Subjectivity Mining: Assignment 2

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 this assignment, the aim is to work with different annotated corpora and investigate whether their characteristics have influence on the classification performance. Therefore, classification models have to be trained and tested on the different annotated text data, which are the Gibert, TRAC and OLID datasets from respectively Gibert et al. [1], Kumar et al. [2] and Zampieri et al. [5]. The paper is organised in the following way. First, the data is described in Section 2, shedding light on the extra train data that is used. Next, Section 3 presents the experimental setup, including preprocessing steps and the models' settings. This is followed by the results per corpus in Section 4. Subsequently, error analyzes will be provided of two experiments in Section 5. At last, conclusions are drawn in Section 6, which also covers the future work.

2 Data

For this assignment, the three datasets of the latter assignment will be considered again. All datasets are in the VUA-format, i.e., a standard format for annotated corpus data, which includes at the most basic level the following files: trainData.csv, testData.csv and devData.csv. For now, the development file is ignored. Additionally, there is extra training data available, which can added to the traindata of Gibert, TRAC and OLID. Finally, lexicon data is provided that can be used to examine whether the model improves by modifying the traindata using the lexicons. In the following sections we will elaborate further on the aforementioned datasets and how the extra data will be used.

Table 1: Gibert

Label	Trainset	Testset
Hate	957	239
NoHate	957	239
Total	1914	478

Table 2: TRAC

Label	Trainset	Testset
OAG	2708	630
CAG	4240	144
NAG	5051	142
Total	11999	916

Table 3: Zampieri

Label	Trainset	Testset
OFF	4400	240
NON	8840	620
Total	13240	860

2.1 Gibert

The annotations used by Gibert and colleagues [1] only differenciate between Hate speech and Non Hate speech. According to their guidelines, a sentence is considered hate speech if "it contains a deliberate attack directed towards a specific group of people motivated by aspects of the group's identity". On the other hand, sentences that do not adhere to the just mentioned definition of hate speech are annotated as Non Hate.

As can be seen in table 1, the dataset included a total of 1941 entries, with a perfect balance between the two annotations. Indeed, 957 of these were annotated as hate while just as many as nonhate. The extra data was fairly simple to implement, as it was annotated in the same way as Gilbert's guidelines, with slight differences in the choice of wording between "noHate" and "nothate". Only minor adjustments to the format of the data was necessary to convert it into VUA format.

2.2 TRAC

The annotations applied in the TRAC paper (Kumar et. al. [2]) yield: overt aggression (OAG), covert aggression (CAG) and non-threatening aggression (NAG). Overt aggression refers to any speech/text that is expressed openly and is considered to be aggressive. In contrast, covert aggression is an indirect attack against the victim and is often packaged as (insincere) polite expressions (through the use of conventionalised polite structures) [2], e.g. satire and rhetorical questions. Both traindata and testdata have been made available in VUA format for this task. The traindata only consists of Facebook posts, while two versions of testdata are available that contain either Facebook posts or Twitter posts. For this task, only the testdata with Facebook posts are considered. In table 2 the distribution of the data is shown. As can be seen, the traindata is highly imbalanced due to the number of Overt Aggression (OAG) posts. As a consequence, it is expected that the model(s) will encounter difficulties in predicting this class. In addition, we expect a low performance classifying the Covert Aggression label due to the lack of emotional expression that can be read from a Facebook post.

Finally, it is asked to include extra traindata. As this dataset contains only the labels Hate and NoHate, we chose to address the imbalanced traindata. Advantageously, posts tagged with Hate can be considered identical to posts tagged with Overt Aggression (OAG). Therefore, we chose to randomly sample 2000 hateful messages from the extra traindata and include them in the traindata with the label 'OAG'.

2.3 OLID

The annotation scheme of Zampieri et al. [5] consists of three layers. In the first layer the labels Offensive and NonOffensive are considered. 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." [5]. The given OLID dataset in VUA format is only satisfied with this first layer and therefore we will not go into the rest of the annotation scheme. There are a total of 13240 and 860 tweets in respectively the train and testdata, as can be seen in table 3. Furthermore, class imbalance can be an issue with only 4400 offensive tweets in the traindata.

