Advanced NLP-monsoon'20

Bias detection in news articles

TEAM 10: CORUSCANT

Sahil Bhatt Srinath Nair

Problem statement

The goal of this task is to detect bias in news articles. It is a binary classification problem that outputs whether or not the content in the news is biased. The biased or hyper-partisan reporting of news is done in a way that strongly favors one position (mostly political) and would be in fierce disagreement with the opponents. Hyperpartisan news reporting often involves either stretching the truth or breaking it with fake news and are often spread quickly due to its highly sensational content. The task is to detect this hyperpartisan language in news articles.

Literature review

Martin Potthast et al. (2017) used a corpus of 1,627 fact checked articles containing both hyperpartisan news from the left-wing and the right-wing, and mainstream publishers. The work tried to find the similarities in the writing style of the left and right wing publishers. The study revealed that the writings of left-wing and right-wing reporters have a lot more in common than any of the two have with the mainstream. Furthermore, they showed that hyperpartisan news can be distinguished well by its style from the mainstream (F1 = 0.78).

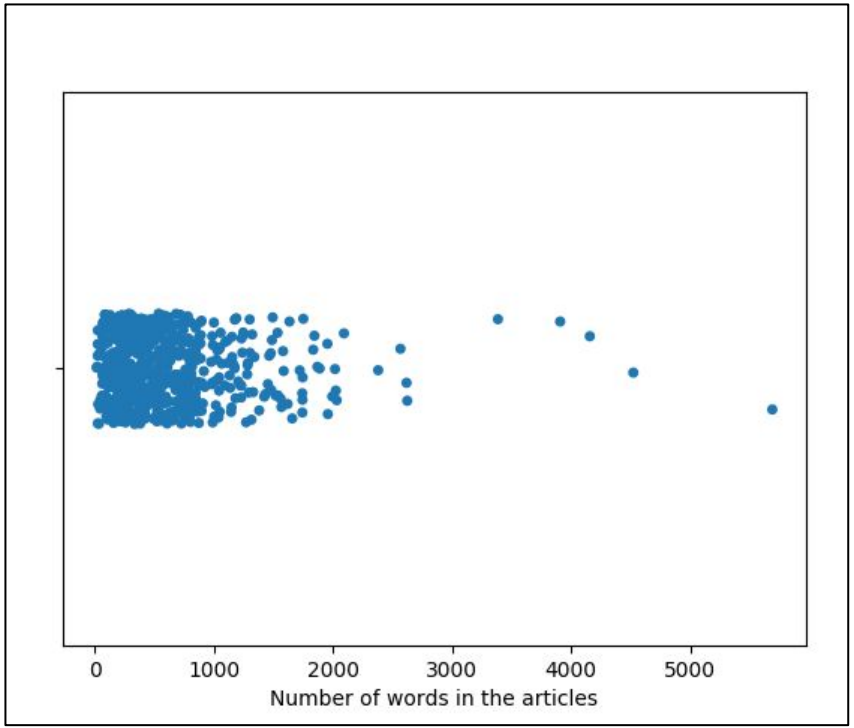
Team Bertha-von-Suttner in the SemEval 2019 task 4 Hyperpartisan News Detection task used sentence representations from averaged word embeddings generated from the pre-trained ELMo model with Convolutional Neural Networks and Batch Normalization for predicting hyperpartisan news. The final predictions were generated from the averaged predictions of an ensemble of models. With this architecture, their system ranked first place, based on accuracy, the official scoring metric.

