Tom Jumbo-Grumbo at SemEval-2019 Task 4: Hyperpartisan News Detection with GloVe vectors and SVM

Chia-Lun Yeh[1], **Babak Loni**[2], **and Anne Schuth**[2]
[1]TU Delft, The Netherlands
[2]De Persgroep
c.yeh-1@student.tudelft.nl
{babak.loni, anne.schuth}@persgroep.net

Abstract

In this paper, we describe our attempt to learn bias from news articles. From our experiments, it seems that although there is a correlation between publisher bias and article bias, it is challenging to learn bias directly from the publisher labels. On the other hand, using few manually-labeled samples can increase the accuracy metric from around 60% to near 80%. Our system is computationally inexpensive and uses several standard document representations in NLP to train an SVM or LR classifier. The system ranked 4th in the SemEval-2019 task. The code is released for reproducibility[1].

1 Introduction

Bias is the inclination or prejudice for or against one person or group. News articles that contain extreme bias fail to provide fair and multifaceted views for readers and can create polarization within the society (Bernhardt et al., 2008). A system that can detect bias in news articles is thus relevant, especially in a time where an increasing number of people consume news from online sources that might not be trustful.

The SemEval-2019 task aims to detect hyperpartisan news given the text of the news article, where hyperpartisan news is defined to be an article that overtly favors a side or view. The details of the task can be found in Kiesel et al. (2019). We are provided with a dataset of two parts. The first part is labeled by the publishers (e.g. if a publisher is decided to be a hyperpartisan source, all its articles are labeled as hyperpartisan), and split into a training and validation set with no overlapping publishers (which we will refer to as training-1 and validation-1). The second part is crowd-sourced and labeled per article (which we will call training-2).

Due to the large number of labeled samples, we decide to use a supervised classification approach, where features are extracted from the text and used to train a classifier. Bag-of-words (BoW), TFIDF weighting, and n-grams have been shown to be strong baselines (Hu and Liu, 2004; Wang and Manning, 2012). Other features such as Part-Of-Speech (POS), counts of sentiment and bias words have also been studied (Liu, 2012; Mukherjee and Weikum, 2015). In a similar setting, Potthast et al. (2018) uses features such as n-gram of characters, readability scores, dictionary, and the ratio of quoted words to separate hyperpartisan news from the mainstream. They trained a random forest classifier and achieved an accuracy of 75%.

Kulkarni et al. (2018) build a neural network to predict the political ideology of news articles to be either left, right or center. They combine information from the headlines, the links within an article, and the content. They use a CNN (Kim, 2014) for the headlines, a Node2Vec (Grover and Leskovec, 2016) to model the links and a hierarchical attention network (HAN) (Yang et al., 2016) to extract features from the content. They compare the model with several baselines, including a BoW LR model, a fully-connected feedforward network, and networks with only the individual components. Their proposed model performs the best. However, their system is trained and evaluated on only data with publisher labels. They randomly split them into training and testing sets, with overlapping publishers.

The main contribution of the paper is two-fold. First, we analyze the problem of using the dataset labeled by publishers, concluding that it is difficult due to the noisy labels. Second, we train SVM classifiers with different representations: TFDIF, doc2vec and GloVe pre-trained vectors. The 300-dimensional GloVe vectors obtain the best cross-validation accuracy as well as the performance metrics on the official test data.

[1] https://github.com/chialun-yeh/SemEval2019

Proceedings of the 13th International Workshop on Semantic Evaluation (SemEval-2019), pages 1067–1071
Minneapolis, Minnesota, USA, June 6–7, 2019. ©2019 Association for Computational Linguistics

This paper is organized as follows. In section 2, we describe the data pre-processing. In section 3, we present the two systems that we devise and explain how one motivates the other. In section 4, we present the performance of the final system. We outline our main conclusions and future work in section 5.

2 Pre-processing

Since the articles are collected from online news platforms, they contain texts that are irrelevant to the news itself. We use the following three steps to clean the data:

(a) Remove online usage including links, hashtags, @-tag, and advertisements.

(b) Remove parentheses, brackets, and curly brackets that contain additional information because the usage is often specific to publishers.

