Detecting Media Bias in the Coverage of U.S. Presidential Elections Using a Hybrid BERT-CNN-BiLSTM Model

Abstract

Recognizing political bias in the media one consumes is essential for being an informed citizen. One common situation where media outlets might lean towards a certain end of the political spectrum is during the elections. This paper aims to contribute to understanding political bias in written media, particularly in the context of pivotal events. We will present both a BERT+LSTM and a BERT+CNN+BiLSTM model for detecting such biases during U.S. presidential elections. These models classify news articles in 3 categories: favoring the Left, favoring the Right, and neutral. Their performance will be evaluated on 3 custom datasets consisting of articles extracted from U.S. newspapers, as well as on the "SemEval-2019 Task 4" Dataset (Kiesel et al., 2019). The performance on the latter dataset will determine how well our models generalize to other types of media biases.

Our **hypothesis** is that combining the BERT (Bidirectional encoder representations from transformers) (presented in Devlin et al., 2019) encoder with a BiLSTM (Bidirectional Long Short-Term Memory) model – using the former's output as input to the latter, and potentially adding CNN layers between them – will create a media bias detector specialized in analyzing the written coverage of the U.S. elections that achieves state-of-the-art performance.

1 Definitions

1.1 Types of Biases

Political bias is a bias involving the slanting or altering of information to make a political position seem more attractive.

Other types of bias include cognitive bias, social bias, economic bias, cultural bias, and gender bias. Some of them have partisan purpose (racial bias, political bias), whereas others do not (economic bias).

1.2 Particularities of political bias

Political bias has some particularities that make creating a model specifically for this type of bias appealing. For example, Sánchez-Junquera (2021) claims it is different than social bias, arguing that political bias detection involves detecting emotional tones (e.g.: left-wing-biased texts refer to right-wing leaders in a negative way), while social bias is more subtle and requires a deeper understanding of the context (e.g.: bias towards immigrants is affected by whether texts on immigrants contains the phrases "problem," "unemployment," or "people," "we need to give a soluton").

2 Modeling the Experiments

2.1 Datasets

Lazaridou et al., 2016 builds a dataset consisting of political articles about the world and the UK using Guardian and The Telegraph articles. Similarily, we will extract articles from American newspapers to build two datasets with articles about the 2012, 2016, and 2020 elections. Additionally, we will build and use an extended version of the NewB (introduced in Wei, 2020) dataset.

2.1.1 Allsides.com

We will use the chart on Allsides.com to determine the political orientation of a newspaper. It organizes newspapers into 5 categories: left, lean left, center, lean right, and right.

2.1.2 Lean-left and lean-right newspapers dataset

The first dataset will use articles from lean-left and lean-right newspapers. We will extract sentences about all aspects of the presidential race. We will consider the sentences to be slightly biased towards the newspaper's orientation. We will use American newspapers, both Democrat- (e.g. New York Times) and Republican-leaning (e.g. Wall Street Journal). By training on this dataset, we increase the models' ability to recognize subtler biases. During the test stage, we expect weaker results on this dataset, compared to the left and right dataset described below, due to the aforementioned subtleties.

2.1.3 Left and right newspapers dataset

We will build a similar dataset using articles from left and right newspapers. We can say with a higher confidence that the sentences extracted from these have a certain bias.

2.1.4 Extended NewB

NewB (Wei, 2020): collection of over 200,000 sentences about Republican (right-wing) politician Donald Trump extracted from 9 mainstream newspapers. The sentences are labeled by source. We expect the sentences to lean in the same direction on the political spectrum as the newspapers they originated from. We will extend it by adding sentences about Clinton, Biden, or Obama extracted from the same 9 newspapers.

2.2 Models

2.2.1 Parameters for BERT

For all three models, we will subsequentially try the parameters that the authors of BERT recommend in Devlin et al., 2019:

a) *Batch size*: 16 and 32
b) *Learning rate*: 5e-5, 3e-5, 2e-5
c) *Number of epochs*: 2, 3, and 4

2.2.2 BERT plus a classification output layer

This will be the simplest model we will try. Devlin et al., 2019 claims that it is enough to add an additional output layer to BERT to create well-performing models. We will add a dense output layer that predicts the class choosing from 3 options: "left," "neutral," and "right."

