LISAC FSDM-USMBA Team at SemEval-2020 Task 12: Overcoming AraBERT's pretrain-finetune discrepancy for Arabic offensive language identification

Hamza Alami[1], **Said Ouatik El Alaoui**[1,2],
Abdessamad Benlahbib[1], **Noureddine En-nahnahi**[1],
[1] LISAC Laboratory, Faculty of Sciences Dhar EL Mehraz (F.S.D.M),
Sidi Mohamed Ben Abdellah University (U.S.M.B.A)
[2] Ibn Tofail University, National School of Applied Sciences, Kenitra, Morocco
hamza.alami1@usmba.ac.ma, s_ouatik@yahoo.com,
abdessamad.benlahbib@usmba.ac.ma, noureddine.en-nahnahi@usmba.ac.ma

Abstract

AraBERT is an Arabic version of the state-of-the-art Bidirectional Encoder Representations from Transformers (*BERT*) model. The latter has achieved good performance in a variety of Natural Language Processing (NLP) tasks. In this paper, we propose an effective *AraBERT* embeddings-based method for dealing with offensive Arabic language in Twitter. First, we pre-process tweets by handling emojis and including their Arabic meanings. Next, to overcome the pretrain-finetune discrepancy, we substitute each detected emojis by the special token *[MASK]* into both fine tuning and inference phases. Then, we represent tweets tokens by applying *AraBERT* model. Finally, we feed the tweet representation into a sigmoid function to decide whether a tweet is offensive or not. The proposed method achieved the best results on *OffensEval 2020: Arabic* task and reached a macro F1 score equal to 90.17%.

1 Introduction

Negative social media behaviors including cyberbullying, trolling and offensive language are intended to hurt or embarrass a victim. Prevention of antisocial internet use is then a protection for internet users that let them live a positive experience through the internet without any type of harassment. Zampier et al. (2019; 2020) introduced a shared task for identifying and categorizing offensive language in social media. The task is divided into three sub-tasks including Sub-task A: Detect if a post is offensive (OFF) or not (NOT); Sub-task B: Categorize the offense type of an offense post as targeted insult (TIN), targeted threat (TTH), or untargeted (UNT); and Sub-task C: Identify the offense target as individual (IND), group of people (GRP), organization or entity (ORG), or other (OTH). The three sub-tasks were run for English while only sub-task A was run for Arabic, Danish, Greek, and Turkish.

The majority of OffensEval 2019 (Zampieri et al., 2019) participants used deep neural network models such as Bidirectional Encoder Representations from Transformers (*BERT*) (Devlin et al., 2019) to build their offensive language detector. These models have the capability to learn and automatically extract complex features from raw data, thus it has achieved some state of the art results in different Natural Language Processing (NLP) tasks.

In this paper, we follow the same path by building an Arabic offensive language detector based on *BERT* more specifically *AraBERT* (Antoun et al., 2020). Our method overcomes *AraBERT*'s pretrain-finetune discrepancy by including the special token *[MASK]* during fine tuning and inference phases. We obtained promising results since we achieved 90.17% macro F1 score on test set and we ranked on top of the participants list.

This paper is organised as follows: Section 2 describes the proposed method; Section 3 presents the experimental settings; Finally, Section 4 concludes and outlines future directions.

Proceedings of the 14th International Workshop on Semantic Evaluation, pages 2080–2085
Barcelona, Spain (Online), December 12, 2020.

2 Method

Our method is composed of three main steps including 1) tweet preprocessing; 2) representation; and 3) classification.

