Detecting and Identifying Bias-Heavy Sentences in News Articles

Nicholas P. Hirning
Stanford University
Stanford, CA 94305
nhirning@stanford.edu

Andy Chen
Stanford University
Stanford, CA 94305
asjchen@stanford.edu

Shreya Shankar
Stanford University
Stanford, CA 94305
shreya1@stanford.edu

Abstract

This project investigates the advantages of using neural networks to approach the task of identifying biased sentences in news articles. In particular, convolutional neural nets (CNN) were used to classify articles by the news provider that published them and a bidirectional RNN was used to extract the sentences that the classifier found important for this prediction. Overall the system showed promise, with the CNN classifier outperforming a baseline bag-of-vectors model to achieve 84% accuracy and the bidirectional RNN achieving 99.6% consistency with the classifier prediction.

1 Introduction

After the 2016 U.S. Presidential Election, the public has taken special interest in media and news provider bias. For instance, the recently coined term "fake news" refers to highly biased or blatantly false published information. [1] With current events, it is useful and informative to design a system that can pick out which sentences and words in a news article are particularly "bias-heavy." This can inspire more objective news coverage on the side of news providers, as well as helping consumers identify and avoid bias. This project studies the application of neural networks for the construction of such a system.

2 Task definition

The task breaks down into two categories:

(1) Defining a metric for identifying bias in a news article. This can be accomplished by trying to classify articles in some way that is not directly related to the news covered by that particular article, since this reveals an unintentional bias in the information. We will attempt to classify each article by its news provider.

(2) Identifying what words, phrases, or sentences within that article induce that bias. This can be accomplished by testing what subset of the contents of the article can be removed without altering the classification in (1).

3 Background and related work

Previous research in the Stanford NLP group has explored topic classification of news articles, primarily with LDA, multinomial Naive Bayes, clustering algorithms, and support vector machines [6]. There has also been research conducted on detecting bias-inducing words in news articles [8].

In looking for ways to implement the neural network classifier, we came across the idea of a convolutional neural network with max-pooling. Convolutional neural networks have been explored in sentiment classification tasks [4]. Research has also been done in character-level convolutional neural networks [2]. This research has been extended to sentence classification [3]. We adopted these classification methods along with a max-pooling scheme for our own convolutional neural network. Our input word vectors were pre-trained using the GloVe model [7].

In the past, attention-based models have been used to explain neural predictions. Recently, a paper on analyzing rationale of neural network predictions was published [5], which inspired us to determine the rationale or indicator phrases for our news provider predictions. The model highlights key words or phrases in the input text that are supposed to justify the classifier neural network's predictions.

3.1 Classifying articles

In order to identify bias, we attempted to identify the news provider that published an article using just the body of the article as input. For this purpose, we collected roughly $10,000$ total news articles from five different news providers (approximately $2,000$ per provider) over the past six months:

- The New York Times
- CNN
- Reuters
- Fox News
- Breitbart

We shuffled and segregated this data to create test and dev data sets that each contained $2,000$ articles and a train set with $6,000$ articles. The task is then to develop a model which, given the body of the news article, returns a prediction about which of the five providers published the article.

In order to test the difficulty of this task and provide initial error bounds for later comparison, we implemented a simple baseline bag-of-vectors model. For this baseline, each article is condensed to a single vector by using GloVe vectors (dimension 300) for each word, simply averaging every vector in the article (after removing any words not represented in the GloVe). These article vectors are the inputs to a two-layer feed-forward neural network of the following form:

$$z_1 = xW + b_1 \tag{1}$$
$$z_2 = f(z_1) \tag{2}$$
$$z_3 = z_2U + b_2 \tag{3}$$

where if d is the dimension of the GloVe vectors, h is the dimension of the hidden layer, and n is the number of news providers used (five for this project), then $W \in \mathbb{R}^{d\times h}, b_1 \in \mathbb{R}^h, U \in \mathbb{R}^{h\times n}, b_1 \in \mathbb{R}^n$. The activation function f was the softplus function, which is a smooth approximation to a linear rectifier and is defined as follows: $\text{softplus}(x) = \ln(1 + e^x)$. Our prediction is then given by $\text{argmax}_i z_{3,i}$.