2.2.3 BERT together with a LSTM

BERT will take the raw text as input and generate contextual embeddings. BERT's embeddings will be fed into a LSTM. LSTMs capture context over longer sequences, as they incorporate elements called "gates" specifically for this purpose (Siami-Namini et al., 2019).

We will feed their output into a dense classification layer that will choose from the three output classes.

2.2.4 BERT, CNN, and BiLSTM

BiLSTM is an upgrade to LSTM. As opposed to LSTMs, BiLSTMs look both in the future and in the past, while LSTMs only exploit historical context (Liu et al., 2019). By combining BERT and a BiLSTM, we hope to obtain a model whose understanding of the input is better than those of the individual models.

The raw input text will go through BERT. Then will follow several CNN layers. Using this approach, Kaur et al., 2023 obtained a classification model that “performs better than the state-of-the-art baseline approach.” The output of the CNN layers will be taken as input by the BiLSTM. In the end, a dense classification layer will choose the output class.

2.3 Experiments

We will train each model on all three datasets. Based on the metrics obtained, we will adjust their hyperparameters (e.g.: number of epochs, learning rate) to improve each model’s performance. After optimizing the models, we will compare the best-performing one with state-of-the-art political bias detection models.

2.4 Evaluation Metrics

We will use the following metrics to compare the performance of the models: *accuracy, precision*, *recall,* and *macro F1* (suggested by Baly et al., 2020).

2.5 Comparisons with Other Approaches

As we mentioned above, we will compare our best-performing model with state-of-the-art political bias detection models. These state-of-the-art models include: the Transformers-based Dbias model (Raza et al., 2022) and the LSTM-based model proposed in Baly et al., 2020.

We will compare how they perform both on our custom datasets, and on datasets that are widely used, such as the “SemEval-2019 Task 4” Dataset (Kiesel et al., 2019).

Note that the performance on some datasets will determine how well our models generalize to political bias in general, as the most popular datasets for political bias do not specifically focus on U.S. elections.

In some cases, we might need to adjust our model to compare it with others. For example, the Dbias model proposed in Raza et al., 2022 has a binary output (biased or not). We can transform our 3-class output into a binary one by considering “left” and “right” to be biased, and “neutral” to be unbiased.

2.6 Mathematical Model

We present a mathematical model for our most complex network (BERT+CNN+BiLSTM), as the others are just simplified versions of it.
Let X be the input text. First, it is passed through the pretrained BERT.

$$H = \mathrm{BERT}(X)$$

$$F_k = \mathrm{ReLU}(\mathrm{Conv}(H, C_k)), \quad k \in \{2, 3, 5\}$$

Then we apply three convolutions with different kernel sizes (2, 3, and 5).

$$C = [F_2; F_3; F_5]$$

Afterwards, we concatenate the results of the convolutions and pass them through the BiLSTM, whose output is then passed through the fully connected layer. In the end, as we said, we apply softmax.

$$H_{\mathrm{BiLSTM}} = \mathrm{BiLSTM}(C)$$

$$Z = W \cdot H_{\mathrm{BiLSTM}} + b$$

We use the cross-entropy loss function. Letting *y=the true probability distribution* (a value of 1 and the others 0) and *Y=the predicted probability distribution*, this function has the following formula:

$$\mathcal{L} = -\sum_{i=1}^{N} y_i \log(Y_i)$$

3 Case Study

3.1 Creation of the Left and Right Newspapers Dataset

We have so far only created and used the left and right dataset. We have used neither the lean-left and lean-right dataset, nor the Extended NewB dataset yet.

The end goal of the data gathering was to obtain a CSV file with three columns: *Text, Year* (of the elections), and *Leaning* (“left,” “right,” or “center”).

We collected data from the 2012, the 2016, and the 2020 elections. We skipped the 2024 one because one of the parties changed candidates late in the race.

For each of the 3 leanings, we used *allsides.com* to choose a representative newspaper. We chose Jacobin (left), AP News (central) and Newsmax (right) due to them being free to access and having an extensive archive.

