Embedding-based Approaches to Hyperpartisan News Detection

Pengyu Chen
pchen@cs.toronto.edu
Karthik Mohan
karthik.mohan@mail.utoronto.ca

Abstract

In this report, we describe our systems in which the objective is to determine whether a given news article could be considered as hyperpartisan. Hyperpartisan news is news that takes an extremely polarized political standpoint with an intention of creating political divide among the public. We attempted several approaches, including n-grams, sentiment analysis, as well as sentence and document representation using pre-tained ELMo. Our best system using pre-trained ELMo with Bidirectional LSTM achieved an accuracy of around 83% through 10-fold cross-validation without much hyperparameter tuning.

1 Introduction

1.1 Hyperpartisan News

The rise of social media and online communication has enabled people to share information with a large audience. The mask of anonymity and the lack of regulations and quality control allow malicious users to spread fake news in a destructive speed. Hyperpartisan news is a type of fake news that is typically highly polarized (hyper-partisan), emotional, and untruthful to mislead the public. Hyper-partisan articles mimic the form of regular news articles; however, they are one-sided and highly polarized in the sense that opposing voices are either deliberately ignored or attacked (Kiesel et al., 2019). Being able to detect whether a source is hyper-partisan can bring us one step closer to solving the automated fake news detection problem.

1.2 Related Work

The analysis of political orientation, bias, and misinformation has attracted significant attention, especially after the 2016 US presidential election. Various machine learning approaches have been made. Pla and Hurtado (2014) proposed an SVM model for political tendency identification using features such as n-grams and part-of-speech tags. Preotiuc-Pietro et al. (2017) used a linear regression model to characterize the political groups of users through language use on Twitter. In order to solve fake news problem, knowledge-based (Lee et al., 2018) and style-based approaches were proposed (Potthast et al., 2018). Furthermore, Barron-Cedeno et al. (2019) present a publicly available real-time propaganda detection system for online news using features such as n-grams and lexicon features. Some fake news datasets are publicly available, but they are often too small to be suitable for deep learning methods.

Last year, the organizers of SemEval2019 (Kiesel et al., 2019) released a large-scale dataset with over 1M articles in Task 4, Hyper-partisan News Detection, where data are labeled with the tagset hyperpartisan, not-hyperpartisan. Considering that many previous works have not utilized deep learning methods for hyperpartisan detection, one of our goals is to narrow down this gap by exploring how well deep learning can handle this task. We employed feed-forward neural networks and Convolutional Neural Networks (CNN). We also employed various types of features including sentence embeddings, n-grams, and sentiment and emotion features.

The rest of the report is organized as follows: data and task definition are described in Section 2, Section 3 describes our methods, followed by experiments and results in Section 4. Finally, we discuss future works and conclude this report in Section 5.

2 Data

In this project, we use the data provided by SemEval2019 task 4 (Kiesel et al., 2019). The task is set up as a binary classification problem where news articles are labeled with the target {hyperpartisan, not-hyperpartisan}.

Two types of dataset are provided, pertaining how labels were obtained. The first dataset (*by-article* corpus) has 1,273 articles, each labeled manually by 3 annotators at the article level (Vincent and Mestre, 2018). Out of the 1,273 labeled articles, only 645 were released by the organizer (238 hyperparsitan and 407 non-hyperpartisan), whereas the other 628 (50% hyperpartisan and 50% not) were reserved private for the evaluation during the competition. One of the major challenges is the range of article sizes. The maximum, mean, and minimum numbers of tokens in the by-article set are: 6470, 666, 19 respectively, making it difficult to directly input word representations as features to the neural network. As a compromise, sentence representations are used for each article by averaging the word embeddings of the corresponding sentence.

The second, larger dataset (*by-publisher* corpus) contains 754,000 articles which were automatically labeled based on a categorization of the political bias of news publishers. This dataset was split into a training set of 600,000 articles and a validation set of 150,000 articles. At first, we expected the use of by-publisher training data can help classify the by-article set, but it turned out that though much larger than the by-article set, its labels contain significant amount of noisy introduced by the labeling program (Kiesel et al., 2019). The utilization of the by-publisher set seems more difficult than expected and therefore we decided not to use it.