At last, extra data - labeled as Hate/noHate - will be added to the OLID traindata. With the intention to give the model a performance boost, it is to blunt to map Hate as Offensive and NoHate as NonOffensive. Simply because the definitions of both annotation schemes are slightly different. It will probably not benefit the models performance. Tweets annotated as Hate in the extra data are offensive anyway, but NoHate annotated tweets can be offensive either. Therefore, only hateful tweets from the extra data were added - labeled as Offensive - to the regular OLID traindata. An additional advantage is the less class imbalance after adding more Offensive labeled tweets to the traindata.'

2.4 Lexicon

Next to adding extra training data to the model, the influence of using lexicon data is investigated. Lexicon datasets contain information about a collection of words that can be used to classify hate speech. In this paper, the lexicon from the Hatebase platform[4] is used to examine the change in performance for the three above mentioned datasets.

[4] `https://hatebase.org/`

3 Experimental setup

To effectively plan and manage the process of developing a machine learning classification task we decided to systematically design and run the different experiments. The process is divided in four procedures: text preprocessing, machine learning model selection, feature representation and parameter optimization with regard to the machine learning models. The model- and parameter-options applied to find the optimal model for each dataset are exhibited in sequential order in table 4. A Python script has been provided for this task with several pipelines that can be modified for experimentation. After finding the optimal baseline model, it is examined whether the model performs better by adding lexicon data and extra training data.

Evaluation of the performance of hate speech (and also other related content) detection typically adopts the classic Precision, Recall and F1 metrics [6]. Precision is a measure of result relevancy, while recall is a measure of how many truly relevant results are returned [5]. As we are dealing with class imbalance, the metric used to determine the optimal settings is the F1 macro average score (the macro average score of the harmonic mean between Recall and Precision). This metric is insensitive to the imbalance of classes and therefore treats them all as equal.

Table 4: Experimental options

1. Model Selection				
Naive Bayes	SVM			
2. Feature Representation				
Count Vectorizer	TFIDF	N-Grams	Stop-Words	Word-Frequencies
3. Text Preprocessing				
Tokenization	Normalization	Lowercasing	Lemmatizing	
4. Parameter Optimization				
Regularization parameter C (SVM)				

3.1 Model selection

Two machine learning models have been adopted for the classification task: the Multinomial Naïve Bayes (NB) and the Support Vector Machine (SVM). Rana and colleagues [3] performed a study where they compared the accuracy of the two above mentioned options. Their conclusion was in accordance with the data we gathered, showing a slightly superior score of the SVM pipeline across all datasets (whereas we only employed text preprocessing).

3.2 Feature representation

Consecutive to the model selection, we had to make a choice between the two proposed ways to convert the text to numbers so that we could use it for analysis: Count Vectorizer and TF-IDF. Both methods have been tested with several options. The first option was to modify to n-gram parameter. The n-gram parameter tells the program whether to interpret the words/tokens as unigrams (1,1), bigrams (2,2), both unigrams and bigrams (1,2), trigrams (3,3) ... etc. For this task all possible combinations of n-grams between one and four (also counting all ranges between these numbers) have been tested. In order to improve the score further more, we attempted to polish our traindata by removing all uninformative elements that might interfere with the performance of the model. A variety of methods have been used for this task. Firstly, modifying the parameter max_df and/or min_df among the vectorizer options for preprocessing the text. The max_df parameter would make the model ignore all words that exceed a certain frequency. The values of the frequency applied during testing ranged from 0.1 to 1 with a step-size of 0.1. The min_df parameter

[5] `https://scikit-learn.org/stable/auto_examples/model_selection/plot_precision_recall.html`

would make the model ignore words/tokens that are strictly lower than the given number. Secondly, we attempted to use the parameter stop_words that is available in the vectorizers. If the parameter is set to 'english', a built-in stop word list for English is used. However, the performance of all models did not benefit from using the stop_words parameter. In some cases it ended up reducing its efficiency. In addition, when there was no improvement using the max_df and the stop_words parameter, we manually examined the most common words obtained from the whole trainset. If words/characters with a high frequency-count seemed uninformative in our opinion, we removed them from the trainset and examined if it improved the model.

