Politically-biased news detection using Machine Learning Techniques

Sahiti Cheguru

Student, B.Tech, Dept of CSE,

Gokaraju Rangaraju Institute of Engineering and Technology, Bachupally, Hyderabad, Telanagana

Jawaharlal Nehru of Technological University Hyderabad, Telangana

sahiticheguru2000@gmail.com

Dr.Y. Vijayalata
Professor, Dept of CSE
Gokaraju Rangaraju Institute of Engineering and Technology, Bachupally, Hyderabad, Telanagana

Jawaharlal Nehru of Technological University Hyderabad, Telangana

vijaya@ieee.org

Abstract. News articles have a relatively high trustworthy rate as compared to other platforms when it comes to gathering knowledge and information. However, biased media coverage leads to spreading of wrong information that manipulates people's perceptions. This project joins the powers of neural networks and computational linguistics to detect political bias persisting in news articles. The research progresses with the idea of binary classification of articles by grouping them as biased and non-biased. Deep learning tools such as Bag-of-words, Word embeddings, and Doc2Vec along with Long Short-Term Memory (LSTM) networks are used to identify bias in a training dataset of 600,000 articles and a validation dataset of 150,000 articles.

Keywords: Natural Language Processing, Machine Learning, Media Bias

1 Introduction

Slanted news coverage, can strongly impact the public perception of the reported topics[1]. Given that news articles are the primary source of information for a broad demographic spectrum of readers, it is imperative that unbiased news is more accessible to everyone. Media bias can be in the political domain which involves specifically choosing to highlight some events, parties and leaders. Bias can be detected in these articles by observing the unclear assumptions, loaded language, or lack of proper context[2]. On a different note, evidences such as editors stating that the coverage does not reflect their own ideas but those of the people upon whom the papers depend for revenue. It suggests a connection between consumers' prior beliefs and media firms' slant [3]

This project aims to examine the political bias prevailing in news articles and identify bias indicators by dealing with a large noisy dataset. This noise stems from fact that the machine learning models used tend to learn publisher-specific traits instead of potential individual bias in the articles. The research progresses with the idea of binary classification of articles by grouping them as biased and non-biased. A word vector specific technique called "word embedding aggregation" [4] is used to extract the features. The article text is aggregated into embeddings using Word2Vec embeddings and De Boom et al.'s word embedding aggregated sentence embeddings. The goal in using Doc2Vec is to avoid a neural network learning publisher-specific lexical features, and to instead be able to generalize with the aggregated document vectors. Deep learning tools along with Long Short-Term Memory (LSTM) networks are used to identify bias in a training dataset of 600,000 articles and a validation dataset of 150,000 articles.

2 Corpus

The dataset is drawn from the large corpora published by FactCheck.com and BuzzFeed. The training dataset consists of 600,000 articles a validation dataset of 150,000 articles with half being politically biased.

Test Dataset

These test datasets were obtained via web scraping from news providers focusing on entertainment. The test dataset, here referred to as byarticle dataset includes 638 articles with no publisher overlap with any of the given corpora. The other, like the training and validation sets, was labeled overall by publisher, here referred to as bypublisher-test dataset, with a total of 4000 articles, also including publisher overlap with other datasets.

Preprocessing the data

The noisy dataset was cleaned through tokenization of texts using the Natural Language Toolkit (NLTK). Special characters, double spaces, more than three dots in a row, and any failures in character translation were replaced or removed. Regular expressions were used for some of these tasks, as well as for removing img or html tags and URLs. Another list of phrases was summarily removed from each text: those which were likely byproducts of the articles' retrieval from their websites. These included
"Continue Reading Below...", "Image Source:", "Opens a New Window", and so on.

3 Methodology

After collecting and pre-processing the data, the following NLP tools are used to extract the required insights from it.

Bag-of-Words Unigram Model

The bag-of-words model is used to determine the term frequency of the words of the research interest that indicate bias. The bag-of-words model is commonly used in methods of document classification where the (frequency of) occurrence of each word is used as a feature or training a classifier[5].

