Subjectivity Mining Assignment 4 Group AI 5

Tom Pelletreau-Duris

Vrije Universiteit Amsterdam

1 Introduction

As the ubiquity of hate speech escalates constantly within our social media-heavy society, it is of paramount importance to improve the models we use in detecting this hate speech for the purpose of counteracting the rise of it. When looking at hate speech detection, four key factors seem to enhance the performances of model detection. First the diversity of the data that feed the models. As there are undoubtedly differences in our many cultures, there too will be differences in the forms and appearances of hate speech. Because of this, including examples of hate speech from a variety of sources, both in terms of culture and media, is integral in attempting to identify as many distinct of these forms of hate speech as possible [1]. Second, the abundance of data seems to be substantial in improving model accuracy [2]. In this paper we will only use short training datasets but with pre-trained or re-trained models that have been configured with large scale datasets. Third, the ability to use different datasets in tandem with each other seems to increase the scalability of the models making them more context-independant and thus more efficient on cross-domain tasks [3]. In this report we evaluated a number of different models using two different datasets OLID [3] and HASOC [4], training them both in-domain and cross-domain, to see whether results improve or at least stay similar in comparing these training methods. Four, the combination of models predictions in meta-models seem to improve even more the performance of hate speech detection models. The use of multiple models on a variety of hate speech data sources could then be ideal in an attempt to discover the most efficient methods for detecting hate speech across multiple domains.

First of all, we will describe the data used for evaluating the models, which comes from three unique sources. Afterwards, we will outline the different methods used in model evaluation, which will lead us to the results of our research question. *Which combination of hate-speech detection models with specific tuning characteristics are the best in in-domain and cross-domain settings ?*. Finally, we will evaluate the results of our models' performances, and discuss the possibility of others to build on top of our work.

2 Data

The data consists of two datasets, namely the OLIDv1 dataset [3], as well as the HASOC dataset [4]. The datasets are composed by english-language tweets,

however, the HASOC dataset also includes content obtained from facebook. Each dataset originally used a different representation for the annotated content, which were processed to use the same annotation. In both datasets were included the labels '0' and '1', for representing that a tweet or message is not offensive, or offensive, respectively. Concerning the collection and proportions of the data in terms of offensiveness, each dataset was collected using a different approach. The OLID dataset was collected by the use of keywords and constructions that commonly appear in tweets that are offensive, such as 'she is', or 'to:BreitBartNews'. The use of mentions of left and right news sources were also included as these also often attract offensive tweets. Finally, the data consists 50% of tweets containing political keywords on controversial topics, and 50% of non-political keywords. Overall the data was accumulated so that about 30% of it was offensive, as determined by the annotations of a group of experts. [3]. The HASOC data was collected in largely the same way, using keywords and hashtags to obtain offensive tweets, however also the history of the authors of the tweets was also used to decrease bias. Furthermore in this case, annotation was not done by experts, but by juniors. [4].

When looking at the actual size of each dataset, we can see that originally, the OLID dataset is much larger than the HASOC, containing consirably more information. To be able to compare both dataset, it was selected a smaller subset of this dataset, containing the same size and label distribution as the HASOC dataset.

3 Methods and Experimental setup

Ensemble methods have been shown to be one of the most robust techniques in hate speech detection and are now regarded as the top-ranked strategies in NLP. Because they rely on combining multiple models to produce improved results, by definition the implementation can be done in different ways. We differentiate three different families of ensemble learning algorithms : Bagging ensembles, Boosting ensembles and stacking ensembles. On one hand, bagging will mainly focus at getting an ensemble model with less variance than its components whereas boosting and stacking will mainly try to produce strong models less biased than their components (even if variance can also be reduced). Also, given the size of our training data is seems more interesting to focus on boosting and stacking strategies as well as hard majority voting and soft-majority voting techniques which do not necessary need to divide the dataset in sub-parts.

First of all, we trained individually our four base models. We used two transformer-based pre-trained models BERT and RoBERTa, a re-trained model HateBERT and a ML model. We compare their results with macro-average precision, recall and F1-score. Because ensemble strategies can benefit from the diversity of base models, we also develop a correlation matrix using the Pearson correlation coefficient which statistically measure the linear correlation between two sets of data. Because we found out that *** is very redundant with *** and offer worse f1-score on our results, we decided to keep only three different

models which make it easier to aggregate results. Indeed, 4 models can have 2 Vs 2 while 3 models always have a majority.