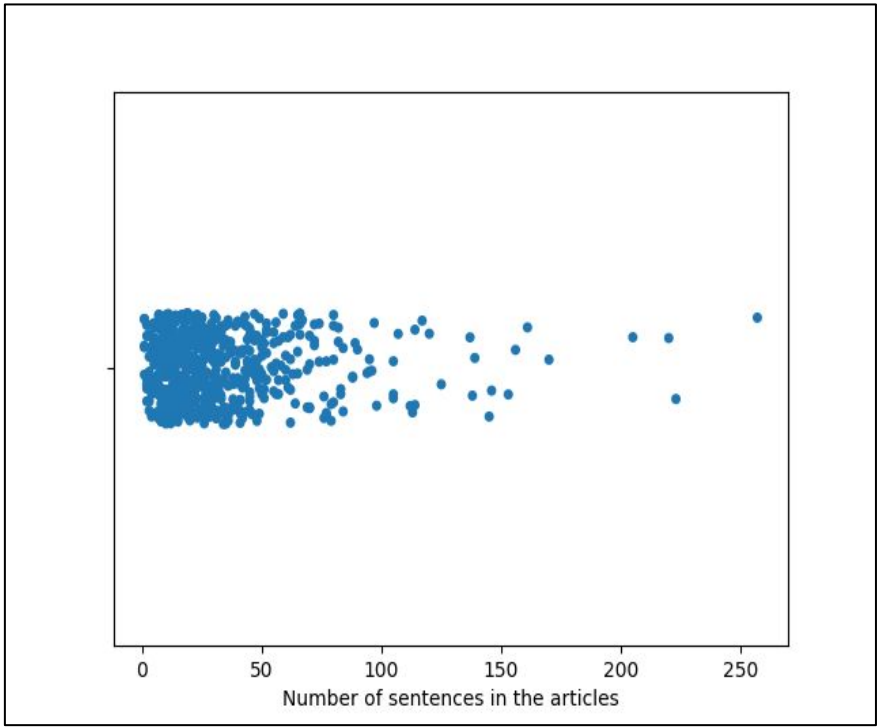
Data processing

The dataset we used was the publicly available dataset of the shared task Semeval task-4. The dataset contains 645 articles. We split the dataset into train and test set by a 80:20 ratio.

The XML files were processed into tsv files which were used to generate the ELMo embeddings and s-BERT embeddings.

