Fake News Classification

May 10, 2019

陳君彥, b04703091 b04703091@ntu.edu.tw
陳柔安, b04701232 b04701232@ntu.edu.tw
蕭法宣, b04705007 b04705007@ntu.edu.tw

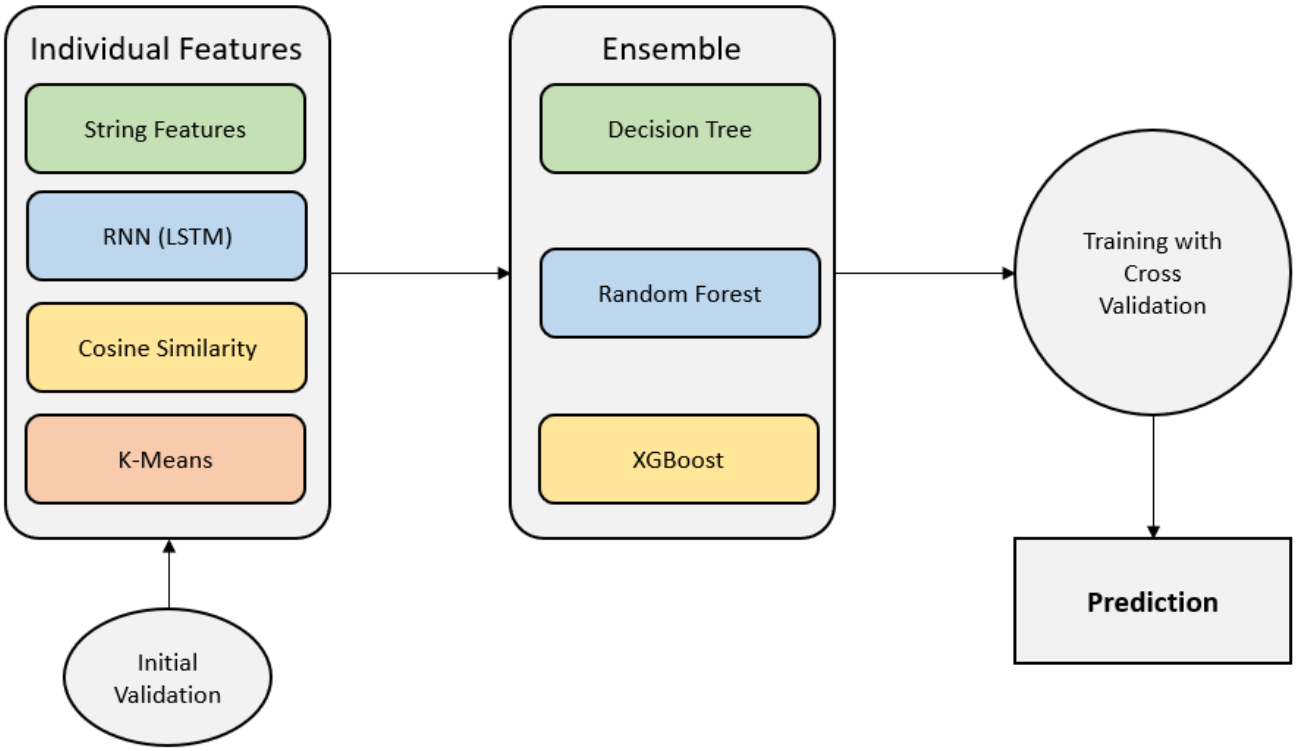

Figure 1: Outline of our prediction method. Extracting useful features of the dataset and ensembled with 3 different aggregation methods. Final model is trained and chosen with cross validation and applied onto the testing data set to make our final prediction.

Division of Work

陳君彥, b04703091:

- Generation and testing of embedding based features.
- Generation and testing of k-means embedding model.
- Creation of written report.

陳柔安, b04701232:

- Creation of ensemble method scripts.
- Further testing of RNN models.
- Testing and submission of final prediction.

蕭法宣, b04705007:

- Initial testing of RNN models.
- Generation and testing of string based features.
- Code project cleanup and sorting.

