BERT-based Ensemble Approaches for Hate Speech Detection

Khouloud Mnassri, Praboda Rajapaksha, Reza Farahbakhsh, Noel Crespi
Telecom SudParis, IMT, Institut Polytechnique de Paris, 91764 Palaiseau, France
{khouloud.mnassri, praboda.rajapaksha, reza.farahbakhsh, noel.crespi}@telecom-sudparis.eu

***Abstract*—With the freedom of communication provided in online social media, hate speech has increasingly generated. This leads to cyber conflicts affecting social life at the individual and national levels. As a result, hateful content classification is becoming increasingly demanded for filtering hate content before being sent to the social networks. This paper focuses on classifying hate speech in social media using multiple deep models that are implemented by integrating recent transformer-based language models such as BERT, and neural networks. To improve the classification performances, we evaluated with several ensemble techniques, including soft voting, maximum value, hard voting and stacking. We used three publicly available Twitter datasets (Davidson, HatEval2019, OLID) that are generated to identify offensive languages. We fused all these datasets to generate a single dataset (DHO dataset), which is more balanced across different labels, to perform multi-label classification. Our experiments have been held on Davidson dataset and the DHO corpora. The later gave the best overall results, especially F1 macro score, even it required more resources (time execution and memory). The experiments have shown good results especially the ensemble models, where stacking gave F1 score of 97% on Davidson dataset and aggregating ensembles 77% on the DHO dataset.**

***Index Terms*—hate speech detection, BERT, deep neural networks, Twitter, ensemble learning.**

I. INTRODUCTION

Twitter is used to disseminate hate speech, especially with the anonymity provided [1]. Thus, any strategy to identify such content is critical in today's world for keeping the internet a safe environment. Detecting online hateful content is the first step in developing a system that flags such items and take right actions. Human annotators are employed by social media corporations to erase these samples and users can flag anything they find harmful to the public. But these procedures are time consuming and depend on human judgment. Thus, automated hate speech detection approaches have been a major concern in this era. To this end, in early research, many attempts were built upon machine learning algorithms and different features extraction techniques, but in recent years, significant performances were obtained by using Transformer-based Language Models [2]. Bidirectional Encoder Representations from Transformers, BERT, has achieved state-of-the-art results in many NLP tasks [3]. Moreover, Neural Network - NN approaches managed to reduce feature engineering and are implemented and coupled with the representation of texts as word vectors through word embedding models. However, machine learning and NN models required to have a larger corpus of datasets to train, but BERT-based models work with a small number of labeled data and sometimes. Hence, to use the advantage of both approaches, we integrate them together to build a deep architecture in order to achieve considerable performances in hate speech classification. In addition, we use Ensemble learning approaches to improve model performances further [4]. In this context, we employed and fine tuned BERT, combined it with deep neural networks and merged obtained models via ensemble learning, in order to detect hate speech in Twitter base data.

The main contributions of this paper are: 1) Presenting transfer learning approach by integrating BERT with Multi Layers Perceptron (MLP), Convolutional Neural Networks (CNN) and Long short-term memory (LSTM) for hate speech detection, in order to explore this mix of models performs better in text classification tasks compared to neural networks or BERT alone. 2) Applying several ensemble learning approaches to improve the performance of these models. 3) Creating new corpora, by fusing multiple public and labeled datasets to get large-size and more balanced corpora. 4) Comparing the integrated models with baseline models and, 5) Comparing model performances through their memory utilization and runtime parameters. We managed to get benefit from the contribution of the combination of BERT and deep neural networks in the text classification task, dealing with the scarcity and the imbalanced data. In addition, we exploited that ensemble learning approaches enhanced our deep models' ability in the classification of hate speech content.