In a second hand, the experiment aims to find the best ensemble strategy given our training datasets OLID and HASOC datasets. To do so, we will compare **Hard majority voting strategy** to **Soft majority strategy** and **Stacking ensemble strategy**. Because the stacking ensemble strategy rely on the training of a meta-model based on the predictions made by base models, and other only aggregate their results, the stacking techniques seems one of the most promising strategies now available. In particular the Hard majority voting strategy consist of the selecting of the label that is most often predicted. It is a simple aggregation vote, if two over three outputs are 1 and only one is 0, then the results will be 1. As the strategy's name indicates, the threshold is strictly based on the majority. The soft majority strategy is based on the probabilistic aggregation of labels. It sums up the probability mass assigned per class label and chooses the label with the highest probability as the ensemble's prediction. In other words, it chooses the class label that is most likely predicted. In contrast to that, a hard majority voting would choose the label that is most often predicted. Finally, the stacking ensemble strategy is based on a meta-learning algorithm that is going to affect a weight to each base model based on the training and the confidence it puts in each. For carrying out this experiment, we prepared the training data for the meta-model. We used a k-fold cross validation methods with 5 folds in order to obtain the predictions for the three pre-trained models. After obtaining the results, we added new features that could be useful in the training, these were the number of characters and number of words in the text. After this steps, the data was fed into a SVM to obtain the predictions. To analyse the results, we trained our three different emsembling strategies with our base-models with an in-domain experiment, training and testing the models with the same OLID dataset[3]. Then we conducted the Cross-domain experiment with the same models, training on the HASOC dataset[4] but testing on the OLID dataset. The Cross-domain experiment would help us evaluate how models behave in a more general scenario where the data is unknown. In particular, how the models behave when decontextualized. For example the data from the HASOC dataset also includes comments from Facebook whereas the OLID dataset only contains data from Twitter.

Finally, we compared the results of the three different ensembling methods and focused on false negative and false positive. We aslo used the AUC metrics to have a fast understanding of each ensemble method results.

- **Preliminary analysis :** Training of four base models, BERT, RoBERTa, HateBERT and ML model with 1e-5 as learning rates and 1 as number of epochs. Then we selected three models out of the four based on their similarity. The correlation matrix helped us have the less redundant set possible.
- **In-Domain experiment :** Hard voting strategy, soft voting stratgey and stacking ensemble methods compared on the in-domain experiment with OLID dataset.

– **Cross domain experiment :** Hard voting strategy, soft voting stratgey and stacking ensemble methods trained on HASOC data, tested on OLID data

In these 3 experiments the overall performance of the models is quantified with macro-averaged precision, recall, F1-scores and AUC.

4 Results

4.1 Preliminary analysis

After running the four models with the data, we obtained the performance and correlation of these.

	BERT	Hate	Roberta	MLPC
BERT	1.000000	0.626300	0.730565	0.465885
Hate	0.626300	1.000000	0.605966	0.362550
Roberta	0.730565	0.605966	1.000000	0.461262
MLPC	0.465885	0.362550	0.461262	1.000000

Table 1. Correlation of the result on dev set

BERT:	0.6530426884650318
ROBERTA:	0.6973684210526315
HATEBERT:	0.6058718861209964
MLPC:	0.68

Table 2. Performance on the dev set

From the result depicted above we made some conclusions about which models to use in the next stages of this research. In the correlation matrix it can be seen that all models have some correlation with each other. But the correlation coefficient is not close to 1 so there are significant differences in the results from the models. Especially when comparing the MLPC to the other models the difference is significant. Since the BERT models use the same architecture, they differ from each other more than expected. Presumably, ensemble methods using these models could perform better than individually.

When comparing the F1 scores of the models it can be seen that all models perform somewhat similar. The model with the worst performance is HATEBERT, for this reason, we decided to exclude it from our ensemble approach.