2.1 Tweet Preprocessing

The aim of this step is to extract new features from emojis and then tokenize tweets in order to use *AraBERT* model (Devlin et al., 2019; Antoun et al., 2020). Given that a tweet may contain words and emojis, we handle emojis in a special way. Thus, our preprocessing pipeline follows these phases: 1) Detect emojis: if emojis exist, we extract the position and the meaning of each emoji; 2) Substitute emojis with the special token *[MASK]* and translate emojis meanings from English to Arabic. By substituting emojis with *[MASK]*, we overcame the pretrain-finetune discrepancy of *AraBERT* (Yang et al., 2019). The latter can be explained by the fact that the special tokens such as *[MASK]* used by *AraBERT* during pretraining are absent from specific datasets at finetuning step; 3) Concatenate the emojis-free tweets with their respective emojis Arabic meanings. The special token *[CLS]* is added to the head of each sentence and the special token *[SEP]* delimits the sentence and the emojis Arabic meanings; 4) Tokenize the output sentence. All the words except the special tokens are segmented by Farasa segmenter (Abdelali et al., 2016) and then tokenized with *AraBERT* tokenizer. If the tweet is free of emojis it is directly passed to Farasa segmenter and *AraBERT* tokenizer. Figure 1. illustrates the flowchart of our tweet preprocessing step.

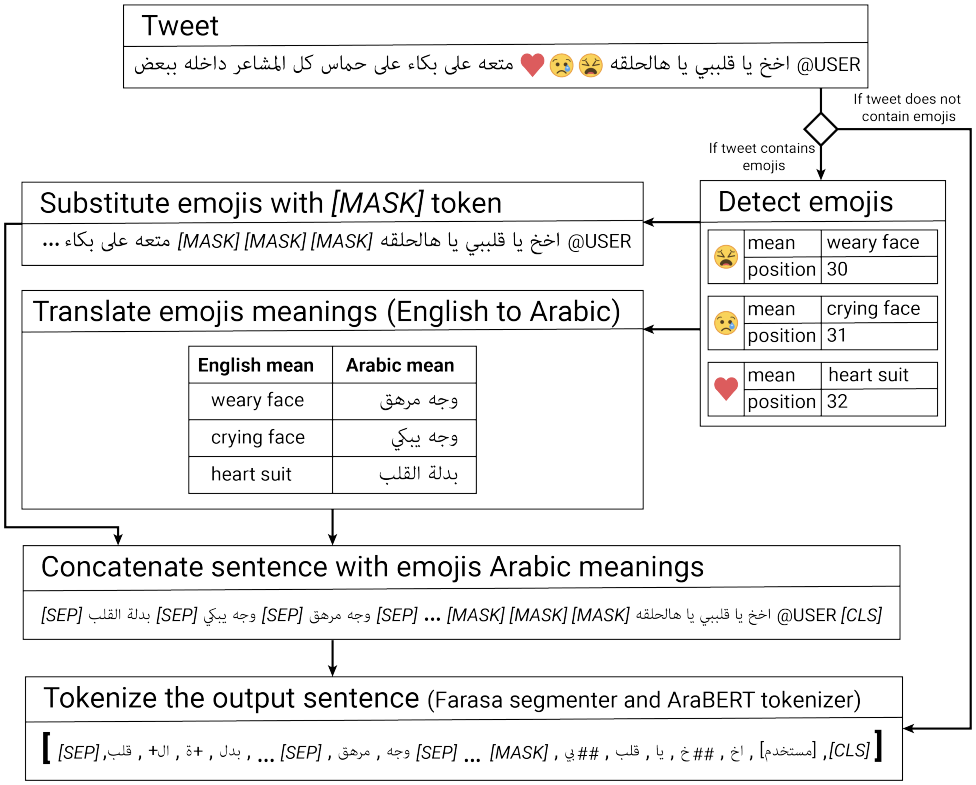


Figure 1: Tweet preprocessing flowchart.

2.2 Tweet Representation

After preprocessing a tweet, we segment the obtained tokens into two segments. The first one contains tweet tokens, while the second segment contains the tokens of the Arabic meanings of detected emojis. The inputs of *AraBERT* model are then the indices and the segments of tokens. The *AraBERT* model (Antoun et al., 2020) is applied to compute token's embeddings. This model has the same configuration as *BERT BASE* model (Devlin et al., 2019) which is 12 encoder blocks, 768 hidden dimensions, 12

attention heads, 512 maximum sequence length, and a total of ∼110M parameters. The model is trained on two objectives: 1) Masked Language Model where the model is trained to predict a masked token; and 2) Next Sentence Prediction in which the model is optimized to predict if the second sentence follows the first sentence. *AraBERT* was pre-trained on 70 million Arabic sentences, corresponding to ∼24GB of text. Finally, a tweet representation is computed by applying global max pooling function on top of *AraBERT* tokens representation. Figure 2 shows the flowchart of tweet representation step.

