Combining Textual Features for the Detection of Hateful and Offensive Language

Sherzod Hakimov[1,2], Ralph Ewerth[1,2]

[1] TIB - Leibniz Information Centre for Science and Technology, Hannover, Germany
[2] Leibniz University Hannover, L3S Research Center, Hannover, Germany

Abstract

The detection of offensive, hateful and profane language has become a critical challenge since many users in social networks are exposed to cyberbullying activities on a daily basis. In this paper, we present an analysis of combining different textual features for the detection of hateful or offensive posts on Twitter. We provide a detailed experimental evaluation to understand the impact of each building block in a neural network architecture. The proposed architecture is evaluated on the *English Subtask 1A: Identifying Hate, offensive and profane content from the post datasets* of *HASOC-2021* dataset under the team name *TIB-VA*. We compared different variants of the contextual word embeddings combined with the character level embeddings and the encoding of collected hate terms.

Keywords

hate speech detection, offensive language detection, abusive language detection, social media mining

1. Introduction

The detection of hateful, offensive and profane language has become a significant research challenge with the widespread usage of social media. Certain groups of people become targets of cyberbullying activities on a daily basis on many social networks such as Facebook, Twitter, or Instagram [1]. There have been many efforts by the research community and social media companies such as Facebook[1] and Twitter[2] to the scope of hate speech and tackle the problem. In general, hate speech is defined as *a language used to express hatred towards a targeted group or individuals based on specific characteristics such as religion, ethnicity, origin, sexual orientation, gender, physical appearance, disability or disease* [2, 3, 4, 5, 6, 7, 8].

In this paper, we analyze the effects of combining multiple textual features to detect hateful, offensive or profane language expressed in the tweet text. We evaluated our approach on the *Hate Speech and Offensive Content Identification in English and Indo-Aryan Languages* (HASOC) challenge datasets[3]. We submitted our solution to the *English Subtask 1A: Identifying Hate, offensive and profane content from the post* [9] of the HASOC-2021 [10] challenge series. The task involves classifying a given tweet text whether the content is hateful, offensive, or profane

Forum for Information Retrieval Evaluation, December 13-17, 2021, India

sherzod.hakimov@tib.eu (S. Hakimov); ralph.ewerth@tib.eu (R. Ewerth)
0000-0002-7421-6213 (S. Hakimov); 0000-0003-0918-6297 (R. Ewerth)

[1] https://www.facebook.com/communitystandards/hate_speech

[2] https://help.twitter.com/en/rules-and-policies/hateful-conduct-policy

[3] https://hasocfire.github.io

language or not. We proposed a combination of multiple textual features based on neural network architecture and evaluated different configurations. Our experimental evaluation is performed on all three datasets: HASOC-2019 [11], HASOC-2020 [12], HASOC-2021 [10].

The remainder of the paper is structured as follows. In Section 2, we describe the proposed model architecture. In Section 3, the experimental setup, challenge datasets, as well as evaluations of model architectures are described in detail. Finally, the Section 4 concludes the paper.

2. Approach

Our model architecture is built on top of three textual features that are combined to predict whether a given text contains hateful, offensive or profane language. The neural network architecture is shown in Figure 1. Input tokens are fed into *BERT*, *Character* and *Hate Words* encoders to extract feature-specific vector representations. Once each feature representation is extracted, the outputs are fed into separate components to obtain one-dimensional vector representations. These vectors are concatenated and fed into three different blocks to obtain binary class probabilities. Each block is composed of a linear layer, batch normalization and a *ReLU* activation function. The source code and the resources described below are shared publicly with the community[4]. Next, we describe the textual encoders in detail.

BERT Encoder: We used a pre-trained BERT [13] model to obtain contextual 768-dimensional word vectors for each input token.

Character Encoder: Each input token is converted into vector representation based on the one-hot encoding of characters in English. We only use letters (a-z) to obtain a sequence of character-level vectors.

Hate Words Encoder: We collected a list of hate terms by combining the dictionary provided by Gomez et al. [14] with additional online dictionaries[5]. We manually filtered out terms that do not express hate concepts and obtained a list of 1493 hate terms. The list contains a variety of terms with different lexical variations to increase the coverage of detecting such terms in tweets, e.g., *bitc**, *border jumper*, *nig***, or *chin**. This encoder outputs a 1493-dimensional vector, a multi-hot encoding of hate terms in input tokens.

