news headlines and bodies as a feature. First converting article headlines and bodies together to tf-idf vector representations and then building a tf-idf lookup dictionary that has each word as key and the word's tf-idf as value. The table may look like: ***{'planes': 0.015, 'tribunal': 0.346, 'em': 0.081, 'hbo': 0.043, 'murdered': 0.018}***. The lookup table serves as a tf-idf searching map for us to retrieve the tf-idf value of a single word. GloVe is an unsupervised learning algorithm for obtaining vector representation of words (Pennington et al., 2014). The content of GloVe vectors look like this: ***'odessey': array([0.48395,-1.5084,-0.20004, 0.76474, -0.89189, -0.10859, 0.0076879, -0.45622 , 0.40059 , 0.6946, …])***. We are using glove.6b.50d.tx[4] as a base vector representation which contains word vectors that each has a shape of length 50. The final vector representations of headlines and bodies are computed using Equation 1. From the pre-built tf-idf table, we project each word in headlines and project each word in bodies onto the GloVe vectors. At this point, we have two vectors representations of length 50 where one represents the news headline, and the other represents the new body. By using the two vectors of length 50, we compute the cosine similarity between them.

Cos sim=Tf-idf(word) * GloVe(word) / total tf-idf

Equation 1: Computation of document to glove

TF vectorization
TF-IDF vectorization
Cosine similarity
Kl-Divergence
N-gram Overlap
Polarity
Jaccard similarity

Table 2: Potential features

Cosine similarity	81.0%
N-gram Overlap	79.0% (-2%)
Kl-Divergence	79.4% (-1.6%)
Polarity	74.5%(-6.5%)

[1] https://www.nltk.org/api/nltk.tokenize.html

[2] https://scikitlearn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html#sklearn.feature_extraction.text.TfidfVectorizer.transform

[3] https://competitions.codalab.org/competitions/16843#results

[4] https://www.kaggle.com/pkugoodspeed/nlpword2vecembeddingspretrained?select=glove.6B.50d.txt

Table 3: Accuracies reached by adding more features repeatedly after re-sampling

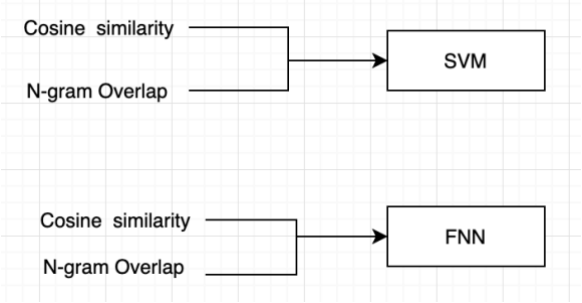

Figure 2: Features and models

- **N-gram Overlap**
  We are using 2-gram overlap to measure how many times a 2-gram that occurs in an article headline re-occurs in the article body. The frequency is calculated using a tf vector representation. The tf lookup table serves as a map to retrieve a word's (n-gram) frequency in a document. Each pair of headline and body are calculated to find out to what extent a body is related to the headline. The final overlap value is calculated using Equation 2.

overlap = [(2-grams occurrence in the headline that reoccurs in the body) / (article length)] to the power of (1/e)

Equation 2: Computation of overlap

3.3 Combined models

As Figure 1 illustrates, two models are working together to complete the entire training and testing job. At the beginning of the development, we employ a set of conventional classifiers; among these classifiers, we try to pick the best one that reaches both high accuracy and high leader board score. Figure 3 is the accuracy distribution of several different conventional classifiers. Figure 4 is the test score reached by using these conventional classifiers. As they show, the accuracies and scores reached are ordinary; not a single classifier reaches a very high accuracy and a very high score at the same time. Inspired by the FNC-1's winner team SOLAT in the SWEN's (Baird et al., 2017) combined approach, we switch to a 2-step approach to obtain the final result.

