Identifying Political Bias with Weak Supervision

Ramya Balasubramaniam
UC Berkeley School of Information
ramya.girish@berkeley.edu

Daniel Elkin
UC Berkeley School of Information
dan.elkin@berkeley.edu

Andres Borrero
UC Berkeley School of Information
andresborrero@berkeley.edu

Abstract

In this paper, we seek to determine the political bias of text using a semi-supervised approach known as weak supervision. This method allows us to combine a labeled but relatively small dataset, the Ideological Books Corpus (IBC), with a much larger dataset consisting of New York Times articles and Reddit comments with the goal of classifying text as liberal or conservative. We find that a weakly-supervised model matches the highest reported accuracy of predicting the political sentiment of articles in the IBC, even when including neutral labels in the data.

1 Introduction

The task of identifying positive or negative sentiment in text has been widely studied (Mntyl, 2018), but relatively little attention has been given to identifying political bias in text. A model that could identify such bias would have broad applications, such as evaluating the ideological bent of news outlets or flagging articles for review in sources of purportedly unbiased information, such as Wikipedia (Recasens, 2013). The model could be used to extract biased sentences from larger corpora as a preprocessing step for other machine learning tasks that require biased text as input. Finally, if successful, the weakly-supervised model described below could be extended to classification problems in other domains.

2 Background

A deep learning approach was first applied to political bias detection by Iyyer et al., who used a recursive neural net (RNN) to classify the ideology of articles in a labeled subset of the Ideological Books Corpus (IBC) in 2014 (Iyyer, 2014). The authors reported an accuracy of 69.3 percent using an RNN, an improvement of about six percent on various bag-of-words models that they used as baselines. The authors suggest that more complex RNNs might further improve results, but we hypothesize that increasing the amount of data available for training may also yield better accuracy. We next discuss methods for expanding the Iyyer dataset, which is the largest of its kind but consists of only approximately four thousand manually-labeled sentences, using weak-labeling.

Weak supervision is a technique for increasing the amount of available training data without resorting to further manual-labeling and has been described by Dehghani et al. in various papers in 2017 (Dehghani, 2017). As described further below, the method consists of assigning uncertain, or weak, labels to a large corpus, which is then used as the input to some classifier, which is known as the target network. A separate network is trained using a smaller, manually-labeled

dataset. The latter network is known as the confidence network and is responsible for determining how likely the weak label is to be correct and thus how much weight we should assign to the example in the target network. Via this interaction, we are able to combine a small dataset in which we have high confidence in the labels with a much larger dataset whose labels are less certain. The augmented dataset is then used for the classification task.

3 Methods

Our proposed methodology involved **Data Extraction** from large corpora, **Data Preprocessing**, **Creation of Labels** for weakly annotated data using Weak Supervision, **Creation of Political-bias Sentence Classifier** using Convolution Neural Networks and **Comparative Analysis of results** with the ones obtained by Iyyer et al. [3].

3.1 Data Extraction

Two large corpora for extracting sentences constituting the weakly annotated dataset (hence, referred to as U): ***BigQuery Reddit corpus*** consisting of Reddit comments and ***The New York Times corpus***.

The Reddit comments were extracted using BigQuery, which is syntactically similar to SQL, from threads with known ideological biases, such as those whose subjects are Donald Trump or Bernie Sanders. For the NYT corpus, we extracted six years (1987-1992) worth of news from XML files using specific packages in python. The extracted sentences were given a more structured format and saved into CSV files for convenience and re-usability.

3.2 Data Pre-processing

The comments/sentences from the aforementioned large corpora and Ideological Book Corpus (IBC) were pre-processed using standard steps in Natural Language Processing (NLP). The pre-processing steps that we used were *Tokenization*, *Canonicalization*, *Conversion to IDs* and *Padding Sentences of unequal length with zeros*. For this step standard functions in NLTK package and pre-defined python classes were used.

During this stage, it was noted that the NYT corpus and IBC dataset have identical encodings (ASCII).However, the Reddit posts had four different types of encodings (ASCII, UTF-8, Windows-1252 and ISO-8859-1). The vocabulary and language structure of the sentences in NYT corpus and IBC dataset are formal as opposed to the vocabulary and language structure used in Reddit posts (which was found to include profanities and expletives). Moreover, the conservative and liberal biases could not be attributed to these some of posts as they were found to be racially charged and way more extremist in nature than the sentences that we intend to classify (ones found in news reports and formal text). So, we used NYT corpus data to train our final model based on Convolutional Neural Network.