3.3 Text prepocessing

Texts contain usually a great deal of noise and uninformative components such as stop-words and/or special symbols. Reducing the noise in the text should help improve the performance of the algorithm and speed up the process. The tokenization can be applied to split the text into smaller units to make it more interpretable by a computer. If normalization is chosen, it modifies the text using the TweetTokenizer from the nltk Python library. The normalization consists of three parts: preserving the case (a flag indicating whether to preserve the capitalisation of text, reducing the length (a flag indication whether to replace repeated character sequences of length 3 or greater with sequences of length 3) and stipping handles (a flag indicating whether to remove Twitter handles of text) [6]. At last, it is examined whether lowercasing and/or lemmatizing the words improves the performance of the model.

3.4 Parameter optimization

After the text preprocessing, model selection, and feature representation options were selected to search for the optimal model, parameters of the models were examined that could further improve the performance. The Naïve Bayes model is known for its simplicity, yet well performing classifier. Therefore we chose to not apply any parameter optimization for this model. The SVM model has many parameters that can be optimized. For this task we chose to apply parameter optimization for the regularization parameter C with values ranging from (0.001, 0.01, 0.1, 1, 5, 10). These values are chosen by inspiration of a paper that applied SVM for hate speech and offensive content detection [4].

4 Results

4.1 Gibert

The parameters used for the count section when training from the Gibert set were the following: max_df = 0.5, ngram_range = (1,2). The SVM regularization parameter is set to C=0.5. The model performed better with the use of Tokenization, Normalization and Lowercasing. Lemmatizing decreased the performance. The combination of these provided a macro average F1-score of 0.76, as can be seen the classification report in table 6. Here below are visible the five most common features that determine whether the text will be interpreted as hateful when learning from the Gibert dataset. The features on the left are from the original training set, in the middle there are the ones arising from the concatenation of the extra data with it. Lastly, on the right the ones from original set with lexicon added:

Table 5: Gibert - Top five features

TRAIN SET	TRAIN + EXTRA SET	TRAIN + LEX
hispanic not	muslim women	is hate
and hispanic	us women	youtube
other	women trans	other
if they you	my nigga	if they
you are	trans women	you are

[6] `https://www.nltk.org/api/nltk.tokenize.casual.html`

We can see a major difference between the two columns, which just indicates that the extra set contained a higher rate of overtly offensive language than the original. Contrarily to expectations, the addition of this additional data did not improve the performance of the model, but rather decrease its macro F1 score from 0.76 to 0.73. Likewise, the use of lexicon data also did not provide any benefit to the model. Figure 1 represents the confusion matrix for the optimized model, graphically shows the high degree of correct predictions in both the Hate and NoHate category.

Table 6: Classification report Gibert with optimal settings

	precision	recall	F1-score
Hate	0.76	0.75	0.75
NoHate	0.75	0.76	0.76
macro avg	0.76	0.76	0.76

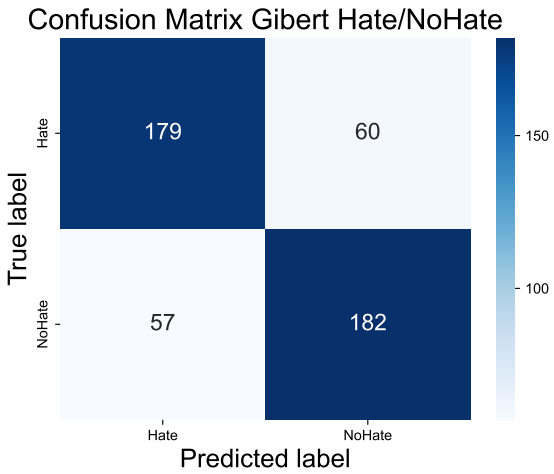

Fig. 1: Confusion table Gibert with optimal settings

4.2 TRAC

The optimized model is constructed by performing all steps given in table 4. The first step was to choose the model. SVM obtained a slightly higher score in terms of F1 macro score compared to the Naïve Bayes. The chosen model performed better with the use of Tokenization, Normalization and Lowercasing. Furthermore, applying the SVM with Count Vectorizer yielded in an increase of performance of 0.05 F1 macro compared to SVM with TFIDF. The only settings for Count Vectorizer that significantly improved the model was by applying both unigrams and bigrams (n-gram = (1,2)). In addition, as the use of the stop_words and max_df parameters decreased the performance of the model, the word-frequencies of the traindata had been examined. It was noticed that the first six words with the highest frequencies could be uninformative for the task of classifying aggressive text. The six words and their frequencies are: 'the' (7570), 'to' (6918), 'of' (5567), 'is' (5466), 'and' (5432) and 'in' (4525). Removing these words increased the performance of F1 macro with 0.02. Finally, the parameter optimization part of the model resulted in a regularization parameter C of 0.01.