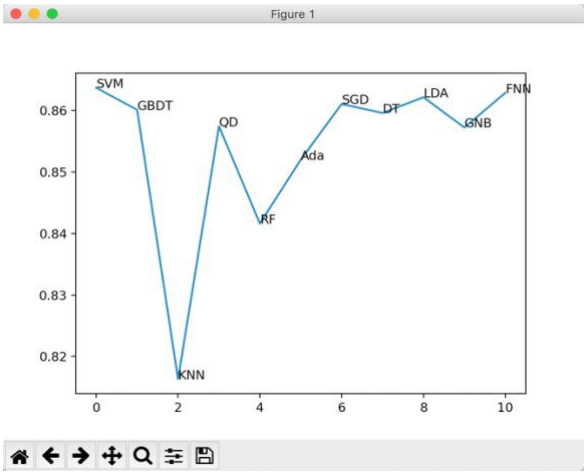

Figure 3: Comparison of accuracy of classifiers

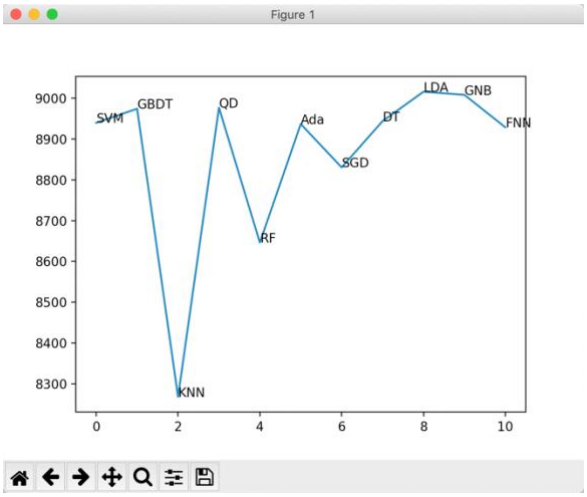

Figure 4: Comparison of the score of classifiers

The author (Baird et al., 2017) employs a combined approach where the final predictions are generated by using both GDBT and neural network. Their final prediction was ensembled based on a 50/50 weighted average between GDBT and CNN. Our final predictions are determined by using both SVM and FNN as Figure 1 shows.

A conventional classifier is employed to distinguish if an instance has a stance of 'related' or 'unrelated'. The binary classification reaches a very high accuracy (over 90%). In this part, SVM is not the only choice that performs the best since the binary classification between 'related' and 'unrelated' is expected to be much easier than classifying related news to 'agree', 'disagree', or 'discuss'. There are many other substitutions reaches above 90% accuracy like SVM. A feed-forward neural network aims to perform a trinary

classification. SVM and FNN are training and testing the same input data set. Predictions generated by the SVM contains [0, 1] labels which map to [related, unrelated] and predictions generated by the FNN contains [0, 1, 2] labels which map to [agree, disagree, discuss]. The final predictions keep all 'unrelated' labels untouched and swapping 'related' to one of [agree, disagree, discuss] generated by the FNN.

An accuracy of 86.5% and a leader board score of 9054 is reached by using this combined approach.

3.4 FNN

Our feed-forward neural network is a simple 3-layer neural network. The input layer uses 'relu' as the activation function, and 'softmax' as the final layer since output are multi-label. The hidden layer uses 'relu' as the activation function. Table 4 shows the hyperparameter configuration of the neural network. Figure 5 shows the schematic diagram of the feed-forward neural network.

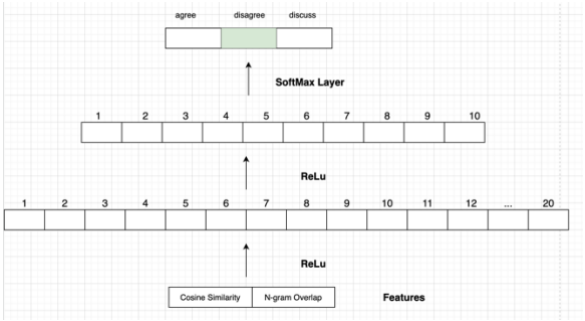

Figure 5: Schematic diagram of the neural network

Dropout rate	0.3
l2 regularization	0.0001
Batch size	5
epochs	50

Table 4: Hyperparameters of FNN

4 Experiments

4.1 Dataset

The dataset of the task is provided by FNC-1 organization (Rao and Pomerleau, 2017); it can be downloaded from the GitHub[5] page. The original data set contains 49972 instances that each row represents a news article. Each row contains:

1. Headline
2. Body ID
3. Stance.

As Table 1 shows, the data set is imbalanced. To build our classification models, we under-sample the data set; each stance(label) contains 840 instances.

4.2 Evaluation & Result

The performance of the system is evaluated using the score function provided by FNC-1. The detailed testing performance is summarized in Figure 6. Before re-sampling, it is noteworthy that the classification of 'unrelated' and 'discuss' is quite good since those two labels take a very large portion (over 90%) of the input training set.