II. LITERATURE SURVEY

Transfer learning: The deep learning models rely on the adoption of Neural Networks -

978-1-6654-3540-6/22

NN coupled with conventional word embedding techniques, which achieved effective performance but often not as efficient as a transformer's [5]. Recent methodologies have gradually changed approaches from RNNs to self-attention and transformers [6] in many NLP tasks. Google's BERT [3] adapted Transformers in 2018, that can condition both left and right context to pretrain deep bidirectional representations from texts. Chiril et al [7] built a BERT-based muti-task hate speech detection method that outperforms a system trained on a single topic-generic dataset. Moreover, Kovács et ai [8] suggested a model of a conjunction of RoBERTa and FastText incorporated with CNNs and RNNs and achieved 63% F1 score. Malik et al. [9] conducted a review of 14 shallow/deep classifiers, driven by a variety of word representation approaches. They resulted that coupling BERT, ELECTRA, and Al-BERT with NNs outperforms other approaches. Their best models were BERT+CNN and ELECTRA+MLP giving F1 macro score of 76% on Davidson [10] dataset. Moreover, Mozafari et al. [11] used BERT-based methods (BERT+Bi-LSTM, and BERT+CNN) to detect hate speech and achieved significant performances.

Ensemble learning: This is a machine learning approach that involves training several models together in a given task. An ensemble is made up of a group of learners known as base/weak learners, trained for the same problem, then integrated to improve results [4].

Badjatiya et al. [12] ensembled Embedding, LSTM and gradient boosted trees to determine whether a tweet is racist, sexist, or neither in a 16k dataset and achieved 93% F1 score. Plaza-del et al. [13] built a vote ensemble classifier including SVM, LR and Decision Tree(DT) to classify hate tweets in English and Spanish and achieved 44% F1 score. Agarwal et al. [14] proposed a Stacking classifier to extract word embedding with RNNs, then, they used SVM, DT, MLP, kNeighbors, ELM, with LR as the meta-classifier and F1 score of 73%. Aljero et al. [15] got F1 score of 97% with their stacking ensemble that combines SVM, LR, and XGBoost.

According to the recent researches that have been held using ensemble learning, we find that the majority of them didn't implement transformers in their experiments (whether for word embedding or as classifiers). Moreover, most of the recent studies were focused on assembling only machine learning classifiers. The classification process was also binary detection of hateful/abusive content.
Based on the state-of-the-art on hate speech detection using transformers-based transfer learning and ensemble learning, in this work we propose several strategies to the same task by integrating best of both approaches.

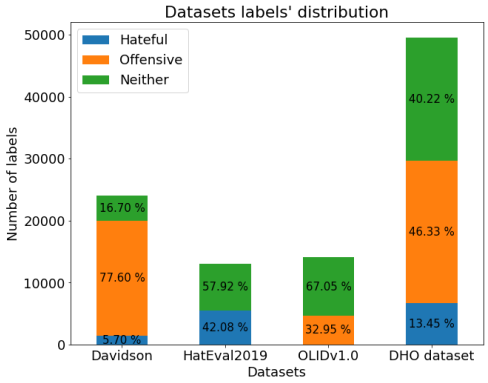

Fig. 1. Datasets labels distribution.

III. Methodology

We aim to build deep learning hate speech detectors by fine tuning and integrating BERT with several NNs, and to enhance hate speech detection performance via ensemble learning. We work on multi-label classification task to distinguish between hateful, offensive and normal content from Twitter.

A. Dataset

Our analyses are based on 3 publicly available datasets: **Davidson** [10], **HatEval** [16], and **OLID** [17]. **1) Davidson:** created by Davidson et al. [10] (2017). It includes 24783 tweets and 3 labels (hateful, offensive, and neither), that were generated using Figure Eight crowdsourcing[1]. These tweets were selected from 85.4M archive tweets, focusing on HateBase keywords (hatebase.org), and annotated by 3 people. For the rest of the paper, we refer this dataset as *Davidson* dataset. **2) HatEval:** contains 13000 tweets about immigrants and women, and generated for the SemEval2019 Task5 [16]. The majority of the tweets came from the AMI corpus[2] and the dataset was labeled via Figure Eight crowdsourcing[1].