The goal is to maximize the accuracy of our classifier (defined as the number of articles correctly classified divided by the total number classified). However, it is difficult to train a loss related to accuracy (since it is either one or zero for a single training example). We computed the loss by

applying softmax to the output vector z_3 and computing cross-entropy loss. If the one-hot vector representing the actual news provider class is $\mathbf{y}$, then the loss can be written as

$$L = -\sum_i y_i \cdot \log \frac{\exp(z_{3,i})}{\sum_j \exp(z_{3,j})} \tag{4}$$

This model has four hyperparameters: hidden layer dimension h, learning rate l, and training epochs e. For the following configurations of learning rate and hidden size, we ran 100 epochs with a batch size of 50 on the train data set and evaluated on the development data set:

h	$l = 0.01$	$l = 0.1$	$l = 1$	$l = 10$
25	0.522	0.576	**0.588**	0.258
50	0.527	0.576	**0.609**	0.257
75	0.530	0.579	**0.597**	0.258
100	0.535	**0.572**	**0.572**	0.258
125	0.529	**0.582**	0.547	0.258
150	0.532	0.569	**0.605**	0.258

Figure 1: Baseline bag-of-vectors model hyperparameter determination for hidden layer dimension h and learning rate l; entries are accuracies on the development set; GloVe dimension of 50

These were trained with GloVe vectors of dimension 50 to reduce the runtime. This determined that a learning rate of $l = 1$ was fairly optimal. Then, fixing the learning rate, we ran different numbers of epochs with a GloVe dimension of 50 to determine the optimal number of epochs and hidden dimension.

h	$e = 200$	$e = 400$	$e = 600$	$e = 800$	$e = 1000$	$e = 1200$	$e = 1400$	$e = 1600$	$e = 1800$	$e = 2000$
25	0.725	0.725	0.720	0.709	0.716	0.722	0.727	0.730	0.713	0.719
50	0.711	0.724	0.729	0.736	0.741	0.732	0.738	0.745	0.745	0.743
75	0.709	0.748	0.728	0.735	0.747	0.749	0.752	0.753	0.753	0.753
100	0.704	0.716	0.730	0.749	0.742	0.744	0.744	0.744	0.745	0.745
125	0.734	0.748	0.744	0.742	0.744	0.748	0.749	0.752	0.750	0.746
150	0.737	0.743	0.745	0.752	0.758	0.752	0.751	0.756	0.757	0.756
175	0.713	0.728	0.713	0.746	0.745	0.745	0.744	0.746	0.744	0.744
200	0.723	0.737	0.737	0.731	0.731	0.744	0.750	**0.762**	0.761	0.760
225	0.702	0.719	0.719	0.725	0.740	0.742	0.743	0.746	0.749	0.748
250	0.724	0.725	0.708	0.714	0.732	0.743	0.742	0.742	0.744	0.743

Figure 2: Baseline bag-of-vectors model hyperparameter determination for hidden layer dimension h and number of epochs e; entries are accuracies on the development set; GloVe dimension of 300

Thus, we determined that the optimal hyperparameters were $l = 1, e = 1600, h = 200$. With this configuration, the run on the test set yields an accuracy of 75.8%. This bag-of-vectors models sets the baseline for our future algorithms.

3.2 Identifying rationale

In order to identify what sentences or phrases in the article are particularly biased, for every article we attempt to identify a minimal subset of the article which yields the same prediction from our classification model. We used the same data set for this step as in the classification step described above.

One intuitive way to do this is to train a system that outputs a weight for each sentence in an article representing the importance of that word to the classification (the weights can also be computed per word as in [5], but in our case there are enough sentences per article to effectively compute weights per sentence — this also removes the need to incentivize picking consecutive words). These values are then used as masking weights for the words in each article, and the resulting words are input into the classifier. The loss is determined by how close the original output of the classifier is to this modified output. These weights can also be viewed as thresholds where all sentences below a certain threshold are removed. Finally, regularization must be applied to keep the model from assigning large weights for every word (as this would trivially result in a very low loss).