1 Introduction

The goal of the WSDM 2019 Fake News Classification [1] challenge is to indentify and tags pairs of news title on whether the two titles are 'unrelated' to each other, 'agreed' with each other, or 'disagreed' with each other. The data is given in simplified chinese, with machine translated english version. The first title of the pair was a known 'fake news' article, and so the challenge could lend a hand in identifying misinformation in the real world.

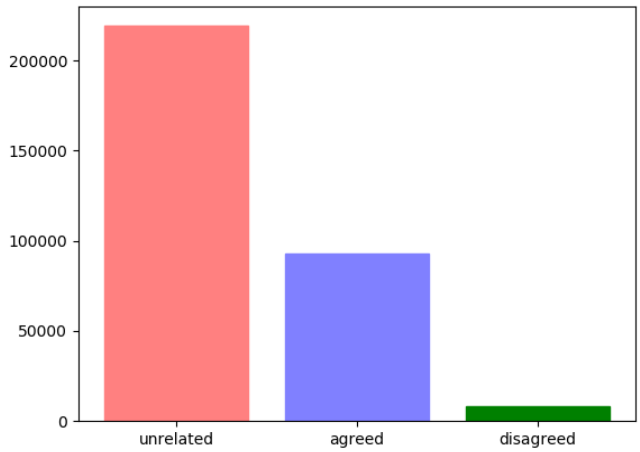

Figure 2: The occurence of each label in the training data set

The data is distributed as shown in Figure 2, with unrelated holding the majority, and with very little data labeled disagree. This proved problematic as it tended to push our models to predict a label as unrelated. Even with the challenge's given weights of 1/16, 1/15, 1/5 for scoring, it still struggled to make a visible difference.

2 Method

We approached this problem as summarized in Figure 1, first by attempting to extract meaningful features in our dataset, testing each individual features, and then combining them in ensemble machine learnin methods to create our final strong prediction. We used cross validation to pick our final trained model, with drop-out and early stopping for regularization.

2.1 Preprocessing

Our dataset is given to us with the titles uncut in chinese, as well as a machine translated english version of the title. As the machine translated version is not fully accurate, we performed most of our analysis on the chinese titles.

Tokenization The chinese titles were cut with jieba [2] with stopwords removed, and the english titles were cut with nltk.word_tokenize [3], with case removed and contractions expanded.

Embedding Model We trained both a chinese and english embedding model using the gensim.Word2Vec [4] using the CBOW embedding model. The training was done on the original training data. The CBOW model uses

2.2 Individual Features

2.2.1 String Features

Overlap Ratio, Partial-Overlap Ratio, Tokenset Overlap Ratio We calculate these three ratios to provide some basic numbers on the similarity between the two titles. These ratios mainly serve to separate titles that are "unrelated" to those that are "related" ("agreed" + "disagreed").

"Rumor" Word Count As the first title in our dataset is always "fake news", we noticed that there are specific "rumor" words that indicate a title is likely to be fake news. Counting the occurence of such words in the second title gives us some indication on whether the second title is also a rumor, lending to us some indications onto whether the titles agree or disagree.

2.2.2 Embedding Features

Sentence Similarity With the trained embedding model, we calculated the combined word vectors for each sentence to create a sentence vector, then calculated a sentence similarity score through cosine similarity. This assisted in the differentiation of "related" and "unrelated" titles.

Noun Object Distance With the trained embedding model, we attempt to calculate the distance between nouns of the two titles, as we imagined that the nouns are more likely to be the focus subject of each title. However, due to the poor performance of jieba's POS tagging, we elected to use the english titles, and picked out the nouns using the POS Perceptron Tagger in NLKT. [5]

2.2.3 K - Means, Transitivity

As the classification for each title are wholly individual, we realized that we can obtain transitivity relations between titles. i.e. if Sentence A agrees with Sentence B, and Sentence C disagrees with Sentence A, we can arrive at the conclusion that Sentence B also disagrees with Sentence C.

Using this property, we tested out a modified k-means model on the sentence vectors acquired through embedding, with the node distances being the cosine similarity of the titles.

