News Bias Classification using LSTM

Sanika Mhadgut
B.Tech Data Science
MPSTME, NMIMS
Mumbai, India
sanika.mhadgut@gmail.com

Gayathri Shrikanth
B.Tech Data Science
MPSTME, NMIMS
Mumbai, India
gayathri.shrikanth@gmail.com

***Abstract*—Over the past decade, political campaigns have been increasingly waged on news sites and there have been ongoing concerns regarding how one's political opinions and bias affect the information people read online. Though the current methods are fast, automated, and scalable compared to the cumbersome manual classification technique, there are only a few approaches systematically analyze media bias. Bias classification has always been a widely debated and disputed topic which requires lots of context and current information about the political system which the existing Machine learning models fail to identify and perform poorly on unseen data. The objective of this research was to take a different technical approach to solve the problem of detecting political bias using the Long Short Term Memory model. The models were trained on various news articles scraped from the web. Results showed that the LSTM model for bias detection achieved training accuracy 0.96 and testing accuracy of 0.93. We concluded that LSTM models offer an attractive solution for a major bottleneck associated with machine learning and text mining when there is a lack of high-quality annotated examples but the problem with a prediction accuracy on new data still exists.**

I. Introduction

Bias is a deviation from its true value, it is a prejudice against or in favour of a thing which is unfair to that person or group. Media bias can refer to deviating coverage amounts across event types or skewed representation of the events[1]. Because of this it has become very necessary to check for the bias in the media posts, so that it doesn't affect anyone and is accurately identified whenever it is presented.Media feeds containing biased political information tampers the information environment and influences peoples decisions.

This research was motivated by how Deep learning algorithms can identify news media bias in any literature. Our aim was to deploy a model which swiftly identifies the nature, amount and direction of bias in a given article. Bias can be classified as left bias, right bias, neutral bias. The issue of media bias has been addressed by comparing and contrasting established research on the topic in the social sciences with the state-of-the-art technical approaches from computer science.

Articles written by newspapers often tend reflect the authors' inherit point of view especially in the political spectrum. When the general public get fed with biased news or information supporting or opposing their political viewpoints, it creates a dogma that hinders them from a subjective, neutral information environment. The writer of the post may not create a direct impact but his choice of words in the post may be such that the reader might indirectly get influenced. It is generally identified that there are three types of bias - left, right, neutral. Left wing is usually opposed to the social hierarchy, followed by people who are usually liberal. Right wing people usually claim that social/economical inequality is natural and inevitable, which is necessary and beneficial for society, they are generally conservative Neutral or Least bias refers to articles that are neither left nor right biased.

Bias in a post can be identified using the choice of words of the writer. Use of spin words and phrases, words giving a dramatic effect or words that portray someone in a negative light indicate bias. Example of such words are serious, Refuse, Crucial, Turn up the heat, Critical, Offend, Even though, Monumental, Finally, Refusing to sat, Dodged, Admission, Came to light, mocked, Lashed out, Erupted etc.Successful identification of such words is one of the basic methods to detect bias.

The outline of this article is structured as follows: In Sect. II, we have described the State of art, Research gaps, scope of the research paper. In Sect. III, developed a conceptual understanding of the Mathematical model of how LSTM model classifies the bias. In Sect. IV, briefly introduced the most important approaches for preprocessing of the text in a flowchart. In Sect. V, describes the dataset and metrics used for performance, along with details about the envirnoment used. In Sect. VI, explained results in detail with a discussion of main findings and analysed the forms of media bias, to automatically identify the form of bias. In Sect. VII, we conclude the article with future recommendations to improve the analysis.

II. Literature Survey

Much research in the NLP community has been focused on sentiment analysis, fake news detection,topic classification for news articles and media bias detection primarily with LDA, multinomial Naive Bayes, clustering algorithms, and support vector machines. Newer methods like LSTM and Convolutional Neural Networks (CNN) have also appeared as models showing promising results in the field of natural language processing. One such approach used was to find the polarity of each individual word in the article and gauge its overall average sentiment. The extremely positive or negative articles were classified as biased. Recasens et. al. detected language bias [6], using a word dictionary curated by Wikipedia editors to ensure articles conform to Neutral Point of View (NPOV) rules. A team of student researchers from Yale University developed a plugin called OpenMind to counter Fake News. The plug-in used existing sentiment analysis technology to analyze an article, identify the major players and any political slant and suggest the reader other stories on the same topic with an alternate viewpoint. However this project did not give information about the amount and type of bias present in the article. BS Detector, a Chrome extension with similar functionality used a curated list of unreliable news sources to flag online articles as being fake news or otherwise unreliable. Here the detection of bias was based on reliability of the source of the article rather than the content. Iyyer et al. developed a Recursive Neural Network (RvNN) model [7] to create an Ideological Book Corpus (IBC) , which labels sentences and phrases based on their political ideologies. Their neural network was proven to outperform contemporary methods such as bag-of-words models and hand-designed lexica.

