Subjectivity Mining: Assignment 4

Group AI-11

Matthias Agema[1] (2707650), Noa van Mervennée[2] (2710633), and Matteo De Rizzo[3] (2749303)

[1] Business Analytics, Vrije Universiteit, 1081 HV Amsterdam
m.j.agema@student.vu.nl
[2] Artificial Intelligence, Vrije Universiteit, 1081 HV Amsterdam
n.van.mervennee@student.vu.nl
[3] Artificial Intelligence, Vrije Universiteit, 1081 HV Amsterdam
m.de.rizzo@student.vu.nl

1 Introduction

Due to the rapid growth of social media and subsequently offensive ellipsis on the internet, the need for automatic classification of hate speech related content is a must nowadays. In the previous assignment we have shown the strength of transformer-based models in comparison to the conventional machine learning model SVM, when it comes to detect offensive content for both in- and cross-domain experiments. This paper picks up where we left off at the last one. Now transformers models will be combined into ensembles, by using three combining strategies in the field of stacking ensembles. The ensembling strategies, which we will discuss and experiment on in the rest of the paper, are hard majority voting, soft majority voting and stacked generalization.

Our aim is to retrieve further insights and get comfortable with applying these ensembles, which showed state-of-the-art results for the hate-speech detection task. To our knowledge, a comparative study of BERT, HateBERT and BERTweet combined into different ensembles for both in- and cross-domain has not been performed before. Therefore, it is an interesting research for the field of detecting hateful content in Twitter messages, as occurring in the OLID testdata. Especially, for the consideration of strategy for ensembling these three transformers models.

To sum up the goal of this paper, we have conducted the following research question: How, in terms of ensemble strategies, can BERT, HateBERT and BERTweet be combined, yielding the best results for both in- and cross-domain experiments? Our hypothesis is that all ensembles will outperform each of the transformers models. When it comes to ensembling strategies, our expectation is that stacked generalization will become the best ensemble in this research. This anticipation comes from its ability to give single transformers more influence on a case-by-case basis, by applying a meta-learning algorithm.

The rest of the paper is organised as follows. First, the data is described in Section 2, shedding light on data that is used for in-domain and cross-domain experiments. Related work can be found in Section 3. Next, Section 4 presents the methods that we used. Based on this, the experimental setup will be discussed in Section 5, including preprocessing steps and the ensembles' settings. This is followed by the results per model in Section 6. Subsequently, error analyzes will be provided of both in- and cross domain experiments in Section 7 and is provided with a quantitative analysis. At last, conclusions are drawn in Section 8, which also covers the future work.

2 Related work

Ensemble learning. An ensemble is made up out of a group of learners trained for the same problem [12]. The aim is to produce improved results by combining multiple models. Markov and Daelemans [6] evaluated deep learning approaches both under in- and cross-domain hate speech detection conditions. They combined a SVM approach with the deep learning models through a simple majority-voting ensemble, i.e. hard majority voting, which selects the label that is most often predicted. In their ensemble a combination of BERT, RoBERTa and SVM was established, which improved the results over all individual model results. We have chosen to not include SVM in our ensembles, due to the underperforming SVM model in our previous research of the third assignment.

Meta-model. Mnassri et al. [7] ensembled combinations of BERT-MLP, BERT-CNN and BERT-LSTM by using 4 ensembling strategies (soft voting, maximum voting, hard voting and stacking). They used 3 publicly available datasets, i.e. Davidson, HatEval and OLID, for in-domain and merged data analyses. For the meta-model, an implementation of the stratified k-fold cross-validation technique was used in combination with a linear regression classifier (logistic regression) as meta learner. Their stacking approach outperformed all other ensembles, giving highest precision and F1-score. Therefore, this paper is used as inspiration for the meta-learning algorithm in the stacked generalization ensemble.

3 Data

For this study, two datasets have been used, OLID and HASOC. In-domain experiments will be performed on OLID, whereby models will be trained on a subset of OLID's traindata. Therefore, the sizes of both labels in the traindata of both datasets are equal as can be seen in tables 1 and 2. With respect to the cross-domain experiments, classifiers will be trained on the HASOC traindata. At last, results in terms of performance are measured only on OLID's testdata. In this section we will elaborate on both datasets.