About the dataset



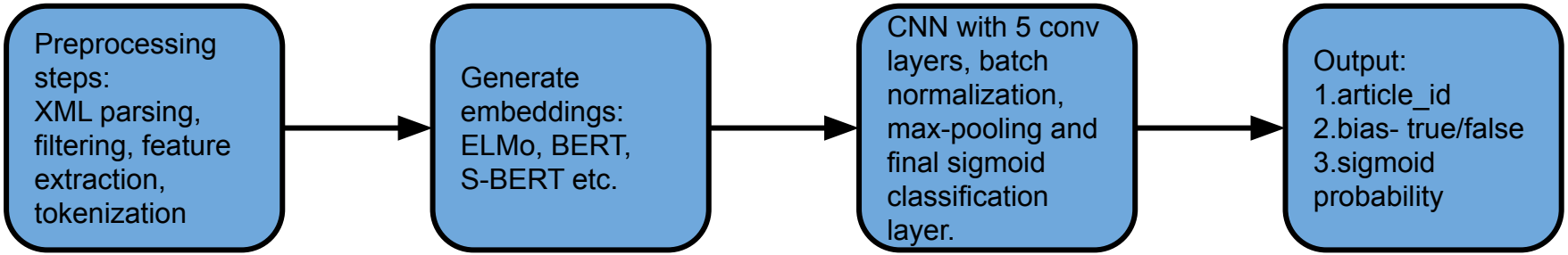


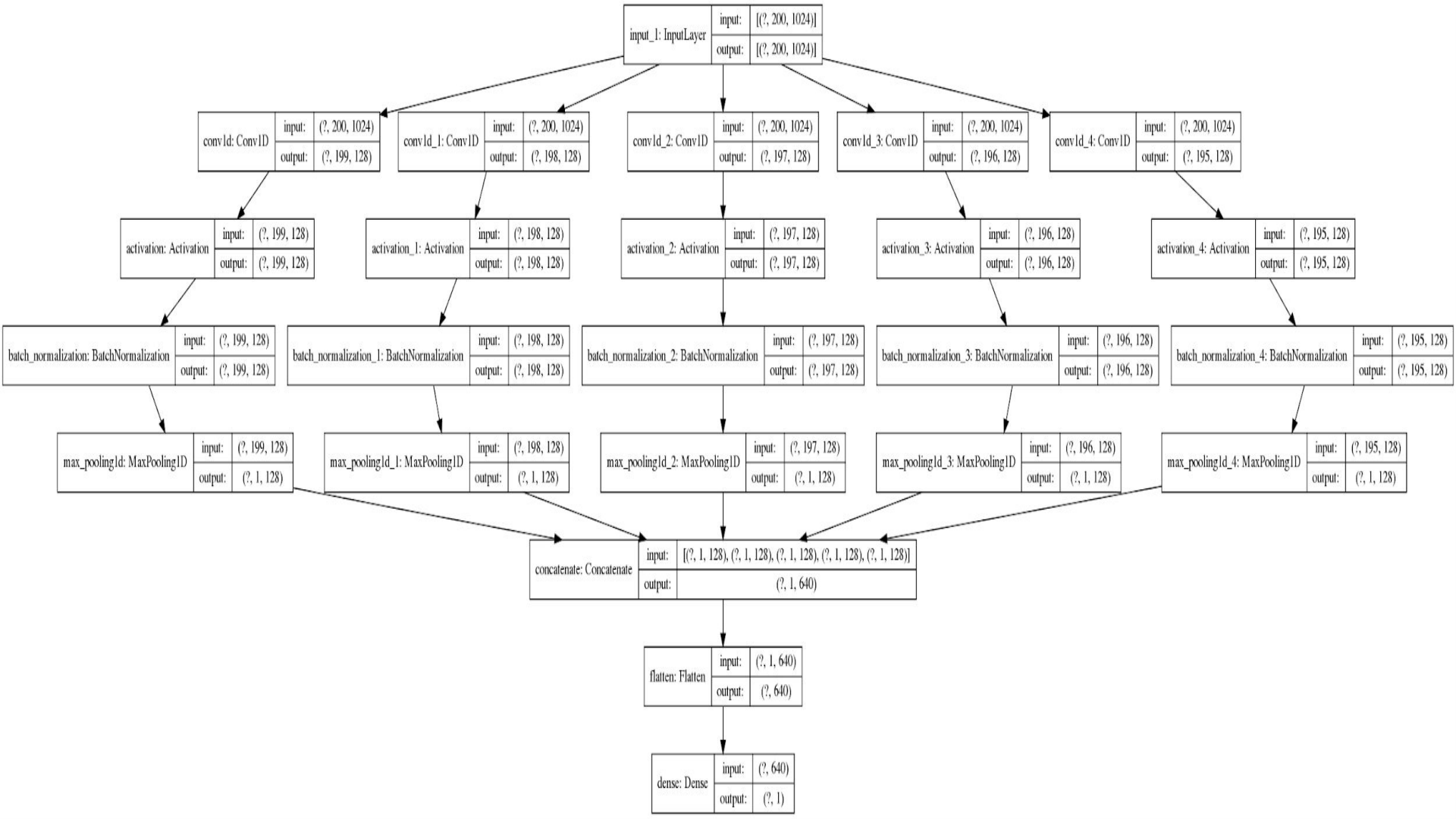
Model Architecture - BASELINE

Our baseline paper, whose performance we tried to replicate, used a convolutional neural network (CNN) to train the classifier. The inputs to the CNN were ELMo embeddings generated using AllenNLP. This baseline model used 5 convolutional layers followed by a ReLU activation function. This was followed by batch normalization and max-pooling of the output. The outputs were then combined to form an input to a fully connected layer to get a single output. A sigmoid function is then employed for the binary classification task.

This was replicated by us with the only difference being that we used Simple-Elmo package to generate the ELMo embeddings as opposed to AllenNLP whose ELMo module was removed in early 2020. The dataset used for training was also a little different in our case.