The annotators detect if a tweet is hateful, aggressive, and whom it is directed (individual or group). In this paper, we refer this dataset as *HatEval2019*. **3) OLID:** Offensive Language Identification Dataset, created by Zampieri et al. [17] and composed of 14100 tweets. A 3-level hierarchical annotation was used, with the first one determines if a tweet is offensive. In this paper, We refer this dataset as *OLID* dataset. **4) DHO:** generated by merging **D**avidson, **H**atEval2019 and **O**LID datasets to build a large data corpus for our analysis. Data statistics of these corpora are illustrated in Figure 1. Compared to *Davidson*s,

[1]https://appen.com/figure-eight-is-now-appen/
[2]https://groups.inf.ed.ac.uk/ami/corpus/

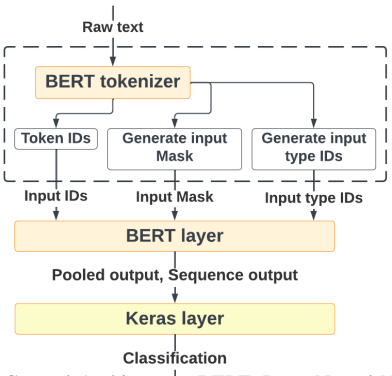

Fig. 2. General Architecture: BERT+Deep Neural Networks.

DHO is more balanced, it is used to demonstrate the resilience and generalization of our models.

B. Deep models

Trained on large amounts of data [3], BERT will certainly lead to excellent performances when using for downstream tasks. It also returns different vectors and contextual embedding for the same word, which extract more information from texts. In the other hand, deep learning networks provide many advantages for NLP, where CNN and RNN are majorly used for text classification. Therefore, we implemented 4 models integrating BERT with other popular NN models such as **MLP**, **CNN** and **LSTM**. At first, we assess the contextual information derived from BERT. We fine-tune them using our datasets to get its contextual representations and then, ensemble models with several ensemble learning techniques: aggregation and stacking, aiming to improve performance and robustness, and to get better classification. The Figure 2 shows the general architecture of our models: Text data needs to be transformed to numeric token ids then arranged in several Tensors before being input to BERT-model, here, TensorFlow Hub provides a matching BERT-preprocessor (tokenizer) for each of the BERT models, which implements this transformation using TF.text library[3]. Our Bert-Model you will return 512 dimension embedding for each token: 'sequence output' and 'Pooled output', which, will be fed into the created NN (Keras layer).

In this work, we used BERT Tensorflow Hub[4] to compute vector-space representations of datasets, to implement 4 different deep models.

1) BERT baseline: (Figure 3.a) We used BERT uncased L-4 H-512 A-8 model: 4 hidden layers (L), 512 hidden size (output size of 512 dimensions) (H). We took the pooled output and integrated a dropout and a dense layer (Softmax activation).

[3]https://www.tensorflow.org/text/tutorials/classify_text_with_bert

[4]https://tfhub.dev/google/collections/bert/1

2) BERT+MLP: (Figure 3.b) MLP is a type of traditional neural networks that are highly adaptable. They consist in 3+ layers of neurons. 2 dense and one dropout layers are added to BERT's pooled output. **3) BERT+LSTM** (Figure 3.c): LSTM is a RNN that update hidden layers using memory cells, and appear to be effective in sequential learning long-range text dependencies. We integrated into BERT's sequence output, 2 LSTM layers (512 units) followed by a dropout and a dense layer. **4) BERT+CNN** (Figure 3.d): CNNs are deeper and sparsely connected, allowing to efficiently find patterns in noisy texts. Here, 2 CNN (conv1D with ReLu activation), a Global Max Pooling and 2 dense layers are integrated into BERT's sequence output.

C. Ensemble Learning

Since Neural networks are nonlinear, they have high variance and low bias, being very sensitive to noisy data. Thus, there is no guarantee to exhibit low generalization error when predicting. Hence, we implemented ensemble learning to improve model performance by reducing these issues' effects. We used 4 ensembling methods combining BERT+MLP, BERT+CNN and BERT+LSTM, with stacking and aggregation: **1) Soft Voting** Or averaging, merges several fine tuned models trained on the same dataset. We took the average of predicted class probabilities of each individual classifier C_j and then, used argmax to obtain the final class as shown in equation 1. The predicted probabilities were treated equally ($\omega_j = 1$ for each jth model).

$$\hat{y} = argmax_{i \in \{1,...c\} j \in \{1,...m\}} \omega_j \hat{p}_{ij} \qquad (1)$$

i is the class value in data and the class probability $\hat{p}$ is $\forall i$, $\hat{p}\,(y = i) = \sum_{j=1}^{m} \frac{\hat{p}(y=i|C_j)}{m}$.