Table 1. Data distribution OLID

	Trainset	%	**Testset**	%
NOT	3591	61.4	620	72.1
OFF	2261	38.6	240	27.9
Total	5852		860	

Table 2. Data distribution HASOC

	Trainset	%
NOT	3591	61.4
HOF	2261	38.6
Total	5852	

3.1 OLID

The original Offensive Language Identification Dataset (OLID) contains over 14.000 English tweets. This dataset is annotated by Zampieri and colleagues for offensive content using a three layered annotation scheme. In the first layer A, the goal is to discriminate tweets between offensive and non-offensive. Where offensive content is defined as: "Posts containing any form of non-acceptable language (profanity) or a targeted offense, which can be veiled or direct. This includes insults, threats, and posts containing profane language or swear words." [10]. Layer B distinguishes targeted insults from untargeted ones. At last, if posts were targeted, in layer C they categorized the targets of insults/threats in three categories. In this paper, we focus on layer A, whether posts are offensive or not. For our in-domain experiments a subset of OLID's traindata was used. The data is preprocessed resulting in a binary label, whereby 0 indicates a non-offensive tweet and 1 corresponds to an offensive message.

3.2 HASOC

The Hate speech and Offensive Concent Identification (HASOC) in Indo-European languages track was introduced in 2019 by Mandl et al. [5]. Three datasets were developed for three different languages, namely: English, Hindi and German. For our experiments we only make use of the English tweets and Facebook messages. HASOC was inspired by OffensEval from Zampieri et al. [11] and comprises three sub-tasks, where the data was created for. The annotation architecture looks like the from Zampieri and colleagues, but differs at some points. In the first layer, they distinguish two labels for a binary classification task, namely: Hate and Offensive (HOF) and Non- Hate and offensive (NOT). They labeled posts as HOF if they contain any form of non-acceptable language such as hate speech, aggression and profanity, which is quite similar to OLID's rule for layer A, all others get NOT. Next, in layer B all HOF labeled cases are further categorized into three classes, (i) hate speech (ii) offensive and (iii) profane, which corresponds reasonable with the third layer from OLID's annotation scheme. At last, layer C considers the type of offense, targeted or untargeted. We will again focus on the first layer, for which a binary classification can be performed.

Preprocessing steps are already done by labeling hate and offensive as 1, and a 0 corresponds to non-hate and non-offensive messages. Only traindata is used from HASOC, because of the cross-domain experiments that we are going to perform on this dataset.

4 Methodology

In this paper, three pre-trained transformer-based models (BERT, HateBERT, and BERTweet) have been combined with three ensembling strategies to compare results across different domains. The models have been chosen based on the evaluations of the last assignment. In the previous assignment, it was obvious that BERTweet outperformed BERT and SVM. However, concerns about the generalizability of the model arises when this model has to classify offensive content coming from areas other than Twitter. Hence, it might be of interest to consolidate different transformer models, pre-trained on data from other scopes, to subsequently strengthen the results regarding the weaknesses of its individual models. Given that ensemble strategies combine multiple models to develop optimal results, they should provide superior outcomes than the transformer-based models individually. The ensembling strategies that will be used for this purpose are Hard majority voting, Soft majority voting and finally Stacking ensemble.

4.1 BERT

The first chosen model is a pre-trained language model, called BERT. BERT stands for Bidirectional Encoder Representations from Transformers, which is designed to pre-train deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers [3]. As a result, the pre-trained BERT model can be fine-tuned with just one additional output layer to create state-of-the-art models for a wide range of tasks [3].

Pre-training The model is pre-trained on the BooksCorpus (800 million words) and English Wikipedia (2,500 million words). Pre-training had two supervised tasks: (i) masked language modeling and (ii) next sentence prediction. In task (i), 15% of all tokens in each sentence are masked at random, then the model predicts the masked tokens. For task (ii), two masked sentences are concatenated, which may or may not have been consecutive in the corpus. It is now up to the model to predict whether those sentences followed each other or not. Both tasks (i) and (ii) are essential for BERT to find context in two directions and associate with relations between sentences.

4.2 BERTweet

As the OLID dataset consists only of texts originating from Twitter, we chose to implement BERTweet as our second pre-trained language model. Tweets are recognized by their short length, use of abbreviations, emojis and informal grammar, making it very difficult to use the content for a classification task. However, BERTweet is the first public large-scale pre-trained language model for English Tweets [8]. The BERTweet uses the same architecture as BERT, which is, as mentioned, trained with the masked language modeling. The pre-training procedure is based on RoBERTa [4] which optimizes the BERT pre-training approach for more robust performance [8]. RoBERTa is significantly improved compared to BERT by training the model longer, with bigger batches over more data; removing the next sentence prediction objective; training on longer sequences; and dynamically changing the masking pattern applied to the training data [4].