We chose between 2 and 6 search terms for each election. For example, for 2012 (Obama vs Romney) we chose “barack obama 2012 elections,” “barack obama rally 2012,” “mitt romney 2012 elections,” and “mitt romney presidential rally.” Then, we searched articles containing these phrases using the web scrapping Python library BeautifulSoup. To get articles from Newsmax, we used Google, as Newsmax’s search engine was not of satisfactory quality. Once we got an article, we searched for the names of the candidates inside the article and extracted fragments of 2-4 sentences around that occurrence of their names. Finally, we appended to the dataset an entry that has that 2-to-4-sentence-long fragment as *Text*, and the leaning of the newspaper (as per *allsides.com*) as *Leaning*.

In the end, we obtained a dataset with 948 entries: 316 for each leaning.

3.2 Implementation of the Models

3.2.1 Technologies used

To build, train and evaluate the three models, we used Google Colab notebooks with T4 GPUs. We used machine learning Python libraries such as Torch, SkLearn, and data manipulation libraries such as Pandas and Numpy. To load the pretrained BERT models we used the Transformers library made available by Hugging Face.

3.2.2 Implementation Details

We are first loading the data from the CSV file and splitting it into train and evaluation datasets. Then, we set hyperparameters such as learning rate, the number of epochs, and the batch size. For each of our 3 architectures, we tried many combinations of these hyperparameters, as described above. Additionally, in the two more complex architectures we used the Adam optimizer, the cross-entropy loss function (recommended for classification tasks), and we trained the model sequentially on chunks of data to avoid memory issues.

For the simple BERT model, we do not have much control over the architecture.

For the BERT+LSTM model, we had a BERT layer similar to the simple model followed by a LSTM layer with hidden size 128. Finally, we had a fully connected layer with 128 input and 3 output neurons.

For the BERT+CNN+BiLSTM model, we added three CNN layers between the BERT encoder and the LSTM layer. We apply each CNN layer in parallel, then concatenate them. We also made the LSTM bidirectional. We set the hidden size of the BiLSTM layer to 256, and that of the CNN layers to 128. The CNN layers have different kernel sizes (2, 3, and 5), in order to capture varied textual patterns. To prevent overfitting, we added dropout with a 0.5 rate. Finally, a softmax layer is applied to make the 3 output values sum up to 1, as if they were probabilities. For the classification result, the model chooses the leaning with the highest probability.

3.3 Results

3.3.1 Metrics

As stated above, we measured these metrics: *accuracy, precision, recall,* and *macro F1*.

We will show the metrics of our models. In another subchapter, we will compare our model with state-of-the-art models.

model / metric	**Accuracy**	**Precision**	**Recall**	**Macro F1**
Simple BERT (2 epochs, batch size=8, lr=5e-5)	0.71	0.73	0.71	0.71
Simple BERT (3 epochs, batch size=8, lr=3e-5)	0.74	0.77	0.74	0.73
BERT+LSTM (3 epochs, batch	0.78	0.83	0.80	0.77

size=16, lr=5e-5)				
BERT+LSTM (4 epochs, batch size=32, lr=5e-5)	0.87	0.89	0.86	0.87
BERT+CNN+BiLSTM (4 epochs, batch size=16, lr=5e-5)	0.87	0.90	0.87	0.87
BERT+CNN+BiLSTM (4 epochs, batch size=16, lr=3e-5)	0.90	0.90	0.90	0.90

Note: “lr” stands for “learning rate.”

We can see that, when trained for 3 epochs, the simple BERT model already achieves a decent accuracy (74%). When trained for 2 epochs, it achieves an accuracy that’s 3 percent lower. The BERT+LSTM model achieves a greater increase in accuracy when its number of epochs is increased: from 78% when trained for 3 epochs to 87% for 4 epochs. We see that the second architecture’s best-performing model achieves higher numbers in all metrics than the first architecture’s best-performing model. Furthermore, the BERT+CNN+BiLSTM model with batch size=16 and learning rate=5e-5 achieves slightly higher numbers than the BERT+LSTM model with the same hyperparameters.

Therefore, overall, the third – and most complex – architecture is the best-performing one, whichever metrics we make this judgment upon. The BERT+LSTM architecture is coming close. Both architectures highly improve upon the simple BERT model.