Model Architecture - Workflow




input_1: InputLayer
input: [(?, 200, 1024)]
output: [(?, 200, 1024)]
conv1d: Conv1D
input: (?, 200, 1024)
output: (?, 199, 128)
conv1d_1: Conv1D
input: (?, 200, 1024)
output: (?, 198, 128)
conv1d_2: Conv1D
input: (?, 200, 1024)
output: (?, 197, 128)
conv1d_3: Conv1D
input: (?, 200, 1024)
output: (?, 196, 128)
conv1d_4: Conv1D
input: (?, 200, 1024)
output: (?, 195, 128)
activation: Activation
input: (?, 199, 128)
output: (?, 199, 128)
activation_1: Activation
input: (?, 198, 128)
output: (?, 198, 128)
activation_2: Activation
input: (?, 197, 128)
output: (?, 197, 128)
activation_3: Activation
input: (?, 196, 128)
output: (?, 196, 128)
activation_4: Activation
input: (?, 195, 128)
output: (?, 195, 128)
batch_normalization: BatchNormalization
input: (?, 199, 128)
output: (?, 199, 128)
batch_normalization_1: BatchNormalization
input: (?, 198, 128)
output: (?, 198, 128)
batch_normalization_2: BatchNormalization
input: (?, 197, 128)
output: (?, 197, 128)
batch_normalization_3: BatchNormalization
input: (?, 196, 128)
output: (?, 196, 128)
batch_normalization_4: BatchNormalization
input: (?, 195, 128)
output: (?, 195, 128)
max_pooling1d: MaxPooling1D
input: (?, 199, 128)
output: (?, 1, 128)
max_pooling1d_1: MaxPooling1D
input: (?, 198, 128)
output: (?, 1, 128)
max_pooling1d_2: MaxPooling1D
input: (?, 197, 128)
output: (?, 1, 128)
max_pooling1d_3: MaxPooling1D
input: (?, 196, 128)
output: (?, 1, 128)
max_pooling1d_4: MaxPooling1D
input: (?, 195, 128)
output: (?, 1, 128)
concatenate: Concatenate
input: [(?, 1, 128), (?, 1, 128), (?, 1, 128), (?, 1, 128), (?, 1, 128)]
output: (?, 1, 640)
flatten: Flatten
input: (?, 1, 640)
output: (?, 640)
dense: Dense
input: (?, 640)
output: (?, 1)

Implementation - BASELINE+

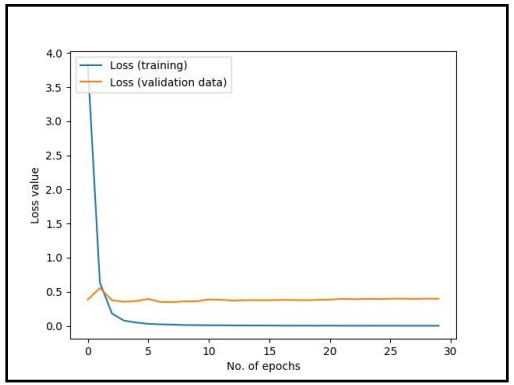
For bettering the performance of our replica of the baseline, we decided to use embeddings from s-BERT which gives sentence level embeddings and is a modified version of the pre-trained BERT network.

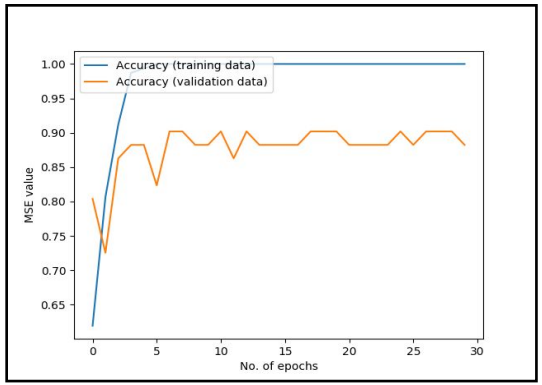
S-BERT: How is it different from BERT?

Sentence-BERT (SBERT), a modification of the pretrained BERT network that use siamese and triplet network structures to derive semantically meaningful sentence embeddings that can be compared using cosine-similarity. This reduces the effort for finding the most similar pair from 65 hours with BERT / RoBERTa to about 5 seconds with SBERT, while maintaining the accuracy from BERT.