4.2 In-Domain Experiment

After training the models and testing them with data from the same source, we obtained the results shown in Table 3, 4 and 5. For the hard majority voting, we can observe in Table 3 a F1 value of 0.84 for the accuracy, which overall represents a good performance. However, when looking at the individual classes, we obtained 0.9 for no-hate and 0.69 for hate, suggesting that the models have more difficulties recognizing hateful content.

	precision	recall	f1-score	support
0.0	0.93	0.87	0.90	662
1.0	0.63	0.77	0.69	198
accuracy			0.84	860
macro avg	0.78	0.82	0.79	860
weighted avg	0.86	0.84	0.85	860

Table 3. Performance of the hard majority voting ensemble

In the case of the Soft Majority voting, it is shown on Table 4 that the F1 value for the accuracy is 0.83, being almost equal than the Hard majority. As expected, we can also see a bigger difficulty for recognizing hateful content like in the previous case.

	precision	recall	f1-score	support
0.0	0.91	0.86	0.89	662
1.0	0.63	0.72	0.68	210
accuracy			0.83	860
macro avg	0.77	0.79	0.78	860
weighted avg	0.84	0.83	0.83	860

Table 4. Performance of the soft majority voting ensemble

Finally, the Table 5 shows the F1 values for the Stacking ensemble. It can be seen that the accuracy was very similar to the previous cases, being placed at 0.84. It was also encountered the same regarding the recognition of the classes.

	precision	recall	f1-score	support
0	0.87	0.91	0.89	620
1	0.74	0.66	0.70	240
accuracy			0.84	860
macro avg	0.81	0.79	0.80	860
weighted avg	0.84	0.84	0.84	860

Table 5. Performance of the stacking ensemble

In this case, for the three methods the results obtained were very similar in all of the cases. The accuracy of the ensemble was very high for recognizing non hateful comments. Even though the hateful comments were not always recognized, the overall performance of the ensembles turned out to be good.

4.3 Cross Domain

For this experiment it the data from the train set was obtain from a different source than the test set. For this reason it would be a more general model and the performance might be expected to be lower. For the hard voting ensemble, we can observe in Table 6 a F1 value of 0.79 for the accuracy, 0.86 for no-hate and 0.57 for hate. These results are not placed far from the in-domain results, reflecting the good performance of the different models.

	precision	recall	f1-score	support
0.0	0.91	0.82	0.86	686
1.0	0.49	0.68	0.57	174
accuracy			0.79	860
macro avg	0.70	0.75	0.72	860
weighted avg	0.83	0.79	0.80	860

Table 6. Performance of the hard majority voting ensemble on cross domain data

Table 7 shows the results obtained for the soft majority voting in the cross domain. It was obtained 0.69 for the F1 value of the accuracy, 0.78 for no-hate and 0.48 for hate. In this case, the models performed worse than in the prior cases, presenting a higher difficulty of recognizing hateful content.

	precision	recall	f1-score	support
0	0.77	0.80	0.78	593
1	0.51	0.46	0.48	267
accuracy			0.69	860
macro avg	0.64	0.63	0.63	860
weighted avg	0.69	0.69	0.69	860

Table 7. Performance of the soft majority voting ensemble on cross domain data

Finally, Table 8 shows the results for the stacking voting. The F1 for accuracy was 0.71, which performs better than the soft majority but worse than hard majority.

We can see that for the cross-domain data, there is more variance in the results depending on the method. For this reason, it is more important in this case to select a suitable voting method to obtain better results.

	precision	recall	f1-score	support
0	0.79	0.82	0.80	620
1	0.48	0.42	0.45	240
accuracy			0.71	860
macro avg	0.63	0.62	0.62	860
weighted avg	0.70	0.71	0.70	860

Table 8. Performance of the stacking majority voting ensemble on cross domain data

5 Analysis

When comparing the results of the ensemble methods, there are a few takeaways that are clear. Firstly, when looking in-domain, we can see that the hard majority voting method provided a slightly better weighted average F1-score than the soft majority voting method. The most likely reason for this is that in some cases there was some model that with more confidence (i.e. higher probability) predicted one label, while the other models actually would have concurred on another label using the hard majority voting strategy. This was likely involving predictions on cases where the probability difference between predicting a 0 or a 1 was very small, thus the models being more unsure about which label to give an instance.

Next, when considering cross-domain results, we again in all cases see a drop-off in performance in comparison to in-domain trained models. Again we believe this is likely due to a difference in the data between the datasets, and as such either this difference either makes it difficult for the models to make predictions on the hasoc dataset, or the models are slightly overfitting to the training data, making them again unable to predict as accurately on the testing data. Interestingly, we do see a significant difference, for all ensemble methods, in f1-score between the prediction of 0 and 1 labels, with the prediction of 1 labels suffering greatly. This would likely again be due to the difference in the data, as the models would have difficulty correctly identifying hate on a dataset consisting only of tweets when the training data also contained facebook comments.

