Detect Hyperpartisan News and Bias jointly by Multi-Task Learning

IRE-Team 8 - Project Report

Samudraneel Dasgupta (20171065)
Sajal Asati (20171183)
Amit Jindal (2019201037)
Rakesh Kumar Vemula (2019701027)

Introduction

The problem is, given a News article, we have to report whether it is hyperpartisan or not and if it is then also detect the kind of hyper partisan bias the news article has using multi task learning. The idea is to jointly learn shared embeddings and then use them for multi-task prediction.

Hyperpartisan news is news that takes an extreme left-wing or right-wing standpoint. If one is able to reliably compute this meta information, news articles may be automatically tagged, this way encouraging or discouraging readers to consume the text.

Dataset

Link to dataset : SemEval 2019 Task 4: Hyperpartisan News Detection
The dataset contains 2 types of files:

- Files containing the articles in XML format.
- Files containing the ground truth in XML format. Ground truth files contain information about whether the article is hyperpartisan, and if yes then it contains the article's bias.

There are two files containing the article data.
One file in which the articles are annotated on an article basis. It contains a total of 645 articles 37% (238) are hyperpartisan and 63% (407) are not.
The other part is a larger collection of articles, it is labelled on the basis of the bias of it's publisher. It contains a total of 750,000 articles. Half of total (375,000) are hyperpartisan and half are not. Half of the articles that are hyperpartisan (187,500) are on the left side of the political spectrum, half are on the right side.
This data is again split into a training set (80%, 600,000 articles) and a validation set (20%, 150,000 articles). It has been ensured that there is no intersection of publishers between the training and validation set so that the model does not classify based on publisher bias.

Example of articles in the dataset :

***Title** : Liberals wailing about gun control, but what about abortion?*
***Body** :*
In response to Joyce Newman's recent letter about a conversation about guns: According to the National Right to Life Organization, approximately 600,000 babies are murdered every year by Planned Parenthood with more than 52 million murdered since Roe v. Wade. This makes Planned Parenthood the biggest mass murderer in the history of the world. Is she willing to have a serious conversation about that? Where is her outrage over that? More people die every year from overdoses or auto accidents then from guns. More people die every year from obesity then from guns. Where is her outrage over those issues? The left's obsession with gun "control" is just that, control. It has always been about Democrats wanting to control every aspect of your life. They support Planned Parenthood but go ballistic when a gun is used to kill someone. It's the old game of "don't pay any attention to what's going on over there, but look what's happening here."

The above article is an example of a **hyperpartisan** article (extreme right wing in this case).

***Title** : McMaster: Potential for War With North Korea Grows Daily*
***Body** :*
During a Saturday speech in California, White House national security adviser HR McMaster said the possibility of war with the Hermit Kingdom is "increasing every day, which means that we are in a race, really, we are in a race to be able to solve this problem." And as far as problems go, he believes there are none bigger for the US, and there's "not much time left," per CNN. He called on China to step up to the plate and choke the flow of oil into North Korea, noting, "you can't shoot a missile without fuel." He specified that he and President Trump agree a total oil embargo would "be appropriate at this point." He reiterated the call for China to step up while appearing on Fox News Sunday, but added, "If necessary, the president and the United States will have to take care of it." Sen. Lindsey Graham set his sights on North Korea, too, while appearing on CBS' Face the Nation Sunday, conveying his belief that it's time to pull the families of the 28,500 US troops stationed in South Korea out of the country, reports the AP. Graham, a member of the Senate Armed Services Committee, also noted he would press the Pentagon to refrain from sending dependents to South Korea going forward.

The above article is an example of a **non-hyper partisan** article.

Baseline Model Choices

We considered 2 simple machine learning models to know what is the baseline accuracy/performance we can achieve.

