Team FalsePostive at HASOC 2019: Transfer-Learning for Detection and Classification of Hate Speech

Kaushik Amar Das[1][0000−0002−6314−2730] and Ferdous Ahmed Barbhuiya[1]

[1]Indian Institute of Information Technology, Guwahati
Bongora, Assam, India
kaushikamardas@tuta.io
ferdousa@gmail.com

Abstract. This paper presents the results obtained by using a convolutional neural network (CNN) and bi-directional long short-term memory (BiLSTM) network on the three different datasets provided in HASOC 2019. The neural networks presented in this paper were first trained to classify whether the document provided in the dataset is hate speech or not. And then, the networks were fine-tuned for the subsequent subtasks of fine-grained classification of hate speech and finding out the type of offense.

Keywords: Transfer learning · Neural network · Hate speech

1 Introduction

Social media platforms such as Facebook and Twitter have increased in popularity over the recent decades and only continue to rise in popularity. While such platforms are used by people for constructive purposes, they have also become a convenient medium for dissemination of hate speech. As such, there is a dire need to possibly limit and curb the spread of hate speech in online social media platforms [2].

According to Davidson et al. [6] hate speech is *language that is used to expresses hatred towards a targeted group or is intended to be derogatory, to humiliate, or to insult the members of the group.* In their work, they conclude that classifying hate speech is a difficult task, as we tend to classify it based on our own subjective biases.

The HASOC[1] 2019 [10] event presents the shared task of classifying hate speech. The shared task is further divided into three subtask A, B and C. Subtask A comprised of classifying the document into two categories, Hate and Offensive

[1] Hate Speech and Offensive Content Identification in Indo-European Languages (2019) Workshop

and Non - Hate and offensive. The items classified as hateful and offensive in subtask A are to be further classified into three fine-grained classes namely Hate speech, Offensive, Profane for subtask B. In subtask C, we further classify the hateful and offensive items of subtask A into two types of offenses, Targeted Insult and Untargeted. For the above tasks, the organizers provided datasets created from Facebook and Twitter posts. The datasets were provided in three different languages, English, German and code-mixed Hindi. The organizer define not hateful or offensive (NOT) as posts that do not contain any hate speech or offensive content. Hate and Offensive (HOF) as posts that contain hate, offensive and profane content. Hate speech (HATE) as posts that contain hate speech. Profane(PRFN) when post contain profane words. When posts contain threats contain insult/threat to an individual, group or others, it is labelled Targeted (TIN). And when the post contains general profanity that are not targeted but contain non-acceptable language, it is labelled Untargeted (UNT).

2 Related Work

Baruah et al. [1] trained BiLSTM models with and without attention to detect hate speech against immigrants and women on twitter. This task was a part of the SemEval 2019 workshop [2]. They found that BiLSTM without attention performed better than the one with attention. Indurthi et al. [8] evaluated the performance of various sentence level embeddings for the same task. They trained various simple machine learning models on these embeddings and found that Google's Universal Sentence Encoder [4] coupled with SVM (with RBF Kernel) outperformed all other models for the task. For the same task Ding et al. [7] used a capsule network on top of a stacked BiGRU network. They used the word level embeddings provided by FastText [9] to first convert the words into vector representations. Nobata et al. [11] used a regression model and studied the performance of different features like word2vec, word n-grams, etc. for detecting hate speech. They also developed an abusive language corpus from annotated user comments. Davidson et al. [6] worked in identifying the challenges in hate speech detection such as detecting hate speech when hate words might not be used in the text.

3 Data

This section discusses more details about the datasets mentioned in Section 1. The number of posts provided per label for English, German and code-mixed Hindi is given in Table 1. The posts marked as HATE is further categorized as given in Table 2 and Table 3. As apparent from the above-mentioned tables, the datasets given is imbalanced.