3.3.2 Inference Examples

Let us look at how our best model (BERT+CNN+BiLSTM trained for 4 epochs with batch size=16 and learning rate=3e-5) performs on some short input texts.

Visibly biased sentences are usually classified correctly: “Trump is the best” is right-leaning, while “Democrats are champions of equality and social justice.” is left-leaning. The model also correctly classifies sentences that are more subtle: “It's best to vote for the Republican candidate, as he will conserve America's values.” (right), “Progressive policies are the way forward for America.” (left). However, it does not obtain the correct answer for this neutral sentence: “Neither Democrats nor Republicans have all the answers.” Note that the BERT+LSTM model correctly classifies it as neutral.

3.3.3 Comparison with DBias

One of the state-of-the-art models that we should compare our models with is DBias (Raza et al., 2022). This Transformer-based model has been downloaded 70,000 times on pepy.tech. It is not able to classify texts as “left” or “right” leaning. Instead, it offers a binary output (biased or unbiased). In order to compare our model with it, we will map our model's "left" and "right" output values to "biased," and "center" to "unbiased".

On our over-900-article-long left and right newspapers dataset, DBias obtains 74% accuracy, opposed to the 90% accuracy of our model. Note that DBias's performance is similar to what we obtained with the simple BERT model.

To best compare our model with DBias, we should also use datasets that are widely used in political bias detection, such as the SemEval-2019 Task 4 dataset, or on datasets used by Raza et al., like the MBIC dataset they created. As mentioned previously, testing on these broader datasets will also tell us how well our models generalize to political or media bias in general.

4 Related Work

4.1 Survey of Other Approaches

4.1.1 Text classification using architectures similar to ours

Taken individually, BERT achieves state-of-the-art results on many text classification datasets, while LSTMs and CNNs achieve state-of-the-art results in the branch of sentiment analysis (Otter et al., 2019).

Additionally, BERT+CNN+BiLSTM architectures have been used for several classification tasks. However, no significant studies have been conducted on how they can be used to detect political bias in the media. Hence, the present paper takes **an approach that has not been used before** for this natural language processing application.

Notable usages of BERT+CNN+BiLSTM architectures include detecting fake news (Alghamdi et al., 2022) and performing sentiment analysis on movie reviews (He et al., 2024 and Gupta et al., 2022).

4.1.2 Political bias or ideology detection

To the author's knowledge, as of November 1, 2024, no studies have been conducted on building a neural network-based model specifically designed to detect bias in written-text media coverage of U.S. presidential elections.

There have been a considerable number of studies on detecting either political bias in written media or political ideology of certain people, without focusing on the U.S. elections. Their approaches vary:

a) *Bayesian classifier* (Fang et al., 2015) or *Bayesian Hidden Markov Models* (Sim et al., 2013)

b) *statistical models such as support vector machines* (Lin et al., 2006)

c) *recursive neural networks* (Iyyer et al., 2014)

d) *fine-tuning Transformers models, such as BERT, on custom datasets* (Raza et al., 2022)

e) *LSTMs* (Baly et al., 2020)

Some of the papers enumerated above use classic methods (Fang et al., 2015, Sim et al., 2013, Lin et al., 2006), without creating a large neural network. This is a very different approach from ours. The latest state-of-the-art models in political bias detection are based on neural networks. The models that are most **similar** to ours are the Transformers-based model proposed by Raza et al., and the LSTM model proposed by Baly et al.

The main **difference** between our models and the Transformer models proposed by Raza et al. is that our models have CNN and LSTM layers following BERT. We expect this to make our models perform better, as we expect BERT's encodings to benefit from the feature extraction provided by CNN and LSTM networks.

Baly et al. create two separate models: one using BERT, and the other using LSTM. They do not combine them into a single model, nor do they add CNN layers.

4.2 Areas Expected to Improve Upon

Due to our models focusing on U.S. presidential elections, we expect them to obtain better results on data about these particular events. We expect to perform better both when doing a 3-class classification (left/right/center) and when doing a binary classification (biased/unbiased). We expect to improve upon all the metrics we test (accuracy, precision, recall, macro F1).