1. Logistic Regression

2. Multinomial Naive Bayes

Following is the performance of both the algorithms for our tasks:

Task : Detect whether article is Hyperpartisan or not **i.e True/False**

For Training Set

Model	Overall Accuracy	Class 0			Class 1		
		Precision	Recall	F1 score	Precision	Recall	F1 score
Logistic Regression	92.77	93.252	92.22	92.73	92.314	93.32	92.81
Multinomial Naive Bayes	82.57	87.21	76.32	81.4	78.96	88.81	83.59

For Validation Set

Model	Overall Accuracy	Class 0			Class 1		
		Precision	Recall	F1 score	Precision	Recall	F1 score
Logistic Regression	91.214	91.53	90.92	91.22	90.97	91.57	91.26
Multinomial Naive Bayes	81.5	85.88	75.44	80.32	78.06	87.57	82.55

For Test Set

Model	Overall Accuracy	Class 0			Class 1		
		Precision	Recall	F1 score	Precision	Recall	F1 score
Logistic Regression	**58.64**	65.93	35.75	46.36	55.92	81.52	66.34
Multinomial Naive Bayes	**57.21**	66.76	28.744	40.18	54.599	85.69	66.7

Task : Detecting what kind of Bias is present:

Multinomial Naive Bayes

Score	Set	Class 0	Class 1	Class 2	Class 3	Class 4
Precision	Training	65.639	54.00	99.13	76.33	99.91
	Validation	65.12	52.82	98.89	76.40	99.66
	Test	36.72	28.73	0	34.21	0
Recall	Training	80.19	88.94	6.35	60.36	3.29
	Validation	79.31	87.72	5.79	59.33	3.54
	Test	27.99	85.84	0	8	0

F1 score	Training	72.19	67.20	11.94	67.41	6.37
	Validation	71.51	65.93	10.94	66.79	6.83
	Test	31.77	43.05	0	12.96	0

Logistic Regression

Score	Set	Class 0	Class 1	Class 2	Class 3	Class 4
Precision	Training	92.51	82.72	82.30	90.35	88.03
	Test	38.75	28.17	26.98	57.26	17.37
Recall	Training	94.14	89.243	78.97	87.75	71.73
	Test	18.58	66.55	15.34	39.22	5.28
F1 score	Training	93.32	85.86	80.60	89.03	79.05
	Test	25.127	39.59	19.56	46.55	8.10

Improvements

- After the baseline models, we decided to try out LSTM. First approach we took was to use the CUDNN LSTM model. We got a training accuracy of 98% but a validation accuracy of only 56% on the hyperpartianship task on this model, clearly signalling overfitting.
- To solve this we needed to add regularisation, however there is no way of adding regularisation on CUDNN LSTM. So we decided to use a simple LSTM model and used dropout to implement regularisation.
- We implemented the LSTM model with a dropout of 0.2% in the beginning. Results we got was a training accuracy of 93% and validation accuracy of 59%, signalling an improvement.
- We then started to tweak with different parameters and also tried out the GloVe word embedding to reach our final model.

Final Model

We have used the LSTM Neural network as our final model because it had better results compared to other models such as logistic regression, naive bayes classifier.

LSTM:

Long Short Term Memory (LSTM) is a type of Recurrent Neural Network. RNN can model a sequence of data so that each sample can be assumed to be dependent on previous ones. But RNN cannot process very long sequences. For example, look at this article

“ ***Virat Kohli*** is an Indian cricketer and the current captain of the India national team. A right-handed top-order batsman, Kohli is regarded as one of the best contemporary batsmen in the world. ***He*** plays for Royal Challengers Bangalore in the Indian Premier League, and has been the team's captain since 2013”

In RNN, each word in form of word embeddings will be given to the network. But, by the time '***He'*** comes RNN considers it as a different token. Actually, ***Virat Kohli*** and ***He*** refers to the same person. This problem we face more often in RNNs. This problem can be solved by using LSTMs.