3 Methodology

3.1 Features

3.1.1 N-Grams

The first obvious approach for the task was employing classic ML classifiers using the simple n-gram features. We built a baseline system using unigram and bigram features over a range of term frequency cutoff. The performance peaked when the cutoff is near 10 - 15. We tested a variety of classifiers including Logistic Regression, SVM, Random Forest, and Gradient Boosting Tree.

3.1.2 Emotionality (Sentiment Analysis)

Valence (sentiment) and polarity analysis is another approach that seems suitable for the task assuming that hyperpartisan news typically involves a high intensity of valence, subjectivity, and polarity because of its biased nature. We used the following features: polarity, subjectivity; and positive, negative and neutral scores, which are obtained from textblob and NLTK text analysis.

3.1.3 Word Embeddings

Word embeddings are commonly used in many NLP tasks in recent years because they are found to be useful representations of words and often lead to better performance. The input to neural networks is a set of pretrained text representations such as Word2Vec (Mikolov et al., 2013), Glove (Pennington et al., 2014), ELMo (Peters et al., 2018), or BERT (Devlin et al., 2018). In our system we use ELMo embeddings which have the advantage of modeling polysemy and morphological features that word-level embeddings could omit. We represent the article as a sequence of sentence embedding by averaging the word embeddings of that sentence. Considering the range of article size, only the first 250 sentences of an article and the first 250 tokens of a sentence was used for each article representation. We also tried the pretrained BERT for word embedding. The drawback of

Table 1: N-grams Models

Models	Accuracy
Logistic Regression	0.7535
SVM	0.7380
Random Forest	0.7534
XGBoost	0.7752
Ensemble	0.7690

using BERT is that we were limited to a sequence length of 512, hence we had to restrict the article sequence length to 512 and continue the training process. The Accuracy is 0.81. However this may not be a good embedding model since we are abruptly restricting the article to 512 word embedding.

3.2 Classifiers and Model Architecture

3.2.1 Feedforward Neural Networks

We first built a system using only fully connected layer with a sequence of sentence embeddings as input as described above. Batch Normalization (Ioffe and Szegedy, 2015) and dropout (Srivastava et al., 2014) was used for better generalization. The network has two hidden fully connected layers with ReLU activation and an output layer with sigmoid activation.

3.2.2 Convolutional Neural Networks

Emoloying convolving filters over neighboring words to encode information about the article seems to be another good idea. Our CNN system consists of 5 parallel convolutional layers with filter sizes 2, 3, 4, 5, 6. After each convolutional layer follows a ReLU activation function, batch normalization layer, and max pooling. The 5 parallel outputs are concatenated and fed into a fully connected layer with a sigmoid non-linearity.

3.2.3 Long Short Term Memory Networks (LSTMs)

Long Short Term Memory Networks (Hochreiter and Schmidhuber, 1997) have been successfully applied to many text classification problems. RNNs are good summarizer of sequantial information such as language yet suffer from gradient issues. LSTMs solve this issue to some extent. A variation of LSTM is Bidirectional LSTMs (Bi-LSTMs), which summarizes information from both left to right and vice versa. We investigate LSTMs and Bi-LSTMs on this task.

3.2.4 Other Variety of Classifiers

We also used a variety of traditional machine learning classifiers using n-grams or emotionality scores as input, including Logistic Regression, SVM, Random Forest, Gradient Boosting Tree, and their ensembles.

4 Experiments and Results

4.1 N-gram Models

Table 1 shows the results of our system using unigrams and bigrams with a frequency cutoff 12. Numbers appear in articles are all replaced by a token <num>. Gradient Boosting Trees turns out to be the winner, achieving an 0.775 average accuracy over 10-fold cross validation. In the second place is the ensemble of the four classifiers, indicating that the prediction of all the four models are almost identical so we can't benefit from ensembling.

4.2 Sentiment Analysis

Table 2 shows the results of our system using polarity, subjectivity, and valence scores. The results are much worse than n-gram systems. There could be three reasons for the poor performance. First,

Table 2: Sentiment Analysis Models

Models	Accuracy
Logistic Regression	0.6372
SVM	0.6310
Random Forest	0.6881
XGBoost	0.7021
Ensemble	0.6975

Table 3: Neural Networks

Models	Accuracy
Feedforward	0.8280
CNN	0.8095
LSTM	0.8218
Bi-LSTM	0.8372

the methods used to generate the scores were not very optimal. Second, to fully leverage this methods, we should use more hand-crafted lexicon-based features. Finally, we don't have enough training data to build our own sentiment analysis model.