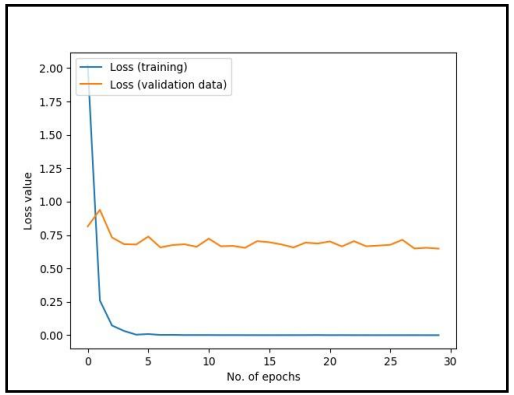
Results

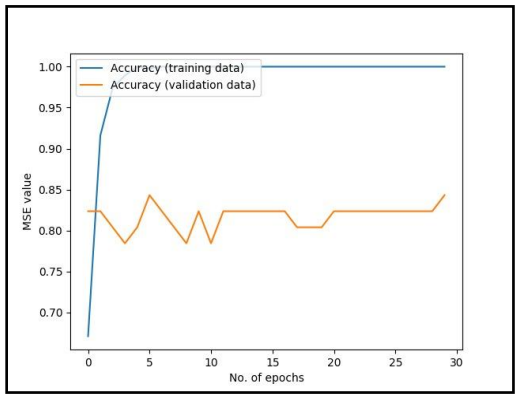
ELMo





S-BERT





Results

MODEL	*F1-score*	*Correct classification confidence*	*Incorrect classification confidence*
ELMo	0.752	0.845	0.741
S-BERT	**0.822**	0.892	0.749

We devised a new metric to compare our models by measuring average confidence scores of classification and misclassification.
These average confidence scores were computed by taking the mean of all sigmoid outputs for our predictions - both for correct and incorrect classifications.

For **ELMo**, the average confidence in the correct classification was 0.845, while for misclassification it was 0.741.
For **s-BERT**, the average confidence in the correct classification was 0.892, while for misclassification it was 0.749.

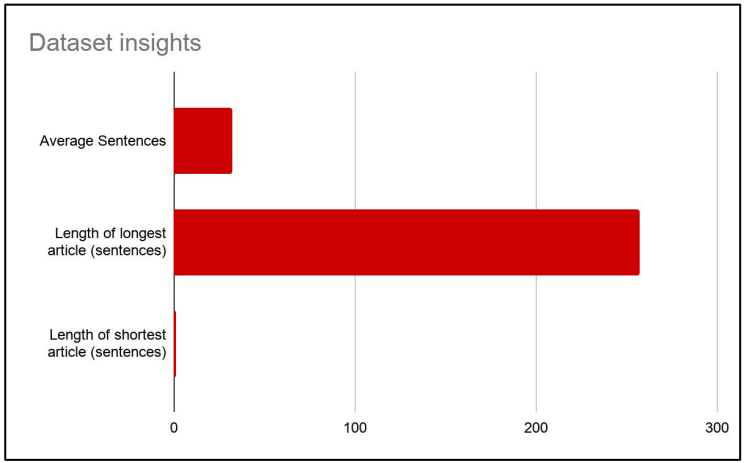
Qualitative Analysis

Average number of words per article:

587

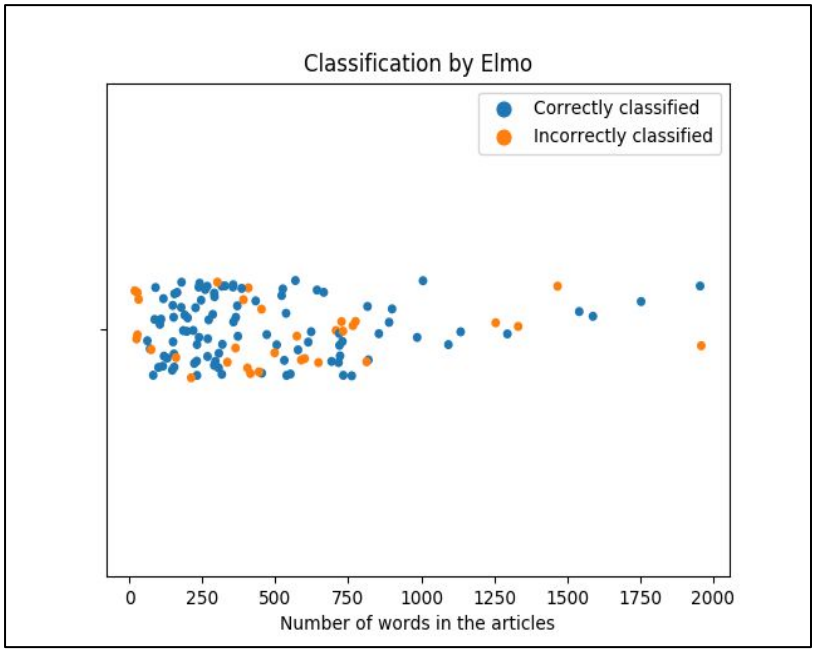
Average number of words per article (BERT-ELMo mismatch):

527

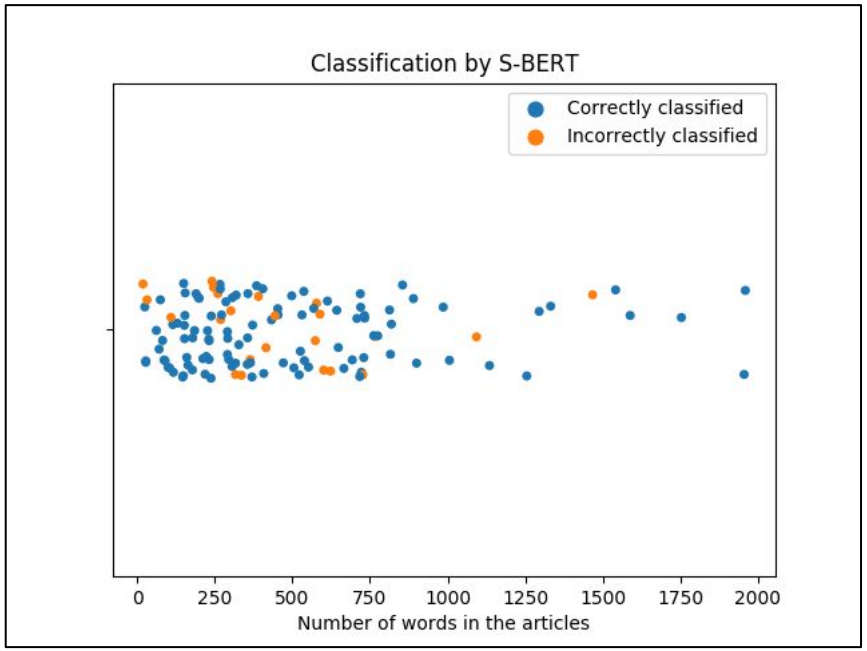


As a sort of sanity check for the ELMo and S-BERT embeddings we generated, we have used t-SNE and PCA to reduce dimensionality and visualize the embeddings on a small subset of the dataset using an interactive map using Plotly **(link to interactive map: ELMo , S-BERT)**. We observe the similarity in the articles close to each other.