First, the plain text is converted to a sequence words using a tokenizer and further reduced into their lemmas. To feed this data to the logistic regression model, each term was transformed into a vector that can be placed in the input layer of a neural network. Scikit-learn is used to create a vocabulary of the most common 4000 words to achieve this in the overall corpus (all datasets), excluding stopwords. Vocabulary words from an exception list was also excluded. This exceptions list was formed by counting all words in the training, validation datasets, and gathering those words which appeared five times more often (relative to the size of the corpus) in one set than in the other.

Feature Assignment

Each term is then characterized with a set of features. In order to capture syntactic relations, dependency and part of speech tags were used in this approach; to keep lexical information involved without assigning inflated importance to publisher-specific words, existing lexicons developed by researchers in the field are used.

Feature	Weight
Adjectives	2.04
Adverbs	4.76
Pronouns	−2.87
Nouns	−0.32
Proper Nouns	−1.04
Strong Subjectivity	3.28
Clues Weak Subjectivity	2.85
Clues Auxiliaries	−2.10
Passives	−2.83
Average Sentence	−0.01
Length Average Word	−0.01
Length Exclamation	0.26
Marks Question Marks	0.40
Multiple Punctuation	0.05
Anger Words	0.06
Fear Words	−2.09
Sadness Words	−0.94
Joy Words	−4.39

Table 1. Features chosen for the second linguistic approach, and their weights according to the trained model.

Part of speech tagging and dependency parsing were both done using spaCy, simultaneously with the tokenization and lemmatization mentioned in the description of the last approach. A Ridge Classifier model from scikit-learn was fit to the vectorized training dataset.

LSTM

Recently, owing to the breakthrough in the field of computational science, deep learning or deep neural network (DNN) methods based on artificial neural networks have received a growing interest both academically and practicality from scientists[6]. Long Short Term Memory networks (or LSTMs) are one of the state-of-the-art DNNs which are capable of learning long-term dependencies.

The text is vectorized in order to be used in the machine learning model. To do so, skip-gram word embeddings on the entire dataset are trained. Embeddings of 50 dimensions were trained using the Python package gensim, for 10 epochs, including words in the vocabulary which appeared in the corpus over five times.

Texts were first transformed into arrays of the shape (100, 50), wherein 100 was the cutoff or maximum text length and 50 was the dimensionality of the word embeddings. Texts shorter than 100 words were padded with zero-vectors to keep the shape consistent to feed into the network. The model consists of a single LSTM layer with 50 units, followed by a dropout wrapper with a keep probability of 0.75 to help prevent overfitting. Next is a standard feedforward neural network output layer. AdamOptimizer was used with a 0.001 learning rate as well as softmax cross-entropy loss for optimization. All LSTM models were trained using Tensorflow for approximately 2 epochs.

Voting System

Three LSTMs as described in the previous section were trained: one each on the training, validation and test datasets. Predictions from each LSTM on each article in each dataset were then collected. The articles which all three LSTM models correctly labeled were pulled into a new dataset labeled agree. This dataset, in total size 162,046 articles with 37% biased and 63% unbiased labels, was what the final model was trained on. Once the new datasets were compiled from the voting system based on the originals, a new LSTM with the same architecture was trained on the combined data.

Doc2Vec

In order to obtain neural network-based document embeddings, the doc2vec algorithm is used that learns features from the corpus and provides a fixed-length feature vector as output. Then, the output is fed into a machine-learning classifier[7]. A doc2vec model is developed using genism to minimize the model's problems in learning publisher's tendencies and not generalizing bias. Doc2vec creates word embeddings as word2vec does, but then can generate a single vector of a certain number of dimensions for any document, by aggregating the word vectors and therefore representing the concept of the whole text. The goal in utilizing this process was that the data fed to the model would carry less information about the specific and potentially publisher-specific syntactic constructions and vocabulary, and more information about the idea of the document as a whole, and biased (or not) relation to its topic.

The doc2vec model is trained over all given datasets, only counting words that appeared in the corpus 5 times or more, 10 times (that is, for 10 epochs). The resulting vector size for the document was set to 50. Total training time took approximately 2 hours and 50 minutes.