2.3 Ensemble

With the above individual weak features of our models, we tested out the following ensemble methods: Decision Tree, Random Forest, XGBoost, and used the earlier features as the inputs to create our prediction model. Through validation testing, XGBoost performed the best and was chosen as our model for the final submission, reaching.

2.4 RNN Models

We noticed the high performance of RNN based models from the submissions of other participants, and decided to further test the following models.

- LSTM
- GRU
- LSTM-bidirectional
- GRU-bidirectional
- LISTM + Word2Vec
- Multilayer GRU

Most notable was the strong overfitting of these models on our dataset, with the in-sample and validation accuracy having a difference of over 10% on several of the models, even with early dropout. The best performing Multilayer GRU achieved a 75% validation accuracy, and a 73.5% accuracy on the submission.

3 Experiments

Before incorporating each feature into our ensemble model, we individually tested some of the features to see if they were valid features that can be used.

3.1 Embedding

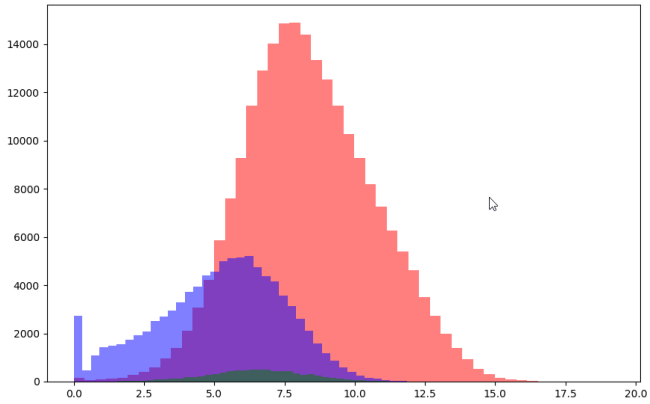

Figure 3: Distribution of ”agreed” (blue), ”disagreed” (green), ”unrelated”(red) with regards to sentence similarity distance

Window	Vocababulary	Accuracy
2	60	0.775235
2	70	0.775923
2	80	0.775693
5	60	0.773631
5	70	0.773828
5	80	0.773926
10	60	0.773173
10	70	0.773500
10	80	0.773959

Figure 4: Finding the best parameters for CBOW model.

We tested if sentence similarity could be used as one method to distinguish ”unrelated” from ”related”. This also helped us pick the optimal CBOW parameters. The distribution shows that ’unrelated’ follows a different distribution to that of ’related’ titles. Using sentence similarity alone we were able to achieve an validation accuracy of 77%, though this resulted in the model largely guessing ’agreed’ for everything else that isn’t ’unrelated’. Hence we needed a different method to separate them.

