A NLP Approach to Understanding Variation in Political News

Stephen M. Lee

August 28, 2019

Contents

"If you can't measure it, you can't improve it."

— Peter Drucker

1 Introduction

The 2016 United States Presidential Election, to many, raised questions as to the reliability of their news sources. For example, Allcott and Gentzkow [2017] estimate that the average US adult read and remembered about one, and possibly up to several, fake news articles during the election period. While they make no statements about the effects of this exposure, the implications are certainly thought provoking.

Since then, many social media companies and other institutions have begun public campaigns to combat the perceived threat from fake news, including explicit efforts from Google and Facebook to remove "fake news sites", as documented in Allcott and Gentzkow [2017].

With a goal of classifying articles by their news source, I scraped several thousand news articles from Fox, Vox, and PBS News. By some estimations,[1] these three news sites represent distinct categories of news: Fox as a conservative right opinion; Vox as a liberal left opinion; and PBS as the center primary source news position. Figure 1 shows a graphic representation of this news bias.

I first calculate the top words, 2-gram, and 3-gram frequencies to better understand the dataset, and then train a bidirectional, long-term short-term memory (LSTM) recurrent neural network using the GloVe pretrained word embeddings to predict the source of news.

This work fits into existing literature by bridging gaps between the cutting edge of computer science and economics. On the computer science front, bidirectional LSTM recurrent neural networks provide some of the most accurate results for predicting sequential data, whereas the economic work on media bias focuses more on inference based analyses. In particular, Gentzkow and Shapiro [2010] find that readers prefer to consume "like-minded" news—meaning they want to have their prior beliefs validated—and that the profit maximizing response from news companies can account for around 20% of the variation in political slant or bias. In this same direction, Gentzkow and Shapiro [2006] also find that a theoretical Bayesian consumer will effectively reinforce their beliefs of a given news source quality when they read something that confirms their priors. Together, these works suggest that political news bias may be a tactical response of competing news firms to segment the consumer market according to their heterogeneous beliefs, and, further, that this polarization may be self-reinforcing. In fact, Gentzkow and Shapiro [2008] go as far as to suggest that competition in information markets may actually be counterproductive in achieving balanced and unbiased news. These effects could explain the relative

[1] The website https://www.adfontesmedia.com/ provides a visual representation of media bias. Their methodology is well documented, and the outcome is largely consistent with other sources and anecdotal descriptions of bias.

neutrality of news sources like the BBC that are somewhat insulated from the competitive pressures faced by Fox and Vox news companies.

If these news companies are in fact biased by construction, can we train a computer to detect and classify the differences based on the language alone? Regardless of the answer, there are substantial implications. If yes, then we can create computer software to help classify news based on language content. This would enable news consumers to effectively "locate" their news consumption in the same way that a GPS helps to locate yourself on a map, or a calorie tracking app helps to locate your dietary health. However, if the answer is no, then it provides a clear direction for further study in the field of natural language processing and machine learning insofar as additional context is needed to parse the perceived differences.

In this case, I find that a relatively simple, bidirectional, LSTM recurrent neural network can in fact correctly predict the source of an article with very high accuracy. I then use the result of this trained network to build a web app that can allow for a copy-and-paste interface to this classification model. Depending on the extent to which the underlying model enables transfer learning, this web app could ideally serve as a first check on the inherent bias in political news before you read.