Pre-training As mentioned in the paper of Liu et al. [8], the model is trained using a 80GB corpus of 850 million English Tweets. The Tweets consist in total of 16 billion tokens where each Tweet has at least 10 and at most 64 tokens. First, Tweets from 01/2012 to 08/2019 have been downloaded. The Tweets are tokenized using the "Tweet Tokenizer" from the NLTK library and use the emoji package to translate emoticons into text. Normalization is performed by anonymizing user-data with @USER and webpage related data with HTTPURL. At last, retweeted Tweets are filtered out resulting in a corpus with 845 million Tweets. More Tweets originate from 01/2020 to 03/2020 and are related to the COVID-19 pandemic. The same procedures have been applied to these Tweets, resulting in the remaining trained corpus of 5 million English Tweets. Finally, all 850 million Tweets are segmented with subword units resulting in 25 subword tokens per Tweet on average.

4.3 HateBERT

HateBERT is a re-trained BERT model for abusive language detection in English [1]. It was trained on RAL-E (the Reddit Abusive Language English dataset), containing more than a million reddit posts from banned communities for being offensive, abusive or hateful that have been collected in a collaboration between the University of Groningen, the University of Turin and the University of Passau. The paper of Caselli et al. concluded that the use of HateBERT outperforms the BERT `base-uncased` model for tasks related to abusive, offensive or hateful language. The datasets tested for HateBERT are OffenseEval, AbuseEval and HatEval.

4.4 Hard and Soft majority voting

The first two ensemble methods used in this paper are voting classifiers. These belong among the simpler ensembling strategies. They consist in a combination of classification models, generate prediction for each one and subsequently use voting to choose the optimal solution. In majority voting ensembles, the chosen prediction for every instance of the data is the one that achieves more than half of the total votes. It can also happen that none of the predictions receives the required amount of votes, which results in the impossibility to determine a stable prediction. Since this assignment required three classification models to form a combination for the ensemble strategies, the vote will never lead to a tie. The two ensemble techniques can be differentiated between Hard and Soft voting: they both work in similar way, but Hard Voting is slightly simpler than its counterpart. For Hard Voting, the final prediction simply depends on majority of the amount of predicted labels amongst the three transformer models, while Soft Voting keeps into account the sum of the predicted probabilities of each three models. As we can see in the following equation, for Hard voting, the predicted class $\hat{y}$ is determined by the mode (the most frequently occurring value in a group) of each classifier model (C):

$$\widehat{y} = mode\{C_1(x), C_2(x), \ldots, C_m(x)\} \tag{1}$$

For this assignment, the value for m is set to three (three classification models that form a combination). Soft voting, on the other hand, predicts the class $\hat{y}$ by choosing the highest value between the sums of the probabilities p for each class (offensive and non-offensive), with a different weight w multiplied to each model if necessary. That means, if the sum of the predicted probabilities for one tweet regarding the offensive class yields a higher score than the sum of the predicted probabilities for the non-offensive class, the offensive class will be voted for. As no weight is requested for this assignment, and due to the limitation of time, the weight has been set to 1. This leads to the following equation applied for the Soft Voting technique:

$$\widehat{y} = \arg\max_i \sum_{j=1}^{m} w_j p_{ij}. \tag{2}$$

4.5 Stacking ensemble

A stacking ensemble uses machine learning models in order to determine the best way to combine the predictions of contributing ensemble members. Stacking makes use of different models to create intermediary predictions. Consequently, these predictions are analyzed by a new machine learning model, called the meta-learner model, in order to make a final classification between the available classes. An advantage of the incorporation of a meta-learner model is the availability to add additional features. These features might be beneficial for understanding the strengths and weaknesses of the individual models, and consequently act upon them. For this research, various features have been added to the data that might be relevant for improving the detection of hateful content. Before running the intermediary prediction through the meta-model, the traindata for the meta-learner is obtained by conducting a k-fold cross validation on the in-domain and cross-domain traindata. In the k-fold procedure, we first shuffle our data to ensure that the inputs and outputs are in random order. We take this precaution to ensure that none of our inputs are biased. After that, we divided the data into $k = 5$ equal parts, each called fold. After that, one fold is used for testing and the remaining folds for training each of the three models. This process is repeated k times so that we can test the models on a different fold each time we start a new iteration. By doing this, we can ensure that every k-part testset is only utilized once which should give the best probability to improve the performance. After this, the

trainset for the meta-model is formed by combining the predicted labels from the three transformer models collected during the k-fold cross validation procedure. The testset consists of the actual labels belonging to the traindata-olid and traindata-hasoc.
Furthermore, we added various features to each entry of the resulting trainset, these being:

- the number of characters
- frequency of uppercase characters
- number of special symbols
- number of tokens
- number of emoticons
- number of hateful words

The number of characters consists of the length of the message, while the frequency of uppercase characters is calculated by dividing the amount of uppercase characters by the total number of characters. The number of special symbols have been determined by counting the amount of non-digit/non-alphabetic characters. Furthermore, the number of tokens have been established by the use of Spacy [4]. With the use of the Enlish *en_core_web_sm* trained pipeline, the spacy model distinguishes the tokens from the text. During establishing the number of tokens, each token is compared to the list of hateful lexicon words to determine the number of hateful words. The list of hateful lexicon words are extracted from *hatebase_dict_vua_format.csv*, made available for this course. At last, the number of emoticons have been extracted for each message with the use of emoji library in python. Lastly, when it was time to choose between the many possibilities for our meta-model, we have relied on previous research to guide us. Ahad and colleagues [2] have studied stacking ensemble methods and came to the conclusion that when using Logistic regression as meta-model, the results outperforms the alternatives. Their approach was tested on a variety of problems, among which binary classification, which was our reason for selecting it for our paper. Logistic regression is a classification model mainly used for binary or linear classification problems, despite its multinomial capabilities [9]. It is a transformation of linear regression with the use of a sigmoid function:

$$F(x) = \frac{1}{1 + e^{-(\beta + \beta_1 x)}} \tag{3}$$

5 Experimental setup

The objective of this research is to investigate the generalizability of several methods for automatic classification of hate speech. To this end, the three models will first be trained on the in-domain setup (OLID dataset), and then with the same parameters they will be trained on the cross-domain setup (HASOC dataset). Then, we will be combining the outcome of the three models into three different ensemble models, in order to compare their efficacy in the classification task across the setups.The ensemble methods that will be used in this paper are hard validation, soft validation and a stacking method, containing a logistic regression function (as described in section 4 All models and ensembles will be evaluated on the OLID test dataset. In order to evaluate them we will be using macro-averaged precision, recall, and F1-score.

5.1 Hyperparameters transformer models

For the first pre-trained transformer model BERT, we implemented the default hyperparameters of $\text{BERT}_{\text{BASE}}$[5]. The model adopts the Adam optimizer with a learning rate of 0.0001. $\text{BERT}_{\text{BASE}}$ applies 12 layers, a hidden size of 768 and a number of self-attention heads of 12, as stated in the paper of Devlin et al. [3]. The BERTweet model is initialized with the $\text{BERTweet}_{\text{BASE}}$ model [6] architecture that implements the same architecture as the $\text{BERT}_{\text{BASE}}$ and the same pre-training procedure as RoBERTa [8], resulting in 135 million parameters [7]. At last, the HateBERT model adopts a learning rate of 0.00005. The model re-trains the English BERT base-uncased model retrieved from the hugging face Transformers library by appling the Masked

[4] https://spacy.io
[5] https://huggingface.co/bert-base-cased
[6] https://huggingface.co/vinai/bertweet-base
[7] https://datquocnguyen.github.io/resources/NVIDIA_GTC_Talk.pdf

Language Model (MLM) objective, which results in a shifted BERT model [1]. They verify in their paper that HateBERT has shifted towards the abusive language phenomena by using the MLM on five template sentences of the form "[someone] is a(n)/ are [MASK]".

5.2 Feature selection meta model

After the retrieval of the predictions during the k-fold cross validation procedure, the trainset for the meta-model is created. As mentioned, in an attempt to improve the results, various features have been added to this trainset that might be beneficial in improving the detection of offensive tweets. This addition finalizes the traindata for the meta-model. The Logistic Regression is adopted with the default regularization strength of 1.0. The maximum number of iterations *max_iter* taken for the model to converge is set to 10000.

6 Results

Table 3 shows the pearsons correlation coefficient regarding the predictions of the individual transformer models. As can be seen from the model, the correlations yield approximately 0.70 between all models. These results are rather high which indicates a strong positive correlation. With a high correlation, it is expected that the combination of the models will not contribute heavily to an improvement of the results. On the other hand, as the correlation is not entirely a perfect correlation, it might be sufficient enough to incorporate some strengths from one model that may be lacking from another model.