The top five features regarding the label Covert Aggression (CAG) and Overt Aggression (OAG) are visible in tables 7 and 8. Both labels show similar results when adding extra data and lexicon data. Remarkable are the exact similar features for CAG in table 7 of the basic trainset and the trainset + lexicon data. From this it is expected that adding lexicon data will not contribute to the performance of classifying content labeled as CAG. The words from both tables also represent the labels accurately, e.g., 'lol' and 'haha' from table 7 are often used to express saracasm or when embarrassing someone. In addition, most words in table 8 reflect the meaning of aggressive content accurately.

Table 7: Top five features CAG

TRAIN	TRAIN+EXTRA	TRAIN+LEX
haha	lol	haha
bike	black money	bike
lol	bike	lol
reservation	modiji	reservation
modiji	reservation	modiji

Table 8: Top five features OAG

TRAIN	TRAIN+EXTRA	TRAIN+LEX
idiots	trans	hell
stupid	shame	hate
bloody	women	stupid
shame	gay	bloody
idiot	fuckg	shame

The classification report and the confusion matrix of the optimized model are shown below. As expected, the F1 macro score of CAG is tremendously low (0.36). From the confusion matrix we can conclude that the model mainly predicts the label NAG, although the the true label is CAG. Unfortunately, the F1 macro score for OAG is also very low (0.46). The confusion matrix shows that it predicts the same amount for each label given the true label OAG, which could indicate randomness in predicting the label OAG. At last, the label NAG obtained the highest F1 macro score (0.71). From the confusion matrix we can see that the model predicts the label NAG rather well.

Table 9: Classification report TRAC with optimal settings

	precision	recall	F1-score
CAG	0.26	0.58	0.36
NAG	0.81	0.64	0.71
OAG	0.55	0.40	0.46
macro avg	0.54	0.54	0.51

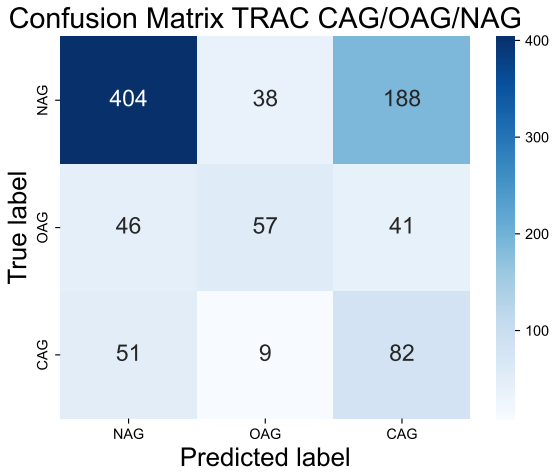

Fig. 2: Confusion table TRAC with optimal settings

At last, extra train-data and lexicon-data is added (seperately) to the optimized model. Before evaluating the results of both tasks, the word-frequencies have been re-analyzed to examine whether the uninformative words with the highest frequencies remained the same, and could therefore be removed again. This was indeed the case, and therefore we deleted the same words again from the traindata. Adding the extra traindata did not contribute to an improvement of the performance of the model. The expectation was that the label OAG would obtain higher results, as we labeled the extra hateful posts with the label OAG to counter the class imbalance. Unfortunately, the F1 macro score for OAG remained the same, as did the average macro F1 score. In addition, the performance of the model with the lexicon data added did also not increase or decrease with respect to the F1 macro average score.