To balanced the datasets, it was shuffled once and then balanced using a simple interweaving algorithm. The interweaving algorithm for two labels is given

in Algorithm 1. The dataset is not shuffled again. This was done with the intuition that when feeding data in batches to the models, the number of samples of each category would be balanced in each epoch so that the model learns from each class evenly. Before training any models, preprocessing was applied on the dataset. The preprocessing applied is similar to the one applied by Davidson et al. [6]. The preprocessing steps are as follows.

1. Hashtags were segmented.
   For example *#buildthewall* is segmented into *build the wall*.
2. URLs were removed.
3. Redundant symbols were removed.
4. Redundant whitespaces were removed.
5. The @ symbol from user handles was removed.
6. *RT* prefixes were removed.

Algorithm 1: Interweaving

```
input : M ← List of majority samples , m ← List of minority samples
output: B ← Interweaved balanced dataset
Init B ← Empty list ;
for i ← 1 to |M| do
    Append M[i] to B;
    if i >=|m| then
        Init r ← random integer between 1 to |m| ;
        Append m[r] to B;
    else
        Append m[i] to B;
return B
```

Table 1. Dataset statistics for subtask A

Language	NOT	HOF	Total
English	3591	2261	5852
German	3412	407	3819
Code-Mixed Hindi	2196	2469	4665

4 Experimental Settings

4.1 Embeddings

The preprocessed posts mentioned in Section 3 were then converted into vector representations using pre-trained embeddings via an embedding layer. If any

Table 2. Dataset statistics for subtask B

Language	HATE	PRFN	OFFN
English	1143	667	451
German	111	86	210
Code-Mixed Hindi	1237	676	556

Table 3. Dataset statistics for subtask C

Language	UNT	TIN
English	220	2041
Code-Mixed Hindi	924	1545
German	–	–

words in the posts were not found in the embeddings, a zero vector of appropriate dimensions was used. For tokenizing the sentences, NLTK's [3] TweetTokenizer was used. The embedding used for the three languages is given in Table 4.

Table 4. Embeddings used

Language	**Pretrained Embeddings**	**Dimentions**
English	FastText English	300
Code-Mixed Hindi	FastText Hindi	300
German	GloVe [12]	200

4.2 Models Used

In this section, the neural network models used are discussed. All the models have been implemented using Keras [5]. And for other tasks such as evaluating the performance of the models and generating train-validation-test splits, Scikit-Learn has been used.

Stacked BiLSTM This model was used for all the subtasks of the English dataset. It was first trained for subtask A. And then the model was further fine-tuned for subtask B and subtask C. The model architecture is given in Table 5. The bidirectional outputs were merged by multiplying. Fine-tuning was done by resetting the weights of the 9^{th} and 10^{th} layer and replacing the final output layer with a new dense layer of appropriate units. All other layers were frozen. The model was trained using Adam optimizer coupled with a cross-entropy loss function for 20 epochs. The batch size used was 32.

CNN This model was used for all subtasks of the German and code-mixed Hindi dataset. Similar to the stacked BiLSTM, this model was first trained for

Table 5. Stacked BiLSTM Architecture

No.	**Layer**	**Hyperparams**	**Regularizers** (recurrent, activity, bias)
1.	Input Layer	–	–
2.	Embedding Layer	–	–
3.	BiLSTM (Returns Sequences)	128 ReLu Units	1E-3, None, None
4.	Batch Normalisation	Default	–
5.	Dropout	Rate 0.5	–
6.	BiLSTM	128 ReLu Units	1E-3, None, None
7.	Batch Normalisation	Default	–
8.	Dropout	Rate 0.2	–
9.	Dense	256 ReLu Units	–
10.	Batch Normalization	Default	–
11.	Dropout	Rate 0.5	–
12.	Dense	Softmax Units	–

subtask A and then fine-tuned for subtask B and C. This time, fine-tuning was done by replacing the final layer by a dense layer of the appropriate number of units, i.e, number of categories to classify. Then all the layers were retrained using a small learning rate (0.0005) and decay (0.000005). The optimizer used was *rmsprop* and the same cross-entropy loss function. The batch size used is 32 and trained for 20 epochs.