2) Maximum voting Consider the maximum prediction probability from the models. We have $C(x) = \hat{p}$: the probability distribution over the c classes y where:
$\hat{p} = \{\hat{p}(y = 0|C), \hat{p}(y = 1|C), ...\hat{p}(y = m|C)\}$. Thus, the max value ensemble result is the class with the maximum probability among the classifiers, as illustrated in the equation 2.

$$\hat{y} = argmax_{i \in \{1,...c\}} \hat{p}\,(y = i|C) \qquad (2)$$

3) Hard voting Employs the principle of majority voting (of an odd number of classifiers), it takes the predictions of each model and output the most frequent class. We predict the class $\hat{y}$ via majority vote of each of the m classifiers as shown in equation 3.

$$\hat{y} = mode\,\{C_1, C_2, ...C_m\}\,. \qquad (3)$$

4) Stacked Generalization ensemble Or stacking, is an integration strategy [18]. It

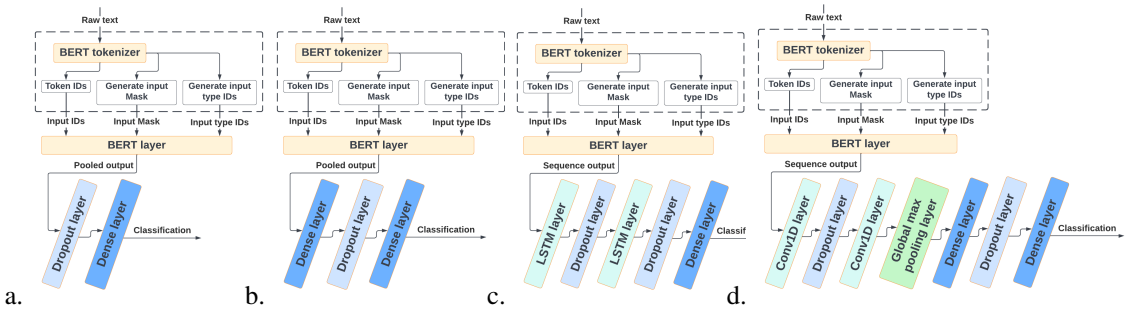

Fig. 3. The architectures of integrated deep models: (a.) BERT baseline, (b.) BERT+MLP, (c.) BERT+LSTM, (d.) BERT+CNN

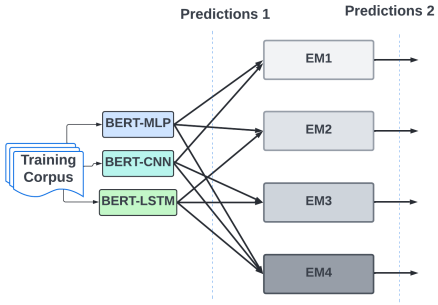

Fig. 4. Different **E**nsemble learning **M**odels (**EM**): **EM1**: BERT-MLP + BERT-CNN, **EM2:** BERT-MLP + BERT-LSTM **EM3:**, BERT-CNN + BERT+LSTM **EM4:** and BERT-MLP + BERT-CNN + BERT-LSTM

frequently combines heterogeneous trained base learners by training a *meta-model* to output a prediction based on the predictions of the base models. We have implemented the Stratified k-fold cross-validation technique to partition the training set between the models, keeping the same ratio of 10% for the validation set. Given the training set $D = \{D_1, D_2, D_3\}$ where D_j and $\bar{D_j} = D \backslash D_j$ are respectively the training set and its corresponding test set, each base t^{th} learner $h_t^{(j)}$ is trained on a D_j, then, their predictions z (equation 4) are fed as training set into the meta learner (Linear Regression classifier in our case) which will predict the final predictions.

$$z = stack\left\{h_t\left(D\right)\right\}. \tag{4}$$

All ensemble models are depicted in Figure 4 which presents the architecture of the integration of multiple deep learning models to form 10 different ensemble models. These ensembles have 2 prediction levels: *Predictions1*: the predictions of each model and *Predictions2*: the output after ensembling them using different scenarios. EM1 is the ensemble of BERT-MLP and BERT-CNN, based on these 2 levels. As Figure 4 depicts, EM2, EM3 and EM4 are implemented by the same technique.