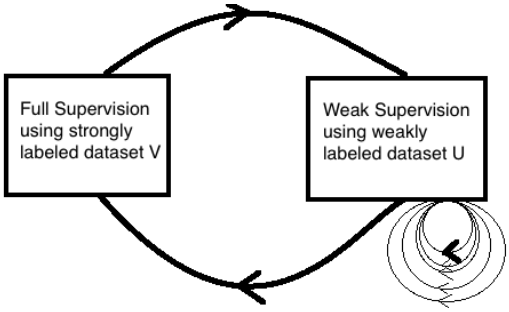
3.3 Creation of labels

The crucial part of our implementation involved creating a suitably large dataset consisting of strongly labeled sentences considered as gold standard (annotated by human experts) and weakly labeled sentences that were annotated using a weak annotator.

Following Dehghani et al., we use a simple lexicon-based approach in which we maintain a dictionary of words that indicate conservative or liberal bias and assign a label according to the counts of these words in the sentence. As there is no equivalent to SentiWordNet for political bias, however, we constructed this dictionary using K-means clustering. The other approaches that we considered, such as using a Naive Bayes model or assigning labels based on assumptions about the ideology of the source of the examples turned out to be unworkable. In the former case, the only labeled dataset we had access to was the IBC corpus, and we wanted to maintain independence between our weak and strong datasets. In the latter case, assumptions about the source could not be extended to the examples in the IBC corpus, and so it was not possible to assign weak labels to these examples, as required for the confidence network and described in section 3.3 below.

For the K-means clustering, we used the td-idf tokenizer to define a bag of words model for our combined dataset (IBC and NYT corpus data). K-means clustering was applied for various values of k or the number of clusters (k=2,3). Individual words appearing in each cluster were examined and correlated with the sentences in the dataset to finally determine the correct label to associate with the cluster. Once the correct labels had been linked

Figure 1: Alternating between Full and Weak supervision



to the clusters, we populated our new dictionaries used for identifying liberal and conservative biases (our weak annotator).

Next, the method of **Weak Supervision** was used to correct the weak labels assigned to the latter set of sentences using the strongly labeled set of sentences. In the proposed method for Weak supervision[4], there were essentially two steps that were alternated in order to maximize the correctness of the labeling process. First, the strongly labeled dataset V was subjected to training in the **Full Supervision Step** and then, the weakly labeled dataset U was subjected to training in the **Weak Supervision Step**. During each process of alternation, we carried out Full Supervision and Weak Supervision steps in the ratio r (as suggested in [4], r = 1:10). Our weak supervision approach could be effectively described using the following diagram:

During the Full supervision, each instance $\boldsymbol{x_j}$ in the strongly labeled dataset V was sent through a feed-forward neural network along-with the weak labels generated using the weak annotator $\hat{\boldsymbol{y_j}}$ and a confidence score $\boldsymbol{c_j}$ was calculated for each instance using a pre-defined confidence scoring scheme.

Initially, a very simple binary scoring scheme was used, in which confidence scores of one were given to the examples in the IBC dataset if the true labels matched the weak labels and confidence scores of zero for the examples where it didn't match. Later, absolute differences between the true and weak labels were calculated and used as confidence scores.

The network formed by using IBC dataset, that we call *Confidence Network* was trained to predict a confidence score $\tilde{\boldsymbol{c_j}}$. The loss function $\boldsymbol{L_c}$ calculated the cross-entropy loss between the target confidence score calculated using the true label $\boldsymbol{y_j}$ of an instance and the weak label $\hat{\boldsymbol{y_j}}$, and the predicted confidence score calculated using the predicted label $\tilde{\boldsymbol{y_j}}$ predicted by the confidence network and the weak label $\hat{\boldsymbol{y_j}}$.