3. Experimental Setup and Results

In this section, we present the challenge datasets, details about data preprocessing and model parameters, and finally explain the experimental setup along with the obtained results.

3.1. Datasets

Our model architecture is built for the HASOC-2021 [10] *English Subtask 1A: Identifying Hate, offensive and profane content from the post* [9]. In Table 1, we provide the number of data points for HASOC-2019[11], HASOC-2020 [12], and HASOC-2021 [10] editions of the challenge. The

[4] https://github.com/sherzod-hakimov/HASOC-2021---Hate-Speech-Detection

[5] https://www.noswearing.com/dictionary & https://hatebase.org/

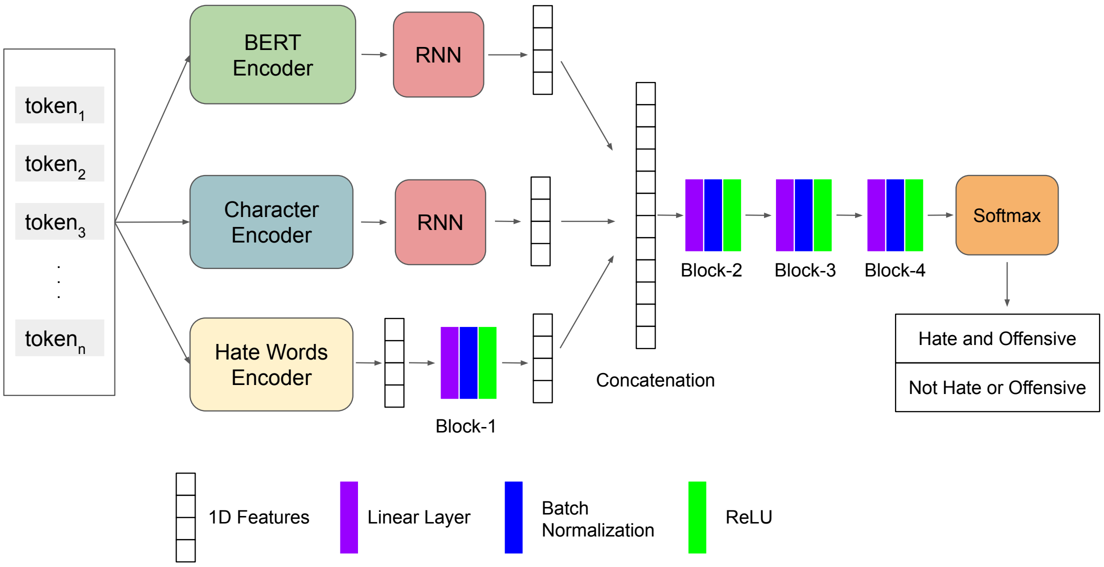


Figure 1: The model architecture combines character, hate words, and BERT embeddings that outputs probability of a given text being hate and offensive or not.

datasets include tweet text as input data and two target labels: *Hate and Offensive (HOF)* and *Not Hate or Offensive (NOT)*.

The number of training samples for the 2019 and 2021 editions are not equally distributed among the two classes. To overcome the class imbalance issue, we applied the oversampling method to the training splits. We randomly selected a certain number of data points for the minority class (HOF for 2019, NOT for 2021) and duplicated them to equalize with the number of data points for the majority class.

Table 1
Distribution of data points for train and test splits for *English Subtask 1A* for all editions of the HASOC datasets. **HOF**: Hate and Offensive, **NOT**: Not Hate or Offensive

Dataset	**Train**		**Test**	
	HOF	NOT	HOF	NOT
HASOC-2019 [11]	2261	3591	288	865
HASOC-2020 [12]	1856	1852	807	785
HASOC-2021 [10]	2501	1342	765	456

3.2. Data Preprocessing

There are several challenges with working text from Twitter. In many cases, the tokens are written in different forms to save space, capitalized, mixed with numbers etc. We apply the following text preprocessing steps to normalize the tweet text: 1) remove hashtags, URLs, user tags, retweets tags using Ekphrasis, 2) remove punctuations, 3) convert tokens into lowercase.

3.3. Model Parameters

In this section, we provide details about all parameters of the buildings blocks in the model architecture show in Figure 1.