Table 3. Pearsons correlation coefficient

	BERT	**BERTweet**	**HateBERT**
BERT	1.00	0.72	0.71
BERTweet	0.72	1.00	0.71
HateBERT	0.71	0.71	1.00

The models' and ensembles' performance in terms of precision, recall and F1-scores, for the in- and cross domain is shown in respectively table 4 and 5. All metrics are divided over the two different classes non-offensive (NOT) and offensive (OFF) and are depicted in bold if that particular score is best in that row. As mentioned in the experimental setup, section 4, all models are tested on the OLID testdata.

Table 4. Metrics in-domain experiments for transformers and ensembles (trained on OLID data)

In-domain		**BERT**	**HateBERT**	**BERTweet**	**Hard voting**	**Soft voting**	**Stacking**
Prec.	NOT	0.86	0.86	**0.90**	0.86	0.88	0.88
	OFF	0.73	0.73	0.72	0.73	**0.76**	0.75
	macro	0.79	0.80	0.81	0.79	**0.82**	0.81
Recall	NOT	0.91	0.91	0.89	0.91	0.91	**0.92**
	OFF	0.62	0.62	**0.74**	0.62	0.69	0.66
	macro	0.76	0.77	**0.81**	0.76	0.80	0.79
F1	NOT	0.88	0.89	0.89	0.88	**0.90**	**0.90**
	OFF	0.67	0.67	**0.73**	0.67	0.72	0.71
	macro	0.78	0.78	**0.81**	0.78	**0.81**	**0.81**

Table 5. Metrics cross-domain experiments for transformers and ensembles (trained on HASOC data)

Cross-domain		**BERT**	**HateBERT**	**BERTweet**	**Hard voting**	**Soft voting**	**Stacking**
Prec.	NOT	0.83	0.81	0.85	0.83	0.84	**0.86**
	OFF	0.65	0.69	0.76	0.65	**0.78**	0.76
	macro	0.74	0.75	**0.81**	0.74	**0.81**	**0.81**
Recall	NOT	0.89	0.93	0.93	0.89	**0.94**	0.93
	OFF	0.51	0.42	0.58	0.51	0.55	**0.60**
	macro	0.70	0.68	0.75	0.70	0.74	**0.76**
F1	NOT	0.86	0.86	**0.89**	0.86	**0.89**	**0.89**
	OFF	0.57	0.53	0.66	0.57	0.64	**0.67**
	macro	0.71	0.69	0.77	0.71	0.77	**0.78**

7 Analysis

Between the three ensemble methods we used in this paper, the Soft Voting and Stacking obtained approximately similar results regarding the in-domain set up in terms of precision, recall and F1 as can be seen in table 4. As the aim of this paper is to gain further insights into the generalizability issues in the area of hate speech, table 5 was the deciding factor on choosing the optimal ensembling technique: the stacking method. Therefore, the analysis section will cover extensively this one method, in both in- and cross-domain in a quantitative way, while only focusing on the cross-domain when discussing the qualitative analysis.

7.1 Quantitative analysis

Despite the better/equally well performance of the Soft Voting ensemble method regarding the models trained on the OLID data, this method has not been classified as the optimal ensemble method. This method outperforms the Stacking method insignificantly for the OFF and macro precision score with a difference of 0.01, and an improvement of 0.01 regarding F1-score of the OFF label. However, the cross-domain experiments provided significant improvements to choose the Stacking ensemble method as the optimal ensemble technique. Shown in table 4, it is remarkable that BERTweet outperforms all other models with a slight improvement. Especially the recall score (0.74) for the Offensive class is noticeable obtained by the BERTweet model trained on the OLID traindata.

The results for the cross-domain setup show that the Stacking method is the optimal model. However, these results are also approximately similar as for the results of the BERTweet model. With a F1 score for the OFF class of 0.67 and the macro F1 score of 0.78, the Stacking method is only with a value of 0.01 superior compared to the BERTweet model. Contrary to our expectations, the ensemble methods we applied to the three models did not significantly improve our results for the in-domain experiments. Nonetheless, the difference in performance regarding the in-domain and cross-domain experiment are conspicuous. The experiments conducted for the previous assignment showed a large drop in performance for the cross-domain set up. As can be seen in table 5, it is remarkable that the performance drop regarding the Stacking method is almost insignificant. Aside from the recall score, the model is able to predict the offensive class approximately equally well for the in-domain as for the cross-domain.

7.2 Qualitative analysis of cross-domain setup

In this section, a certain amount of errors that were performed by the Stacking method are analysed on a qualitative level. Out of the 860 entries in the test-set predicted on the HASOC traindata, this approach resulted in 45 false negative predictions and 97 false positives (see figure 6.