4.3 Neural Networks

Table 3 illustrates the average accuracy over 10-fold cross validation obtained on the systems of feedforward neural networks, CNN, and LSTM. Dropout is used for feedforward networks and LSTM and batch normalization is used for feedforward networks and CNN. We used the default Adam as the optimizer and binary cross-entropy as the loss function. We can see a huge improvement by leveraging the complexity of neural networks. The four systems are comparable but Bi-LSTM slightly outperforms the others.

4.4 Overfitting Investigation

Overfitting has been a problem throughout the entire experiments. This is reasonable considering the size of training size is only 645. Thus, we conducted another experiment on how the number of training examples will influence performance. We randomly chose 80 examples as test set and trained feedforward networks using 50, 100, 200, 350, and 565 examples from the rest of the dataset. Figure 1 illustrates the result, suggesting that our system could perform even better if given a larger training set.

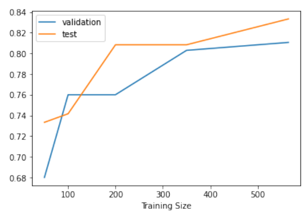

Figure 1: Training size vs. performance

5 Discussion and Conclusion

In this report, we present our systems on hyperpartisan news detection. We trained a set of models using a variety of features including n-grams, sentiment scores, and word embeddings. The Bi-LSTM system using ELMo embedding acquired a 83.72% accuracy through 10-fold cross validation without much hyperparameter tuning. We also investigated into the overfitting problem and concluded that the potential of deep learning methods has not been fully exploited due to the small training set.

A few things could be done in future work. First, it would be helpful to have a larger sized dataset to handle overfitting issues. Second, we haven't tuned hyperparameters for each network. Finally, we should try other state-of-the-art models and more expressive network architectures.

References

[1] Johannes Kiesel, Maria Mestre, Rishabh Shukla, Emmanuel Vincent, Payam Adineh, David Corney, Benno Stein, and Martin Potthast. 2019. SemEval–2019 Task 4: Hyperpartisan News Detection. In *Proceedings of The 13th International Workshop on Semantic Evaluation*. Association for Computational Linguistics.

[2] Ferran Pla and Lluis-F Hurtado. 2014. Political tendency identification in twitter using sentiment analysis techniques. In *Proceedings of COLING 2014, the 25th international conference on computational linguistics*: Technical Papers, pages 183–192.

[3] Daniel Preotiuc-Pietro, Ye Liu, Daniel Hopkins, and Lyle Ungar. 2017. Beyond binary labels: political ideology prediction of twitter users. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics* (Volume 1: Long Papers), volume 1, pages 729–740.

[4] Nayeon Lee, Chien-Sheng Wu, and Pascale Fung. 2018. Improving large-scale fact-checking using decomposable attention models and lexical tagging. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1133–1138.

[5] Martin Potthast, Johannes Kiesel, Kevin Reinartz, Janek Bevendorff, and Benno Stein. 2018. A Stylometric Inquiry into Hyperpartisan and Fake News. In *56th Annual Meeting of the Association for Computational Linguistics* (ACL 2018), pages 231–240. Association for Computational Linguistics

[6] Alberto Barron-Cedeno, Giovanni Da San Martino, Israa Jaradat, and Preslav Nakov. 2019. Proppy: A system to unmask propaganda in online news. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence* (AAAI'19), AAAI'19, Honolulu, HI, USA.

[7] Emmanuel Vincent and Maria Mestre. 2018. Crowdsourced Measure of News Articles Bias: Assessing Contributors' Reliability. In *Proceedings of the 1st Workshop on Subjectivity, Ambiguity and Disagreement (SAD) in Crowdsourcing*, pages 1–10.

[8] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

[9] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

[10] Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365.*

[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *CoRR*, abs/1810.04805.

[12] Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167.*

[13] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958.

[14] Sepp Hochreiter and Jurgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.