Table 6. CNN Architecture

No.	**Layer**	**Hyperparams**	**Regularizers** (kernel, activity, bias)
1.	Input Layer	–	–
2.	Embedding Layer	–	–
3.	1D Convolution	128 Filters of size 5	1E-3,1E-3,1E-3
4.	1D MaxPooling	Pool size 5	–
5.	1D Convolution	128 Filters of size 5	1E-3,1E-3,1E-3
6.	GlobalMaxPooling	–	–
7.	Dense	128 Relu Units	–
8.	Batch Normalisation	Default	–
9.	Dropout	Rate 0.5	–
10.	Dense	Softmax Units	–

5 Results and Discussion

The primary and secondary metric of evaluation for HASOC is weighted and macro *f1*. The official results on the test set as published by the organisers for each dataset is given in Table 12, Table 13 and Table 14. For the English dataset, the BiLSTM model performed quite poorly achieving a *macro average f1* of only 0.61. The model became biased on the posts that were not hate-speech as is apparent from the confusion matrix shown in Table 7. On further fine-tuning this model for subtask B and C, its performance degraded even further scoring *macro average f1's* of 0.28 and 0.36 respectively. The CNN model used for the Hindi subtasks achieved a *macro average f1* of 0.76 which is around 0.05 shy from the best performing model. However, this model also lost a lot of performance on fine-tuning for subtask B and subtask C, achieving *macro average f1* of 0.26 and 0.58 respectively. Both models suffered from false positives and false negatives. NLTK's TweetTokenizer is not built for the Hindi language, hence it seems to perform character-level tokenization on the Hindi words. In the subtasks for the German language, the CNN model achieved a *macro average f1* of 0.52, while the best-achieved score is 0.61.The model was able to classify only 20 samples as HOF. This could be due to the huge skew in the training set as shown in Table 1. Oversampling by interweaving did not prove to be an effective method for balancing the dataset, especially when the imbalance is large. In subtask B, the models performed poorly in classifying the OFFEN and PRFN labels. In the German dataset, it completely failed to classify the PRFN class and performed poorly on the OFFEN class. The same performance degradation can be observed in subtask B and subtask C.

Table 7. Confusion Matrix for Subtask A

	Subtask A					
	English		German		Hindi	
	HOF	NOT	HOF	NOT	HOF	NOT
HOF	135	153	20	116	469	136
NOT	196	669	74	640	185	528

6 Conclusion

The problem of hate speech has become increasingly more prevalent. People that post hateful tweets always find news ways to skirt around detection systems. With the velocity at which content is generated on social media, it is not feasible to manually flag every post for toxicity. As such, it is of utmost importance to develop automated systems that detect and purge hate speech and other toxic

Table 9. Confusion Matrices for Subtask B

	English			
	HATE	NONE	OFFEN	PRFN
HATE	34	75	3	12
NONE	160	639	7	59
OFFEN	10	58	1	2
PRFN	27	50	2	14

	German			
	HATE	NONE	OFFEN	PRFN
HATE	4	27	10	0
NONE	24	641	47	2
OFFEN	1	70	6	0
PRFN	0	18	0	0

	Hindi			
	HATE	NONE	OFFEN	PRFN
HATE	30	98	30	32
NONE	92	392	106	123
OFFEN	33	90	25	49
PRFN	29	114	36	39

Table 10. Confusion Matrix for Subtask C

	Subtask C					
	Hindi			English		
	NONE	TIN	UNT	NONE	TIN	UNT
NONE	297	392	24	639	193	33
TIN	211	313	18	156	75	14
UNT	19	41	3	27	14	2

Table 12. Results for English Dataset

	English							
	Subtask A				Subtask B			
	Precision	Recall	F1-Score	Best F1	Precision	Recall	F1-Score	Best F1
Macro Avg.	0.61	0.62	0.61	0.79	0.29	0.29	0.28	0.54
Weighted Avg.	0.71	0.70	0.70	0.84	0.62	0.60	0.60	0.75

