TakeLab at SemEval-2019 Task 4: Hyperpartisan News Detection

Niko Palić, Juraj Vladika, Dominik Čubelić, Ivan Lovrenčić,
Maja Buljan, Jan Šnajder
Text Analysis and Knowledge Engineering Lab
Faculty of Electrical Engineering and Computing, University of Zagreb
Unska 3, 10000 Zagreb, Croatia
{name.surname}@fer.hr
borat-sagdiyev team

Abstract

In this paper, we demonstrate the system built to solve the SemEval-2019 task 4: Hyperpartisan News Detection (Kiesel et al., 2019), the task of automatically determining whether an article is heavily biased towards one side of the political spectrum. Our system receives an article in its raw, textual form, analyzes it, and predicts with moderate accuracy whether the article is hyperpartisan. The learning model used was primarily trained on a manually prelabeled dataset containing news articles. The system relies on the previously constructed SVM model, available in the Python Scikit-Learn library. We ranked 6th in the competition of 42 teams with an accuracy of 79.1% (the winning team had 82.2%).

1 Introduction

The ability to quickly, precisely, and efficiently discern if a given article is hyperpartisan can prove to be beneficial in a multitude of different scenarios. Should we, for example, wish to evaluate if a certain news publisher delivers politically biased content, the best way to do so would be analyzing that very content. However, the sheer amount of articles modern news companies produce nowadays asks for an automated approach to the problem.

Spotting bias in text is both a well-known and challenging natural language problem. As bias can manifest itself in a covert or ambiguous manner, it is often hard even for an experienced reader to detect it. There was some research done on similar issues before (Doumit and Minai, 2011), but none specifically on the subject of hyperpartisan news.

The system described in this paper was built for Task 4 of the SemEval-2019 competition. The goal of the system, as set by the task, is to predict, as accurately as possible, whether a given article is hyperpartisan. While there were other criteria for evaluating the performance of the model (precision, recall, F1), we decided to optimize the program for the accuracy criterion, as the rankings were based solely on this measure. Accuracy, in this context, indicates the ratio of correctly predicted articles to the total number of articles. The final model reached an accuracy of 79.1%, which presents a decent score, considering the complexity of the problem and the available technology.

The system we built is based on the SVM model publicly available in Python's SciKit-Learn library (Pedregosa et al., 2011). Our model was trained on a handful of carefully chosen features derived from the given dataset and our understanding of the nature of bias. The dataset was split into a high-quality, manually labelled set of articles, and a large, but sub-par set of automatically labelled articles. By experimenting with the datasets, models, and features, we managed to create a system which ranked 6th in this competition.

2 Dataset Description

The dataset, which was provided by the task organizers, was divided into two separate clusters. The first and larger cluster consisted of one million news articles labelled solely by the political affiliation of the *publisher*. The second and much smaller cluster consisted of one thousand news articles labelled by people who read and evaluated them.

There was a substantial difference in labelling between these two datasets, so the quality and accuracy of the smaller dataset greatly overshadowed the abundance of articles in the larger dataset. Furthermore, the articles mostly came from large U.S. news publishers, such as: Fox News, CNN, Vox, etc. This predominance of prominent U.S. news networks and the substantial difference in quality largely impacted our feature

Proceedings of the 13th International Workshop on Semantic Evaluation (SemEval-2019), pages 995–998

design and, ultimately, our model selection.

The task itself was not divided into subtasks, but the submission on these two different datasets was regarded as a different subtask. The final test set, on which the main leaderboard is made, consisted solely of the articles from the smaller and more accurate dataset.

3 Model Description

Model selection and feature design were vastly influenced by the radical difference in quality between the two given datasets. As we mentioned, the larger, publisher-based dataset had a high number of mislabelled articles. For example, articles about diet tips, weather forecasts, or some other non-political news were often labelled as hyperpartisan news, solely on the fact that the publisher is classified as a generally biased news source. This mislabelling often caused our models to wrongly guess on what really indicates hyperpartisanism in news articles. Since the larger dataset often gave us relatively low accuracy, we decided to try a different approach.

Our first submitted model was trained only on the smaller and more accurate dataset. We decided to use SVC with GridSearch to maximize our accuracy on such a small dataset. Our best accuracy on the validation set, after cross-validation and with all features in place, was 77.9%. Furthermore, for our second submitted model, we decided to use a self-learning method to acquire more quality data from the larger dataset. Our first step was to train a logistic regression model on one-half of the smaller dataset, and, once trained, we tested that model on the larger dataset. All correctly classified articles were added to a new dataset. Once we extracted and combined all high-quality articles, we again used the SVC model. Our final accuracy on this model was also 77%.

4 Features

Our main tool used for processing news articles was word2vec[1] (Mikolov et al., 2013). Each word was converted into a vector, and all the word vectors generated from an article were then summed up. To prepare the dataset for word2vec, we used stemming and lemmatization tools from NLTK toolkit (Bird et al., 2009), and eliminated standard English stopwords. Also, in our earlier stages of development, we used chunking to get more meaningful information from the text.

We ultimately ended up with a 300-dimensional vector for every article. This was the result of passing the preprocessed text into word2vec. Below, we elaborate on some other features we introduced to our model.

Publication date
Hyperpartisanism, or extreme bias, is a classification closely related to politics. As we were dealing predominatly with U.S. news outlets, we reasoned that news articles occurring around certain dates could demonstrate more hyperpartisanism. For example, months leading up to and following annual U.S. elections could be a mild indication of hyperpartisanism. Our first intuition was to only include months as a feature, since, in the U.S., certain election processes take place yearly. With further tests, we found that, if the year was also included, the accuracy improved by over 2%. That improvement could mean not only that elections produce more hyperpartisan news, but the type of the election, and maybe even the winner, could have an impact on news bias.