Building a supervised model for bias detection requires a data set with article-level labeling. The approach of manually labeling and classifying each news article as left, right and least biased requires a lot of resources and is many times not possible. For training the model itself the minimum requirement of labeled samples is in the thousands. Class imbalance also has to be taken into consideration while creating the data set. Bias identification may be highly subjective to the topic and person thus verifying articles will commonly require considerable time from an expert. As the data set creation itself takes a lot of time most of the available labeled articles are outdated for analyzing contemporary articles, due to shifts in topics and news cycle.When the model is trained on outdated data it has been observed that though it performs well on the labeled test set, prediction on current news articles is not accurate. An alternative approach to creating labels was through distant supervision, where labels were generalized based on the source. Labeling approach based on sites such as mediabiasfactcheck.com and allsides.com app has been shown promising in recent misinformation detection work (Horne et al. 2018; Baly et al. 2018). The traditional Machine Learning and Neural Network approaches cannot use its reasoning about previous references in the article to decide if there is any bias at later points in the article.RNN's addressed this issue by having networks with loops in them, allowing information to persist. LSTM are a special kind of RNN, capable of learning long-term dependencies.

In this paper, we have tried to train a Long Short Term Memory model that tells us the probability and type of bias in an article.It helps the user make an informed decision of whether he wants to read the article or not. Our model was trained on different news articles scraped from the web. The bias classification labels were assigned based on the classification done by mediabiasfactcheck.com. The articles were predominantly political in nature, from the 2016 US presidential election period.

III. Mathematical Model

Our proposed model has 8 layers.

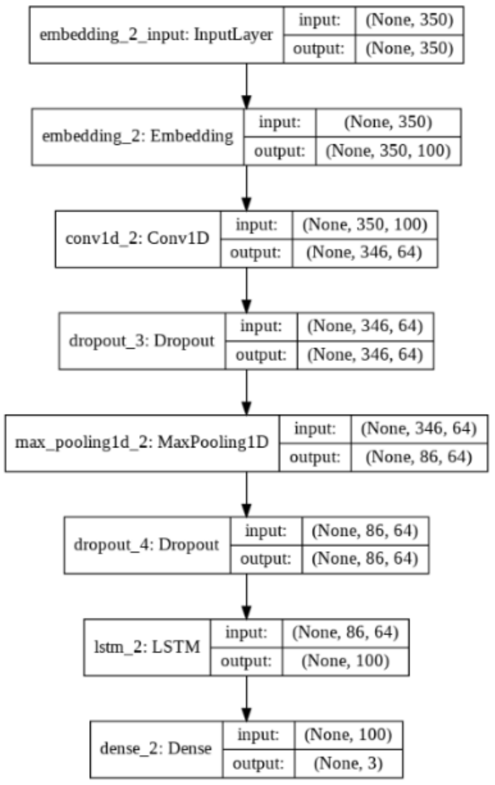


Fig. 1. LSTM Layers

A. Embedding Layer

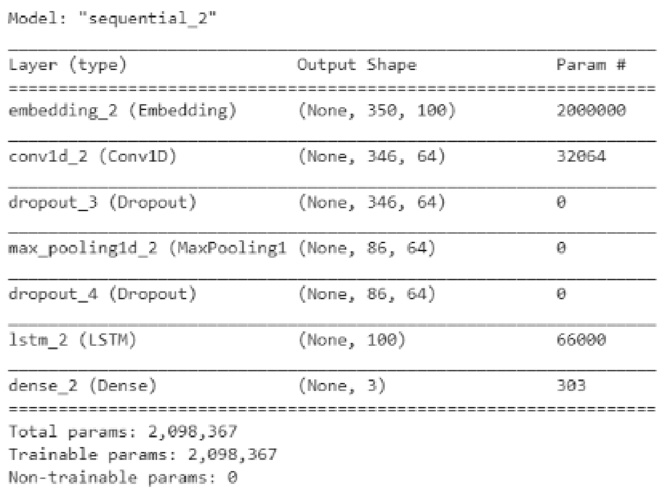
The input data fed to the embedding layer must be integer encoded which is done using the Keras Tokenizer API.This layer is initialized with random weights and will learn an embedding for all of the words in the training data set.The embedding layer turns positive integers (indexes) into dense vectors of fixed size. The input dimension is 20000, size of the vector space in which words will be embedded is 100 and length of input sequences is 350.