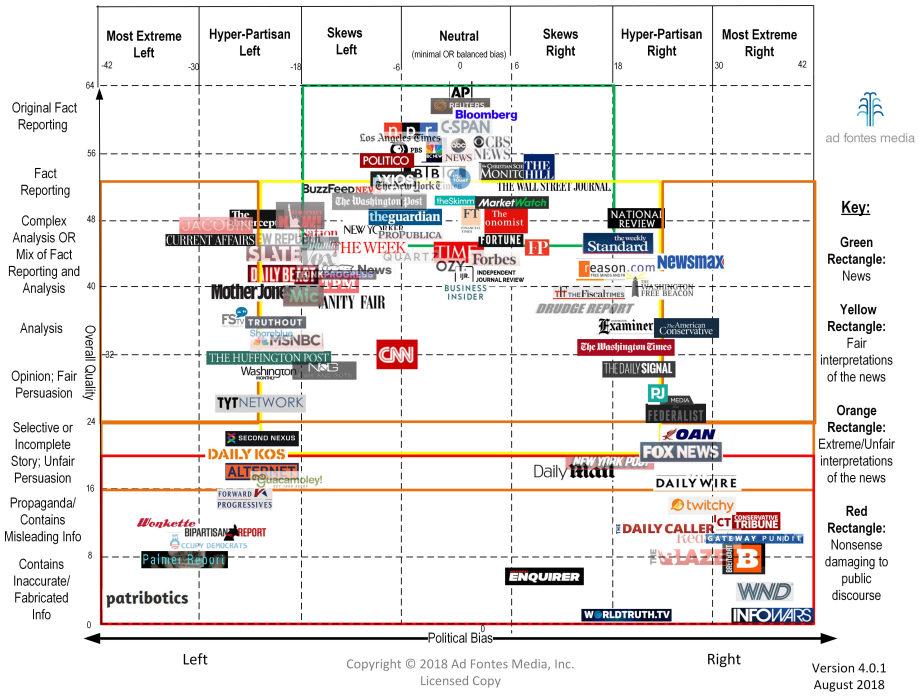

Figure 1: A graphical depiction of the bias in various political news sources.

2 Data

I mined political news articles from the websites of Fox News, Vox News, and PBS News. Importantly, I restricted focus to only URLs that contained an explicit reference to politics i.e. articles from their respective political sections. This allowed me to collect articles that were as similar as possible to each other in order to try and limit the chances of spurious predictive results. Intuitively, the motivation behind this is to facilitate classification based only on sentiment or semantics, rather than subject matter differences. To better understand how similar the content of these articles are, I construct n-gram tokens and count the frequency of their occurrences. In Table 1, we see the most frequent one, two, and three-gram phrases from the collected corpus.[2]

Looking carefully at the most common words and phrases we see substantial similarity in terms of topic. Regardless of the source, "Trump" is the most used word. Perhaps surprisingly, the top four most used words for PBS and Fox news are exactly the same, and in the same order. Beyond that, we see phrases "white house", "president Donald Trump", and "senate majority leader" appear in the top ten most frequent phrases for each news source. Together, this suggests that, topic wise, the corpus for each source are comparable.

In addition to subject, wording, and phrasing, I also check for grammatical and structural differences with a simple lexical analysis. Using the University of Pennsylvania tagset, I count the percent of adjectives and adverbs in each article and calculate the average for each news source. I find very similar use of adjectives and adverbs for Fox and PBS, and a slightly more frequent use with Vox. Intuitively, the goal is to understand, on average, how descriptive the language is for each source. Toward this end, I included comparatives (e.g. better, worse, greater) and superlatives (e.g. best, worst, greatest) in the count.

Finally, I calculate the average number of words per article by source, and the average number of words per sentence, again grouped by news source. Here I find that PBS writes the shortest sentences, while Vox writes the longest. This measure is relevant when considering the average number of words per sentence as a proxy for complexity, as introduced in Flesch [1948]. Table 2 summarizes these descriptive statistics.

Table 2: Summary statistics by news source.

Source	*Documents*	*Average word count*	*Percent adjectives*	*Percent adverbs*	*Average words / sentence*
Fox	661	686.2	6.6 %	3.4 %	20.1
PBS	1739	654.3	6.6 %	3.2 %	18.0
Vox	1027	1332.8	7.3 %	4.6 %	21.3

[2]The associated counts of each n-gram phrase are shown in Table [NEED REF] in the Appendix.

Table 1: Most frequent words and phrases, by news source.