4.3 OLID

With respect to model selection, SVM outperformed NB based on the basic settings for both models. Hence, there is chosen to further optimize the SVM models settings for this corpus. To begin with the data preprocessing steps, `std_prep` showed up with best results, without taken lexicons into account yet. Preprocessing steps considered in the `std_prep` function are Tokenization, Normalization and Lowercasing. Lemmatizing

- by the use of Spacy - did not yield to a significant performance lift. Next feature representation, SVM in combination with Count Vectorizer outperformed TFIDF. This is tested in combination with a variety of n-grams, as mentioned in Section 3.2. The models performance, in terms of macro F1-score, was best with a combination of unigrams and bigrams, i.e., n-gram = (1,2). Additionally, a `min_df` parameter equal to two is used in combination with the aformentioned Count Vecotrizer settings. The best option for the regularization parameter is C = 0.1.

In table 10 the top five features are shown with respect to the different datasets used for training the model. After adding the extra training data, there is only one change in the top five features, implying that this will not have a big impact on the models performance. Same holds for the lexicon data, there are some slight changes but no groundbreaking difference in performance will be expected, which will be elaborated further on in the next paragraph.

Table 10: OLID - Top five features

TRAIN SET	TRAIN + EXTRA SET	TRAIN + LEX
bitch	bitch	fuck
fuck	fuck	shit
shit	shit	fucking
fucking	stupid	stupid
idiot	idiot	disgusting

The classification report and confusion matrix of the optimized model are shown below in respectively table 11 and figure 3. It can be seen that the model almost predicted true Offensive tweets equally wrong (111) as right (129). For the true NonOffensive tweets this gap between wrong and right predicted values is much larger, i.e., recall of 0.89.

Table 11: Classification report OLID with optimal settings

	precision	recall	F1-score
NON	0.83	0.89	0.86
OFF	0.65	0.54	0.59
macro avg	0.74	0.71	0.72

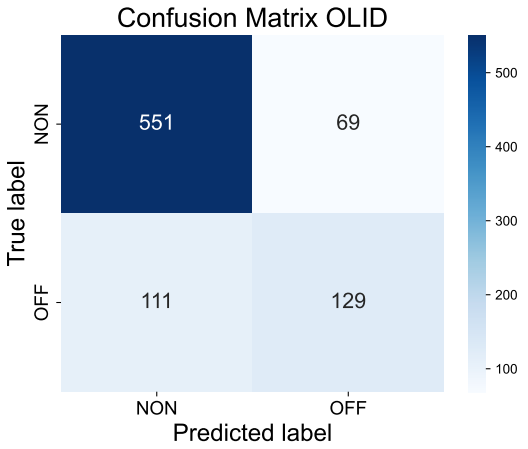

Fig. 3: Confusion table OLID with optimal settings

After adding lexicon data into the training process, the optimal model did barely changed. Performance in terms of macro F1-score is exactly the same and when looking at the new confusion matrix, there is only at max a plus/minus of 1 per category. Hence, that probably means that only two tweets are assigned to another class through the model.

At last, the extra traindata did not yield to a higher macro F1-score for the optimal model. Now more tweets are predicted as Offensive, but those did not match the true label. Hence, the macro F1-score decreased with 0.01.

5 Analysis

For this section, two experiments were chosen to perform an error analysis. Both analysis are performed for the TRAC and OLID dataset on the optimal model (i.e. with parameter optimization and without the extra dataset or lexicon data included). The analysis consists of evaluating the classification reports, an examination on the confusion matrices and the discussion of some falsely classified content.

5.1 TRAC

Overt Aggression classified as Non-Threatening Aggression / Covert Aggression The confusion matrix, shown in figure 2 displays an approximately equally distributed number of predictive labels amongst all three labels considering the true OAG label, which explains the recall score of 0.4. In contrast, the precision score is quite high (0.55), indicating a relatively high proportion of correctly predicted OAG labels from all predicted labels. The low recall score might be caused by the fact that the NAG label has more instances. After inserting the additional data with the OAG labels, it was noted that the recall score improved by 0.13, while the macro F1 score remained the same due to the decreasing precision score. In addition, as mentioned in section 2.2, it was expected that the classification of the label CAG would present difficulties, which might explain the amount of predictive CAG labels while the actual label is OAG.

Class imbalance One example of a false prediction that might be the cause of the class imbalance is placed in example 1. If the data would be evenly distributed, it is expected that the word "bitch" would be correctly identified as overt aggression and not as non-threatening aggression.