4 Approach

We explain the approach separately for the two components of this project.

4.1 Classification

For the classification, we chose to use common techniques involving a combination of convolutional neural nets (CNNs) and k-max pooling [3][4]. The articles were tokenized into sentences using the Python Natural Language ToolKit (NLTK) and then the words were replaced with GloVe vectors of dimension 300 as in the bag-of-vectors baseline model. Sentences with more than a certain number of words are removed, and the rest are used as rows in a rank-3 array. Thus, each article is turned into a rank-3 array where each row represents a sentence (each entry is a GloVe vector). A specific identifier was used to pad the end of each row. The inputs were fed through a linear layer and then into a repeated loop of applying a CNN followed by a k-max pooling operation. Finally, the resulting output of this process was passed through a final linear layer and dropout was applied to this layer. Let $x_{i,j}$ denote the jth GloVe vector in the ith sentence of the input article (indexed starting at 1), suppose d_i is the length of the ith sentence, suppose we are applying a CNN filter of width n and k'-max pooling and we apply this combined CNN/pooling operation a total of t' times. Then, the entire process can be summarized by the following:

$$(z_1)_{i,j} = x_{i,j}W + b_1 \tag{5}$$

$$\begin{cases} (z_{3,1})_{i,j} & = (z_1)_{i,j} \\ (z_{2,t})_{i,j} & = \begin{bmatrix} (z_{3,t})_{i,j} & (z_{3,t})_{i,(j+1)} & \cdots & (z_{3,t})_{i,(j+n)} \end{bmatrix} F \text{ for } j+n \leq d_i \text{ and } 1 \leq t \leq t' \\ (z_{3,t})_{i,j,k} & = \max_{l=0,1,\ldots,k'} (z_{2,t-1})_{i,j+l,k} \text{ for } 1 < t \leq t' \end{cases} \tag{6}$$

$$(z_4)_i = \frac{1}{d_i - t'(n+k'-2)} \sum_{j=1}^{d_i - t'(n+k'-2)} (z_{3,t'})_{i,j} \tag{7}$$

$$z_5 = \sum_i [(h_i \circ (z_4)_i)U + b_2] \tag{8}$$

where the curly braces indicate the repeated CNN/max-pooling operation (where $z_{3,1}$ is the "base case"). Intuitively, the CNN step takes n consecutive vectors in a sentence and outputs a new vector by applying its filter F (because it must operate on n consecutive vectors, this reduces the effective overall sentence length by $n-1$). Similarly, the k'-max pooling step takes k' consecutive vectors and takes the maximum of each component to construct a new vector (for the same reason as before, this reduces the effective overal sentence length by $k'-1$). We then average the resulting vectors in each sentence to get one vector per sentence (this is $(z_4)_i$). Finally, we pass the resulting vector through a final linear layer (after applying dropout via the vector h). If the hidden dimension is h, the GloVe vector dimension is g, and the number of news providers is m, then the parameters are $W \in \mathbb{R}^{g \times h}, b_1 \in \mathbb{R}^h, F \in \mathbb{R}^{nd \times d}, U \in \mathbb{R}^{d \times m}, b_2 \in \mathbb{R}^m$. The final loss is computed identical to that in the bag-of-vectors model; see equation (4) with z_5 in the place of z_3.

4.2 Rationale Identification

For rationale identification, we modeled our approach after that described in the paper by Lei et al. which uses a bidirectional RNN model for this purpose [5]. In this case, the articles were again tokenized into sentences and the words were identified with their corresponding GloVe vectors. However, we then averaged the vectors for the words in a sentence and the input to the model consisted of a matrix where each row was the average GloVe vector in a particular sentence (so the number of rows is equal to the number of sentences in a given article). These vectors are initially transformed by a linear hidden layer, and then a bidirectional RNN is applied to the resulting encoded sentence vectors. Each RNN takes in a sentence vector and outputs a vector of identical shape. Thus, after the bidirectional RNN runs, each sentence vector is mapped to two vectors of identical shape which are then concatenated and passed through a final output layer. The output for each sentence is a 1×2 vector. After passing this through a softmax so the two entries sum to 1, we regard the first entry as

the "probability of dropping the sentence" and the second entry as the "probability of keeping the sentence." In practice, the second entry is then used to weight the sentences when passing into the CNN classifier.