Most Common Words

	VOX	*PBS*	*FOX*
1	trump	trump	trump
2	tax	said	said
3	will	president	president
4	people	house	house
5	health	will	new
6	bill	new	will
7	republicans	white	democratic
8	one	senate	democrats
9	new	democrats	told
10	care	campaign	border

Most Common 2-gram Phrases

	VOX	*PBS*	*FOX*
1	health care	white house	white house
2	white house	president donald	new york
3	trump administration	donald trump	president trump
4	donald trump	special counsel	green new
5	tax cuts	supreme court	health care
6	health insurance	attorney general	new deal
7	new york	new york	united states
8	affordable care	justice department	border security
9	tax bill	counsel robert	donald trump
10	federal government	trump said	state union

Most Common 3-gram Phrases

	VOX	*PBS*	*FOX*
1	affordable care act	president donald trump	green new deal
2	president donald trump	special counsel robert	house speaker nancy
3	congressional budget office	majority leader mitch	special counsel robert
4	health care bill	attorney general jeff	partial government shutdown
5	new york times	senate judiciary committee	speaker nancy pelosi
6	majority leader mitch	sarah huckabee sanders	state union address
7	american health care	senate majority leader	new york times
8	leader mitch mcconnell	counsel robert mueller	majority leader mitch
9	corporate tax rate	leader mitch mcconnell	president donald trump
10	senate majority leader	secretary sarah huckabee	senate majority leader

2.1 Challenges

There are several limitations to discuss. The most obvious is the difference in corpus size from each source. In particular, Fox News has fewer documents than either PBS or Vox by quite a large number. Fortunately, there are many well established methods for dealing with imbalanced data, like bootstraping, as in Dupret and Koda [2001].

Second, due to the variability of online formatting, it's worth noting the possibility that, even after cleaning, each source exhibits some subtle idiosyncratic characteristics that could allow a neural network to detect those instead of pure sentiment and semantic differences. To mitigate this, I removed any mention of their own organization, any other common and unique affiliations, and other identifying characteristics.[3]

Finally, each news source shows a difference in the average article length. To overcome this, I limited the article length to a maximum of the first 500 words to ensure that no single source was consistently shorter when fed into the neural network.

3 Models

As a baseline, I use long-term short-term memory (LSTM) neural network. The LSTM architecture was introduced in Hochreiter and Schmidhuber [1997], and extended the Recurrant Neural Network (RNN) to improve lagged information storage for the purpose of predicting sequential data. The key advantage of the recurrent LSTM architecture is the ability for the cell to "remember" relevant lagged values, while "forgetting" less useful ones. Visually, we can see a representation of a single LSTM unit in Figure 2.[4]

[3]For example, I removed any mention of 'Fox' from every Fox News article. Similarly, Fox News cited the "Associated Press" disproportionately often, so I also removed that string. Additionally, PBS News begins each article with location information in the following format: "LOCATION — Start of article...". In this case, I removed the names of the most frequently referenced cities and the following "—" character.

[4]Source: https://colah.github.io/posts/2015-08-Understanding-LSTMs

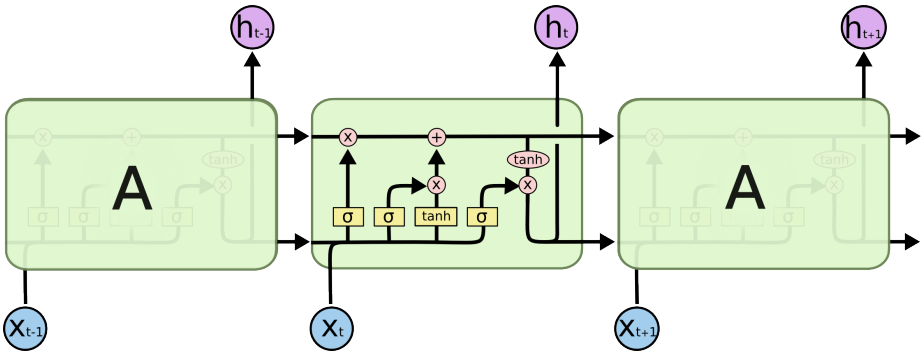

Figure 2: A graphical depiction of a single LSTM cell.