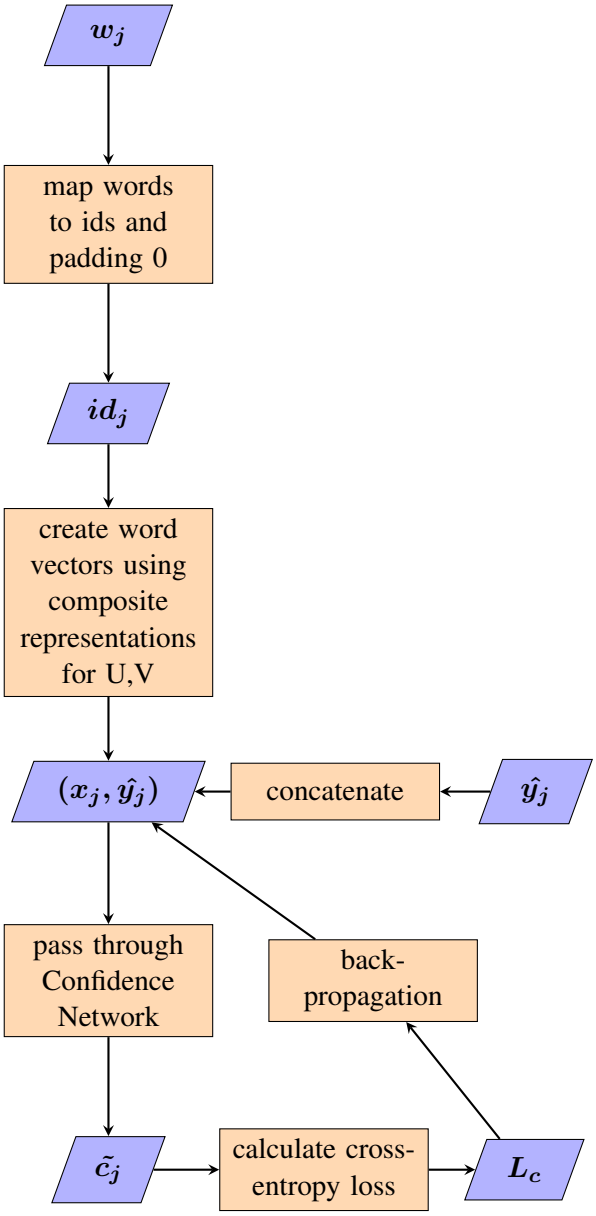


Figure 2: Full Supervision

The weak annotator and the confidence scoring scheme were the crucial hyper-parameters during the weak supervision phase and controlled the accuracy of predicting labels for our weakly labeled dataset U. The average cross-entropy loss for the instances in the Confidence network is given by:

$$\boldsymbol{L_c} = \frac{1}{|\boldsymbol{V}|} \sum_{j}^{j=|\boldsymbol{V}|} \boldsymbol{c_j}(1-\hat{\boldsymbol{c_j}}) + \hat{\boldsymbol{c_j}}(1-\boldsymbol{c_j}) \quad (1)$$

During the Weak supervision, each instance $\boldsymbol{x_i}$

in the weakly labeled dataset U was sent through another feed-forward neural network, which we call *Target Network*. Additionally, to obtain the confidence score for each instance in the U dataset, instances were concatenated with their associated weak labels and sent through the confidence network. The resulting confidence score c_i was multiplied by the corresponding cross-entropy losses L_t for each instances in this network. This allowed us to selectively back-propagate through the Target Network, by allowing us to associate high weights to gradients of instances having high confidence score and low weights to gradients of instances having low confidence score. The final weighted cross-entropy loss for the instances in the Target network is given by:

$$L_w = \frac{1}{|U|} \sum_{i}^{i=|U|} c_i L_{t,i} \tag{2}$$

where $L_{t,i}$ is the cross-entropy loss associated with each instance i

The strong labels for the dataset U were obtained by using the target network that was trained through this selective process of Weak supervision.

3.4 Creation of Political-bias Sentence Classifier

As opposed to the phrase level classification done using **Recursive Neural Networks** in Iyyer et al[3] paper, we used sentence level classification for our approach. Different models were used for implementing the Political-bias classifier. Our baseline model was implemented using the **Naive Bayes algorithm**. The model that was used in or final evaluative and comparative analysis implemented **Convolution Neural Networks**. The results obtained have been tabulated and discussed at length in the following section.