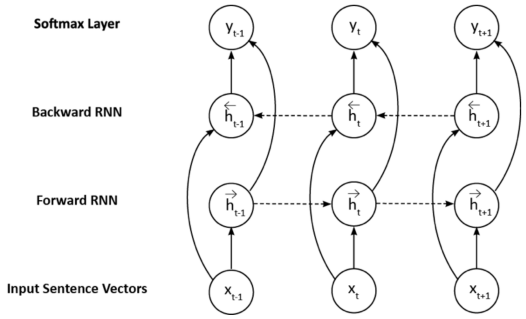

Figure 3: Schematic of bidirectional RNN

Let x_i represent the average of the GloVe vectors in the ith sentence (indexed starting at 1) and let n represent the number of sentences. Then mathematically, this model breaks down as follows:

$$(z_1)_i = x_i W + b \tag{9}$$
$$(z_{2f})_1 = \sigma_f(x_1 W_1 + b_1) \tag{10}$$
$$(z_{2f})_i = \sigma_f(x_i W_1 + b_1 + (z_{2f})_{i-1} U_1) \text{ for } 1 < i \leq n \tag{11}$$
$$(z_{2b})_n = \sigma_b(x_n W_2 + b_2) \tag{12}$$
$$(z_{2b})_i = \sigma_b(x_{n-i+1} W_2 + b_2 + (z_{2b})_{i+1} U_2) \text{ for } 1 \leq i < n \tag{13}$$
$$(z_3)_i = \text{softmax}\left([(z_{2a})_i \quad (z_{2b})_i]\, U + b'\right) \tag{14}$$

where $(z_3)_i$ is the 1×2 vector representing the "probabilities" of dropping or keeping a sentence. The loss is then computed by taking the second entry of $(z_3)_i$, normalizing this vector with the L2 norm, and then using the result as masking weights for the inputs of the sentences to the CNN classifier. Finally, we computing the L2 loss (regularized using the variance of $(z_3)_i$ distribution) between the original output of the CNN classifier and the output when the masked sentences are used as inputs. The intuition behind regularizing with the variance of $(z_3)_i$ is that if the $(z_3)_i$ are very close together (and the CNN classifier biases b_1, b_2 are relatively small) the output will be similar to the result if you kept all of the sentences. If the hidden dimension is h, the GloVe vector dimension is g, then the parameters are $W \in \mathbb{R}^{g \times h}; b \in \mathbb{R}^h; W_1, W_2, U_1, U_2 \in \mathbb{R}^{h \times h}; b_1, b_2 \in \mathbb{R}^h; U \in \mathbb{R}^{2h \times 2}; b' \in \mathbb{R}^2$ (and σ_f, σ_b are activation functions which for our purposes will be set as hyperbolic tangent).

5 Experiments

The CNN classifier model has six hyperparameters: the value for max-pooling, k; the width of the CNN filter, n; the hidden dimension, h; the number of epochs, e; the learning rate $l = 0.01$; and the number of times the CNN/max-pooling operation is performed, t'. We performed hyperparameter determination similar to that for the bag-of-vectors model. Several of the determination tables (showing accuracies) are shown below.

n	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$
1	**0.665**	0.612	0.601	0.647	0.628
2	0.595	**0.629**	0.609	0.605	0.598
3	**0.627**	0.602	0.621	0.590	0.600

n	Accuracy
1	0.6943
2	0.6972
3	**0.7084**
4	0.7079
5	0.6704

Figure 4: Determination of optimal max-pooling value k with $t' = 1$, $e = 10$, $h = 20$, $l = 0.01$ (left) and determination of optimal CNN filter width n with $k = 1$, $t' = 1$, $e = 20$, $h = 40$, $l = 0.01$ (right); GloVe dimension of 50

After these tests, we determined that the optimal hyperparameters were as follows:

$$k = 1 \qquad n = 3 \qquad t' = 1 \qquad l = 0.01 \qquad e = 60 \qquad h = 240 \tag{15}$$