(c) Remove paragraphs that might reveal publisher information. Some publishers use headers and footers of specific patterns in their articles. We try to remove them by discarding the first and last paragraphs from the article if the article has more than two paragraphs, assuming that these two paragraphs have higher probabilities of being headers and footers. This is by no means optimal since the first paragraph often contains important content if it is not a header. Some publishers also inserted short text such as ”read more here” between paragraphs. To remove these irrelevant texts that can reveal publisher pattern, we remove any paragraph with less than ten words. Any article with less than ten words after the cleaning is discarded.

We consider (a) and (b) as basic data cleaning and apply them on all data. On the other hand, (c) is a more aggressive cleaning that is done only on training-1. This is because we have a comparatively large training set where we can afford filtering out information and even entire articles.

3 System Description

3.1 System 1

In the first method, we use training-1 to train our models, validation-1 to choose hyperparameters, and training-2 to test the models. As mentioned earlier, training-1 is labeled by publishers. While a biased publisher publishes more biased articles on average, it is unlikely that all of its articles are biased. Therefore, the labels are noisy, e.g., some labels are flipped. It is, however, difficult to identify the articles that have the wrong labels without manual inspection. We assume that the publisher labels are correlated with true bias labels, thus providing information to learn bias. To have an idea of to what extent this assumption holds, we investigate training-2. We select publishers of whom at least five articles are included in the dataset and whose media bias can be retrieved from Media-Bias/FactCheck[2]. This results in a total of 24 publishers. The publisher bias ratings on the website can be roughly mapped to 7 categories, extreme-left, left, left-center, center, right-center, right, and extreme-right. In Table 1, we list these publishers along with the percentage of the articles that are rated as hyperpartisan by crowd workers. The number of articles per publisher range from 5 to 24. Figure 1 shows the percentage of hyperpartisan articles in each category. We see that left-center and center publishers indeed have considerably less percentage of hyperpartisan articles. However, right-center publishers are almost as biased as right publishers. The observation can be due to the small sample size (the high percentage is caused mainly by the publisher RealClearPolitics). In general, there is a correlation between the publisher and true hyperpartisanship.

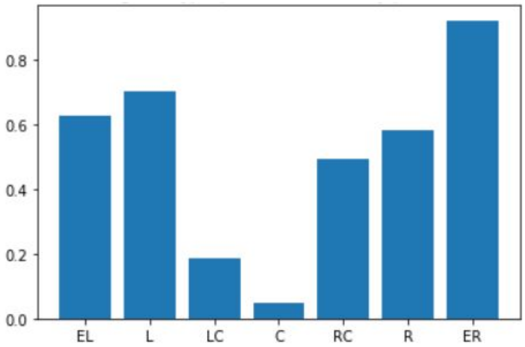

Figure 1: Percentage of hyperpartisan articles in the 7 bias categories: extreme-left (EL), left (L), left-center (LC), center (C), right-center (RC), right (R), and extreme-right (ER).

We use BoW and n-grams (n=1,2) as features, with different weighting schemes, including raw counts, binary, and TFIDF. For BoW and n-grams, the feature dimension is 50K and 500K respectively. We train two classifiers on each representation. The accuracy of the classifiers on validation-1 is listed in Table 2. We include experiments where training-1 is not cleaned with pre-

[2] https://mediabiasfactcheck.com/

processing step (c) to make sure that the step helps the task.

From the result, we observe that adding bigrams doesn't improve accuracy. We use the best model (BoW and an SVM classifier) to predict the articles in training-2. The accuracy is 56%, which is lower than the majority baseline of 63%.

Although we clean the dataset in an effort to prevent the classifier from overfitting on the publisher, it seems that the classifier cannot generalize to unseen publishers, and fails to capture bias. We also experiment with training a CNN (Kim, 2014) with the headlines, and a HAN (Yang et al., 2016) with the content. However, the two models again fail to generalize to new publishers. The observation makes us believe that the publisher labels are too noisy to be used directly to learn true bias. Another possible explanation could be that the publishers have too distinct writing styles so that the classifier focuses much on those features when learning.