More recently, Gers and Schmidhuber [2000], Chung et al. [2014], and Yao et al. [2015] introduce variations on the baseline Hochreiter and Schmidhuber [1997] LSTM model, although Greff et al. [2016] show them to be roughly equivalent.

Much success in natural language processing (and other sequential tasks) has been attributed to so-called bidirectional LSTM networks. As in Wang et al. [2015], I train an LSTM architecture bidirectionally (i.e. forwards and backwards), and compare it to a unidirectional (i.e. forward-only) baseline. Intuitively, we can think of bidirectional feeds as providing additional context for a given word to "know" about what came before it *and* what comes after it. Visually, Figure 3 depicts a bidirectional network during training.

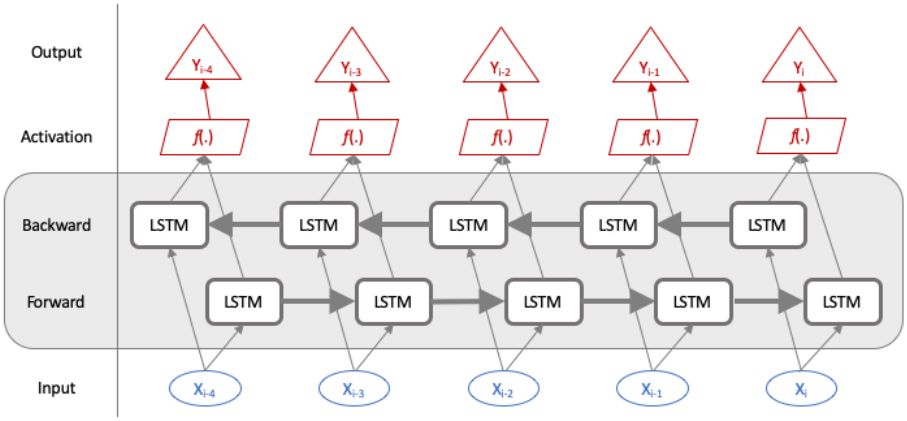

Figure 3: A visual representation of a bidirectional LSTM training.

In order to help understand the advantages of training bidirectionally, consider the following sentence. "The man sat to eat an orange, which, strangely, matched the color of his beard with tremendous accuracy." When we as humans

read that sentence, we can retroactively modify our understanding. In other words, we can update our image of the man even after he was first mentioned. In this example, when reading the word man, it's possible to first imagine a cleanly shaved man with short dark hair based on the little previous context. However, only *after* finishing the sentence, we can update our mental image to a man with long orange hair and a trimmed beard. Similarly, training the neural network both forward and backward can allow for additional context.

3.1 Word Embedding

Using the common crawl 840B Global Word Vector (i.e. GloVe), I mapped each word into its corresponding 300×1 dimensional vector.[5] Since this set of embeddings are case sensitive and unstemmed, I do minimal preprocessing to the text besides the basic cleaning mentioned in Section 2.1.

4 Results

Using bootstrapped data and the setup above, I train and test models with various parameterizations. Notably, for an RNN, we can specify the batch size, dropout rate, recurrent dropout rate, and the number of steps per epoch. Broadly speaking, batch size describes the number of words included in each training group, the dropout rate specifies the probability of ignoring any given entry in the matrix of weights—this helps to prevent the model from overfitting on any specific word when making predictions. The recurrent dropout, similarly, specifies the dropout that occurs between recurrent cells. Finally, using only one epoch, the steps per epoch represents the number of iterations used in training.

Results are summarized in Table 3. We can see that the results favor a larger batch size, and a larger number of iterations. Perhaps surprisingly, we don't see too much gain from increasing the maximum article length from 250 words to 500 words. This suggests that any linguistic or semantic differences are, in general, noticeable from the start.