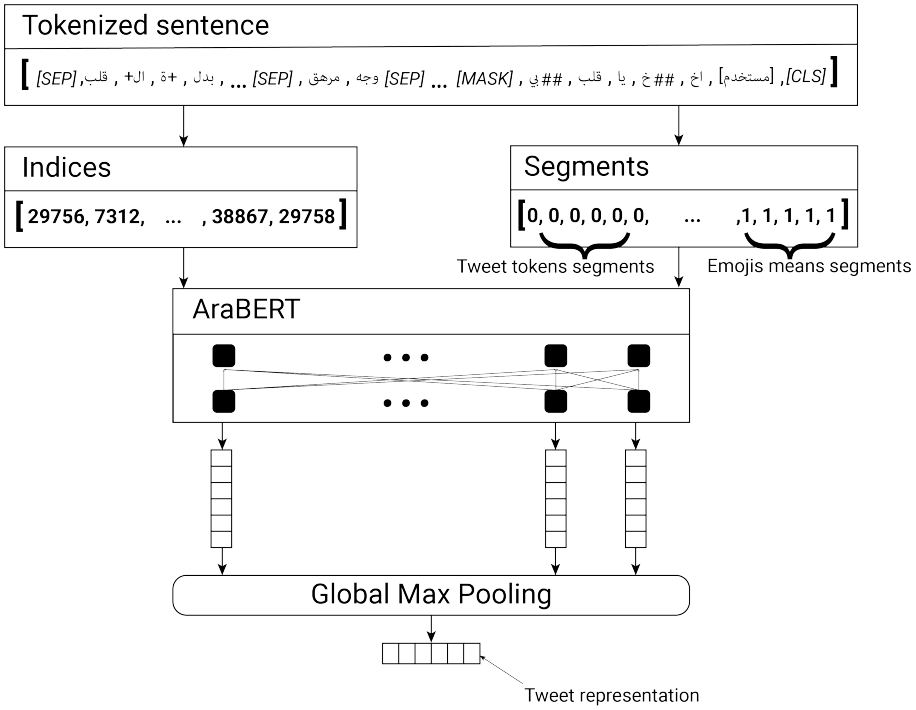


Figure 2: Tweet representation flowchart.

2.3 Tweet Classification

The last step of our method consists in classifying a tweet given its representation. Thus, in order to compute the likelihood of a tweet to be offensive or not, we use a sigmoid function that takes a tweet representation as input and we train the model to optimize the binary cross entropy loss. We notice that during training, *AraBERT* parameters are fine-tuned on this specific task: Arabic Offensive Language identification on Twitter.

3 Experimental results

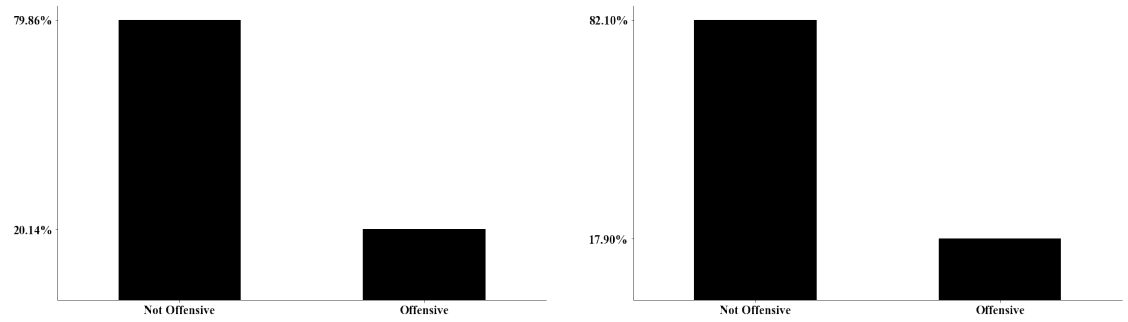
3.1 Dataset

The dataset[1] used in *OffensEval 2020: Arabic* is proposed by Mubarak et al. (2020). It is a large dataset that contains 10000 labeled tweets with OFF or NOT, for offensive and not offensive. The dataset is split as follows: the training set which includes 7000 (70%) of tweets, the development set that contains 1000 (10%) of tweets, and the test set that contains 2000 (20%) tweets. Figure 3. shows the distributions of the train dataset and development dataset. The test labels were kept unknown to evaluate different participants models.

