HCMUS at MediaEval2021: Content-Based Misinformation Detection Using Contextualized Word Embedding from BERT

Tuan-Luc Huynh[1,2],Nhat-Khang Ngo[1,2],Phu-Van Nguyen[1,2],Thien-Tri Cao[1,2],Thanh-Danh Le[1,2]
,Hai-Dang Nguyen [1,2], Minh-Triet Tran[1,2,3]
[1]University of Science, VNU-HCM
[2]Vietnam National University, Ho Chi Minh city, Vietnam
[3]John von Neumann Institute, VNU-HCM
{htluc,nnkhang,npvan,cttri,ltdanh}19@apcs.fitus.edu.vn,nhdang@selab.hcmus.edu.vn,tmtriet@fit.hcmus.edu.vn

ABSTRACT

The FakeNews task in MediaEval2021 explores the challenge of building accurate and high-performance algorithms. Despite the dominance of deep learning approaches in fake news detection, in this paper, we propose different approaches leveraging the advantages of using pretrained BERT family transformers in extracting word embedding. The result from experiments shows that averaging ensemble methods using machine learning classifiers as estimators can achieve up to 0.6478 Matthew Correlation Coefficient(MCC) in the run submission's test set.

1 INTRODUCTION

In the context of social media where information is no longer trustworthy, MediaEval2021 FakeNews: Corona Virus and Conspiracies Multimedia Analysis call participants for solving problem misinformation disseminated in the context of the long-lasting COVID-19 crisis. The first subtask is about Text-Based Misinformation Detection, which is based on tweets on Twitter. The mission is to classify tweets into categories like "promote/support", "discuss" or "not related" regarding COVID-19 related fake news and conspiracy theories. In this paper, we follow the content-based new detection approach. We experiment in both machine learning and deep learning approaches and propose ensemble models of different scikit-learn machine learning algorithms[8]. We also propose some features that are exceptionally effective on these classifiers.

2 RELATED WORK

Fake news detection and classification are no longer new problems; nevertheless, more accurate and efficient models is needed every year because the task is surprisingly challenging. Zhou et al [12], divides fake news into four categories *Knowledge-based, Style-based, Propagation-based, Source-based* and using *Bag-Of-Word* to obtain the frequency of lexicons for classifying fake news. In general, there are two different approaches to this problem: News Content-based learning and Social Context-based learning. Our work is greatly inspired by the previous attempt of Tuan et al.[9]

MediaEval'21, December 13-15 2021, Online

3 APPROACH

3.1 Preprocess

We follow a conventional text preprocess pipeline. Additionally, we also expand contractions; expand internet slang that are popular among tweets; extract URLs domain; convert emojis and emoticons to text. Finally, Ekphrasis[2] library segments words written with no spaces and correct misspellings or typos. As for data augmentation, we follow the augmentation method used by Tuan et al.[9] in his work: EDA[10]. Our data consist of around 1500 sentences. Therefore, according to the paper of Wei et al.[10], we decide to use the following parameters for data augmentation: "–num_aug=8 –alpha_sr=0.05 –alpha_rd=0.1 –alpha_ri=0.1 –alpha_rs=0.1".

3.2 Features

Inspired by the work of Tuan et al[9], we decide to use the COVID-Twitter-BERT-v2 pretrained model provided by Müller et al[7] for extracting word embedding. The preprocessed data are fed into BERT tokenizer's[11] and "max_length" is set to 64, which is an approximation for the mean of the tweets' length. The output of the tokenizer is then fed directly to the pretrained model to obtain all the hidden states. We process the hidden states into 5 different features, which inspired by Dharti's article[4]: concatenate 4 last hidden states (Concat), last hidden state (LHS), sum of 4 last hidden states (Sum), mean of 4 last hidden states (Mean), and sentence embedding (Sentence). All the mentioned features are self-explanatory, except the "Sentence" feature, which is the mean of all 64 tokens of the "LHS" feature.

3.3 Models

We use two models: a dense model and a dense model with a convolutional layer[1] as illustrated in Figure1 for the deep learning approach. The dense model shares the same structure as the convolutional one, except it does not have the convolutional layer[1]. Moreover, the feature can be any feature as described above. For machine learning, we try applying "Sentence", "LHS", and "Mean" features on different classifiers provided by scikit-learn[8].

4 EXPERIMENTS

In the deep learning approach, we set the learning rate for both dense models to be 1e-04. The optimizer for both models is AdamW[6]. The train test ratio is 8:2. Since the dataset is small, we try applying EDA[10] and evaluate using the same method as the non-augmented attempt. The results for the two attempts are illustrated

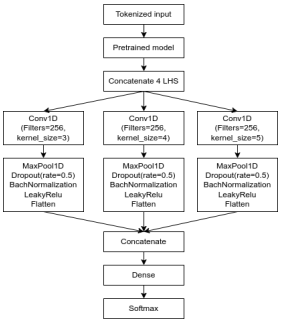

Figure 1: Architecture of Conv1d model

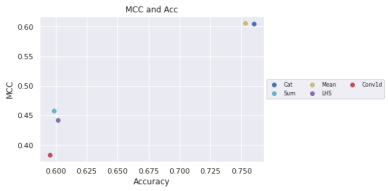

Figure 2: Deep learning: no augmentation

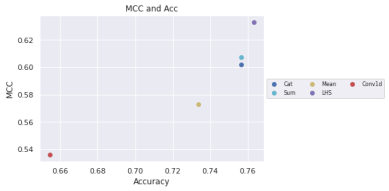

Figure 3: Deep learning: EDA

in Figure2 and Figure3, respectively.

