Team Bertha von Suttner at SemEval-2019 Task 4: Hyperpartisan News Detection using ELMo Sentence Representation Convolutional Network

Ye Jiang, Johann Petrak, Xingyi Song, Kalina Bontcheva, Diana Maynard
Department of Computer Science
University of Sheffield
Sheffield , UK
{yjiang18,johann.petrak,x.song,
k.bontcheva,d.maynard}@sheffield.ac.uk

Abstract

This paper describes the participation of team "bertha-von-suttner" in the SemEval2019 task 4 Hyperpartisan News Detection task. Our system[1] uses 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, our system ranked in first place, based on accuracy, the official scoring metric.

1 Introduction

Hyperpartisan news is typically defined as news which exhibits an extremely biased opinion in favour of one side, or unreasoning allegiance to one party (Potthast et al., 2017). SemEval-2019 Task 4 on "Hyperpartisan News Detection" (Kiesel et al., 2019) is a document-level classification task which requires building a precise and reliable algorithm to automatically discriminate hyperpartisan news from more balanced stories.

One of the major challenges of this task is that the model must have the ability to adapt to a large range of article sizes. In one of the training data sets, the *by-publisher* corpus, the average article length is 796 tokens, but the longest document has 93,714 tokens. Most state-of-the-art neural network approaches for document classification use a token sequence as network input (Kim, 2014; Yin and Schütze, 2016; Conneau et al., 2016). This implies either a high computational cost when a very large maximum sequence length is used to fully represent the longest articles, or alternatively potentially a significant loss of information if the sequence length is restricted to a manageable number of initial tokens from the document.

In this paper, we introduce the **E**LMo **S**entence **R**epresentation **C**onvolutional (ESRC) Network. We first pre-calculate sentence level embeddings as the average of ELMo (Peters et al., 2018) word embeddings for each sentence, and represent the document as a sequence of such sentence embeddings. We then apply a lightweight convolutional Neural Network (CNN), along with Batch Normalization (BN), to learn the document representations and predict the hyperpartisan classification.

Two types of data set have been made available for the task. The *by-publisher* corpus contains 750K articles which were automatically classified based on a categorization of the political bias of the news source. This dataset was split into a training set of 600K articles and a validation set of 150K articles, where all the articles in the validation set originated from sources not in the training set. The second set, *by-article*, contains just 645 articles which were labelled manually. The final evaluation (Potthast et al., 2019) was carried out on a dataset of 628 articles which were also labelled manually.

We created several models based on the two datasets and evaluated them using cross-validation on the *by-article* training set (as the final test set was not available to the participants and it was only available for a maximum of three evaluations). In order to investigate the usefulness of the *by-publisher* training data for training a model that performs well on the manually annotated *by-article* corpus, we experimented with various kinds of pre-training and fine-tuning, and found that any kind of use of the *by-publisher* corpus was actually harmful and decreased the usefulness of the model. A CNN model which used ELMo-based sentence embeddings to represent the article, and was trained on the *by-article* set only,

[1] The code is available at https://github.com/GateNLP/semeval2019-hyperpartisan-bertha-von-suttner

Proceedings of the 13th International Workshop on Semantic Evaluation (SemEval-2019), pages 840–844
Minneapolis, Minnesota, USA, June 6–7, 2019.