Qualitative analysis

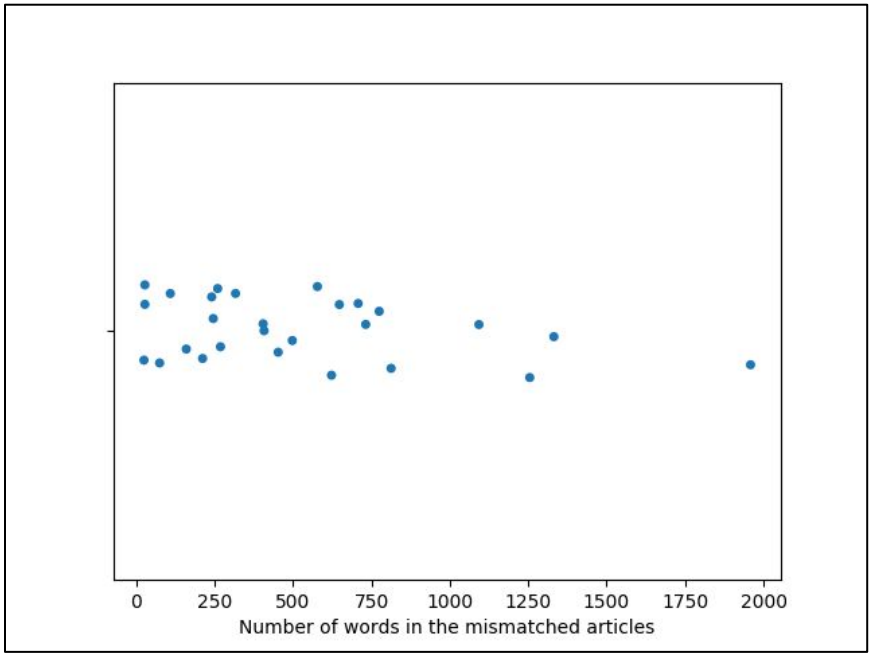


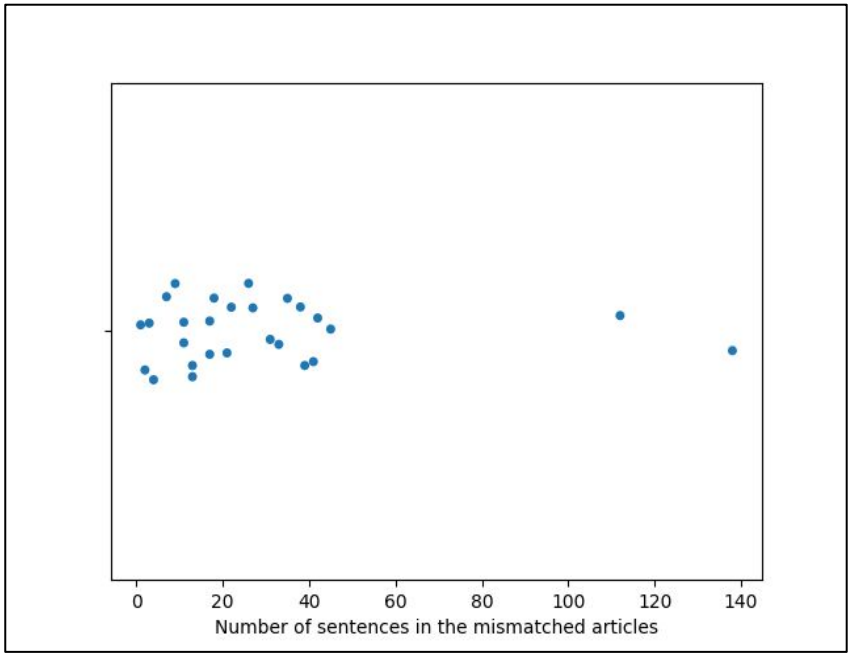
Classification by ELMo



Classification by S-BERT

Qualitative analysis





Classification mismatches between ELMo and S-BERT

CONCLUSION

We have thus replicated the baseline model implemented by the best performing team in the Shared-Task. The model implemented uses a CNN to train the classifier.

The best performing model in the shared task is considered to be the baseline for the hyperpartisan detection task with an F1 score of 0.809.

We used a pre-trained s-BERT model to go past the baseline which we had replicated. The improved model gave an F1 score of 0.822. The test dataset used to evaluate the baseline was not available publicly and hence that makes it difficult for us to compare our results with theirs.

Future work

The CNN that is being used to train the classifier can be replaced with some other Neural Network. We would recommend starting off by trying out RNNs which are used often in NLP tasks.

We would also suggest training with a larger dataset. As far as a generalization of our model goes, we have to keep in mind that the dataset used for training our model was handpicked to be mostly political in nature. The larger dataset that could be used can have more generic articles to help generalize better to all articles that could be input to the model.

Furthermore, this task can be modelled as a multiclass classification problem where the model can be trained to distinguish between left leaning articles, right leaning articles and unbiased(neutral) articles.