1. U mother fuc**** abp editorY not posting about sukuma Son of bitch y diverting issue

Contextual Underneath is another example of obvious overt aggression content that is predicted as non-threatening content. The prediction of example 2 is less surprising, due to the lack of interpretation of "fuc****" and the lack of ability to identify misspelled words ("Bulshit").

2. We have caught your official butcher kalboshen.... So you trying to be innocent victim.... Bulshit india

Covert Aggression classified as Non-Threatening Aggression / Overt Aggression Covert aggression indicates an indirect attack towards an individual or group. As curse-words are not included, the structure and text will show many similarities with content labeled as non-threatening aggression. The confusion matrix 2 confirms this conclusion. As can be seen, a large number of predictions are classified as NAG although the actual label is CAG. However, the relatively high recall score of 0.58 indicates that more CAG labels have been correctly predicted amongst the other predicted values. As expected, a small number of cases with the true label CAG have been predicted the label NAG (9). Furthermore, the precision score of CAG is tremendously low (0.29). This is mainly caused by the imbalance of the NAG label, as the confusion matrix shows 188 cases of NAG, and only 82 and 41 cases of CAG and OAG.

Class imbalance Below is another example of a false prediction. This post is identified as NAG, although the actual label is CAG. It is expected that the post would correctly predicted with the use of 'bad for'. As this might indicate an indirect attack to another person/group in the context of 'bad'.

3. One man show. Dictater are always bad for ony country

Contextual The fourth case of mis-classified content is presented in example 4. For this example, the model lacks the ability to recognize the sound of irony. With more data, it might be that the model will recognize the content correctly by the triple-dots. The fifth case is an example of a lack of understanding. The 'MMS is in slow motion' is an indirect statement that expresses the malfunctioning of a game.

4. He took one week training from Sonia for this speech...
5. How can i increase the speed of playing !! MMS is in slow motion

Non-Threatening Aggression classified as Covert Aggression / Overt Aggression At last, the predicted content of the NAG labels have been examined. As shown in the classification report the results of the precision (0.81), recall (0.64) and F1 score (0.71) are very high compared to the other labels. One of the main reasons is the imbalanced data. Another reason is the fact that it is easier to classify content as non-threatening aggression (due to the lack of aggressive words) than to classify aggressive content.
One case that has the actual label of NAG but is predicted with CAG is example number 6. This example contradict our previous conclusion regarding the triple-dots. Therefore we might conclude that the model does learn from the dots and therefore identifies this case as covert aggression, indicating the difficulty of correctly classifying content as CAG.

6. He took one week training from Sonia for this speech...

5.2 OLID

Offensive classified as NonOffensive False negatives for the Offensive class are the most common errors made by the model. Section 4.3 showed that the recall score for offensive content is 0.54, which is relatively high. This implies that the model predicted many cases as NonOffensive while their true label Offensive was. Probably the most obvious reason is due to class imbalance. What implies that there are to less training examples labeled as Offensive, to recognize every kind of offensive content in the testdata. Another reason is the lack of context or thoughtfulness of the model. At last, a remarkable category is content wherein the writers are offensive towards themselves.

Class imbalance Some examples of clear offensive content are shown in 1 and 2. In these cases the word "niggas" appear, but is classified by the model as NonOffensive nevertheless.

1. #GeoffreyOwens niggas don't really understand...levels to career pursuits
2. Another season for Tom Brady to give it to these niggas

Contextual In case 3, the hashtag "#PedoWood" is mentioned. This refers to a portmanteau of Hollywood and pedophile, implying Hollywood is notorious for pedophilia[7]. In case 4, contextual knowledge is needed to understand that calling someone, and his "ilk", "tone-death" is offensive. Furthermore, in case 5, the pronunciation of getting "rid" of someone is a direct offense. Therefore, contextual information is needed to fully understand this kind of cases.

3. @USER #ThoseThatAreTheLoudest are afraid, JUSTICE is coming #Bono is a scam just like the #ClintonFoundation #PedoWood #Haiti URL
4. #NEWS Jeff Sessions: If you want more death, 'listen to the ACLU, Antifa, Black Lives Matter'" URL #CNBC"
5. #tytlive Trump needs to go because he is a danger to the country and the world. Any way we can get rid of him is good

Self-offensive Even though words as "dickhead" and "fuckup" does appear in cases 10 and 11, the model classified it as NonOffensive nevertheless. An interesting aspect of both cases is the offenses towards the writers selves.