As for the machine learning approach, the detailed result on the test set as illustrated by table 1. Most of the classifiers obtain MCC greater than 0.5, which is better than some deep learning models. Some classifiers even have an MCC score of 0.63, which is as good as the dense model using the augmented "LHS" feature. In the end, this is the main reason why we decide to move from deep learning approach to machine learning approach.

We perform a 5 fold stratified cross-validation to ensure the classifiers work well on new data. Table 2 shows the cross_val_score using MCC as the metric. After the cross-validation process, some classifiers still retain high MCC (E.g. SVC using the "Sentence" feature). According to this cross-validation result, we decide to choose SVC, Logistic Regression using the "Sentence" feature, and classifiers that have an MCC greater than 0.5 using the "Mean" feature as potential classifiers for later BayesSearchCV[5] fine-tuning. Finally, all fine-tuned classifiers are estimators for the voting classifiers for better generalization. We use both soft and hard voting ensemble methods.

5 RESULTS AND ANALYSIS

All the results have MCC greater than 0.6 as illustrated by table 3. The stand-alone SVC classifier in run 5 is the best classifier in term of performance; however, we recommend using the ensemble classifiers for better generalization. Classifiers using "Sentence" feature obtain competitive result in comparison with classifiers using "Mean" feature.

Table 1: Machine learning: MCC metric on test set

Classifier	Sentence	LHS	Mean
Perceptron	0.5645	0.6014	0.6096
SGDHinge	0.5516	0.6252	0.6305
SGDLog	0.5458	0.5899	0.6305
SGDModified_huber	0.5719	0.6125	0.5849
SGDSquared_hinge	0.5115	0.6068	0.5747
SGDPerceptron	0.5661	0.6054	0.5894
LogisticRegression	0.5502	0.6058	0.5946
SVC	0.6279	0.483	0.4739
LinearSVC	0.5251	0.6012	0.6127

Table 2: Machine learning: MCC cross_val_score of some efficient classifiers and features

Classifier	Sentence	LHS	Mean
Perceptron	0.5114	0.4735	0.4926
SGDHinge	0.4875	0.4659	0.4898
SGDLog	0.5127	0.4823	0.5106
SGDModified_huber	0.4888	0.4831	0.5161
SGDSquared_hinge	0.5045	0.4699	0.4889
SGDPerceptron	0.4868	0.4645	0.479
LogisticRegression	0.5141	0.4936	0.5167
SVC	0.5822	0.4008	0.3939
LinearSVC	0.4775	0.4938	0.504

Table 3: Run submission results

Run ID	Classifier	MCC
1	Soft voting "Mean"	0.6145
2	Hard voting "Mean"	0.6282
3	Soft voting "Sentence"	0.6016
4	Hard voting "Sentence"	0.6154
5	SVC "Sentence"	0.6478

6 CONCLUSION

We propose using different features derived from BERT's[3] hidden states for training lightweight and high performance machine learning classifiers in this text classification task. Our approach achieves an average score over 0.6 MCC in the run submission without using any augmentation or extra information. In future works, we will thoroughly experiment on more deep learning models.

ACKNOWLEDGMENTS

This work was funded by Gia Lam Urban Development and Investment Company Limited, Vingroup and supported by Vingroup Innovation Foundation (VINIF) under project code VINIF.2019.DA19

REFERENCES

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. (2015). https://www.tensorflow.org/ Software available from tensorflow.org.

[2] Christos Baziotis, Nikos Pelekis, and Christos Doulkeridis. 2017. DataStories at SemEval-2017 Task 4: Deep LSTM with Attention for Message-level and Topic-based Sentiment Analysis. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. Association for Computational Linguistics, Vancouver, Canada, 747–754.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[4] Dharti Dhami. 2020. Understanding BERT — Word Embeddings. (2020). https://medium.com/@dhartidhami/understanding-bert-word-embeddings-7dc4d2ea54ca

[5] Tim Head, Manoj Kumar, Holger Nahrstaedt, Gilles Louppe, and Iaroslav Shcherbatyi. 2021. scikit-optimize/scikit-optimize. (Oct. 2021). https://doi.org/10.5281/zenodo.5565057

[6] Ilya Loshchilov and Frank Hutter. 2019. Decoupled Weight Decay Regularization. (2019). arXiv:cs.LG/1711.05101

[7] Martin Müller, Marcel Salathé, and Per E Kummervold. 2020. COVID-Twitter-BERT: A Natural Language Processing Model to Analyse COVID-19 Content on Twitter. (2020). arXiv:cs.CL/2005.07503

[8] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[9] Nguyen Manh Duc Tuan and Pham Quang Nhat Minh. 2020. FakeNews Detection Using Pre-trained Language Models and Graph Convolutional Networks. (2020).

[10] Jason Wei and Kai Zou. 2019. EDA: Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks. (2019). arXiv:cs.CL/1901.11196

[11] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Association for Computational Linguistics, Online, 38–45. https://www.aclweb.org/anthology/2020.emnlp-demos.6

[12] Xinyi Zhou and Reza Zafarani. 2020. A Survey of Fake News: Fundamental Theories, Detection Methods, and Opportunities. *ACM Comput. Surv.* 53, 5, Article 109 (sep 2020), 40 pages. https://doi.org/10.1145/3395046