3.2 Experimental settings

We used Tensorflow 2.0 and keras-bert[2] libraries to build and train our models. During training, the number of epochs was 5 and the batch size was 32 for all models. We apply Adam optimizer with a

[1]https://competitions.codalab.org/competitions/23021
[2]https://github.com/CyberZHG/keras-bert



(a) (b)

Figure 3: (a) The distribution of train dataset which contains 7000 labeled tweets; (b) The distribution of development dataset which contains 1000 labeled tweets.

learning rate 10^{-4}, warmup, and weight decay. The maximum input sequence length of *AraBERT* was 128. All our experiments were run on google colab[3].

3.3 Evaluation

In this work, we built three models following the same steps described in Section 2, where each model has a different preprocessing step. The first model, named *vanilla model*, do not apply any emojis preprocessing. The sentence was directly segmented with Farasa segmenter then tokenized with *AraBERT* tokenizer. The second model, entitled *AraBERTEmojisIn*, substitutes emojis with their Arabic meanings. In the last model, *AraBERTEmojisOut*, we substitute emojis with the special token *[MASK]* and create another input segment that contains the Arabic meaning of each emoji within the tweet. The meanings were separated by the special token *[SEP]*. Table 1 illustrates the obtained results according to the precision, recall and F1 score metrics. As illustrated in Figure 4, we also used the ROC curve to compare further the vanilla, *AraBERTEmojisIN*, and *AraBERTEmojisOUT* models. The *AraBERTEmojisOUT* model achieves the best results in terms of macro F1 score, weighted F1 score and the AUC score. This is due to the use of *AraBERT* embeddings, and the way the model handles the pretrain-finetune discrepancy by including *[MASK]* token in training and inference phases.

Table 1: The obtained results of our models on OffensEval 2020: Arabic dataset

		Macro average			**Weighted average**		
	Accuracy	***Precision***	***Recall***	***F1 score***	***Precision***	***Recall***	***F1 score***
Vanilla	93,3%	89,58%	88,10%	88,82%	93,19%	93,3%	93,23%
AraBERTEmojisIN	93,3%	87,83%	89,04%	88,42%	93,41%	93,3%	93,35%
AraBERTEmojisOUT	93.9%	87,11%	91,42%	**89,06%**	94,38%	93,9%	**94,06%**

The prediction results based on our *AraBERTEmojisOUT* model achieved 90.17% in terms of macro F1 score and ranked on the top of *"OffensEval 2020: Arabic"* participations ranking list. In order to illustrate in more details the performance of *AraBERTEmojisOUT* model using of test data, we made the confusion matrix as shown in Figure 5. The model achieved 96% precision and 96% recall on NOT class labels and 83% precision and 85% recall on OFF class labels. This is clarified by the fact that the training set is unbalanced where the NOT samples size is approximately two times the OFF samples sizes.

4 Conclusion

In this paper, we described our method for identifying Arabic offensive language in Twitter. The preprocessing step consists in extracting new features from emojis to enrich the information within tweets. Afterward, *AraBERT* model is applied to build tweets representations by taking into consideration all the words and emojis contained in the tweet. To enhance tweet representation, all *AraBERT* parameters