6. Am I a dickhead ???? Probably yes
7. 28, 27, 25 and 21 but like,, it's still really miserable and unpleasant for us?? And like they even told me how they weren't happy and would have got divorced before I was even born so I'm like cool cool I was literally born into hatred that's awesome no wonder I'm such a fuckup

NonOffensive classified as Offensive False positives for the NonOffensive class are present in lesser numbers than true negatives for the Offensive class. What resulted in a precision of 0.65. The error analysis have find out that these errors come from words mostly associated with offensive content. Both cases 9 and 10 are for sure positive. Through the use of words as "shit" and "fuck" to describe something good is not per definition offensive. However these words appear much often in offensive cases and therefore 9 and 10 are

[7] `https://www.urbandictionary.com/define.php?term=pedowood`

predicted as offensive by the model. Calling someone "nasty", e.g. case 11, is maybe somewhat inappropriate, but context is unknown in this situation. In our opinion, it is doubtful whether this cases should be labeled as offensive or not. Anyway, the annotators should discuss where the limit is on how far you can go to use these kinds of words as an adjective.

8. #Room25 is actually incredible, Noname is the shit, always has been, and I'm seein her in like 5 days in Melbourne. Life is good. Have a nice day.
9. An American Tail really is one of the most underrated animations ever ever ever. Fuck I cried in this scene
10. @USER @USER @USER She is just nasty

6 Conclusion and future work

In this paper, the given pipeline is used to investigate whether dataset characteristics and experimental setup effects automatic classification for three different corpora. Additionally, to the optimal model extra training data and lexicons were added. We have seen that SVM outperforms the NB model for all three datasets. The results, of TRAC data, showed low scores for the classification of OAG. Language on Facebook could be more neat than on Twitter and since we were only considering the Facebook data, this could be a possible explanation. It is a nice future work to investigate whether language on Facebook is indeed more neat, and consequently language on Twitter more aggressive. The error analysis showed that most of the false negatives come from class imbalance, the lack of contextual knowledge and self-offensive content. Moreover, the model showed the lack of interpreting words as aggressive due to misspellings. For future work it is recommended to adopt a method that incorporates this task.

Appendix

Contributions

Noa - optimizing model for dataset Kumar, written chapters: 2.2, 3.0, 3.3, 3.4, 4.2, 5.0, 5.1
Matthias - optimizing model for dataset Zampieri, written chapters: 1, 2.0, 2.3, 2.4, 4.3, 5.2, 6
Matteo - optimizing model for dataset Gibert, written chapters: 2.1, 3.1, 3.2, 4.1

References

1. de Gibert, O., Perez, N., García-Pablos, A., Cuadros, M.: Hate speech dataset from a white supremacy forum. In: Proceedings of the 2nd Workshop on Abusive Language Online (ALW2). pp. 11–20. Association for Computational Linguistics, Brussels, Belgium (Oct 2018). https://doi.org/10.18653/v1/W18-5102, https://aclanthology.org/W18-5102
2. Kumar, R., Reganti, A.N., Bhatia, A., Maheshwari, T.: Aggression-annotated corpus of Hindi-English code-mixed data. In: Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018). European Language Resources Association (ELRA), Miyazaki, Japan (May 2018), https://aclanthology.org/L18-1226
3. Rana, S., Singh, A.: Comparative analysis of sentiment orientation using svm and naive bayes techniques. In: 2016 2nd International Conference on Next Generation Computing Technologies (NGCT). pp. 106–111 (2016). https://doi.org/10.1109/NGCT.2016.7877399
4. Ratan, S., Sinha, S.: Svm for hate speech and offensive content detection. Forum for Information Retrieval Evaluation (december 2021), https://www.researchgate.net/publication/360912255_SVM_for_Hate_Speech_and_Offensive_Content_Detection
5. 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
6. Zhang, Z., Luo, L.: Hate speech detection: A solved problem? the challenging case of long tail on twitter. CoRR **abs/1803.03662** (2018), http://arxiv.org/abs/1803.03662