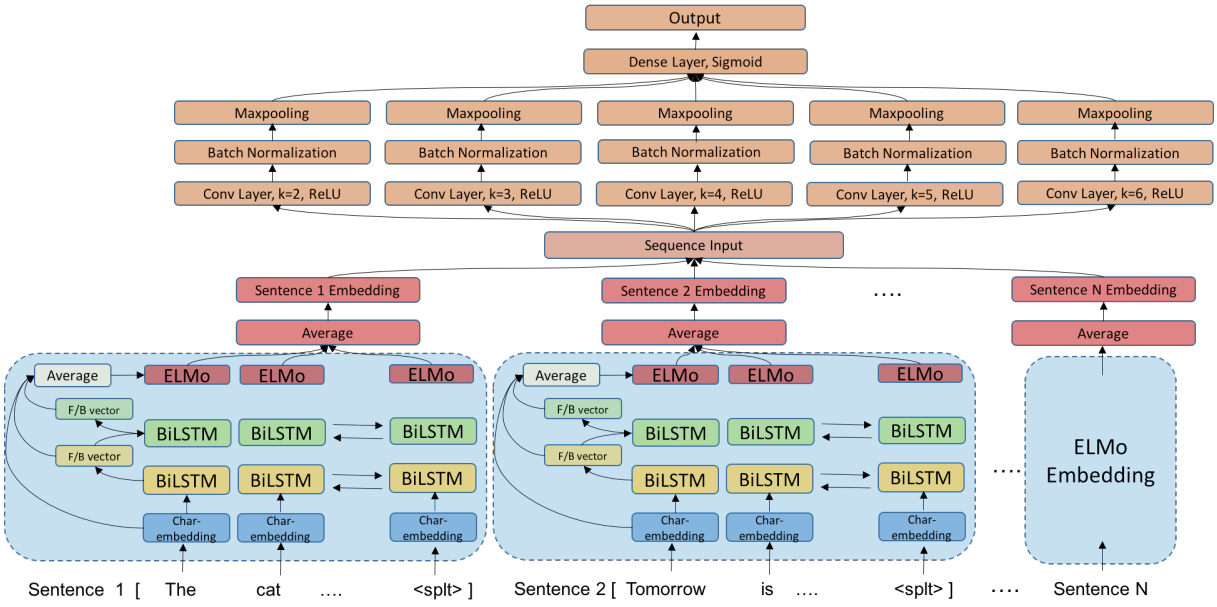

Figure 1: System architecture, *F/B vector* denotes Forward/Backward hidden state from BiLSTM layers.

turned out to outperform all other attempts.

2 System Description

In our model, we represent each article as a sequence of sentence embeddings, where each sentence embedding is calculated as the averaged word embeddings generated from a pre-trained ELMO model. The network consists of 5 parallel convolutional layers with kernel sizes 2,3,4,5,6 and 512 output features, each followed by a ReLU non-linearity, batch normalization, and max-pooling. All the results of the max-pooling layers are combined and go through a final fully connected layer with a sigmoid activation function for the final binary classification. Our model architecture is shown in Figure 1.

2.1 Data

The maximum, mean, and minimum numbers of tokens in the *by-article* corpus are: 6470, 666, 19 respectively, and in the *by-publisher* are: 93714, 796, 10 respectively. This makes it impractical to directly use word level representations as the input for our models. As a simple and easy to calculate compromise between representing the details of the article and as much of a longer article as possible, we represent the article as a sequence of sentence embeddings which are calculated as the average of the word embeddings of a sentence. This can be done using any pre-trained word embeddings and does not require a large training set for training or pre-training, so can be easily applied to even the small *by-article* corpus. To form the input sequence for our network, a maximum of the 200 initial tokens per sentence was used for each sentence embedding and a maximum of 200 sentences was used per article. The title of the article was used as the first article sentence for each document.

2.2 Preprocessing

Our model is character-based, which enabled us to only perform minimal pre-processing. We extract the title and article text from the original XML representation. All the original HTML paragraphs in the text cause a sentence break; the remaining text paragraphs have been split into sentences using `Spacy`. The original case of the text was maintained.

Whitespace is normalized to a single space between tokens; numbers are replaced by a special number token; and all punctuation and other special characters are preserved as input to the pre-trained ELMo model.

2.3 Deep Contextualized Word Representation

Traditionally, the input to CNNs is a set of pre-trained word vectors such as Word2Vec (Mikolov et al., 2013), Glove (Pennington et al., 2014), or Fasttext (Bojanowski et al., 2017). In our model, we use the AllenNLP library to generate ELMo

embeddings, in which the word representation is learned from character-based units as well as contextual information from the news articles. These character-based word representations allow our model to pick up on morphological features that word-level embeddings could miss, and a valid word representation can be formed even for out-of-vocabulary words. Furthermore, ELMo uses two bi-directional LSTM (Gers et al., 1999) layers to learn the contextual information from the text, which makes it capable of disambiguating the same word into different representations based on its context.

We use the original[2] pre-trained ELMo model to output three vectors for each word. Each vector corresponds to a layer output from the ELMo pre-trained model. Then, we take the average of all three vectors to form the final word vector, and compute the sentence vector by averaging the word vectors in the sentence.

2.4 Convolutional Layers

We combine 5 convolutional layers for different kernel sizes. Each layer is then followed by a non-linear activation function ReLU.

2.5 Batch Normalization

Batch Normalization (BN) is a method for reducing internal covariate shift in neural networks (Ioffe and Szegedy, 2015). BN normalizes the input distribution by subtracting the batch mean and dividing by the batch standard deviation, so that the ranges of input distribution between each layer stay the same. This allows the model to have a higher learning rate, so that the training speed is accelerated. It also reduces overfitting by decreasing the dependence of weight initialization between each layer. The original paper suggested that BN should be applied before the activation layer, but we apply it after the activation layer, after observing better performance in our model this way round. We also applied weighted moving-mean and moving-variance to avoid updating the mean and variance so aggressively in the mini-batch during training time.

2.6 Fully Connected Layer

We perform max-pooling on the output of the batch-normalization layers. Then the outputs of the max-pooling for all convolution layers are combined to form the input to a fully connected layer, which maps to a single output, followed by the Sigmoid function for the binary classification task.

[2] elmo_2x4096_512_2048cnn_2xhighway

3 Experiments and Results

The generated ELMo embedding contains three vectors for each word, where each vector corresponds to one of the output layers from the pre-trained model. We average the three vectors to generate word representations which contain morphological and contextual information, and compute the sentence vectors by averaging all the word vectors in each sentence. We take a maximum of 200 words for each sentence and a maximum of 200 sentences for each article. If a document has fewer than 200 sentences, we pad the number of sentences out to 200.