[3]https://colab.research.google.com/

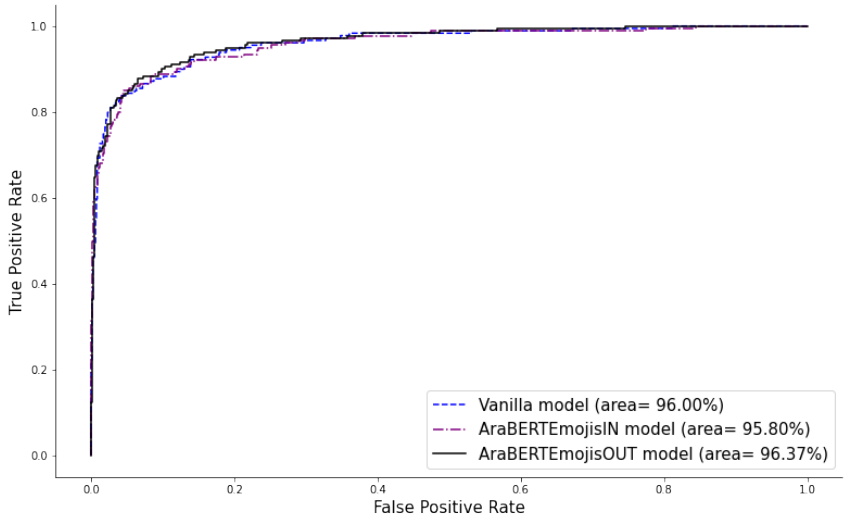


Figure 4: ROC curve and AUC scores for the vanilla, *AraBERTEmojisIN*, and *AraBERTEmojisOUT* models.

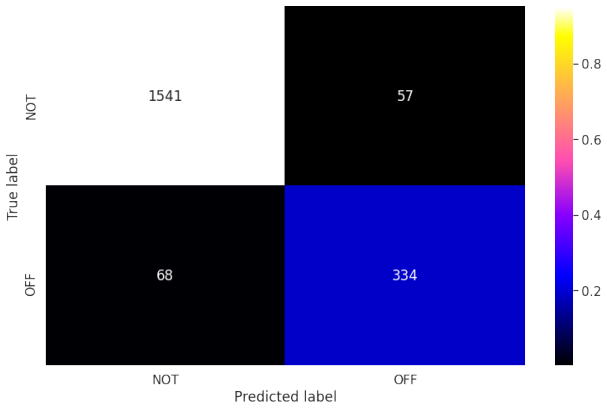


Figure 5: Confusion matrix of *AraBERTEmojisOUT* model.

were fine-tuned on this specific task: Arabic Offensive Language Identification on Twitter. Finally, a tweet label is predicted by applying a sigmoid function to its representation. Our method for *OffensEval 2020: Arabic* ranked on the top of participants list in terms of macro F1 score. To extend our work, we plan to improve tweets representation by applying more advanced word embeddings such as an Arabic version of *XLNet*(Yang et al., 2019).

References

Ahmed Abdelali, Kareem Darwish, Nadir Durrani, and Hamdy Mubarak. 2016. Farasa: A fast and furious segmenter for arabic. In *Proceedings of the Demonstrations Session, NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 11–16. The Association for Computational Linguistics.

Wissam Antoun, Fady Baly, and Hazem M. Hajj. 2020. Arabert: Transformer-based model for arabic language understanding. *CoRR*, abs/2003.00104.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Hamdy Mubarak, Ammar Rashed, Kareem Darwish, Younes Samih, and Ahmed Abdelali. 2020. Arabic offensive language on twitter: Analysis and experiments. *arXiv preprint arXiv:2004.02192*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pages 5754–5764.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019. Semeval-2019 task 6: Identifying and categorizing offensive language in social media (offenseval). In Jonathan May, Ekaterina Shutova, Aurélie Herbelot, Xiaodan Zhu, Marianna Apidianaki, and Saif M. Mohammad, editors, *Proceedings of the 13th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT 2019, Minneapolis, MN, USA, June 6-7, 2019*, pages 75–86. Association for Computational Linguistics.

Marcos Zampieri, Preslav Nakov, Sara Rosenthal, Pepa Atanasova, Georgi Karadzhov, Hamdy Mubarak, Leon Derczynski, Zeses Pitenis, and Çağrı Çöltekin. 2020. SemEval-2020 Task 12: Multilingual Offensive Language Identification in Social Media (OffensEval 2020). In *Proceedings of SemEval*.