3.2 System 2

Due to the observation in system 1, we decide to treat training-1 and validation-1 as unlabeled samples that can be used to train a feature extractor in an unsupervised setting. We then train the classifier using training-2. We use the first part of the data by the following two extractors.

1. TFIDF: The data is used to build vocabulary and record the inverse document frequency. All terms that occur in more than 90% of the documents are discarded, and we kept the most frequent 50K terms.

2. Doc2Vec: The data is used to train a PV-DM model proposed by Le and Mikolov (2014). We discard all terms that occur in less than 10 documents or are shorter than two characters. We train the model for 20 epochs using the implementation of gensim (Řehůřek and Sojka, 2010). When inferencing new documents, the word vectors are fixed and the model is trained for 100 epochs.

In addition, we experiment with using pre-trained word embeddings since the meaning of each word should not differ significantly in different corpora. We use vectors trained with GloVe algorithm (Pennington et al., 2014) on Wikipedia and Gigaword 5 [3]. The vectors are chosen because they are trained on Wikipedia and newswire text, which provides general knowledge and news domain specific usage. We take the vectors of each word in the document and average all the vectors. Stop words are removed, and if the document has more than 1000 words, we average over the first 1000 words (we find this to work better in our case empirically).

We also experiment with a set of features including normalized count of 5 POS tags, 6 readability scores, 8 normalized sentiment and bias word counts according to MPQA and bias lexicons (Wilson et al., 2005; Recasens et al., 2013), number of quotes, words, capitalized words, stop words, and sentences, and average length of words and sentences. This result in a total of 27 features which we call Feat.

For supervised training, we split training-2 into two sets. The first half, with 322 samples, is used to train and choose hyperparameters in a 10-fold cross validation setting. The second half, with 323 samples, is used for testing. We train LR and SVM on the features. Both linear SVM and SVM with rbf kernels are experimented with. We also have some initial experiments of single layer and two-layer neural networks of different hidden layer sizes but the small sample size makes them difficult to generalize.

4 Results

We first train LR and SVM with different GloVe vector dimensions. Table 3 shows the accuracy on the test set. SVM with rbf kernel works consistently better. The best vector dimension is 300.

We then compare different features, including TFIDF, Doc2Vec, GloVe, and the effect of adding Feat. Table 4 shows the accuracy on the test set. It shows that SVM performs better than LR, and only in the case of TFIDF does a linear SVM outperforms kernel SVM. It also shows that the pre-trained GloVe vectors achieve better performance than the vectors that are trained on our data. The ability to generalize might result from the larger corpus that is used to train the vectors. Adding simple lexical and sentiment features hurts the performance.

The three representations are furthered evaluated on another test set (the official test set of the

[3] https://catalog.ldc.upenn.edu/LDC2011T07

category	publisher	doc(%)	category	publisher	doc(%)
extreme-right	thegatewaypundit.com	94.44	left	salon.com	100.00
extreme-right	dcclothesline.com	85.71	left	gq.com	60.00
extreme-left	trueactivist.com	62.50	left	rawstory.com	40.00
right	pjmedia.com	100.00	left	opednews.com	100.00
right	express.co.uk	36.84	left	people.com	20.00
right	opslens.com	100.00	right-center	realclearpolitics.com	92.86
right	insider.foxnews.com	27.27	right-center	circa.com	12.50
right	foxnews.com	50.00	left-center	cbsnews.com	11.11
right	washingtonexaminer.com	57.14	left-center	heavy.com	7.69
right	bizpacreview.com	40.00	left-center	nytimes.com	30.00
right	nypost.com	66.67	center	snopes.com	8.33
right	bearingarms.com	66.67	center	nfl.com	0.00

Table 1: Selected publishers with their bias categories and percentage of biased articles in the dataset.

Features	Classifier	
	LR	SVM
BoW (without (c))	58.83	59.72
BoW	60.67	**60.93**
BoW-binary	60.61	60.68
BoW-TFIDF (without (c))	60.15	59.61
BoW-TFIDF	60.86	60.90
N-grams	60.73	59.13
N-grams-binary	60.18	59.74
N-grams-TFIDF (without (c))	59.65	59.72
N-grams-TFIDF	60.51	60.61

Table 2: Validation accuracy after fine-tuning. Without (c) means that the training set is not cleaned with the pre-processing step (c). Cleaning helps improve accuracy.