[5] This embedding is introduced in Pennington et al. [2014] and uses 840 billion tokens and a case-sensitive vocabulary of 2.2 million words to map words into a corresponding 300×1 dimensional vector.

Table 3: Training results from the bidirectional LSTM, sorted by F1 score.

Article Length	**Batch Size**	**Dropout**	**Recurrent Dropout**	**Steps Per Epoch**	**F1**
250	64	0.1	0.2	1000	0.946
500	64	0.2	0.2	1000	0.944
500	64	0.2	0.1	1000	0.939
250	64	0.1	0.1	1000	0.937
500	64	0.1	0.1	1000	0.937
500	64	0.1	0.2	1000	0.933
250	64	0.2	0.1	1000	0.921
250	32	0.2	0.1	1000	0.910
250	32	0.1	0.1	1000	0.906
250	64	0.2	0.2	500	0.906
250	64	0.2	0.2	1000	0.904
500	32	0.1	0.2	1000	0.904
500	32	0.2	0.1	1000	0.904
500	64	0.1	0.1	500	0.902
500	32	0.1	0.1	1000	0.900
500	64	0.2	0.2	500	0.900
250	32	0.1	0.2	1000	0.897
250	64	0.1	0.1	500	0.897
250	64	0.1	0.2	500	0.897
500	32	0.2	0.2	1000	0.897
250	32	0.2	0.2	1000	0.895
500	64	0.2	0.1	500	0.895
500	64	0.1	0.2	500	0.881
250	64	0.2	0.1	500	0.877
500	32	0.1	0.2	500	0.874
250	32	0.1	0.1	500	0.870
250	32	0.2	0.2	500	0.860
250	32	0.1	0.2	500	0.858
250	32	0.2	0.1	500	0.851
500	32	0.2	0.2	500	0.845
500	32	0.2	0.1	500	0.835
500	32	0.1	0.1	500	0.828

The reported F1 scores are measured across the entire sample by counting the total number of accurate predictions, false positives, and false negatives. Mathematically, the F1 score is the harmonic mean of precision and recall, defined as follows:[6]

$$F_1 = 2 \, \frac{Precision \cdot Recall}{Precision + Recall}$$

[6]For source, see https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html.

Where,

$$Precision = \frac{TruePositives}{TruePositives + FalsePositives}$$

$$Recall = \frac{TruePositives}{TruePositives + FalseNegatives}$$

As an additional exercise for robustness and comparison, I used similar parameters to train an LSTM recurrent neural network unidirectionally, rather than bidirectionally. Interestingly, we see that the best performing unidirectional model still under performed the worst bidirectional model, although on the whole, the unidirectional model dramatically under-performed relative to the bidirectional baseline. Results are shown in the Appendix in Figure 7.

5 Implementation

To make use of this trained model in application, I wrote a simple web interface that enables passing text from an arbitrary news article through a form, and returns values that can be interpreted as predictions of the likelihood that a given article is from Fox, Vox, or PBS News. The commercial use case for this type of app is to provide context as to the type of writing, either before or after the consumer has read the article. Ideally, each user could login and see a history of the news they consume—this could help interested readers identify if and when they are in a political bubble or echo-chamber. Idealistically, it could help balance information consumption.

Extensions of this work would include more volume and more recent news articles as to stay current with changing topics and trends. Additional news sources would also be desirable for readers to chose their set of comparisons such that they are comfortable in assessing the relative balance and neutrality of the set as a whole.

A sample of the interface is shown in Figures 4 and 5 below.