Our models are built by using the Keras library with a Tensorflow backend. All the results are shown in Table 1. The table shows for each model the accuracy obtained on the *by-article* training set, and for the submitted models, the *by-publisher* test set, and the hidden *by-article* test set (which unlike the other two, was not available to participants).

In order to investigate the correlation between the two datasets, we first built the `ESRC-publisher` model which is trained on a randomly selected 100K out of the 750K articles from the *by-publisher* corpus, as it is impractical to generate ELMo embeddings for the entire corpus. We also fine-tuned the `ESRC-publisher` model based on the *by-article* set to obtain the `ESRC-publisher-article` model by freezing the weights of all but the last layer of the model. Finally we trained the `ESRC-article` model only on the *by-article* set, one version without and one version (`ESRC-article-BN`) with the additional batch normalization (BN) layer. The accuracy for the `ERC-publisher` model is from evaluating on the whole `by-article` training set, while all other evaluations on the `by-article` training set were carried out using a 10-fold cross validation. However, because of the very limited size of that corpus, the evaluation part of each fold was also used for early stopping and model selection within each fold.

For the evaluation on the hidden test set, we selected the best three models from the 10-folds, according to the accuracy on the evaluation set of

each fold to form an averaged ensemble model, `ESRC-article-BN-Ens`.

For comparison, the table also shows the results for an earlier version of the model, `GloVe-article`, which used GloVe word embeddings (6 billion words, 300 dimensional) to represent up to the first 400 words of the article and did not use batch normalization.

Models	**By-Article Training**
GloVe-article	0.7963
ESRC-publisher	0.5643
ESRC-publisher-article	0.8189
ESRC-article	0.8182
ESRC-article-BN	0.8387
ESRC-article-BN-Ens	**0.8404**
Submitted Models	**By-Article Test**
GloVe-article	0.7659
ESRC-article-BN-Ens	0.8216
Submitted Models	**By-Publisher Test**
GloVe-article	0.6435
ESRC-article-BN-Ens	0.5947

Table 1: System comparison (accuracy).

The parameters in our models are as follows: we used 5 convolutional layers with kernel sizes ($k = 2, 3, 4, 5, 6$) and 512 output features. The momentum in the batch normalization is set to 0.7.[3] We used the default Adam algorithm as the optimizer, and Binary Cross-Entropy as the loss function. The batch size was set to 32 and the fixed number of epochs used was 30. The final best model after 30 epochs was used.

4 Discussion and Conclusion

The `ESRC-publisher` model performs extremely badly on the *by-article* evaluation data. Even fine-tuning the `ESRC-publisher` model on the *by-article* corpus produces models which perform worse than a model that is trained only on the *by-article* data. This confirms results from earlier experiments with simpler models that any use of the *by-publisher* data only hurts the model. We assume that the algorithm used for assigning the labels to this dataset just does not reflect any information about hyperpartisan articles sufficiently to be helpful. For this reason, the `GloVe-article` model also outperforms the ESRC-article-BN-Ens model on the by-publisher dataset.

A quick manual inspection of the data showed that the source of an article is insufficient by far to identify articles as hyperpartisan or not. It would be interesting to know how the algorithm used for creating the *by-publisher* corpus actually performs on the *by-article* corpus. To get maximum performance on the *by-article* dataset, we therefore decided to completely ignore the *by-publisher* data for our final model. The use of BN also showed significant improvement.

Since we use a CNN with a comparatively large number of parameters in relation to the size of the training set which is rather small, we expect significant variance in the generated models and therefore use the average of an ensemble of several models for the final predictions.

[3] This was determined by exploring values from 0.1 to 0.9 at an earlier stage of the experiments and kept, so it may not be the optimal value.

5 Acknowledgements

Research partially supported by a Grantham Centre for Sustainable Future Scholarship, a Google Faculty Research Award 2017, and projects funded by the European Commissions Horizon 2020 research and innovation programme under grant agreements No. 654024 SoBigData and No. 825297 WeVerify.

References

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Alexis Conneau, Holger Schwenk, Loïc Barrault, and Yann Lecun. 2016. Very deep convolutional networks for text classification. *arXiv preprint arXiv:1606.01781*.

Felix A Gers, Jürgen Schmidhuber, and Fred Cummins. 1999. Learning to forget: Continual prediction with lstm.

Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*.

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 (SemEval 2019)*. Association for Computational Linguistics.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Martin Potthast, Tim Gollub, Matti Wiegmann, and Benno Stein. 2019. TIRA Integrated Research Architecture. In Nicola Ferro and Carol Peters, editors, *Information Retrieval Evaluation in a Changing World - Lessons Learned from 20 Years of CLEF*. Springer.

Martin Potthast, Johannes Kiesel, Kevin Reinartz, Janek Bevendorff, and Benno Stein. 2017. A stylometric inquiry into hyperpartisan and fake news. *arXiv preprint arXiv:1702.05638*.

Wenpeng Yin and Hinrich Schütze. 2016. Multichannel variable-size convolution for sentence classification. *arXiv preprint arXiv:1603.04513*.