However, for detecting political bias in general, without focusing on U.S. elections, we expect state-of-the-art models to surpass our performance. Our models might achieve satisfactory results on this task too, if they generalize well.

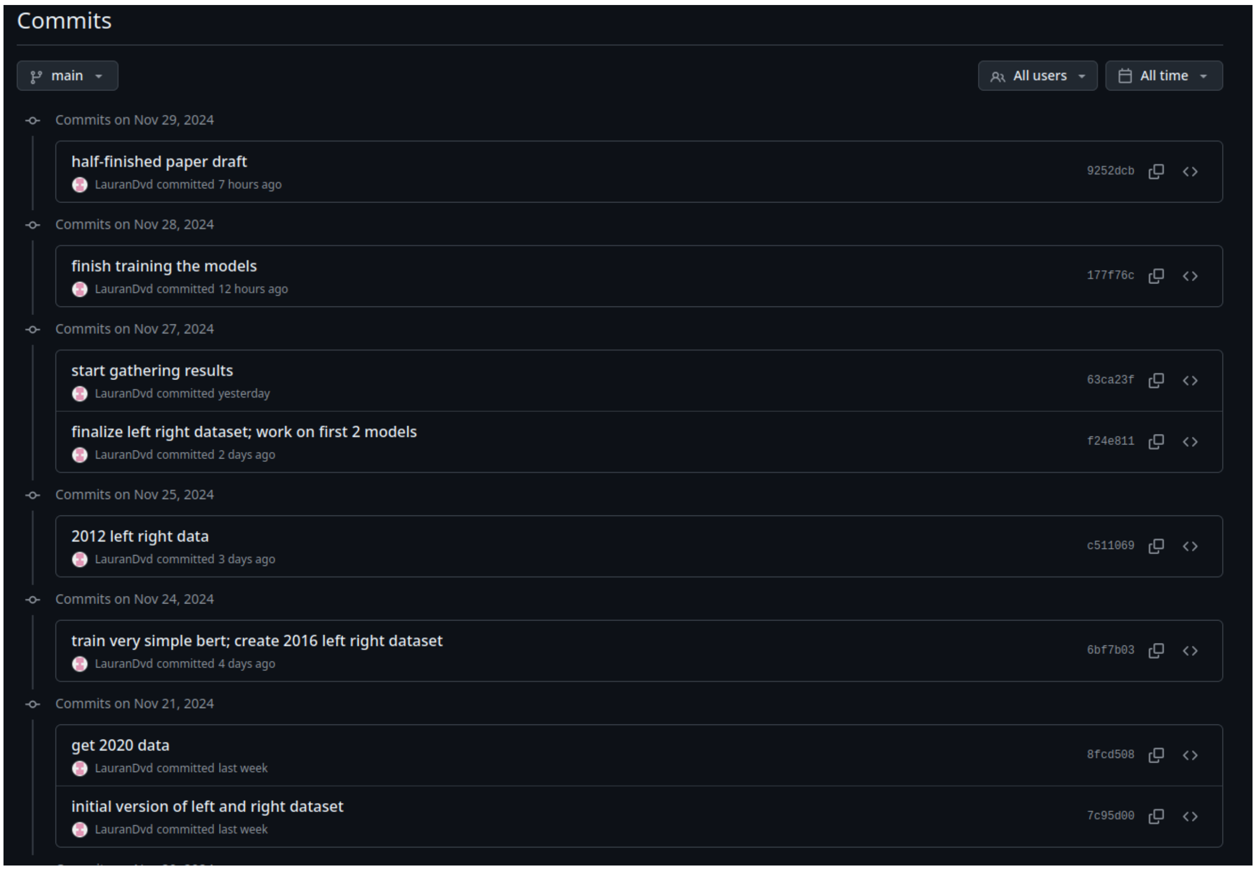
4.3 Benchmark Datasets to Use

We can best compare our models with others by using datasets that are highly used in the literature, like the "SemEval-2019 Task 4: Hyperpartisan News Detection" dataset (Kiesel et al., 2019). Additionally, for comparisons with specific state-of-the-art models that were trained on custom datasets, we will evaluate our models on their datasets too. An example of such a dataset is the over-34,000-article-long dataset used by Baly et al. (2020) for their LSTM model.