Figure 7 shows the classification detail after over sampling the training set. The accuracy of labelling 'agree' increases. The accuracy of classifying 'discuss' is 54.8%, which cannot be considered as satisfactory, neither can 'disagree'. The performance overall on classifying labels 'agree', 'disagree', and 'discuss' is quite poor. But comparing to baseline which has a 75.2% testing accuracy, our combined approach did improve the leaderboard score and testing accuracy.

```
CONFUSION MATRIX:
-------------------------------------------------------------
|           |   agree   | disagree  |  discuss  | unrelated |
-------------------------------------------------------------
|   agree   |     6     |     0     |   1684    |    213    |
-------------------------------------------------------------
| disagree  |     0     |     0     |    525    |    172    |
-------------------------------------------------------------
|  discuss  |     4     |     0     |   3973    |    487    |
-------------------------------------------------------------
| unrelated |     0     |     0     |    333    |   18016   |
-------------------------------------------------------------
ACCURACY: 0.866

MAX  - the best possible score (100% accuracy)
NULL - score as if all predicted stances were unrelated
TEST - score based on the provided predictions

||    MAX    ||    NULL   ||    TEST   ||
|| 11651.25  ||  4587.25  ||  9036.25  ||
```

Figure 6: Confusion matrix

[5] https://github.com/FakeNewsChallenge/fnc-1

CONFUSION MATRIX:

	agree	disagree	discuss	unrelated
agree	908	51	772	172
disagree	242	33	279	143
discuss	1537	86	2447	394
unrelated	336	113	317	17583

ACCURACY: 0.825

Figure 7: Confusion matrix after re-sampling

From the confusion matrix, the classification of 'agree', 'disagree', and 'discuss' is limited to our feature set. The feature selection is not able to draw a better line to distinguish the three stances. Recall from section 3 Table 2 above, every listed feature aims to find some potential relations between a headline and an article body. However, it does not give us a better performance on determining if an instance's stance is 'agree', 'disagree', or 'discuss' when using different feature combinations. Perhaps the goal in the future development is to carry out more in-depth analyses on a linguistic level.

5 Conclusion

The problem of detecting fake news is a very broad topic which includes detecting any kinds of news or, generally speaking, media. Short tweets, personal opinions, official statements, etc. can all be included in this topic. Thought our solution gives a rather acceptable outcome and results, there are still several points that are needed to be solved to make a generalization, for example, overfitting. The input data set provided is not large enough and our neural network is not complex enough to combat this problem and to generalize. Our feature selection is simple and is not good enough to reveal the inner relation between texts. In fact, even though there are studies done by scholars, the solutions or some projects are data (input) oriented. It means sometimes we cannot not, yet, generalize the solutions to apply models to future and unknown date set to make a trustworthy prediction.

Our model, indeed, beats the baseline model[6] by increasing the accuracy of 11%; however, we would rather treat our solution as an enhanced version of the baseline model. We welcome researchers and future students to improve and extend our solution thus far.

[6] https://github.com/FakeNewsChallenge/fnc-1-baseline.git

References

Shaz Akhtar and Abhinav Kumar Jha. "Automatic Stance Detection.", 2017, *Github*, github.com/Abhinav1004/Fake-News-Stance-Detection.

Monther Aldwairi and Ali Alwahedi. "Detecting. Fake News in Social Media Networks." *Procedia Computer Science*, vol. 141, 2018, pp. 215–22. *Crossref*, doi:10.1016/j.procs.2018.10.171.

Sean Baird and Doug Sibley and Yuxi Pan. "Talos Targets Disinformation with Fake News Challenge Victory." *Blog.Talosintelligence*, William Largent, 20 June 2017, blog.talosintelligence.com/2017/06/talos-fake-news-challenge.html

Benjamin Riedel, Isabelle Augenstein, Georgios P. Spithourakis, and Sebastian Riedel. 2017. A simple but tough-to-beat baseline for the fake news challenge stance detection task. *arXiv preprint arXiv:1707.03264*.

Delip Rao and Dean Pomerleau. 2017. *Fake News. Challenge*. fakenewschallenge.org.

Jiawei Zhang and Bowen Dong and S. Yu Philip. "Fakedetector: Effective fake news detection with deep diffusive neural network." *2020 IEEE 36th International Conference on Data Engineering (ICDE)*. IEEE, 2020.

Andreas Hanselowski, et al. "A retrospective analysis of the fake news challenge stance detection task." *arXiv preprint arXiv:1806.05180* (2018).

Jeffrey Pennington and Richard Socher and Christopher D. Manning. "GloVe: Global Vectors for. Word Representation." *GloVe: Global Vectors for Word Representation*, Jeffrey Pennington, Aug. 2014, nlp.stanford.edu/projects/glove.