The feed forward neural network is built using Keras. It included a single fully-connected layer of 64 nodes followed by sigmoid activation. Cross entropy loss and Adam Optimizer were used for optimization. Training was very fast, at about 76 seconds per epoch, and the model was trained for 5 epochs.

4 Results

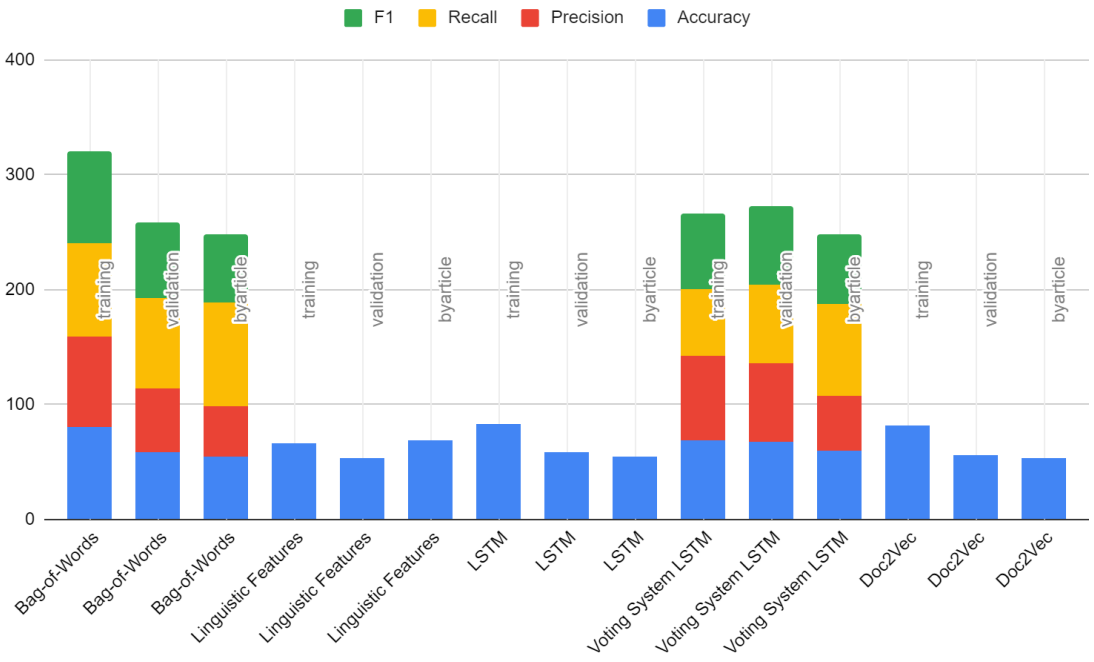

Figure 1. Results on the training datasets.