B. Convolution 1D Layer and Max Pooling Layer

The next layer is a Conv 1D layer with 64 filters, a kernel size of 5 and tanh activation function.1D CNN helps pick up complex patterns in the text sequence. The dot product between a patch of input sequence and the filter is computed. We define 64 filters to identify 64 different features. The output of the conv 1D network layer is a 346 x 64 neuron matrix. The shape is given by the formula

C. Dropout Layer

Dropout is a regularization method where individual nodes are either dropped out with a probability 1-p or kept with probability p, so that a reduced network is left. If a node is dropped its incoming and outgoing edges are also removed.



```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_2 (Embedding)      (None, 350, 100)          2000000
_________________________________________________________________
conv1d_2 (Conv1D)            (None, 346, 64)           32064
_________________________________________________________________
dropout_3 (Dropout)          (None, 346, 64)           0
_________________________________________________________________
max_pooling1d_2 (MaxPooling1 (None, 86, 64)            0
_________________________________________________________________
dropout_4 (Dropout)          (None, 86, 64)            0
_________________________________________________________________
lstm_2 (LSTM)                (None, 100)               66000
_________________________________________________________________
dense_2 (Dense)              (None, 3)                 303
=================================================================
Total params: 2,098,367
Trainable params: 2,098,367
Non-trainable params: 0
```

Fig. 2. Layer Framework

D. Max Pooling Layer

The MaxPool layer helps to reduce dimensions of the input representation, reducing its dimensionality and allowing assumptions to be made about features contained in the sub-regions binned. The maximum value of the subregion is chosen in this case.

E. LSTM Layer

LSTM are a special kind of RNN, capable of learning long-term dependencies.It has 3 types of gates -Input gate, Output gate and the forget gate. Gates in LSTM are made of sigmoid activation functions where the resulting value is between 0 and 1 where 0 means the data is to be forgotten and 1 means the data should be passed ahead.

$$i_t = \sigma(w_i[h_{t-1}, x_t] + b_i) \tag{1}$$

$$f_t = \sigma(w_f[h_{t-1}, x_t] + b_f) \tag{2}$$

$$o_t = \sigma(w_o[h_{t-1}, x_t] + b_o) \tag{3}$$

The following are the equations of the cell state and the output state.

$$\tilde{c}_t = tanh(w_c[h_{t-1}, x_t] + b_c) \tag{4}$$

$$c_t = f_t * c_{t-1} + i_t * \tilde{c}_{t-1} \tag{5}$$

$$h_t = o_t * tanh(c_t) \tag{6}$$

We have defined our LSTM layer to have 100 output features.The resulting feature matrix is passed to the last layer of the neural network.

F. Dense Layer

This is the last layer of our Neural Network model.A dense layer is one in which each neuron in the current layer receives input from all the neurons in the previous layer.The output space dimension is given as 3, as our aim is to classify the text sequence into 3 classes - Left, Right and Least biased.The softmax activation function is used which converts its input into a discrete probability distribution.The class with maximum probability was chosen as the output class.

$$softmax(x)_i = \frac{exp(x_i)}{\sum_j exp(x_j))} eq \tag{7}$$

The optimizer used for the model was adam. As this is a multi class classification problem the categorical cross entropy loss function was used. Back propagation is carried out at each epoch to update the weights such that loss function is minimized.

IV. FLOWCHART AND ALGORITHM

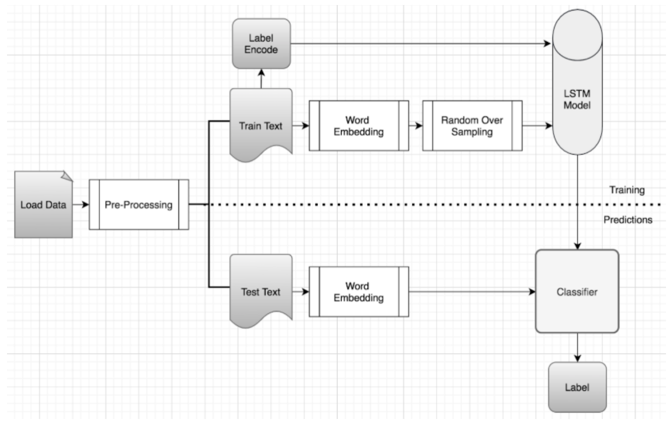


Fig. 3. Flowchart/ Pipeline

A. Pre-processing

Text pre-processing is a very crucial part in Natural Language Processing because the words, sentences, and chars are basic units and parameters which are later on passed to machine learning or deep learning networks/algorithms, where information of the text is retrieved.[3] So it is very important for the basic units to be well pre-processed. Hence, a collection of activities are performed in a pipeline form on the Text Documents.