The pre-processed sentences constituting our U dataset (labeled using *Weak Annotation*) were suitably split into train and dev sub-sections for training the aforementioned models (both Naive Bayes and Convolution Neural Networks). The resulting models were evaluated on Ideological Book Corpus (IBC) dataset. The results of the evaluation was compared to the results obtained by Iyyer et al.[3] by implementing phrase level classification.

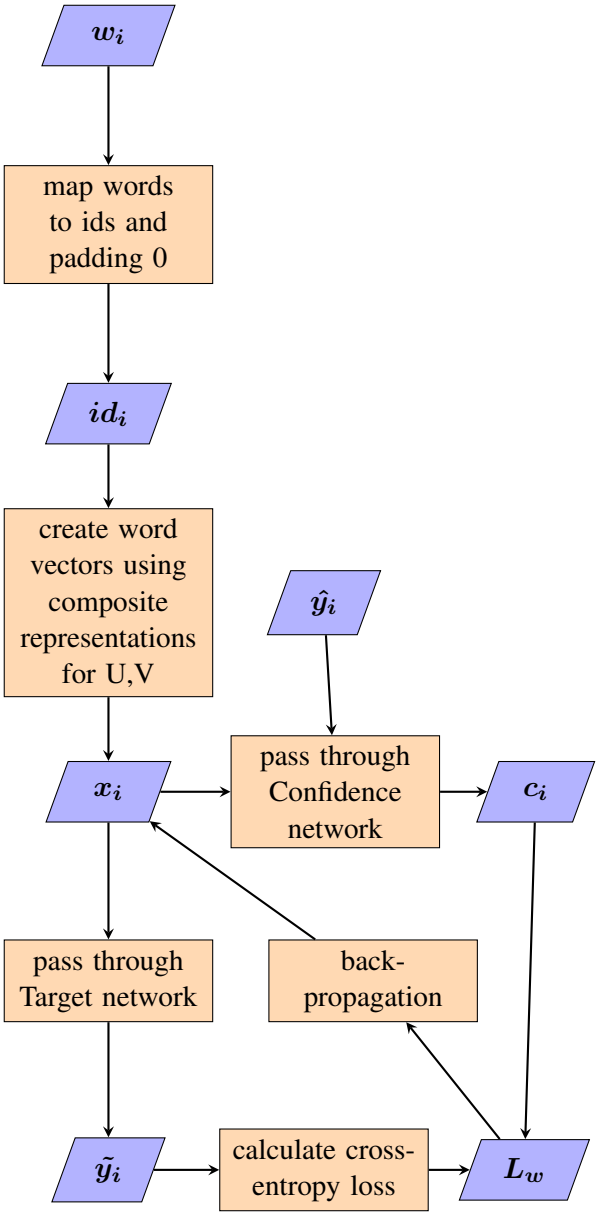


Figure 3: Weak Supervision

4 Results and discussion

In this section we discuss the results from our baseline and models.

4.1 Baseline Models

As a simple baseline, we constructed a Naive Bayes model using raw token counts. This baseline scored a 57.3% when considering all three classes (liberal, conservative, neutral) and 65.4% when considering only sentences with liberal or conservative labels, which is the approach taken by Iyyer et al.

We also constructed a convolutional neural network trained only on a subset of the IBC data, which scored only slightly better than random selection with an accuracy of 51.68% on the two-class classification problem.

Table:1 Baseline Models

Method	No.of classes	Accuracy
Naive Bayes	3	57.3%
Naive Bayes	3	65.4%
CNN	2	51.68%

4.2 Weak Supervision Models

Table:2 Using arbitrarily defined dictionary

Stage	Eval. set	Accuracy
Full Supervision	Dev	69.53%
Weak Supervision	Dev	98.27%
Final CNN	Test(IBC)	13.33%

Table:3 Using dictionary created by K-means clustering

Stage	Eval. set	Accuracy
Full Supervision	Dev	36.23%
Weak Supervision	Dev	92.45%
Final CNN	Test(IBC)	69.33%