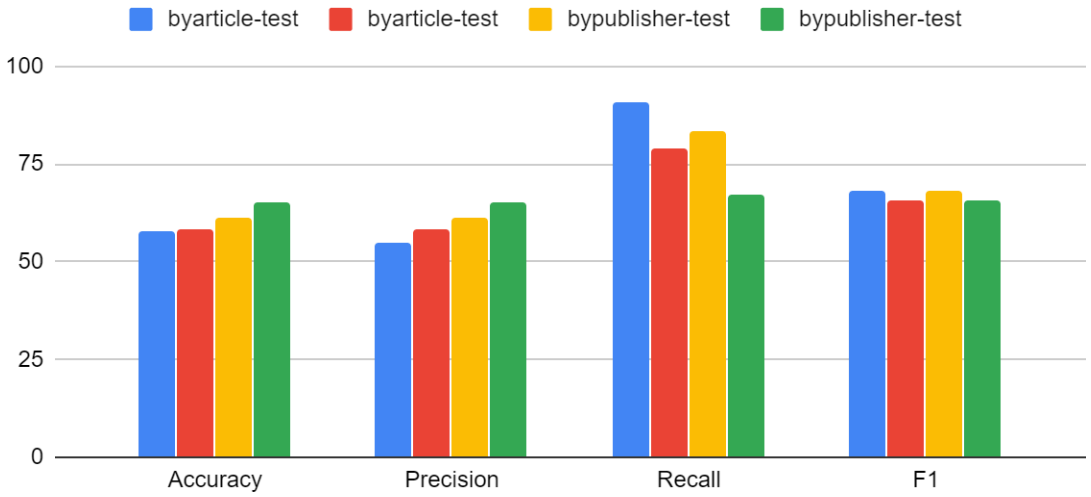

Figure 2. Results on test dataset

Results on available datasets are shown in Figure 1. Generally, the neural network approaches performed better on their training datasets and worse on the validation datasets than the non-deep learning approaches, which seemed to generalize better to other publishers despite higher training error. Interestingly, the results indicate that Linguistic Features approach performed significantly better on the byarticle dataset than the others did.

All results showed higher recall than precision — and except for the case of the Voting System LSTM with the bypublisher-test set, markedly higher. By accuracy, the best result was the Voting System LSTM on the bypublisher-test set.

Overall, the results show a pattern, as briefly mentioned: neural networks seemed to do better on the training dataset and worse on the data belonging to publishers they were not trained on than the bag-of-words and linguistic feature models. This is logical, if the neural networks were able to better pick up on the features that separate publishers from one another, and in doing so generalized worse to other publishers. Essentially: they overfit more strongly to the publishers they were trained on.

Theoretically these articles would all have characteristics common to biased articles of all publishers. When put together, then, the hope was that a model trained on this subset of the dataset would learn those common characteristics and not just the publisher-specific ones. Training the new Voting System LSTM on the pared-down dataset, as expected, reduced training accuracy. Naturally, this could have meant either better generalization to the problem of bias detection or an overfitting to the new dataset instead of just the publishers.

5 Conclusion

This research examines the biased nature of news articles that is commensurate with the bag-of-words model, linguistic features, and also models developed using deep learning. A wide range of approaches were considered to tackle the problem associated with grainy and unstructured datasets. The most onerous task was to train the models characterizing the dataset basing upon the publisher instead of the individual potential bias of the article. Both neural networks with word vectors were used including other machine learning approaches featuring linguistic features, as well as one filtering method for the data using a voting system. Future work involves examining a more rigorous approach to filtering the dataset. This was begun with the Voting System, but could be strengthened. For a better analysis of current results, it would be good to rerun all models for precision, recall and F1 scores, once the computational resources are again available. There are many things that could have gone wrong with this system. Since the three models used for voting did not have very high accuracy on each other's datasets in the first place, the level of noise may not have been reduced at all. Furthermore, the original training dataset was four times as large as the validation dataset. Their subsets after the voting system was applied were equally unbalanced. When combined and used for training the final LSTM, it could have been unbalanced enough that the model learned mostly from the data from the original dataset, and the features from those publishers.

References

1. Hamborg, F., Donnay, K. & Gipp, B. Automated identification of media bias in news articles: an interdisciplinary literature review. *Int J Digit Libr* **20,** 391–415 (2019). https://doi.org/10.1007/s00799-018-0261-y
2. Gangula, R. R. R., Duggenpudi, S. R., & Mamidi, R. (2019, August). Detecting Political Bias in News Articles Using Headline Attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP* (pp. 77-84).
3. Gentzkow, M., & Shapiro, J. M. (2006). Media bias and reputation. *Journal of political Economy*, *114*(2), 280-316.
4. De Boom, C., Van Canneyt, S., Demeester, T., & Dhoedt, B. (2016). Representation learning for very short texts using weighted word embedding aggregation. *Pattern Recognition Letters*, *80*, 150-156.
5. McTear, M. F., Callejas, Z., & Griol, D. (2016). *The conversational interface* (Vol. 6, No. 94, p. 102). Cham: Springer.
6. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. Nature 2015, 521, 436.

7. Markov, I., Gómez-Adorno, H., Posadas-Durán, J. P., Sidorov, G., & Gelbukh, A. (2016, October). Author profiling with doc2vec neural network-based document embeddings. In *Mexican International Conference on Artificial Intelligence* (pp. 117-131). Springer, Cham.