False negatives

False negatives are cases in which our model predicted the message to be offensive while the true label indicated that it was not offensive. One example of this is the following tweet:

1. And EMOJI females b on the same shit URL

In this case, only one of the component models identify correctly this message as offensive, and it is not surprising that it was BERTweet. A reason for this is BERTweet's training on tweet format, which contained a wide use of emoji's. In this message, BERTweet was able to understand to add context to the meaning behind the emoji (a man with darker skin shrugging). Nonetheless, all models perceived the message with high ambiguity, which can be seen in the predicted probabilities that determined the prediction. All the models gave probabilities of that would orbit in the negative values around the 0 for the non offensive class. The probabilities for the non-offensive class yield: BERT (-0.04), BERTweet (-0.05) and HateBERT (-0.02), and for the offensive class: BERT (-0.34), BERTweet (0.01) and (-0.08). BERTweet is the only model that assigns a positive value regarding the offensive class, although minimal. As stated before this could be because of the emoji, that might have created further uncertainty in the model.

Another example is stated below from which the actual label is offensive, although the Stacking method classified as non-offensive. A reason for this behaviour is the hidden word of 'loser' inside the hashtag. The model is therefore not able to recognize the swear-word. All individual models and ensembling models fail to predict this tweet correctly.

2. #SierraBurgessIsALoser She is me when my phone dings EMOJI

False positives

On the contrary of false negatives, false positive are the not offensive entries that have been evaluated as offensive by our model. One example of this is:

1. Are you fucking serious? URL

This specific message was classified incorrectly by all individuals models, and therefore it is not surprising that the Stacking method did as well. Here, the use of the word *fucking* is used to emphasize a certain emotion in the text, but it does not have to be necessarily offensive. For example, it could express surprise just as well as annoyance. Nonetheless, due to the presence of the curse word it can be expected that the models would classify it as offensive, when it is merely informal language. It is noteworthy that HateBERT, the model specifically trained on hateful messages, did not consider this message quite offensive as it gave a somewhat negative probability of prediction on both classes (-0.19 for not offensive and -0.16 for offensive).

Another example of false positive prediction is the following:

2. @USER Well she is Chuck Schumer's cousin or niece or something so she is obviously part of the DNC conspiracy against Nixon. Or part of the Deep State. Or a Zionist plot. Or 'Big comedy'. Or whatever the hell today's conspiracy theory is.

In this case, among the three components of the voting method, only HateBERT's prediction is correct while both BERT and BERTweet predicted this as offensive. When dealing with a complex message like this one, also hard to classify by a person, HateBERT does not jump to conclusions like the other models. Its approach is to keep the probabilities very low, around 0, while giving small negative values to what it considers unlikely (-0.05 for not offensive and -0.23 for offensive). Nonetheless, given the high probabilities given by the other models, it results in the Stacking ensemble method to mislabel the message.