A final run on the test set resulted in an accuracy of 84.0%. In terms of the five news providers, the accuracies broke down as follows:

News Provider	Accuracy
CNN	93.4%
Reuters	87.2%
New York Times	86.5%
Fox News	77.4%
Breitbart	55.8%

Figure 5: Accuracies for specific news providers

Finally, we show the learning curve for the final run on the test set:

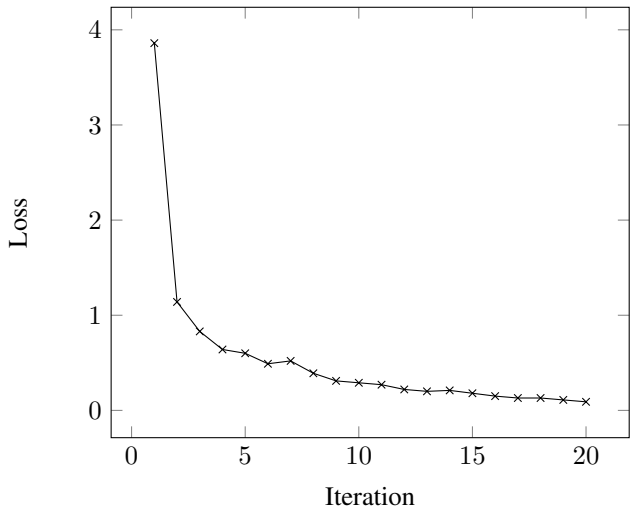

Figure 6: Learning curve showing average loss for first 20 epochs of final run of CNN classifier on test set

As previously described, this trained classifier was then used in training the rationale identifier. This model has four hyperparameters: the regularization constant λ, the hidden size h, the number of epochs e, and the learning rate l. Again, we performed hyperparameter determination and several

of the relevant tables are shown below (the entries of the tables are the percent of results that were consistent with the original output of the CNN classifier).

l	$\lambda = 0.01$	$\lambda = 0.1$	$\lambda = 1$
0.01	0.9967	0.9946	**0.9975**
0.1	0.9620	0.9600	0.9322
1	0.0517	0.9298	0.0520
10	0.0516	0.0517	0.9297

h	Accuracy
180	0.932
210	0.956
240	**0.998**
270	0.954

Figure 7: Determination of optimal regularization and learning rate with $e = 5$, $h = 240$ (left) and determination of optimal hidden dimension with $e = 5$, $l = 0.01$, $\lambda = 1$ (right); GloVe dimension of 300

The optimal hyperparameters were determined to be

$$e = 20 \qquad h = 240 \qquad \lambda = 1 \qquad l = 0.01 \tag{16}$$

With these values, a final run on the test set gave an accuracy of 99.6%. Some examples of the results are shown below.

Most Indicative Sentences for Bias	**CNN Classifier Prediction**
But this year you may be aware of one nominee, the Iranian director Asghar Farhadi, who is among the prominent artists to be affected by President Trump's ban on visas for travelers from predominantly Muslim countries. The race for the Oscar for best foreign-language film usually doesn't get much attention , considering the category's scope. Mr. Farhadi's drama " The Salesman " is one of five films up for the foreign-language film statuette.	The New York Times
"I am with you 1,000 percent," Trump said in a short address to CIA staff after his visit to the agency headquarters in Virginia. (AFP) WASHINGTON, United States – President Donald Trump told the CIA it had his full support on Saturday as he paid a visit to mend fences after publicly rejecting its assessment that Russia tried to help him win the US election.	Breitbart
President Donald Trump is commemorating International Holocaust Remembrance Day by slamming America's door on refugees. On International Holocaust Remembrance Day, President Trump issues an order demanding "extreme vetting" of refugees. Mark Hetfield: this is a tragic case of history repeating itself.	CNN
Hong Kong Tourism Board said it commissioned the territory's only certified Lego professional builder, Andy Hung, to create the (6.5 ft) tall statue.	Fox News
"Tariffs are never good," CEO Lewis Gradon said in a phone interview.	Reuters

Figure 8: Examples of results of bidirectional RNN rationale identifier

6 Conclusion

This project shows promising results in developing a system to extract and identify bias-inducing sentences in news articles. In particular, the ability of the bidirectional RNN to achieve very high consistency with the classifier implies that the limiting factor is the accuracy of the classifier. However, utilizing convolutional neural nets in the classifier also showed promise, outperforming the baseline bag-of-vectors model by nearly 10% and achieving close to 85% accuracy in classifying articles among the 5 news providers.