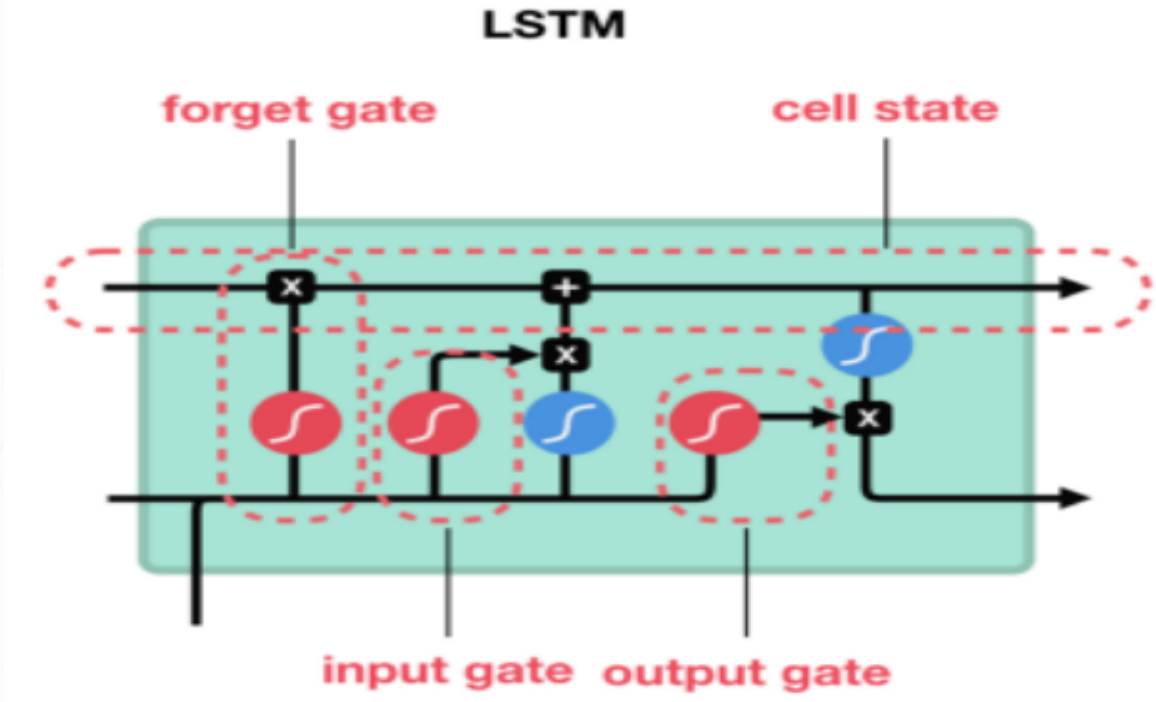
LSTM Architecture:

Cell State:

It has a long term memory usually called as a cell state. The cell state, in theory, can carry relevant information throughout the processing of the sequence. As the cell state goes on its journey, information gets added or removed to the cell state via gates. The gates are different neural networks that decide which information is allowed on the cell state.

Forget Gate:

This gate decides what information should be thrown away or kept. Information from the previous hidden state and information from the current input is passed through the sigmoid function. Values come out between 0 and 1. The closer to 0 means to forget, and the closer to 1 means to keep.



Input Gate:

To update the cell state, we have the input gate. First, we pass the previous hidden state and current input into a sigmoid function. That decides which values will be updated by transforming the values to be between 0 and 1. 0 means not important, and 1 means important. You also pass the hidden state and current input into the tanh function to squish values between -1 and 1 to help regulate the network. Then you multiply the tanh

output with the sigmoid output. The sigmoid output will decide which information is important to keep from the tanh output.