Website referencing
After inspecting a number of news articles in the dataset, we found that the majority of articles reference news sites whose objective political affiliation may be easily determined. Using that fact, we made a list of all known, extremely biased, U.S.-based news sources, and a list of objectively neutral news sources to counter the effect. By simply providing the number of extremely biased and neutral references, we improved the accuracy of our models by 2.3%. This could confirm that it is generally more common for news sources to reference other news sources with whom they share a similar political affiliation.

Sentiment analysis
Our initial assumption was that hyperpartisan articles tend to be more negative and aggressive than non-hyperpartisan articles. We used the sentiment intensity analyzer found in NLTK's sentiment library. The analyzer provides the scores for positivity, negativity, neutrality, and a compound score (i.e. aggregated score). We included these four as features in our model, which proved our theory correct and increased our accuracy results by 1.5%.

[1] Additionaly, we experimented with GloVe (Pennington et al., 2014) and fastText (Bojanowski et al., 2016), but both models were outperformed by word2vec.

Model	Accuracy
Word2vec	0.58370
with added sentiment factor	0.58589
with added quotation counter	0.59874
with added date (month)	0.61360
with added date (month and year)	0.61782
with added NER counter	0.62556

Table 1: Validation results for our first SVC model on the by-publisher dataset, with particular features added one by one.

Model	Accuracy
Word2vec	0.75663
with added sentiment factor	0.76128
with added date (month and year)	0.76285
with added quotation counter	0.76904
with added trigger word counter	0.77981

Table 2: Validation results for our first SVC model on the by-article dataset, with particular features added one by one.

Trigger words
Analyzing the articles, we noticed that the writers of extremely biased news articles were prone to using *trigger words* more often than the writers of neutral news articles. With that in mind, we assembled a list of possible trigger words (containing mostly profanities). For each article, we took the count of *trigger words* in the text, and used it as input in the feature vector.

Named entity recognition
A named entity is a real-world object, such as a person, location, organization, product, etc., which can be denoted by a proper name. We decided to count, and use as another feature, the number politics-related named entities found in the articles. We assumed that the more named entities were found, the more biased an article would be. We used the software library spaCy[2] (Honnibal and Montani, 2017) and its named entity recognition tagger to extract mentions of various named entities, such as organizations, people, locations, dates, percentages, time, and money. We used the counts as inputs in the feature vector. Although counting most of the entity types dropped the total accuracy of the model, one type in particular was beneficial. It is a type called NORP and it denotes nationalities, religious groups, and political groups. Counting only the number of these named entities as a feature increased the model's accuracy slightly, by about 1%.

[2] https://spacy.io

5 Evaluation

First, we trained our model with the much larger but subpar *by-publisher* dataset, and then fine-tuned it using the much smaller but more precise *by-article* dataset.

5.1 Larger Dataset

Labels of the *by-publisher* dataset weren't as precise as the smaller dataset, which is why the accuracy results were lower, in the $58-62\%$ range. Table 1 showcases the results. We can see that adding a sentiment factor as a feature didn't change the overall accuracy for this dataset, but we should mention that it did increase the smaller dataset's accuracy much more. Adding the number of occurrences of biased publisher names quoted in the article increased the accuracy by about 1.3%. The largest increase in accuracy came with adding the date as a feature. Adding only the month as a feature increased the accuracy by 1.5%, but adding both month and year of article publication saw an additional increase of about 0.5%. Adding a counter of named entities (in particular: nationalities, religious groups, and political groups) increased the overall accuracy by just under 1%.

5.2 Smaller Dataset

Finally, the model with all of the features explained above was trained on the *by-article* dataset. While validating our results, the dataset was divided into five parts. 4/5 were used for training, and the remaining 1/5 was used for validation. The datasets were permutated, and, in the end, we took the average of the five permutations. We trained the model using 10 different classifiers. The validation results are shown in Table 3.

6 Conclusion

We described our system for hyperpartisan detection, developed for SemEval 2019 - Task 4. The essence of our system was SVC with GridSearch built on a variety of hand-crafted features. The task itself was a complex NLP problem, as hyperpartisanism detection in text often presents a problem even for an experienced human reader. Our main mission was to extract a few quality features that would help us tackle this convoluted problem. Future work includes experiments with different

Classifier	Accuracy
Logistic Regression	0.70246
SVC	0.76590
SVC GridSearch	**0.77981**
GaussianNB	0.68344
RandomForestClassifier	0.71649
MLPClassifier	0.76117
AdaBoostClassifier	0.70866
LinearSVC	0.69619
GradientBoostingClassifier	0.72242
IsolationForest	0.35153

Table 3: Validation results for the final model on the by-article dataset. Classifier used for submission is in boldface.

Team name	Accuracy
bertha-von-suttner	0.822
vernon-fenwick	0.820
sally-smedley	0.809
tom-jumbo-grumbo	0.806
dick-preston	0.803
borat-sagdiyev	**0.791**
morbo	0.790
howard-beale	0.783
ned-leeds	0.775
clint-buchanan	0.771

Table 4: Final rankings on the main task. Our submission is in boldface.

classifiers, and possibly neural networks. Furthermore, our mission in feature extraction will continue and we hope to find more linguistic features that would help us to improve and upgrade our model.

References

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.".

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*.

Sarjoun Doumit and Ali Minai. 2011. Online news media bias analysis using an lda-nlp approach. In *International Conference on Complex Systems*.

Matthew Honnibal and Ines Montani. 2017. spacy 2: Natural language understanding with bloom embeddings, convolutional neural networks and incremental parsing. *To appear*.

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.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

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.