IV. Experiments and Results

A. Data Pre-processing

We started pre-processing our raw data: 1) Switch tweets to lower case, 2) Delete URLs, 3) Remove users names, 4) Shorten prolonged words (”yeeessss” to ”yes”...), 5) Keep stop words, 6) Remove punctuation marks, unknown uni-codes, and additional delimiting characters, 7) Remove hashtags (#) and correct their texts (e.g, ”#notracism” to ”not racism”), 8) Eliminate tweets of length less than 2, and 9) Remove emoticons.

B. Data analysis platform and evaluation metrics

The implemented models are trained with various fine-tuning strategies (batch size of 32 for 50 epochs) on Colab. We created our custom optimizer with a learning rate of 2e-5, and experimented with AdamW optimizer and Sparse Categorical Cross-entropy loss functions. In order to utilize features from each label, we introduced weights in the training phase. Thus, classifiers performances are measured via macro averaged F1 score, accuracy, precision and recall scores.

C. Results and Interpretations

The results are shown in Table 1 and the best results of each group of the same ensemble and are highlighted in blue and the best overall ones are in red. We set up early stopping with Validation Accuracy as a monitor parameter to prevent overfitting. Following experiments are mainly based on *Davidson* dataset and *DHO* dataset. **Davidson dataset:** As illustrated in Table 1, BERT+LSTM has shown the best results on the test set. Moreover, all the models outperform BERT baseline, which prove the importance of adding NN classifiers for such a task. As for the aggregation ensembles, all the approaches outperformed single models, it shows obviously better results, especially the Soft Voting of BERT+LSTM with BERT+CNN, as well as Hard Voting ensembling (in bold) that outperformed both of the other aggregation ensembles. **DHO dataset:** Models trained and tested on this dataset gave better performance than on Davidson's. Getting the most performed model BERT+MLP. Moreover, aggregation ensembles outperformed each of these single models, getting the best result when ensembling the 2 most performed models: BERT+MLP and BERT+LSTM. All of the 3 ensemble approaches have given almost similar

			Davidson Dataset			Merged Dataset		
		Model	Accuracy	F1 Score	Precision	Accuracy	F1 Score	Precision
		BERT (baseline)	0.8498	0.7209	0.6894	0.79164	0.7571	0.7466
		BERT-MLP	0.876	0.7401	0.7209	**0.8087**	**0.7704**	**0.764**
		BERT-CNN	0.8943	0.7355	0.7379	0.7853	0.74902	0.7391
		BERT-LSTM	**0.8968**	**0.7548**	**0.7397**	0.8013	0.762	0.7559
Ensembler Method	**Soft voting**	EM1	0.9005	0.7558	0.7468	0.8045	0.7672	0.7586
		EM2	0.8943	0.7554	0.7354	**0.8158**	**0.7783**	**0.7724**
		EM3	**0.9047**	**0.7596**	**0.7519**	0.8091	0.7734	0.7641
		EM4	0.9018	0.7589	0.7469	0.812	0.7746	0.7669
	Max value	EM1	0.9009	0.7564	0.747	0.8077	0.7715	0.7622
		EM2	0.8939	0.7549	0.7348	**0.8162**	**0.7786**	**0.7726**
		EM3	**0.9034**	0.7566	**0.7494**	0.8077	0.7715	0.7622
		EM4	0.9026	**0.7588**	0.748	0.814	0.7773	0.769
	Hard voting	EM4	0.9001	**0.7595**	0.7457	0.8085	0.771	0.7628
	Stacking	EM4	0.776	**0.9706**	**0.9430**	0.463	**0.9278**	**0.8654**

TABLE I
EXPERIMENTAL RESULTS FOR THE DAVIDSON DATASET AND DHO DATASET.

results, but we can observe that Maximum value and Hard Voting are the best ones. Unlike BERT-CNN, BERT-MLP and BERT-LSTM gave the best performance on DHO and Davidson datasets respectively, this is related to the type of corpora and the subject we are dealing with: NLP Twitter text classification with small imbalanced dataset. As for the stacking approach, it outperformed all the other ensembles in both datasets, giving highest F1 and precision, even with the lower accuracy, which is not always guaranteed in this ensemble and unexpectedly, Davidson accuracy results were better then DHO's. We refer these results to the meta classifier used (Linear Regression), which is not a complex multi layered model that can correctly deal with the used datasets. And, as David H. et al. stated [18], for almost any real-world generalization problem one should use some version of stacked generalization to minimize the generalization error rate. Moreover, the main reason for getting lower accuracy is not easy to be interpretable since BERT (as any other transformer) is a black box, so it is not easy to understand its functioning. Meanwhile, we refer also these results to the imbalanced corpora used in our experiments. Figure 5 shows the confusion matrix to present clearly the ensembles' classification performance of the best performed models: soft voting EM3 for Davidson, and Max value EM2 for DHO. The classification performance of both models is very good for each class, especially for 'Offensive' and 'Neither' (Highest True Positives) since they represent the biggest percentages of each dataset. Moreover, 'Hateful' class classification error rate is decreased in DHO because this corpora is more balanced.