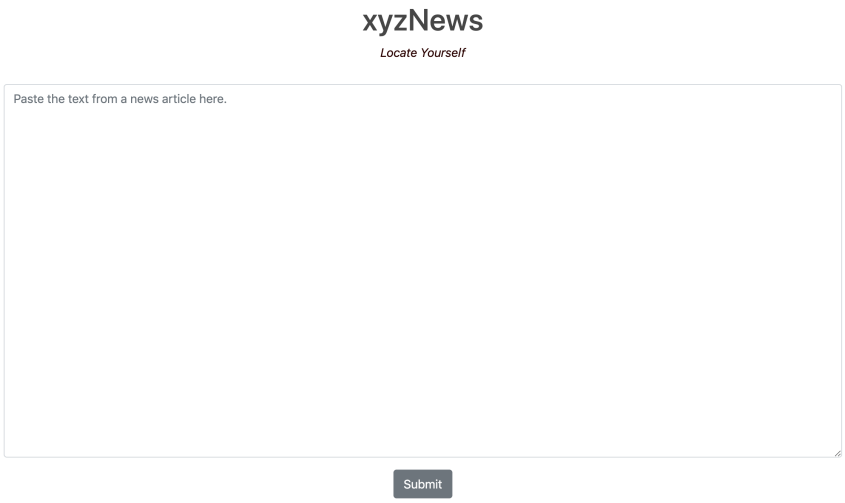

Figure 4: The home landing page for the web app.

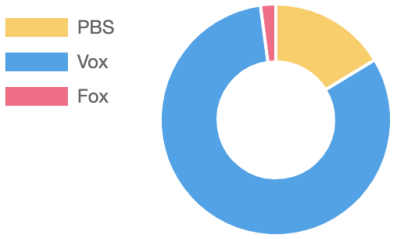

Figure 5: The web display of the prediction results.

6 Conclusion and Discussion

I've shown that a neural network can accurately classify news articles based on common language differences in the underlying news sources. These results have implications for our politically polarized world, where even scientific "facts" are often disputed. While there are plenty of factors that could motivate news companies to intentionally bias their news, Gentzkow and Shapiro [2008] and Gentzkow and Shapiro [2006] suggest that the profit maximizing response of a company is to produce a news product that confirms consumers prior beliefs. Thus, given a society of individuals with heterogeneous preferences and beliefs,

and a competitive news information market, it seems unlikely that biased news (or even fake news) will disappear anytime soon.

A software application based on the ideas presented above could serve as a starting point to measure news consumption—much in the same way that calendars can help to measure our time use, nutrition apps measure our consumption of macronutrients, and GPS measures our geographical position.

7 Appendix

Table 4: Word Frequencies

Num	Vox		PBS		Fox	
1	trump	5446	trump	7811	trump	2510
2	tax	5206	said	7383	said	2009
3	will	4098	president	3997	president	1730
4	people	4013	house	3495	house	1656
5	health	4003	will	3079	new	1569
6	bill	3138	new	2277	will	1414
7	republicans	2655	white	2184	democratic	1137
8	one	2573	senate	2002	democrats	954
9	new	2566	democrats	1963	told	862
10	care	2565	campaign	1933	border	790

Table 5: Top frequencies of two word phrases.

Num	Vox		PBS		Fox	
1	health care	1654	white house	1683	white house	556
2	white house	743	president donald	1297	new york	359
3	trump administration	672	donald trump	1035	president trump	318
4	donald trump	598	special counsel	613	green new	256
5	tax cuts	479	supreme court	584	health care	160
6	health insurance	479	attorney general	499	new deal	151
7	new york	470	new york	491	united states	134
8	affordable care	437	justice department	485	border security	132
9	tax bill	376	counsel robert	405	donald trump	131
10	federal government	365	trump said	369	state union	126

Table 6: Top frequencies of three word phrases.

