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

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 takes in 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 the 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 instance, 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 the 4th task of the SemEval 2019 competition. The goal of the system, as set by the task, is to, as accurately as possible, predict if a given article is hyperpartisan. While there were other criteria for evaluating the performance of the program (precision, recall, F1), we decided to optimize the program for the accuracy criterion, as the rankings were based solely on it. Accuracy in this context stands for the ratio of correctly predicted articles to the total number of articles. The finalized program reached an accuracy of 79.1%, which is solid considering the complexity of the problem and the available technology.

The program we built is based on the SVM Model publicly available in the Python's SciKit-Learn library. 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 huge, but sub-par set of automatically labelled articles. For the sake of the finding the optimal model, we also used a Recurrent Neural Network to extract some more valuable articles from the second set. By experimenting with the datasets, models and features we managed to create a program which ranked 6th in this competition.

2 Dataset description

The dataset, which was provided by 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 larger American news publishers such as: FOX News, CNN, Vox and similar. This predominance of big American news networks and the substantial difference in quality largely impacted our feature designing 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 designing were vastly influenced by the radical difference in the quality of two given datasets. As we mentioned, the larger publisher-based dataset had a large number of mislabelled articles. For example, an article about diet tips or some other non-political news were often labelled as hyperpartisan news, solely on a 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 hyperpartisan 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 have 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 half of the smaller dataset and once trained, we tested that model on the larger dataset.All correctly guessed articles were collected to a new dataset. Once we extracted and combined all quality articles, we again used SVC model. Our final accuracy on this model was also 77%.

4 Features

Our main tool used for processing news articles was Google's Word2Vec. Apart from that, we have used Stanford's GloVe and Facebook's fastText. Both fastText and GloVe performed worse than Word2vec. Furthermore, we used stemming and lemmatization tools from NLTK toolkit. Also, in our earlier stages of development, we used Chunking to get some 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 some other features we added to our model as the project development continued.

Publication date
Hyperpartisanism, or extreme bias, is a classification which is tightly related to politics. As we were dealing with heavily American news outlets, our train of thought led us to believe that news articles around certain dates could be more hyperpartisan. For example, months that are around annual American elections could be a mild indication for hyperpartisanism. Our first impulse was to only include months as a feature, as in United States some elections are every year.With some more testing, we found out that if we also included the year, the accuracy improved by over 2 percent. That improvement could mean that not only do the elections produce more hyperpartisan news, but the type of the election and maybe even the winner, could be impactful on news bias.

Website referencing
By inspecting a good number of news articles in the dataset, we found out that the majority of articles reference news sites for which we can find their objective political affiliation. Using that fact, we have made a list of all known American extremely biased news sources and a list of objectively neutral news sources to counter the effect. So just by providing the number of extremely biased and neutral references we have improved the accuracy of our models by 2.3 percent. This could come from the fact that it is generally more common for news sources to reference other news sources with whom they share a similar political affiliation.

Sentiment analysis
The idea we started with was that hyperpartisan articles tend to be more negative and aggressive than non-hyperpartisan articles tend to be. We used the sentiment intensity analyzer found in NLTK's sentiment library. The analyzer provides the scores for positivity, negativity, neutrality and the compound score (i.e. aggregated score). Those were the 4 features we added, which proved our theory correct and increased our accuracy results by 1.5%.

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
By analyzing the articles, we noticed that the writers of extremely biased news articles were prone to using some *trigger words* more often than the writers of neutral news articles. With that in mind, we assembled the list of some 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 with a proper name. One of our ideas was counting politics related named entities found in the articles. We presumed that the more named entities were found, the more biased an article was. We used the software library spaCy[1] 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, about 1%.

[1] `https://spacy.io`

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 bolded.

5 Evaluation

Firstly we trained our model with the much larger but subpar *by-publisher* dataset and only after that 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 smaller, 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 smaller dataset's accuracy much more. Counting 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 month as a feature increased the accuracy by 1.5%, but adding both month and year in which an article was published as a feature 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 a little less than 1%.

5.2 Smaller dataset

The model with all of the features mentioned above was then finally trained on the *by-article* dataset. While validating our results, the dataset was divided into 5 parts. Four fifths were used for training and the remaining fifth was used for validating. 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.

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 bolded.

6 Conclusion

We described the system for hyperpartisan detection which we developed for SemEval 2019 - Task 4. The essence of our system was SVC with GridSearch built on a variety of handcrafted 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 classifiers and possibly neural networks. Furthermore, our mission in feature extraction will continue and we hope to find some more linguistic features that would help us to improve and upgrade our model.

References

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

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.