BERT Encoder: We experiment with two different variants of BERT models. The first variant is *BERT-base*, which is the default model provided by Devlin et al. [13]. The second variant is *HateBERT* provided by Caselli et al. [15], which is a *BERT-base* model pre-trained further on hateful comments corpus extracted from Reddit. Both variants output a sequence of 768-dimensional vectors for the given input tokens.

Recurrent Neural Network (RNN) Layers: We experimented with different types of RNN layers: Long-short Term Memory (LSTM), Gated Recurrent Unit (GRU), and Bidirectional Gated Recurrent Unit (Bi-GRU). We also experimented with different layer sizes, which are 100, 200, 300.

Linear Layers: The model architecture includes four blocks that are composed of three consecutive layers: linear layer, batch normalization, and activation function (ReLU). The sizes of the linear layers in the *Block-1*, *Block-2*, *Block-3*, *Block-4* are 512, 512, 256, 128 respectively.

Training Process: Each configuration of the model architecture is trained using Adam optimizer [16] with a learning rate of 0.001, a batch size of 64 for maximum of 20 iterations. We use the 90:10 splits for train and validation splits to find the optimal hyperparameters.

Implementation: The model architecture is implemented in Python using the Tensorflow Keras library. The source code is shared openly with the community[6].

3.4. Results

We tested different model configurations as explained above for all three datasets. The results are given in Table 2. The official evaluation metrics for the *English Subtask 1A* [9] is Macro F1-score. Additionally, we included accuracy and weighted F1-score since the number of data points for each class are not balanced for test splits of the datasets (see Table 1). We included the best performing models with the corresponding features. Based on the initial experiments, the choice of Gated Recurrent Unit (GRU) with the layer size of 100 yielded the highest performance on the validation set for three datasets in comparison to other model configurations. Therefore, all model configurations listed in the table below use the *GRU* layer with the size *100*.

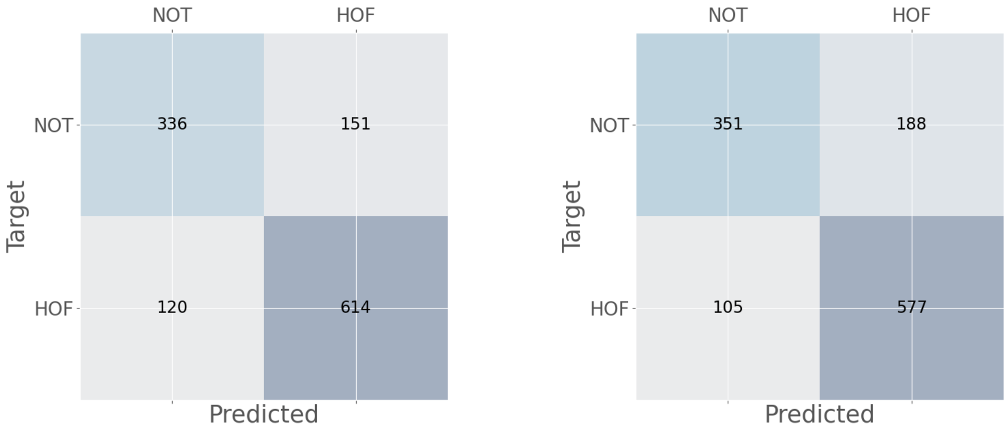
The results suggest that *BERT-base* embeddings have greater a impact than *HateBERT* embeddings. Another important observation is that the feature based on the multi-hot encoding of hate terms (HW) achieves high accuracy and weighted F1-score for all datasets. Specifically, every model configuration that included the HW feature yields the best results on the *HASOC-2020* dataset. Our approach that combines BERT-base, character embeddings, and multi-hot encoding of hate terms achieved a Macro F1-score of 0.77 on *English Subtask 1A* [9] of the HASOC-2021 [10] dataset. We submitted the same model as a team *TIB-VA* to the official challenge. Our model was ranked at the position 33 with the Macro-F1 score of 0.76.