B. Cleaning Data set

Text and the labels were merged into a single data frame. The missing observations were dropped. The target variable was label encoded into its standard format, where Neutral was 0, Left was 1, Right was 2.

C. Tokenization

Tokenization is the process of separating the text into words or phrases called tokens[3]. The tokens were made into lowercase. For more consistency in the documents, punctuation marks, digits were removed. Contractions such as what's to what is, 've to have, n't to not, I'm to I am, 're to are, ‘d to would, ‘ll to will, were fixed using Regex Operations. Unnecessary spaces, new lines in the tokens and the text was removed. HTML parsers were removed using Beautiful soup.

D. Stop words Removal

There are words in the text which occur frequently, but provide no essential information or knowledge while comparing documents. Stop words are words such as ‘and’, ‘are’, ‘this’, ‘the’, ‘is’ etc.[3] Removing stop words reduces the amount of data and also enhances the system performance.

E. Lemmatization

Lemmatization is the process of processing the inflected forms of a single word into its root form so that the text can maintain consistency, and the inflected words can be considered as a single word. For example, “study” is a root lemma of “studies” and “studying”.

F. Word Embedding

Countvectorizer, bag of words and TF-IDF approaches can also be used to generate numeric feature vectors from text data. These methods however have a very high dimensional feature vector matrix thus increasing the required space and complexity. To avoid this we used the TensorFlow (Keras) Tokenizer class to automatically tokenize the training dataset. Here words are represented as an n-dimensional dense feature vector based on their index. The number of columns is equal to the number of words in the longest input sentence.The rest of the sequences were then padded to a fixed size of 350.

G. Overcoming Class Imbalance

The data set had a class imbalance where the left biased class had 9672, right biased class had 8263, neutral bias class 5199. We need to balance the data with ROS(Random Over Sampling) which can help us improve the classification of left, right, neutral imbalanced data. The ROS method consists of randomly duplicating samples from the minority classes which will increase the number of observations in the minority class till the data set has equal number of samples for each class[4]. After Random Over Sampling, all the classes has 9672 number of samples.

V. EXPERIMENTAL STUDY

A. Data set

The data set for detecting media bias was web scraped articles from a hand-picked subset of sites. These articles were labeled as “right centric”, ”left centric” or ”leasedia bias fact check.com.A subset of sources was used as training data and a different unseen subset was used as a validation set. The data set consisted of the following information- Article title, Article text and source. The labels Left, Right and least biased were encoded as 1, 2 and 0 respectively. The train set consisted of 23134 news articles while the test set had 9336 articles.

B. Metrics for Performance

The metrics used to test model performance were training, testing and validation accuracy, loss, precision, recall and F1 score. The amount of correct classifications out of the total classifications made by the model gives the accuracy score of the model A confusion matrix used to identify mis-classified points. Due to class imbalance Precision, recall and F1 score were a more accurate representation of model performance. Recall is the ratio of correctly predicted articles to all articles in the actual bias class. F1 score is the weighted average of precision and recall ranging from 0 to 1 where 1 is considered the best.

$$precision = \frac{TP}{TP + FN} \tag{8}$$

$$recall = \frac{TP}{TP + FN} \tag{9}$$

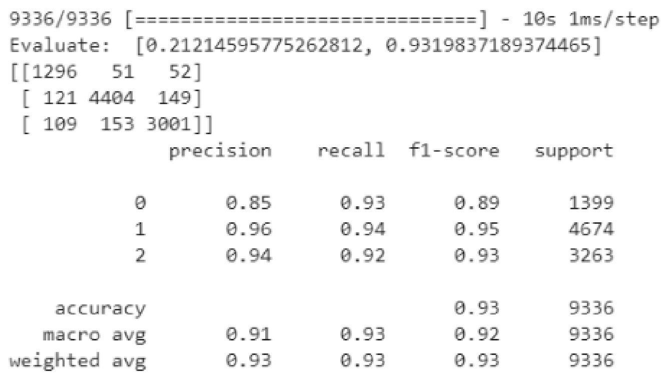
$$F = 2.\,\frac{precision\,.\,recall}{precision + recall} \tag{10}$$

C. Environment

The code was executed on Google Colaboratory a distribution of Python with many scientific libraries pre-installed and GPU enabled. A Python 3.6 development environment was used with numpy 1.12.1, pandas 0.18.1, nltk 3.2.1, scikit-learn 0.18.1 preinstalled.