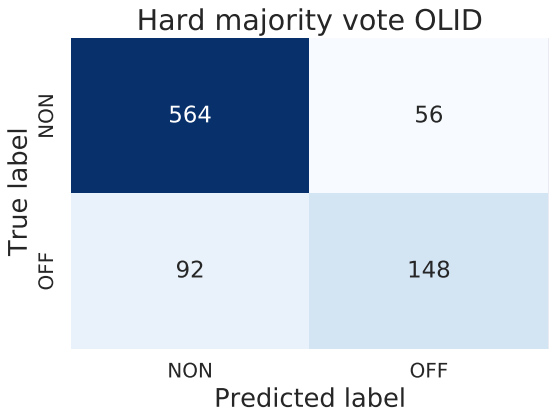

Fig. 1. Confusion matrix Hard Majority voting OLID

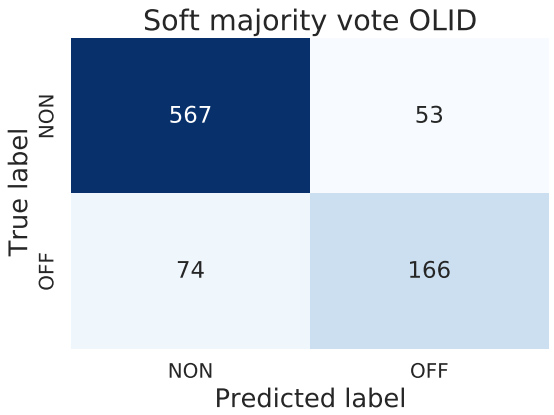

Fig. 2. Confusion matrix Soft Majority voting OLID

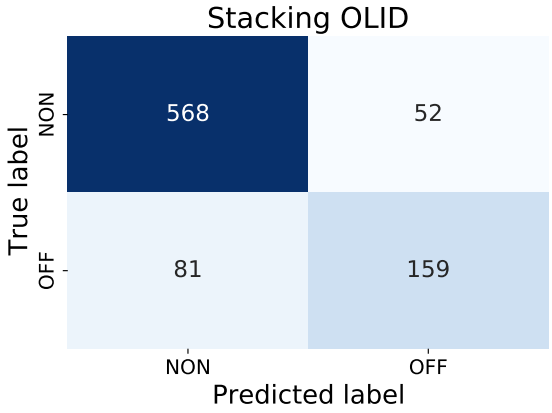

Fig. 3. Confusion matrix Stacking OLID

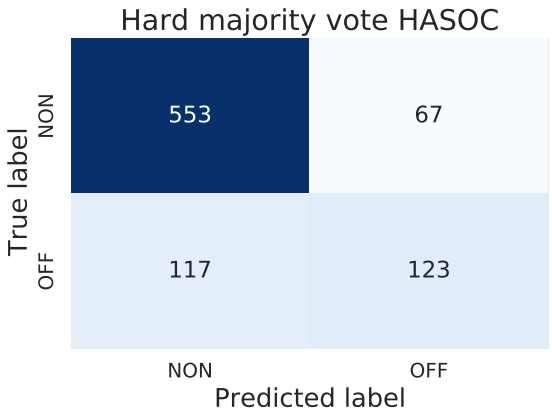

Fig. 4. Confusion matrix Hard Majority voting HASOC

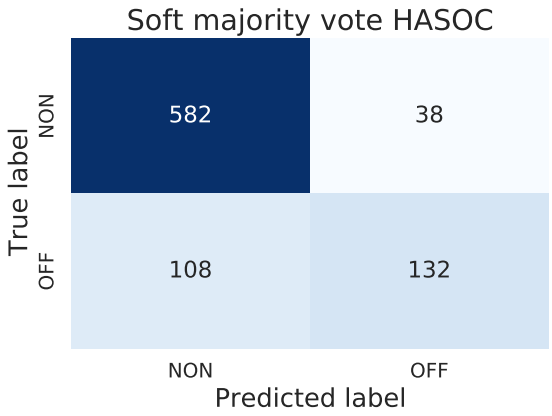

Fig. 5. Confusion matrix Soft Majority voting HASOC

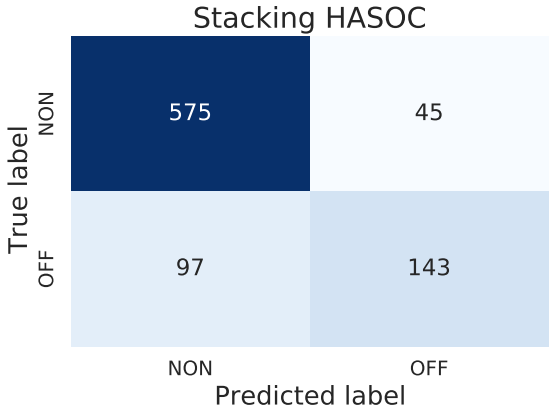

Fig. 6. Confusion matrix Stacking HASOC

8 Conclusion and future work

In this paper we compared three different ensembling methods to compare their effectiveness in different domains. In the hypothesis, we conveyed our expectations that the ensembling strategies would outperform the individual models, and moreover, that the stacking method would present the best results. We assumed this would be the case because of the amount of added features that this method would incorporate However, despite the stacking method performing slightly better than the other ensembling approaches, most of its conclusions were drawn from BERTweets decisions. In both domains BERTweet demonstrated to be the best model by providing the best results. Furthermore, as its strengths provided more advantages, most of these were incorporated into the ensemble, where the other models did not end up contributing much. Thus, the final predictions of the stacking ensemble seem to be almost exclusively determined by BERTweet's contribution.

However, considering that the cross-domain experiment produced similar results, we can conclude that this predominant role of BERTweet did not carry over. Indeed, in this domain the ensemble adopted many of the strengths that are characteristical of the other two models. First of all, while BERTweet only trained on tweets, and therefore specialised on this short kind of format, lexicon and use of emojis, it lacks of adaptability when it comes to longer messages, that do not adhere to the standard structure presented on Twitter. This was an aspect where BERT and hateBERT had an advantage upon, as they were trained on very different data, and provided more generalizability when it comes to cross-domain datasets.