5 Source Code Repository

We used Git and GitHub to manage our work (Python notebooks, text notes and the paper). Please find the current history in the pictures below. Note that the final paper, and some text notes, had not been uploaded before the screen captures were taken.

Link: https://github.com/914LauranDavid/research_lab_5_8/



Commits

Commits on Nov 29, 2024

half-finished paper draft
LauranDvd committed 7 hours ago 9252dcb

Commits on Nov 28, 2024

finish training the models
LauranDvd committed 12 hours ago 177f76c

Commits on Nov 27, 2024

start gathering results
LauranDvd committed yesterday 63ca23f

finalize left right dataset; work on first 2 models
LauranDvd committed 2 days ago f24e811

Commits on Nov 25, 2024

2012 left right data
LauranDvd committed 3 days ago c511069

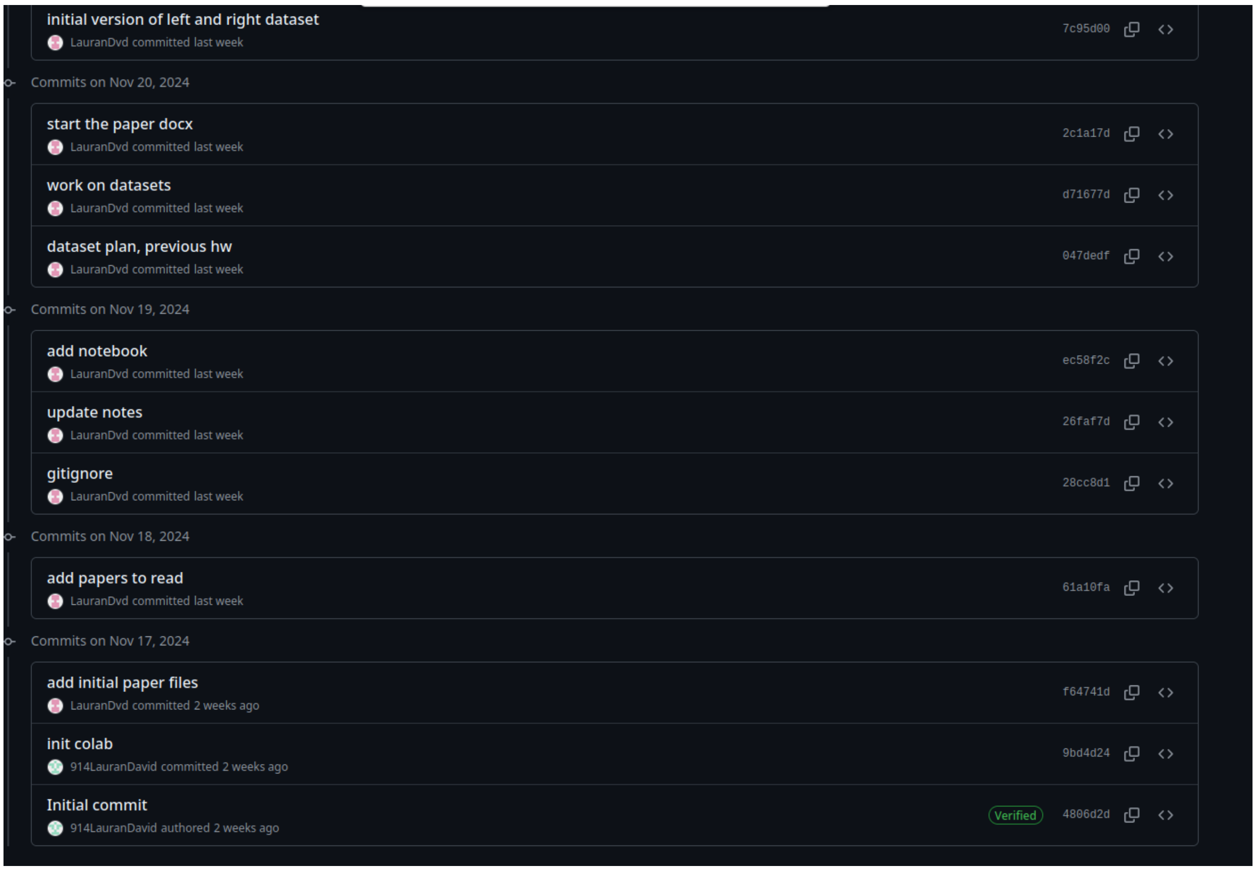
Commits on Nov 24, 2024

train very simple bert; create 2016 left right dataset
LauranDvd committed 4 days ago 6bf7b03