Curiously, the accuracy scores during full supervision fall from **69.53%** to **36.23%** when the weak annotator was changed from an arbitrarily defined dictionary to a more logically constructed K-means clustering applied on IBC and NYT corpus data which was used during our weak supervision stage and to build our final CNN model. On further investigation it was observed that in the former case since most the words defined in the dictionary were neither a part of IBC dataset nor part of NYT corpus, our weak annotator labeled majority of our sentences as neutral. In the latter case our words in the dictionary were taken from IBC and NYT corpus and thus received more labels in the conservative and liberal categories. Unfortunately, this also means that our confidence network learns very little about how to assign a confidence score in the first case. The fact that our confidence network learns more in the second case (when a better weak annotator is defined) is reflected from the scores observed during the final stage of analysis (a jump from 13.33% to 69.33%). All these results/scores were observed on three class problem as opposed to the 69.3% accuracy reported by Iyyer at el on two class problem.

By referring to the previously defined dictionary as arbitrary, we want to stress the fact that the constituent words were chosen keeping the language used in liberal or conservative quotes in a formal settings (news articles and congressional speeches). Our improvised dictionary was meticulously populated with words obtained from the clusters, formed using K-means++ algorithm. This algorithm was applied on IBC and NYT corpus data. The two set of dictionaries that were used in our project have been included in the appendix A.

The hyper-parameters that were tweaked during the training of confidence, target and final CNN networks have been listed along with the various values tweaked in appendix B.

References

Denny Britz. *Implementing a CNN for Text Classification in TensorFlow*. December 11, 2015. Retrieved from `http://www.wildml.com/2015/12/implementing-a-cnn-for-text-classification-in-`

Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. In *EMNLP 2014*.

Mika Viking Mntyl, Daniel Graziotin, Miikka Kuutila. 2018. The Evolution of Sentiment Analysis - A Review of Research Topics, Venues, and Top Cited Papers. In *Computer Science Review*, Volume 27, February 2018, Pages 16-32.

Marta Recasens, Cristian Danescu-Niculescu-Mizil, and Dan Jurafsky. 2013. Linguistic models for analyzing and detecting biased language. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*.

Mohit Iyyer, P. Enns, J. Boyd-Graber, P. Resnik. 2014. Political Ideology Detection Using Recursive Neural Networks. In *Proceedings of ACL 2014*.

Mostafa Dehghani, A. Severyn, S. Rothe, and J. Kamps. 2017. Avoiding Your Teachers Mistake: Training Neural Networks with

Controlled Weak Supervision. arXiv preprint arXiv:1711.00313.

A Weak Annotators: Dictionaries

We used two sets of dictionaries in our project. The first one was a compilation was words that we commonly came across in popular ideologically biased formal text (news articles and speeches):

Table:4 Dictionary created arbitrarily

Liberal	Conservative
'liberal'	'conservative'
'democrat'	'republican'
'pro-choice'	'pro-life'
'agnostic'	'free market'
'abortion'	'catholic'
'gay'	'christian'
'freedom'	'left'
'climate'	'tax cut'
'secular'	'left-wing'
'civil'	'NRA'
'liberty'	'GOP'
'equality'	'fair trade'
'regulation'	'second amendment'
'violence'	
'ACLU'	
'right'	
'far-right'	

For the second one, we compiled the list of words using the K-means clustering algorithm.

Table:5 Dictionary created using K-means clustering

Liberal	Conservative
'abortion'	'house'
'congress'	'committee'
'rights'	'senator'
'anti'	'leader'
'law'	'democrat'
'court'	'approved'
'party'	'majority'
'government'	'chairman'
'women'	'republicans'
'federal'	'democratic'
'caucuses'	'democrats'
'issue'	'plan'
'national'	'voted'
'right'	'years'
'people'	'billion'
'support'	'tax'
'million'	'administration'
'health'	'states'
'public'	'passed'
'city'	'year'
'make'	'new'
'000'	'president'
'caucus'	'today'
'political'	'republican'
'percent'	'senate'
'supreme'	'dole'
'like'	'intelligence'
'groups'	'conference'
'time'	'governor'
'money'	'leaders'
'legislature'	'finance'
'say'	'judge'
'decision'	'banking'
'civil'	'reagan'
'american'	'week'
'life'	'budget'
'mr'	'measure'
'bush'	'assembly'
	'legislation'
	'members'
	'nomination'
	'united'
	'expected'
	'floor'
	'white'
	'campaign'
	'hearings'
	'judiciary'