[6] https://github.com/sherzod-hakimov/HASOC-2021---Hate-Speech-Detection

Table 2
Evaluation results of various model configurations on *English Subtask 1A* of three datasets. The evaluation metrics are accuracy (Acc), Macro F1-score (M-F1), and weighted F1-score (W-F1). The best performing model configurations for each dataset are highlighted in bold. **BB**: word embeddings extracted from a pre-trained *BERT-base* [13] model, **HB**: word embeddings extracted from a pre-trained *HateBERT* [15] model, **CH**: character level embeddings, **HW**: multi-hot encoding of hate words.

	HASOC-2019			**HASOC-2020**			**HASOC-2021**		
Features	**Acc**	**M-F1**	**W-F1**	**Acc**	**M-F1**	**W-F1**	**Acc**	**M-F1**	**W-F1**
BB	**0.78**	**0.70**	**0.78**	0.86	0.86	0.86	0.74	0.73	0.73
HB	0.64	0.61	0.61	0.84	0.84	0.84	0.74	0.73	0.74
CH	0.36	0.36	0.36	0.56	0.51	0.60	0.63	0.59	0.64
HW	0.75	0.61	0.77	**0.89**	**0.89**	**0.89**	0.71	0.71	0.71
CH + HW	0.76	0.59	0.80	**0.89**	**0.89**	**0.89**	0.71	0.71	0.71
BB + HW	0.78	0.68	0.80	0.88	0.88	0.88	0.78	0.76	0.78
BB + CH	0.77	0.70	0.77	0.81	0.81	0.81	0.73	0.72	0.73
HB + HW	0.71	0.65	0.70	**0.89**	**0.89**	**0.89**	0.78	0.77	0.78
HB + CH	0.58	0.56	0.55	0.87	0.87	0.87	0.73	0.71	0.73
BB + CH + HW	0.75	0.69	0.75	**0.89**	**0.89**	**0.89**	**0.79**	**0.77**	**0.79**
HB + CH + HW	0.75	0.69	0.75	0.49	0.33	0.66	0.78	0.76	0.78

We present the confusion matrices in Figure 2 for two models with different pre-trained variants of BERT models: *BERT-base* (BB) and *HateBERT* (HB). Both models were trained with character level embeddings (CH) and multi-hot encoded hate words (HW). We can observe that the model using *BERT-base* embeddings (Figure 2a) makes more correct predictions (614 vs. 577) in detecting hateful content (HOF) when compared with the other model variant (Figure 2b). A similar pattern exists for cases where the target class is NOT, and the model predicts HOF where the model with the BB feature makes fewer mistakes (151 vs. 188) than the other model.



(a) Model with features: BB + CH + HW

(b) Model with features: HB + CH + HW

Figure 2: Confusion matrices for two models evaluated on the *HASOC 2021 - English Subtask 1A*

4. Conclusion

In this paper, we have analyzed model architectures that combine multiple textual features to detect hateful, offensive and profane language. Our experimental results showed that simply using the multi-hot encoding of collected 1493 hate terms yields significant performance. The combination of BERT embeddings, character embeddings, and features based on hate terms achieved the best performance on the *English Subtask 1A*, HASOC 2021 dataset. Another observation of the evaluation is that a variant of the BERT model trained on domain-specific (*HateBERT*) text did not improve the results in comparison to the default pre-trained model variant (*BERT-base*).

References

[1] R. M. Kowalski, G. W. Giumetti, A. N. Schroeder, M. R. Lattanner, Bullying in the digital age: A critical review and meta-analysis of cyberbullying research among youth., Psychological bulletin 140 (2014) 1073.

[2] P. Badjatiya, S. Gupta, M. Gupta, V. Varma, Deep learning for hate speech detection in tweets, in: R. Barrett, R. Cummings, E. Agichtein, E. Gabrilovich (Eds.), Proceedings of the 26th International Conference on World Wide Web Companion, Perth, Australia, April 3-7, 2017, ACM, 2017, pp. 759–760.

[3] T. Davidson, D. Warmsley, M. W. Macy, I. Weber, Automated hate speech detection and the problem of offensive language, in: Proceedings of the Eleventh International Conference on Web and Social Media, ICWSM 2017, Montréal, Québec, Canada, May 15-18, 2017, AAAI Press, 2017, pp. 512–515.

[4] P. Fortuna, S. Nunes, A survey on automatic detection of hate speech in text, ACM Comput. Surv. 51 (2018) 85:1–85:30. doi:10.1145/3232676.