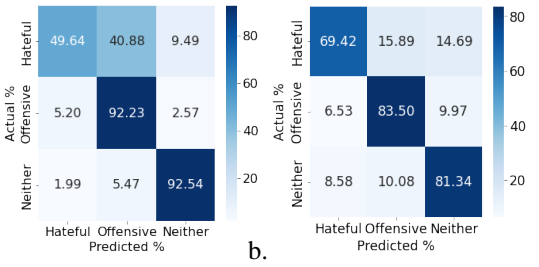

Fig. 5. Confusion matrix of ensembles: (a.) Soft voting EM3 - Davidson, (b.) Max value EM2 - DHO

V. DISCUSSIONS AND FUTURE WORK

A. Discussions and Challenges

Data: As shown in the section above, we worked on a small-size imbalanced datasets, where there is a huge gap between class distribution (in Davidson dataset 'Hateful' label takes 5.77% while 'Offensive' takes 77.34% of all data) Figure 1. This led to many issues we faced during training such as the overfitting problems. **Computational power:** Our proposed approaches require a strong GPU, and even with the use of Google Colab Pro for training, we were restricted to some limitations (Number of BERT and deep learning layers). We ended up getting a 28,765,188-parameter BERT baseline, 29,027,844-parameter BERT-MLP, 28,835,428-parameter BERT-CNN and 32,963,588-parameter BERT-LSTM, which, increases the models' complexity, thus, required more computational resources as displayed in Figure 6.

Obviously, DHO require more resources than Davidson. And even getting less number of parameters, training BERT-CNN took more memory size than BERT-MLP. Although requiring more memory, the ensembles were very faster than baselines' training, which explains their efficiency. The best least resouces consuming models are EM1 and EM3 for both datasets. **Multi-label classification:** We worked on multi-label datasets, which add more complexity to the classifications. When compared with the previous works, our results didn't often overcome the binary classifications on the same datasets. **Stacking ensemble:** Dealing with BERT+NN, we couldn't implement directly the pre-defined stacking ensemble functions of Sklearn library. Thus,

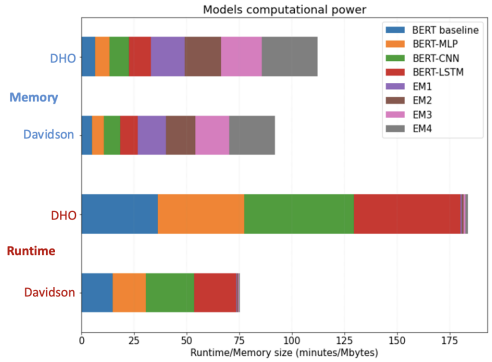

Fig. 6. Runtime and memory models.

we manually elaborated the required functions (Defining: stacked dataset got from the predictions of base learners, the meta learner and, its stacked final predictions).

B. Future work

Various improvements can be added to our study in the future to enhance hate speech detection approach with ensemble transfer learning. It starts with the data where we can work more on the labels' distribution. Taking into consideration balance status, we can use several techniques like data augmentation. Dealing with the dataset bias issue and the granularity of hate speech content, we can get benefit further from K-BERT (using Knowledge Graph) [19], it helps also to handle the issue of the data scarcity, especially in other languages, thus, detecting hateful and offensive content from unlabeled corpora. Adding to that, we can check our models' generalization by testing them on other datasets, from different social media resources like Facebook etc. As for the models' implementation, we can use TensorFlow Hub BERT architecture similar to BERT base Hugging Face model's (L=12, H=768 and A=12), or at least, increase the number of parameters and test their impact on models' performances. Moreover, we can employ other ensemble learning techniques and enhance the stacking ensemble we built, we can also improve the baseline models and build new and more complex deep learning classifiers and combine them with other transformers than BERT and compare their effects.