VI. RESULTS AND DISCUSSION

The above model was run for 3 epochs as it was observed that the model started to over fit once it achieved a training accuracy of 0.96.



```
9336/9336 [==============================] - 10s 1ms/step
Evaluate:  [0.21214595775262812, 0.9319837189374465]
[[1296   51   52]
 [ 121 4404  149]
 [ 109  153 3001]]
              precision    recall  f1-score   support

           0       0.85      0.93      0.89      1399
           1       0.96      0.94      0.95      4674
           2       0.94      0.92      0.93      3263

    accuracy                           0.93      9336
   macro avg       0.91      0.93      0.92      9336
weighted avg       0.93      0.93      0.93      9336
```

Fig. 4. Accuracy Matrix

The LSTM model resulted in a training accuracy of 0.9624 and validation accuracy of 0.9359. The training loss was 0.1078 and the validation loss was 0.2076. When the model was run on an unseen test set it resulted in a training accuracy of 0.93. Left Bias articles were found to have a maximum recall of 0.94 followed by Right bias articles with a recall of 0.93 and least bias articles with a recall of 0.92.Prediction on contemporary news resulted in an accuracy of 0.75.

Class	Recall
Left Bias	0.94
Right Bias	0.93
Least Bias	0.92

Fig. 5. Class Recall for LSTM Model

The only previously implemented model on this data set was the logistic regression model based on lexical n-gram features CLiPS' Google Summer of Code's newsaudit.net [7]. It achieved a recall of 0.82 for Right bias, 0.71 for left bias and 0.33 for least bias.

Class	Recall
Left Bias	0.82
Right Bias	0.71
Least Bias	0.33

Fig. 6. Class Recall for Logistic Regression n-gram model

The new LSTM model showed a drastic improvement in classifying articles with no bias. This might also be due to the Random Over Sampling of the train set before fitting the model. The improvement of recall for the biased classes can be attributed to the LSTM model which maintains a memory state and can thus classify based on different features across the articles considered together.

VII. CONCLUSION

A. Conclusion

Media has a tremendous influence on thoughts, ideas and decisions of people thus it is essential that news media should be fair and accurate. We tackled the problem of news bias detection and also tried to predict the type of bias in the article. Our LSTM model achieved a high training accuracy of 0.96 and validation accuracy of 0.93. Our model was found to accurately predict bias in articles from the same time period as the articles in the training set but when supplied with new contemporary articles its accuracy decreased. Thus we can conclude that LSTM models are an effective way to detect bias using the context of the entire article but having a contemporary data set for training is essential.

B. Future Recommendation

There are many ways in which this model can be improved. Our classifier was trained on data directly labeled by mediabiaschecker.com. Training a model on more contemporary data which has been manually verified or average labels across multiple classification websites might help further increase the accuracy of the model. The model could also be trained to identify bias of articles in different languages and make use of media features in articles for better classification. Deployment of this model as a web browser extension will make it easy to use and help people choose the type of articles they want to read wisely.

References

[1] Predicting Media Bias in Online News CS 229: Machine Learning - Final Project John Merriman Sholar (jmsholar@stanford.edu) Noa Glaser (SuNet ID: noaglasr@stanford.edu) June 6th, 2016

[2] 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

[3] Preprocessing Techniques for Text Mining Dr.S.Kannan, Vairaprakash Gurusamy, Associate Professor, Research Scholar, Department of Computer Applications, Department of Computer Applications, Madurai Kamaraj University. Madurai Kamaraj University.

[4] Johnson, J.M., Khoshgoftaar, T.M. Survey on deep learning with class imbalance. J Big Data 6, 27 (2019). https://doi.org/10.1186/s40537-019-0192-5

[5] Nicholas P. Hirning Andy Chen Shreya Shankar Stanford University Stanford"Detecting and Identifying Bias-Heavy Sentences in News Articles"

[6] Recasens, Marta Jurafsky, Dan. (2013). Linguistic Models for Analyzing and Detecting Biased Language. ACL 2013 - 51st Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference. 1.

[7] CLiPS' Google Summer of Code 2019

[8] Predicting Media Bias in Online News CS 229: Machine Learning - Final Project John Merriman Sholar (jmsholar@stanford.edu) Noa Glaser (SuNet ID: noaglasr@stanford.edu)

[9] Hamborg, Felix Donnay, Karsten Gipp, Bela. (2018). Automated identification of media bias in news articles: an interdisciplinary literature review. International Journal on Digital Libraries. 10.1007/s00799-018-0261-y.

[10] Hochreiter Fakulhmidhuber LONG SHORT-TERM MEMORY Neural Computation