For future research we recommend including models that differ from each other to a higher degree, with more uncorrelated Pearsons coefficients, as they could possess very different strengths and characteristics that an ensemble method could maintain and exploit to provide much better results. Moreover, it would be beneficial to investigate the reason why BERTweet had such a strong influence on the Stacking model, where the other models were mostly undervalued. To test this, the same setup could be tested on more datasets, mainly composed by messages originated on Twitter, so to confirm whether BERTweets' advantage coming from its training was the cause of this phenomenon.

Appendix

GitHub link

`https://github.com/teoderizzo/GrAI-11_assignment4_SubjectivityMining`

Contributions

Noa - 4.2, 4.4, 4.5, 5.0, 5.1, 5.2, 6, 8
Matthias - 1, 2, 3.0, 3.1, 3.2, 4.3
Matteo - 4.0, 4.4, 4.5, 7, 7.1, 7.2, 8

References

1.
2. Abro, A.A., Taşcı, E., Ugur, A.: A stacking-based ensemble learning method for outlier detection. vol. 8, pp. 181 – 185. Balkan Yayın (2020). https://doi.org/10.17694/bajece.679662
3. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: Pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers). pp. 4171–4186. Association for Computational Linguistics, Minneapolis, Minnesota (Jun 2019). https://doi.org/10.18653/v1/N19-1423, https://aclanthology.org/N19-1423
4. Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V.: Roberta: A robustly optimized bert pretraining approach (2019), http://arxiv.org/abs/1907.11692, cite arxiv:1907.11692
5. Mandl, T., Modha, S., Majumder, P., Patel, D., Dave, M., Mandlia, C., Patel, A.: 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. p. 14–17. FIRE '19, Association for Computing Machinery, New York, NY, USA (2019). https://doi.org/10.1145/3368567.3368584, https://doi.org/10.1145/3368567.3368584
6. Markov, I., Daelemans, W.: Improving cross-domain hate speech detection by reducing the false positive rate. In: Proceedings of the Fourth Workshop on NLP for Internet Freedom: Censorship, Disinformation, and Propaganda. pp. 17–22. Association for Computational Linguistics, Online (Jun 2021). https://doi.org/10.18653/v1/2021.nlp4if-1.3, https://aclanthology.org/2021.nlp4if-1.3
7. Mnassri, K., Rajapaksha, P., Farahbakhsh, R., Crespi, N.: Bert-based ensemble approaches for hate speech detection (2022). https://doi.org/10.48550/ARXIV.2209.06505, https://arxiv.org/abs/2209.06505
8. Nguyen, D.Q., Vu, T., Tuan Nguyen, A.: BERTweet: A pre-trained language model for English tweets. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations. pp. 9–14. Association for Computational Linguistics, Online (Oct 2020). https://doi.org/10.18653/v1/2020.emnlp-demos.2, https://aclanthology.org/2020.emnlp-demos.2
9. Subasi, A.: Chapter 3 - machine learning techniques. In: Subasi, A. (ed.) Practical Machine Learning for Data Analysis Using Python. pp. 91–202. Academic Press (2020). https://doi.org/https://doi.org/10.1016/B978-0-12-821379-7.00003-5, https://www.sciencedirect.com/science/article/pii/B9780128213797000035
10. Zampieri, M., Malmasi, S., Nakov, P., Rosenthal, S., Farra, N., Kumar, R.: Predicting the type and target of offensive posts in social media. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers). pp. 1415–1420. Association for Computational Linguistics, Minneapolis, Minnesota (Jun 2019). https://doi.org/10.18653/v1/N19-1144, https://aclanthology.org/N19-1144
11. Zampieri, M., Malmasi, S., Nakov, P., Rosenthal, S., Farra, N., Kumar, R.: SemEval-2019 task 6: Identifying and categorizing offensive language in social media (OffensEval). In: Proceedings of the 13th International Workshop on Semantic Evaluation. pp. 75–86. Association for Computational Linguistics, Minneapolis, Minnesota, USA (Jun 2019). https://doi.org/10.18653/v1/S19-2010, https://aclanthology.org/S19-2010
12. Zhou, Z.H.: Ensemble Learning, pp. 411–416. Springer US, Boston, MA (2015). https://doi.org/10.1007/978-1-4899-7488-4$_2$93, https://doi.org/10.1007/978-1-4899-7488-4_293