VI. Conclusion

For the purpose of hate speech detection on social media, this paper worked on several public datasets to build large more balanced corpora used to train and test transfer learning classifiers. Combining BERT transformer with several deep neural networks, we managed to get heterogeneous multi-label classifiers that successfully detect hateful and offensive content from Twitter. We, then , improved their performance using different aggregating and stacking ensemble techniques, the latter significantly outperformed the baselines and the other ensembles' predictions even with giving low accuracy, which is a future subject to work on, where we aim to develop more powerful and resilient stacking meta-classifiers.

References

[1] D. L. Ascher, "Unmasking hate on twitter: Disrupting anonymity by tracking trolls," 2019.

[2] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz *et al.*, "Transformers: State-of-the-art natural language processing," pp. 38–45, 2020.

[3] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[4] Z.-H. Zhou, "Ensemble learning," in *Machine learning*. Springer, 2021, pp. 181–210.

[5] Y. Tay, M. Dehghani, D. Bahri, and D. Metzler, "Efficient transformers: A survey," *ACM Computing Surveys (CSUR)*, 2020.

[6] B. Yang, L. Wang, D. Wong, L. S. Chao, and Z. Tu, "Convolutional self-attention networks," *arXiv preprint arXiv:1904.03107*, 2019.

[7] P. Chiril, E. W. Pamungkas, F. Benamara, V. Moriceau, and V. Patti, "Emotionally informed hate speech detection: a multi-target perspective," *Cognitive Computation*, vol. 14, no. 1, pp. 322–352, 2022.

[8] G. Kovács, P. Alonso, and R. Saini, "Challenges of hate speech detection in social media," *SN Computer Science*, vol. 2, no. 2, pp. 1–15, 2021.

[9] J. S. Malik, G. Pang, and A. v. d. Hengel, "Deep learning for hate speech detection: A comparative study," *arXiv preprint arXiv:2202.09517*, 2022.

[10] T. Davidson, D. Warmsley, M. Macy, and I. Weber, "Automated hate speech detection and the problem of offensive language," vol. 11, no. 1, pp. 512–515, 2017.

[11] M. Mozafari, R. Farahbakhsh, and N. Crespi, "Hate speech detection and racial bias mitigation in social media based on bert model," *PloS one*, vol. 15, no. 8, p. e0237861, 2020.

[12] P. Badjatiya, S. Gupta, M. Gupta, and V. Varma, "Deep learning for hate speech detection in tweets," pp. 759–760, 2017.

[13] F. M. Plaza-del Arco, M. D. Molina-González, M. T. Martín-Valdivia, and L. A. U. Lopez, "Sinai at semeval-2019 task 5: Ensemble learning to detect hate speech against inmigrants and women in english and spanish tweets," pp. 476–479, 2019.

[14] S. Agarwal and C. R. Chowdary, "Combating hate speech using an adaptive ensemble learning model with a case study on covid-19," *Expert Systems with Applications*, vol. 185, p. 115632, 2021.

[15] M. K. A. Aljero and N. Dimililer, "A novel stacked ensemble for hate speech recognition," *Applied Sciences*, vol. 11, no. 24, p. 11684, 2021.

[16] V. Basile, C. Bosco, E. Fersini, D. Nozza, V. Patti, F. M. R. Pardo, P. Rosso, and M. Sanguinetti, "Semeval-2019 task 5: Multilingual detection of hate speech against immigrants and women in twitter," pp. 54–63, 2019.

[17] M. Zampieri, S. Malmasi, P. Nakov, S. Rosenthal, N. Farra, and R. Kumar, "Predicting the type and target of offensive posts in social media," *arXiv preprint arXiv:1902.09666*, 2019.

[18] D. H. Wolpert, "Stacked generalization," *Neural networks*, vol. 5, no. 2, pp. 241–259, 1992.

[19] W. Liu, P. Zhou, Z. Zhao, Z. Wang, Q. Ju, H. Deng, and P. Wang, "K-bert: Enabling language representation with knowledge graph," vol. 34, no. 03, pp. 2901–2908, 2020.