Num	**Vox**		**PBS**		Fox	
1	affordable care act	222	president donald trump	785	green new deal	143
2	president donald trump	157	special counsel robert	396	house speaker nancy	81
3	congressional budget office	127	majority leader mitch	179	special counsel robert	72
4	health care bill	121	attorney general jeff	139	partial government shutdown	63
5	new york times	115	senate judiciary committee	137	speaker nancy pelosi	58
6	majority leader mitch	114	sarah huckabee sanders	137	state union address	53
7	american health care	97	senate majority leader	131	new york times	51
8	leader mitch mcconnell	96	counsel robert mueller	123	majority leader mitch	47
9	corporate tax rate	95	leader mitch mcconnell	113	president donald trump	43
10	senate majority leader	95	secretary sarah huckabee	108	senate majority leader	41

Table 7: Training results from the unidirectional LSTM, sorted by F1 score.

Article Length	Batch Size	Dropout	Recurrent Dropout	Steps Per Epoch	F1
250	64	0.2	0.1	1000	0.824
250	64	0.1	0.2	1000	0.797
250	32	0.1	0.2	1000	0.766
500	64	0.1	0.1	1000	0.724
250	64	0.2	0.2	1000	0.716
250	32	0.2	0.1	1000	0.703
500	64	0.2	0.2	1000	0.703
250	32	0.2	0.2	1000	0.695
250	64	0.1	0.2	500	0.686
250	64	0.2	0.2	500	0.686
500	64	0.2	0.1	1000	0.678
250	64	0.2	0.1	500	0.670
250	32	0.1	0.1	1000	0.653
250	64	0.1	0.1	500	0.653
250	64	0.1	0.1	1000	0.653
500	32	0.1	0.2	1000	0.644
250	32	0.2	0.1	500	0.640
250	32	0.2	0.2	500	0.628
250	32	0.1	0.2	500	0.619
250	32	0.1	0.1	500	0.607
500	64	0.1	0.2	500	0.607
500	64	0.1	0.2	1000	0.602
500	32	0.2	0.1	1000	0.600
500	64	0.2	0.2	500	0.598
500	32	0.2	0.2	500	0.594
500	32	0.1	0.2	500	0.592
500	32	0.1	0.1	1000	0.584
500	32	0.1	0.1	500	0.579
500	32	0.2	0.2	1000	0.571
500	64	0.1	0.1	500	0.565
500	32	0.2	0.1	500	0.559
500	64	0.2	0.1	500	0.556

References

Hunt Allcott and Matthew Gentzkow. Social media and fake news in the 2016 election. *Journal of economic perspectives*, 31(2):211–36, 2017.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

Georges Dupret and Masato Koda. Bootstrap re-sampling for unbalanced data in supervised learning. *European Journal of Operational Research*, 134(1): 141–156, 2001.

Rudolph Flesch. A new readability yardstick. *Journal of applied psychology*, 32 (3):221, 1948.

Matthew Gentzkow and Jesse M Shapiro. Media bias and reputation. *Journal of political Economy*, 114(2):280–316, 2006.

Matthew Gentzkow and Jesse M Shapiro. Competition and truth in the market for news. *Journal of Economic perspectives*, 22(2):133–154, 2008.

Matthew Gentzkow and Jesse M Shapiro. What drives media slant? evidence from us daily newspapers. *Econometrica*, 78(1):35–71, 2010.

Felix A Gers and Jürgen Schmidhuber. Recurrent nets that time and count. In *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks. IJCNN 2000. Neural Computing: New Challenges and Perspectives for the New Millennium*, volume 3, pages 189–194. IEEE, 2000.

Klaus Greff, Rupesh K Srivastava, Jan Koutník, Bas R Steunebrink, and Jürgen Schmidhuber. Lstm: A search space odyssey. *IEEE transactions on neural networks and learning systems*, 28(10):2222–2232, 2016.

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

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

Peilu Wang, Yao Qian, Frank K Soong, Lei He, and Hai Zhao. A unified tagging solution: Bidirectional lstm recurrent neural network with word embedding. *arXiv preprint arXiv:1511.00215*, 2015.

Kaisheng Yao, Trevor Cohn, Katerina Vylomova, Kevin Duh, and Chris Dyer. Depth-gated lstm. *arXiv preprint arXiv:1508.03790*, 2015.