Features	Dim.	LR	SVM
GloVe	100	72.45	78.33 (rbf)
GloVe	200	72.76	76.78 (rbf)
GloVe	300	72.45	**79.57** (rbf)

Table 3: Accuracy of different GloVe vector dimensions.

Features	Dim.	LR	SVM
TFIDF	50K	77.09	77.71 (linear)
GloVe	300	72.45	**79.57** (rbf)
GloVe + Feat	327	75.85	78.33 (rbf)
Doc2Vec	400	71.83	78.95 (rbf)
Doc2Vec + Feat	427	77.71	75.85 (rbf)

Table 4: Accuracy of our model that is trained using training-2. The majority baseline is 63% accuracy.

task) that is labeled by crowd workers. Since the additional feature set does not improve the performance, it is not further evaluated. In Table 5, the accuracy, precision, recall, and F1-score on the held-out test set are shown. Our classifiers tend to have a higher false negative rate. This can be due to the imbalance in the training data. Further experiments would be required to see whether resampling to have a balanced training set can improve that.

Features	Acc.	Precision	Recall	F1
TFIDF	74.36	80.00	64.97	71.70
GloVe	**80.57**	85.82	73.25	79.04
Doc2Vec	73.89	82.61	60.51	69.85

Table 5: Submission results on the held-out test set, with metrics including accuracy, precision, recall, and F1-score.

5 Conclusion and Future Work

In this paper, we present the system we use to compete in the SemEval-2019 hyperpartisan news detection task. The final model we use is a kernel SVM trained with pre-trained GloVe vectors. It turns out that a simple method which requires the least training time performs the best in this case.

Both system 1 and system 2 have interesting future work to be done. For system 1, it is interesting to correct the labels or filter the articles in order to obtain a cleaner data to learn from. For system 2, we plan to use contextual embeddings (Peters et al., 2018) or pre-trained language models (Radford, 2018; Devlin et al., 2018) to extract representations that are then fed into downstream classifiers. The high performances of the models made them interesting to compare with.

References

Mark Daniel Bernhardt, Stefan Krasa, and Mattias K Polborn. 2008. Political polarization and the electoral effects of media bias. *Journal of Public Economics*, 92(5-6):1092–1104.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Aditya Grover and Jure Leskovec. 2016. Node2vec: Scalable feature learning for networks. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 855–864, New York, NY, USA. ACM.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, pages 168–177, New York, NY, USA. ACM.

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 (SemEval 2019)*. Association for Computational Linguistics.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751. Association for Computational Linguistics.

Vivek Kulkarni, Junting Ye, Steve Skiena, and William Yang Wang. 2018. Multi-view models for political ideology detection of news articles. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3518–3527. Association for Computational Linguistics.

Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ICML'14, pages II–1188–II–1196. JMLR.org.

Bing Liu. 2012. *Sentiment Analysis and Opinion Mining*. Morgan & Claypool Publishers.

Subhabrata Mukherjee and Gerhard Weikum. 2015. Leveraging joint interactions for credibility analysis in news communities. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, CIKM '15, pages 353–362, New York, NY, USA. ACM.

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. Association for Computational Linguistics.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237. Association for Computational Linguistics.

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.

Alec Radford. 2018. Improving language understanding by generative pre-training.

Marta Recasens, Cristian Danescu-Niculescu-Mizil, and Dan Jurafsky. 2013. Linguistic models for analyzing and detecting biased language. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1650–1659. Association for Computational Linguistics.

Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta. ELRA. `http://is.muni.cz/publication/884893/en`.

Sida Wang and Christopher D. Manning. 2012. Baselines and bigrams: Simple, good sentiment and topic classification. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers - Volume 2*, ACL '12, pages 90–94, Stroudsburg, PA, USA. Association for Computational Linguistics.

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT '05, pages 347–354, Stroudsburg, PA, USA. Association for Computational Linguistics.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489. ACL.