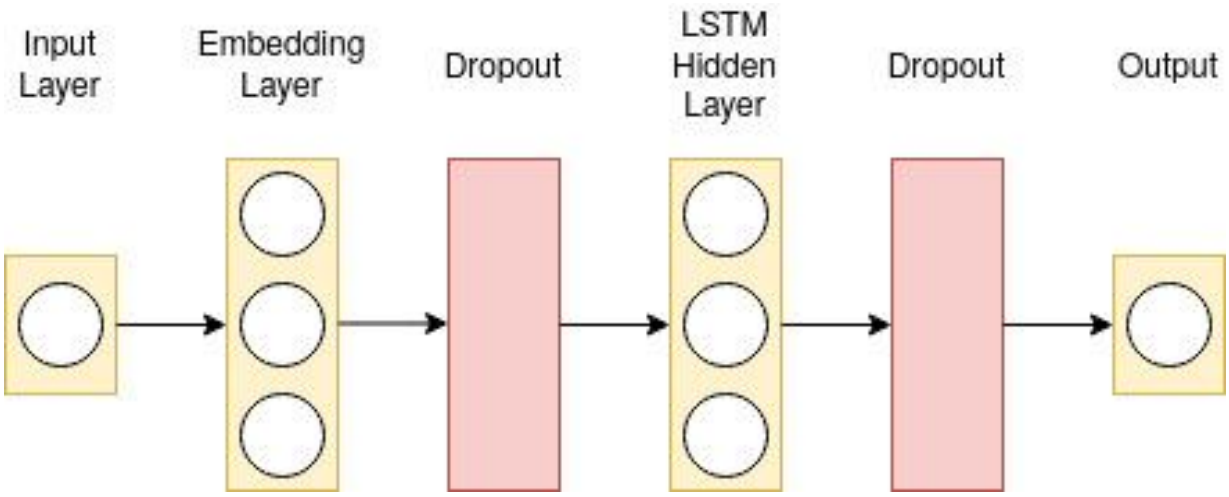
Output Gate:

Lastly we have the output gate. The output gate decides what the next hidden state should be. Remember that the hidden state contains information on previous inputs. The hidden state is also used for predictions. First, we pass the previous hidden state and the current input into a sigmoid function. Then we pass the newly modified cell state to the tanh function. We multiply the tanh output with the sigmoid output to decide what information the hidden state should carry.

Workflow :