	Subtask C			
	Precision	Recall	F1-Score	Best F1
Macro Avg.	0.36	0.36	0.36	0.51
Weighted Avg.	0.64	0.62	0.63	0.76

Table 13. Results for Hindi Dataset

	Hindi							
	Subtask A				Subtask B			
	Precision	Recall	F1-Score	Best F1	Precision	Recall	F1-Score	Best F1
Macro Avg.	0.76	0.76	0.76	0.81	0.26	0.26	0.26	0.58
Weighted Avg.	0.76	0.76	0.76	0.82	0.38	0.36	0.37	0.71

	Subtask C			
	Precision	Recall	F1-Score	Best F1
Macro Avg.	0.35	0.35	0.34	0.57
Weighted Avg.	0.48	0.47	0.46	0.74

Table 14. Results for German Dataset

	German							
	Subtask A				Subtask B			
	Precision	Recall	F1-Score	Best F1	Precision	Recall	F1-Score	Best F1
Macro Avg.	0.53	0.52	0.52	0.61	0.27	0.27	0.27	0.35
Weighted Avg.	0.75	0.78	0.76	0.79	0.73	0.77	0.75	0.77

content. It is hoped that the system models developed in this study can shed some light on the task of detection of hate speech and its finer modalities. But it leaves much to be desired.

References

1. Baruah, A., Barbhuiya, F.A., Dey, K.: Abaruah at semeval-2019 task 5 : Bi-directional lstm for hate speech detection. In: SemEval@NAACL-HLT (2019)
2. Basile, V., Bosco, C., Fersini, E., Nozza, D., Patti, V., Pardo, F.M.R., Rosso, P., Sanguinetti, M.: Semeval-2019 task 5: Multilingual detection of hate speech against immigrants and women in twitter. In: SemEval@NAACL-HLT (2019)
3. Bird, S., Klein, E., Loper, E.: Natural Language Processing with Python. O'Reilly Media (2009)
4. Cer, D., Yang, Y., yi Kong, S., Hua, N., Limtiaco, N., John, R.S., Constant, N., Guajardo-Cespedes, M., Yuan, S., Tar, C., Sung, Y.H., Strope, B., Kurzweil, R.: Universal sentence encoder. ArXiv **abs/1803.11175** (2018)
5. Chollet, F., et al.: Keras. https://keras.io (2015)
6. Davidson, T., Warmsley, D., Macy, M., Weber, I.: Automated hate speech detection and the problem of offensive language. In: Proceedings of the 11th International AAAI Conference on Web and Social Media. pp. 512–515. ICWSM '17 (2017)
7. Ding, Y., Zhou, X., Zhang, X.: Ynudyx at semeval-2019 task 5: A stacked bigru model based on capsule network in detection of hate. In: SemEval@NAACL-HLT (2019)
8. Indurthi, V., Syed, B., Shrivastava, M., Chakravartula, N., Gupta, M., Varma, V.K.: Fermi at semeval-2019 task 5: Using sentence embeddings to identify hate speech against immigrants and women in twitter. In: SemEval@NAACL-HLT (2019)
9. Mikolov, T., Grave, E., Bojanowski, P., Puhrsch, C., Joulin, A.: Advances in pre-training distributed word representations. In: Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018) (2018)
10. Modha, S., Mandl, T., Majumder, P., Patel, D.: Overview of the HASOC track at FIRE 2019: Hate Speech and Offensive Content Identification in Indo-European Languages. In: Proceedings of the 11th annual meeting of the Forum for Information Retrieval Evaluation (December 2019)
11. Nobata, C., Tetreault, J.R., Thomas, A.O., Mehdad, Y., Chang, Y.: Abusive language detection in online user content. In: WWW (2016)
12. Pennington, J., Socher, R., Manning, C.D.: Glove: Global vectors for word representation. In: Empirical Methods in Natural Language Processing (EMNLP). pp. 1532–1543 (2014), http://www.aclweb.org/anthology/D14-1162