There are a variety of ways that the classification and extraction could be improved. The classifier was trained on a very limited data set with only 5 providers, one of which (Breitbart) had only 600 articles. Unsurprisingly, the classifier performed the worst on this news provider. This indicates that performance could be improved by increasing the number of articles in the data set (which would potentially allow an increase in the size of the parameters for better accuracy). Other possible improvements that were not investigated include using dynamic k-max pooling, using average pooling, and/or not applying the convolution neural network a fixed number of times. The extraction could also be modified in a number of ways. First, we created weights for each sentence, though a system could easily be designed to weight different words (and penalize for words being far from each other). There are also different objective functions that could be explored aside from the L2 loss regularized with variance of the weights. Examples include the L1 loss, not normalizing the weights and then regularizing based on the size of the weights (this could perform better in the case that the classifier biases b_1, b_2 are large). Finally, one could explore different activation functions such as ReLu or softplus for the bidirectional RNN that may yield better results than the hyperbolic tangent.

As a final note, it would be of interest to see if these results could be replicated for a data set with far more than 5 news providers. As noted earlier, the classifier could also be trained to recognize different biases; for example, it could be trained to predict the sex, race, or gender of the author. Overall, if similar results can be achieved in these cases, the detection of bias using trained computer systems could become a powerful tool for helping to provide objective news coverage in the future.

Acknowledgments

We would like to thank Arun Chagarthy and the Stanford Computational Linguistics Lab for guidance in this project. We would also like to thank Chris Manning, Richard Socher, and the rest of the CS 224N staff for access to the Microsoft Azure GPUs and their time in the course.

Contribution

All members worked on and contributed significantly to all parts of the project. In particular, Nicholas Hirning and Shreya Shankar worked on the gathering and preprocessing of the data, implementing the CNN classifier, and integrating the CNN classifier into the bidirectional RNN. Andy Chen implemented the bag-of-vectors baseline and the bidirectional RNN.

References

[1] Allcott, Hunt and Matthew Gentzkow. "Social Media and Fake News in the 2016 Election." *The National Bureau of Economic Research (2017)*: n. pag. Web.

[2] Bell, Peter. 0123456789. Auckland, N.Z.: U of Auckland, 1992. *NIPS Proceedings*. Neural Information Processing Systems Foundation, Inc. Web.

[3] Kim, Yoon. "Convolutional Neural Networks for Sentence Classification." *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (2014): n. pag. Web.

[4] Lei, Tao, Regina Barzilay, and Tommi Jaakkola. "Molding CNNs for Text: Non-linear, Non-consecutive Convolutions." *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing* (2015): n. pag. Web.

[5] Lei, Tao, Regina Barzilay, and Tommi Jaakkola. "Rationalizing Neural Predictions." *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing* (2016): n. pag. Web.

[6] Pavitra, R., and P. C D. Kalaivaani. "Weakly Supervised Sentiment Analysis Using Joint Sentiment Topic Detection with Bigrams." 2015 2nd International Conference on Electronics and Communication Systems (ICECS) (2015): n. pag. Baselines and Bigrams: Simple, Good Sentiment and Topic Classification. Stanford University. Web.

[7] Pennington, Jeffrey, Richard Socher, and Christopher Manning. "Glove: Global Vectors for Word Representation." *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (2014): n. pag. Web.

[8] Recasens, Marta, Cristian Danescu-Niculescu-Mizil, and Dan Jurafsky. "Linguistic Models for Analyzing and Detecting Biased Language." *Linguistic Models for Analyzing and Detecting Biased Language*. Stanford University, n.d. Web.