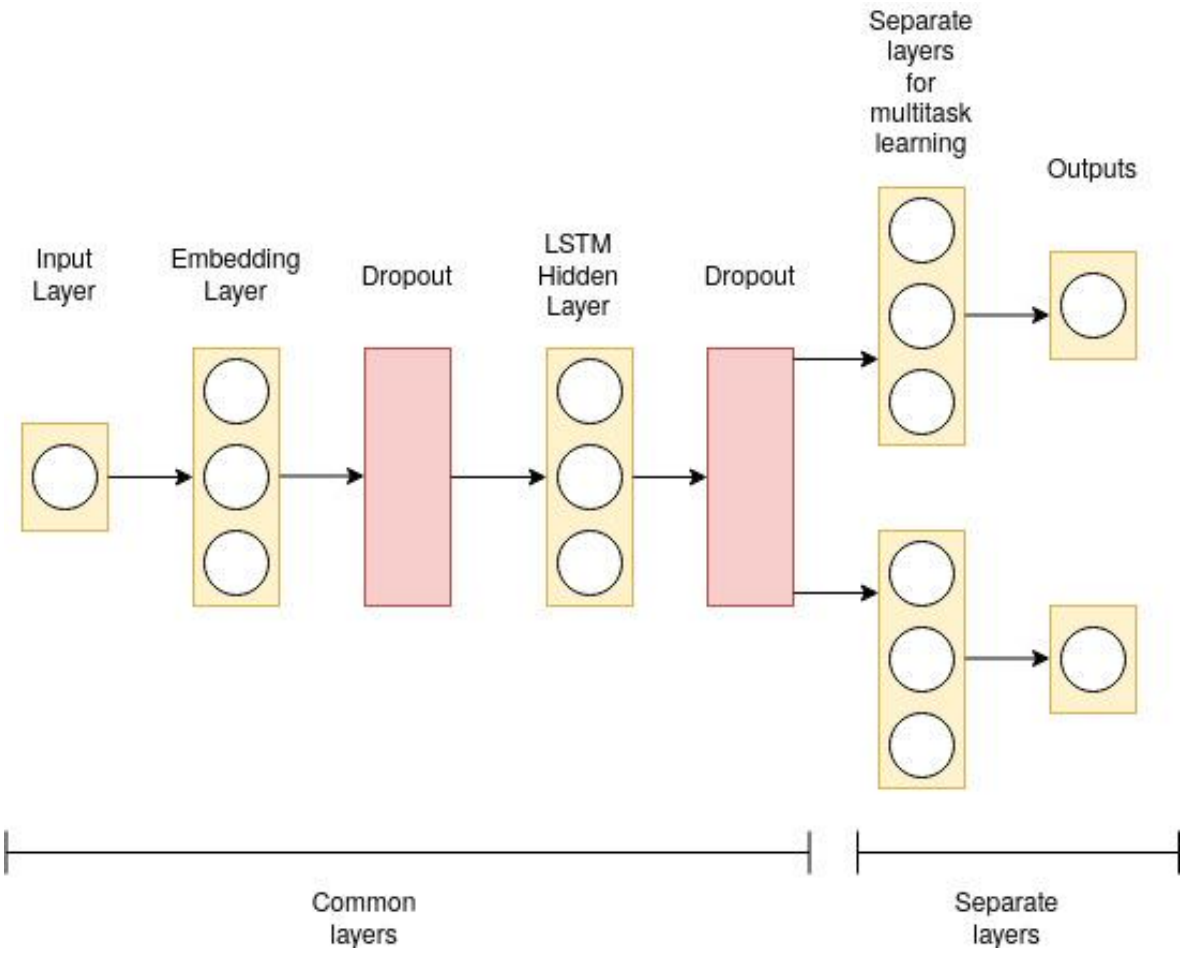
- Firstly we need to convert the article text to vectors. To get the word to vec representation we will convert all text samples in our dataset into sequences of word indices. A word index is simply an integer identifier for that word. We only consider the top 50000 most commonly occurring words in the dataset as part of our vocabulary.
- We have limited the maximum size of each article to 600 words.
- We used word embeddings from pre-trained GloVe. GloVe is an unsupervised learning algorithm for obtaining vector representations for words. Training is performed on aggregated global word-word co-occurrence statistics from a corpus, and the resulting representations showcase interesting linear substructures of the word vector space. It was trained on a dataset of one billion tokens (words) with a vocabulary of 400 thousand words. GloVe is available in embeddings of different vector sizes: 50, 100, 200 and 300 dimensions. We used the 300 dimension embedding.
- After this we will create a weight matrix which is a set of glove vectors for every word in our word index. We will then load this embedding matrix into an Embedding layer. The Embedding layer maps the integer inputs to the vectors found at the corresponding index in the weight matrix.
- Coming on to the LSTM Model, it has an input layer, followed by the embedding layer, then the hidden LSTM layer and finally the output layer. We have also added dropout after the embedding and hidden layer to prevent overfitting on training data. The schematic diagram of the model is given below.

Model without multi-task learning :



Model with multi-task learning :

- We have **2 tasks** at hand - One is to classify whether a news article is hyperpartisan or not, the other is to categorize it's bias (left, left-center, center, right-center, right).
- We can see that both the tasks are closely related, and using multi task learning makes sense because we can benefit from having shared lower-level features to be similar for both in the neural network. So multi-task learning can improve the learning efficiency and also act as a regularizer.
- The difference between the simple LSTM Model and the multitask model is that in addition to the initial shared layers, we have one additional layer at the end which is separate for the two outputs, one is whether the article is hyperpartisan, the other being for detecting bias.
- The schematic diagram is given below.



Final Result (Accuracy on validation set) :

Hyperpartisanship : 81%
Type of Bias : 64%