Commits on Nov 21, 2024

get 2020 data
LauranDvd committed last week 8fcd508

initial version of left and right dataset
LauranDvd committed last week 7c95d00


initial version of left and right dataset
LauranDvd committed last week 7c95d00

Commits on Nov 20, 2024

start the paper docx
LauranDvd committed last week 2c1a17d

work on datasets
LauranDvd committed last week d71677d

dataset plan, previous hw
LauranDvd committed last week 047dedf

Commits on Nov 19, 2024

add notebook
LauranDvd committed last week ec58f2c

update notes
LauranDvd committed last week 26faf7d

gitignore
LauranDvd committed last week 28cc8d1

Commits on Nov 18, 2024

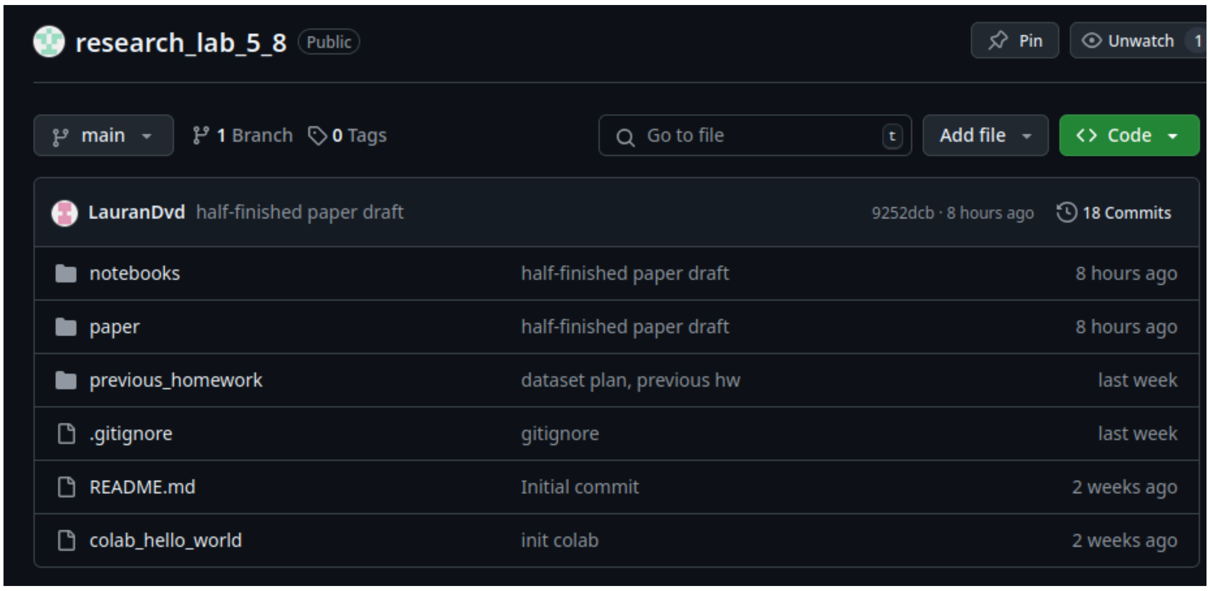
add papers to read
LauranDvd committed last week 61a10fa

Commits on Nov 17, 2024

add initial paper files
LauranDvd committed 2 weeks ago f64741d

init colab
914LauranDavid committed 2 weeks ago 9bd4d24

Initial commit
914LauranDavid authored 2 weeks ago Verified 4806d2d


research_lab_5_8 Public

Pin · Unwatch 1

main · 1 Branch · 0 Tags · Go to file · Add file · Code

LauranDvd half-finished paper draft 9252dcb · 8 hours ago 18 Commits

notebooks	half-finished paper draft	8 hours ago
paper	half-finished paper draft	8 hours ago
previous_homework	dataset plan, previous hw	last week
.gitignore	gitignore	last week
README.md	Initial commit	2 weeks ago
colab_hello_world	init colab	2 weeks ago