[5] W. Warner, J. Hirschberg, Detecting hate speech on the world wide web, in: Proceedings of the second workshop on language in social media, 2012, pp. 19–26.

[6] F. Poletto, V. Basile, M. Sanguinetti, C. Bosco, V. Patti, Resources and benchmark corpora for hate speech detection: a systematic review, Lang. Resour. Evaluation 55 (2021) 477–523. URL: https://doi.org/10.1007/s10579-020-09502-8. doi:10.1007/s10579-020-09502-8.

[7] Z. Zhang, L. Luo, Hate speech detection: A solved problem? the challenging case of long tail on twitter, Semantic Web 10 (2019) 925–945. URL: https://doi.org/10.3233/SW-180338. doi:10.3233/SW-180338.

[8] P. Zeinert, N. Inie, L. Derczynski, Annotating online misogyny, in: C. Zong, F. Xia, W. Li, R. Navigli (Eds.), Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021, Association for Computational Linguistics, 2021, pp. 3181–3197. URL: https://doi.org/10.18653/v1/2021.acl-long.247. doi:10.18653/v1/2021.acl-long.247.

[9] T. Mandl, S. Modha, G. K. Shahi, H. Madhu, S. Satapara, P. Majumder, J. Schäfer, T. Ranasinghe, M. Zampieri, D. Nandini, A. K. Jaiswal, Overview of the HASOC subtrack at FIRE 2021: Hate Speech and Offensive Content Identification in English and Indo-Aryan Lan-

guages, in: Working Notes of FIRE 2021 - Forum for Information Retrieval Evaluation, CEUR, 2021. URL: http://ceur-ws.org/.

[10] S. Modha, T. Mandl, G. K. Shahi, H. Madhu, S. Satapara, T. Ranasinghe, M. Zampieri, Overview of the HASOC Subtrack at FIRE 2021: Hate Speech and Offensive Content Identification in English and Indo-Aryan Languages and Conversational Hate Speech, in: FIRE 2021: Forum for Information Retrieval Evaluation, Virtual Event, 13th-17th December 2021, ACM, 2021.

[11] T. Mandl, S. Modha, P. Majumder, D. Patel, M. Dave, C. Mandalia, A. Patel, Overview of the HASOC track at FIRE 2019: Hate speech and offensive content identification in indo-european languages, in: P. Majumder, M. Mitra, S. Gangopadhyay, P. Mehta (Eds.), FIRE '19: Forum for Information Retrieval Evaluation, Kolkata, India, December, 2019, ACM, 2019, pp. 14–17. URL: https://doi.org/10.1145/3368567.3368584. doi:10.1145/3368567.3368584.

[12] T. Mandl, S. Modha, A. K. M, B. R. Chakravarthi, Overview of the HASOC track at FIRE 2020: Hate speech and offensive language identification in tamil, malayalam, hindi, english and german, in: P. Majumder, M. Mitra, S. Gangopadhyay, P. Mehta (Eds.), FIRE 2020: Forum for Information Retrieval Evaluation, Hyderabad, India, December 16-20, 2020, ACM, 2020, pp. 29–32. URL: https://doi.org/10.1145/3441501.3441517. doi:10.1145/3441501.3441517.

[13] J. Devlin, M. Chang, K. Lee, K. Toutanova, BERT: pre-training of deep bidirectional transformers for language understanding, in: J. Burstein, C. Doran, T. Solorio (Eds.), 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), Association for Computational Linguistics, 2019, pp. 4171–4186.

[14] R. Gomez, J. Gibert, L. Gómez, D. Karatzas, Exploring hate speech detection in multimodal publications, in: IEEE Winter Conference on Applications of Computer Vision, WACV 2020, Snowmass Village, CO, USA, March 1-5, 2020, IEEE, 2020, pp. 1459–1467. URL: https://doi.org/10.1109/WACV45572.2020.9093414. doi:10.1109/WACV45572.2020.9093414.

[15] T. Caselli, V. Basile, J. Mitrovic, M. Granitzer, Hatebert: Retraining BERT for abusive language detection in english, CoRR abs/2010.12472 (2020). URL: https://arxiv.org/abs/2010.12472. arXiv:2010.12472.

[16] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, in: Y. Bengio, Y. LeCun (Eds.), 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings, 2015. URL: http://arxiv.org/abs/1412.6980.