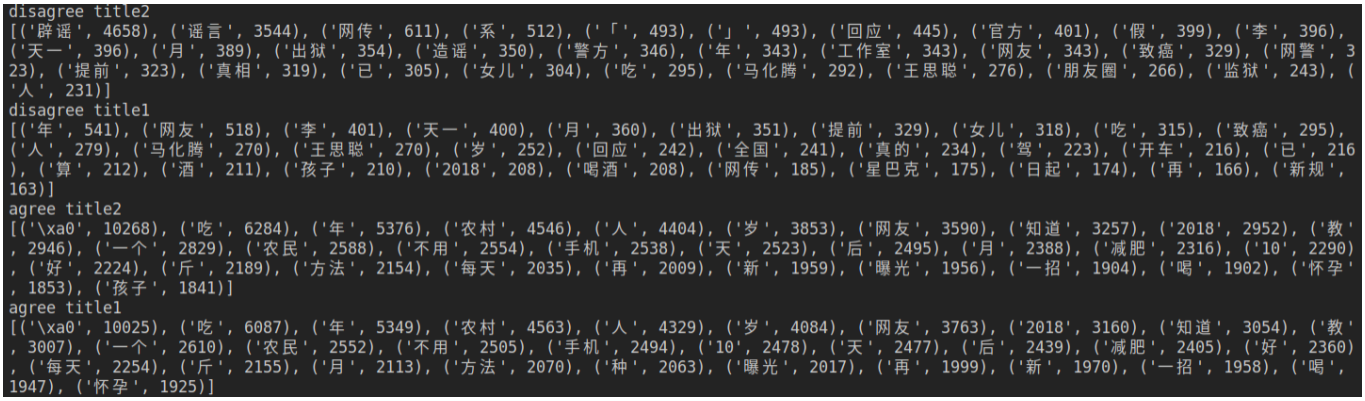

Figure 5: Occurence of words in titles for different labels

3.2 Counting Features

As shown in Figure 5, we noticed some specific words that stood out when looking at 'agreed' and 'disagreed'. With some human knowledge, we picked " 谣", " 官方", " 假", " 真相" and used their occurence count as a feature. This worked out to be the most important in differentiating between related titles.

3.3 Ensemble

Our final submission was the aggregation of the previous features, with Figure 8 being our final results. We tested the addition of each feature, and measured the change in validation accuracy with each additional feature, as shown in Figure 6, and with the importance of each feature as shown in Figure 7

New Feature	Validation Accuracy
String Features	0.81866
+ Sentence Similarity	0.81716
+ Noun Similarity	0.81647
+ K-Means	0.82034

Figure 6: Validation Accuracy for each additional feature

Feature	Importance
overlap ratio	0.498709
partial ratio	0.124975
tokenset ratio	0.04927
rumor word count	0.29825
sentence similarity	0.01066
noun similarity	0.00264
k-means	0.015499

Figure 7: Importance of each feature in final model

Method	Validation	Submission
Decision Tree	0.75037	0.71961
Random Forest	0.80481	0.77659
XGBoost	0.82025	0.79398

Figure 8: Results of the ensemble methods

3.4 RNN

We tested out several different RNN models looking for one that can perform similar or better than our ensemble. While none of the models achieved this, as shown in Figure 3.4, it gave us some additional insight and an idea of what to improve.

However, due to the lack of computing power and time contrainst, we terminated our experiments here, though we surmise that additional layers, and blending with our XGBoost model, the model could perform with a better accuracy.

Model	Validation	Submission
LSTM	0.7448	0.71403
GRU	0.7498	0.71917
LSTM-bidirectional	0.7433	0.71546
GRU-bidirectional	0.7402	0.70804
LSTM + Word2Vec	0.7506	0.71664
GRU Multilayer	0.7742	0.73807

3.5 Failed - Sentiment

We originally intended to include the use of sentiment analysis as a feature. However, upon generating the score, we found out that the scores for the three categories followed almost the exact same distribution, providing absolutely no benefit to our model. We theorize that this is due to the short nature of titles, leading to increased use in "eye catching", "power" words, with little change in sentiment. Combined with the very limit amount of words to extract data from led to all the titles being basically the same.

4 Conclusion

We arrive at the conclusion that the aggregation of different features in our data is an effective method in indentifying and classifying fake news. After discussions with classmates and scouring forums for other people's attempt, we also conclude that this method is also rather efficient.

Our personal machines were all laptop spec'd machines, running the models within an acceptable time frame, with the results being only slightly lower than much more intensive models, such as BERT, which often require much more powerful hardware, and ran upwards of hours when training the models and predicting the results.

References

[1] WSDM. (2019) WSDM - fake news classification. [Online]. Available: www.kaggle.com/c/fake-news-pair-classification-challenge/overview

[2] fxsjy. (2018) Jieba github repository. [Online]. Available: github.com/fxsjy/jieba

[3] E. K. Steven Bird, Edward Loper. (2018) Natural language toolkit. [Online]. Available: www.nltk.org/api/nltk.tokenize.html

[4] R. Řehůřek. (2018) Gensim model. [Online]. Available: radimrehurek.com/gensim/models/word2vec.html

[5] E. K. Steven Bird, Edward Loper. (2018) NLTK - Perceptron Tagger. [Online]. Available: www.nltk.org/_modules/nltk/tag/perceptron.html