B Hyper-parameters

Some of the hyper-parameters that we tweaked are as follows:

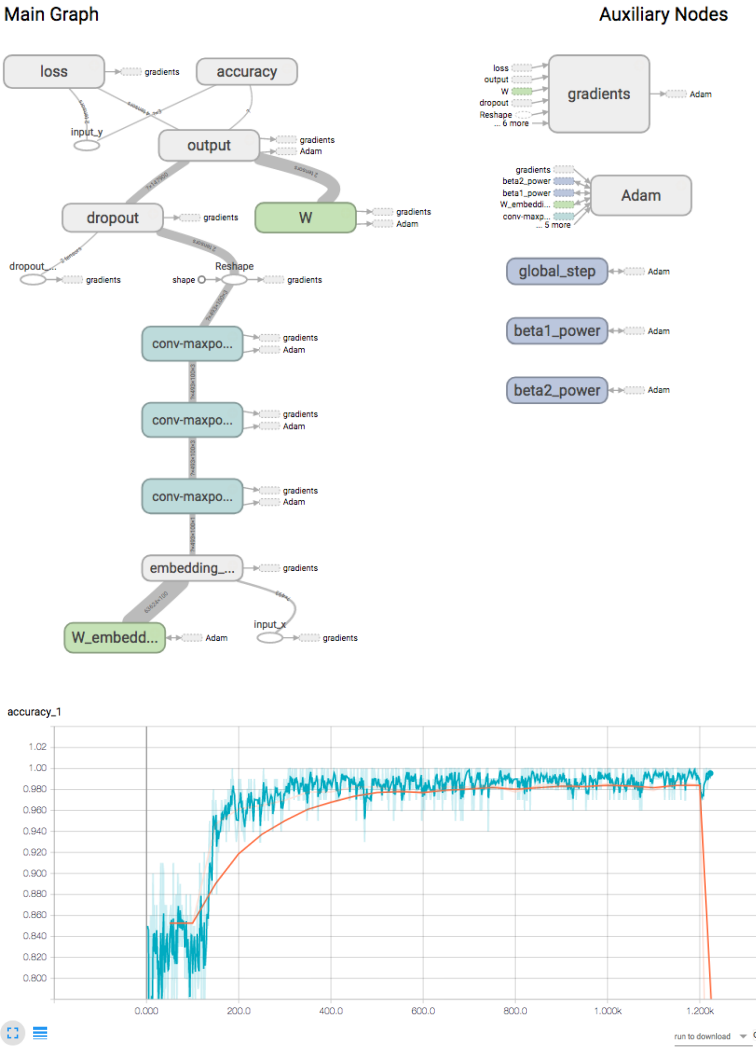
Table:6 Hyper-parameters (Weak Supervision)

Hyper-parameter	values
No. of Epochs (Conf. N/W)	5,10,20
No. of Epochs (Tar. N/W)	50,100,200
Ratio of Epochs(Conf.:Tar.)	1:2,1:5,1:10
Word embedding Dim.	100,125,150,175
Hidden Layer Dim	[12,24] [32,64] [32,64,128]
batch size(conf.N/W)	20,30,50
batch size(tar.N/W)	128,200,250,500

Table:6 Hyper-parameters (Final CNN)

Hyper-parameter	values
No. of Epochs	1,2,5
Word embedding Dim.	100,128
filter widths	[5,5] [5,5,5] [3,3,3]
filter heights	[5,5] [5,5,5] [3,3,3]
channel-in	[1,12] [1,10,12] [1,3,4] [1,3,3]
channel-out	[12,24] [10,12,24] [3,4,5] [3,3,3]
No. of filters	2,3
batch size	100,200,250

C Tensorboard Graphs (CNN)



D Source Code

The source code for this project is available at `https://github.com/dan55/political_bias`. The notebook used to generate the results is `https://github.com/dan55/political_bias/blob/03f5872c6e98ea337655bc8b74e91d724edcdf9b/Bias_sentence_CNN/CNN_attempt2.ipynb`.