References

1. Johannes Kiesel, Maria Mestre, Rishabh Shukla, Emmanuel Vincent, Payam Adineh, David Corney, Benno Stein and Martin Potthast. 2019. SemEval-2019 Task 4: Hyperpartisan News Detection. In Proceedings of the 13th International Workshop on Semantic Evaluation, pages 829–839.
2. Daniel W. Otter, Julian R. Medina, and Jugal K. Kalita. 2019. A Survey of the Usages of Deep Learning for Natural Language Processing. In IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, VOL. XX, NO. X, JULY 2019.
3. Alghamdi Jawaher, Lin Yuqing and Luo Suhuai. 2022. Modeling Fake News Detection Using BERT-CNN-BiLSTM Architecture. 2022 IEEE 5th International Conference on Multimedia Information Processing and Retrieval (MIPR).
4. Aixiang He and Mideth Abisado. 2024. Text Sentiment Analysis of Douban Film Short Comments Based on BERT-CNN-BiLSTM-Att Model. In IEEE Access 12 (2024), pages 45229-45237.
5. Bhart Gupta, P. Prakasam and T. Velmurugan. 2022. Integrated BERT embeddings, BiLSTM-BiGRU and 1-D CNN model for binary sentiment classification analysis of movie reviews. In Multimedia Tools and Applications, volume 81, issue 23, pages 33067 – 33086.
6. Jacob Devlin, Ming-Wei Chang, Kenton Lee and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 4171–4186.
7. Anjie Fang, Iadh Ounis, Philip Habel, Craig Macdonald and Nut Limsopatham. 2015. Topic-centric Classification of Twitter User's Political Orientation. In SIGIR '15: Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 791 – 794.
8. Yanchuan Sim, Brice D. L. Acree, Justin H. Gross and Noah A. Smith. 2013. Measuring Ideological Proportions in Political Speeches. In Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, pages 91–101.
9. Wei-Hao Lin, Theresa Wilson, Janyce Wiebe and Alexander Hauptmann. 2006. Which Side are You on? Identifying Perspectives at the Document and Sentence Levels. In Proceedings of the Tenth Conference on Computational Natural Language Learning (CoNLL-X), pages 109–116.
10. Mohit Iyyer, Peter Enns, Jordan Boyd-Graber and Philip Resnik. 2014. Political Ideology Detection Using Recursive Neural Networks. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1113–1122.
11. S. Raza, D.J. Reji and C. Ding. 2022. Dbias: detecting biases and ensuring fairness in news articles. In Int J Data Sci Anal 17, pages 39–59.
12. Ramy Baly, Giovanni Da San Martino, James Glass and Preslav Nakov. 2020. We Can Detect Your Bias: Predicting the Political Ideology of News Articles. In

Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 4982–4991.

13. S. Siami-Namini, N. Tavakoli and A. S. Namin. 2019. The Performance of LSTM and BiLSTM in Forecasting Time Series. In 2019 IEEE International Conference on Big Data (Big Data), pages 3285-3292.
14. Gang Liu and Jiabao Guo. 2019. Bidirectional LSTM with attention mechanism and convolutional layer for text classification. In Neurocomputing, volume 337, pages 325-338.
15. Kamaljit Kaur and Parminder Kaur. 2023. BERT-CNN: Improving BERT for Requirements Classification using CNN. In Procedia Computer Science, volume 218, pages 2604-2611.
16. Konstantina Lazaridou and Ralf Krestel. 2016. Identifying Political Bias in News Articles. In Bull. IEEE Tech. Comm. Digit. Libr., volume 12, n. pag.
17. Jerry Wei. 2020. NewB: 200,000+ Sentences for Political Bias Detection. In ArXiv abs/2006.03051, n. pag.
18. Javier Sánchez-Junquera. 2021. On the Detection of Political and Social Bias. In octoral Symposium on Natural Language Processing from the PLN.net network 2021 (RED2018-102418-T), 19-20 October 2021, Baeza (Jaén), Spain.