When comparing the in-domain ensemble methods' performance to the individual models', we can see a significant improvement. The average weighted average f1-score of the ensemble methods is 0.84, while that of the individual models was around 0.7. When looking at the cross-domain weighted average f1-scores, the average for the ensemble methods was 0.73, while that of the individual models was around 0.63. While this is not as big of an improvement as with the in-domain results, it is nevertheless also a large boost in performance.

Overall, the ensemble approaches used provided very positive results. We attribute this to the fact that considering multiple models simultaneously dramatically reduces the flaws of each individual model. For example, if one model were to perform badly on a specific type of tweet, or a specific length of text, then the other models would be able to counteract this performance, and using either soft or hard majority methods, the label of the badly performing model would no longer be considered.

5.1 Error Analysis

In the following section an error analysis is performed. This is also called qualitative analysis and in contrary to the quantitative analysis, where the overall performance is analyzed, in qualitative analysis single examples are investigated and patterns in these examples are found. This is done on the model that performed best in the cross domain setup, and in this case this is the hard majority voting ensemble.

The qualitative analysis is performed using the definition of errors provided by van Aken et al.[5] The test set consists of tweets and hashtags are used a lot. These hashtags are seen by the model as one word but often they are a bundle of words without spaces, as seen in the examples below. These errors can be classified as Idiosyncratic and rare words, using the classes defined in paper[5].

690 @USER #MoreBS = MORE #BARRYSOETOROBULLSHIT

771 #Conservatives are #dumberthandogshit is the central point of my tweet

To prevent this error from happening the hashtags should be split up in words. This is actually not an easy task, another machine learning model could be used to analyze the hashtags and provide possible split ups which then could be used in the training and prediction of the data.

The next pattern that was found was the usage of swear words in false positives. Classifiers often learn that there is a close correlation of swear words and toxicity in comments. This can be problematic when non-toxic comments contain swear words. This is common in this dataset and below are a few examples of which the model classified them as hate.

507 #GeoffreyOwens niggas don't really understand...levels to career pursuits

376 7 fucking years. #MyTwitterAnniversary URL

To better predict these cases the improvement in Natural language processing can be of help. Transformers such as BERT use attention to know on what part of the sentence the model should focus on. The use of attention and improvements in these techniques are the solution to this issue.

When looking at the previous error analysis we performed the errors found are similar. But the errors are now clearly classified using the framework provided by van Aken et al.[5]

6 Conclusion and Future Work

There are a few takeaways that have been attained through our experiments. The use of ensemble models in hate speech detection was successful and led to promising results for all ensemble techniques, the results are clearly better

then using a individual model. All techniques performed well and there is no method that significantly outperforms another. However, when the models are evaluated using a cross domain setup there is a significant drop in performance. This drop in performance is expected but it is remarkable that one technique now clearly outperforms the other two. Also we noticed a bigger variance in the results depending on the voting technique used. This suggests that the data used to train and test the models have a big impact on how the different models perform, for this reason it is necessary to carefully select the voting method to be used.

We can also conclude that the models have more difficulties classifying hate than no-hate. Obtaining data from different sources for testing has a big impact in this factor, leading to a drop of performance. The models tend to classify most of the sentences that contain swear words as hate, even though these might not be. Another case are the words without spaces, that might not be recognized by the model and leading to incorrect classification.

It is advised to do further research in when a voting method would be more suitable for a certain type of data. Also it is recommended to discover new ways of processing the data to avoid the problems with the swear words and unrecognized words.

References

1. R. Kumar, A. N. Reganti, A. Bhatia, and T. Maheshwari, "Aggression-annotated corpus of hindi-english code-mixed data," *arXiv preprint arXiv:1803.09402*, 2018.
2. Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," *arXiv preprint arXiv:1907.11692*, 2019.
3. 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.
4. T. Mandl, S. Modha, P. Majumder, D. Patel, M. Dave, C. Mandlia, and A. Patel, "Overview of the hasoc track at fire 2019: Hate speech and offensive content identification in indo-european languages," in *Proceedings of the 11th forum for information retrieval evaluation*, pp. 14–17, 2019.
5. B. Van Aken, J. Risch, R. Krestel, and A. Löser, "Challenges for toxic comment classification: An in-depth error analysis," *arXiv preprint arXiv:1809.07572*, 2018.

7 Appendix

Code available in: `https://github.com/TomPelletreauDuris`