Proceedings of the 15th Conference on Natural Language Processing (KONVENS 2019)

FKIE - Offensive Language Detection on Twitter at GermEval 2019

Theresa Krumbiegel
Fraunhofer FKIE
Fraunhoferstraße 20
53343 Wachtberg
theresa.krumbiegel@
fkie.fraunhofer.de

Abstract

We describe our submissions to the Shared Task on Identification of Offensive Language at GermEval 2019. We take part in all three subtasks, utilizing a Support Vector Machine (SVM) for subtasks 1 and 2, and a Long short-term memory (LSTM) neural net as well as a Convolutional neural net (CNN) for subtask 3. We obtained a macro-F1 score of 75.21 for subtask 1, 55.42 for subtask 2 and 64.20 for subtask 3 on a development set that was split from the overall training set provided by the organisers.

1 Introduction

The interest in systems that are able to classify offensive language on social media platforms such as Twitter has grown over the last years. Several scientific contributions deal with the development of such systems (cf.(Zampieri et al., 2019), (Hakimi Parizi et al., 2019)). As a consequence of this increased interest, critical voices can also be heard regarding the way in which offensive language is detected. Davidson et al. (2019), for example, report on finding racial bias in datatsets that are used to train detection systems and Silva et al. (2016) state that designing an objective definition of hate speech is invariably difficult because of the complex context in which it needs to be integrated.

Within the frame of the GermEval shared task, offensive language is defined as ”hurtful, derogatory or obscene comments made by one person to another” (Ruppenhofer et al., 2018). Three tasks are given regarding the detection of such language. The first of these is a coarse-grained binary classification task that aims at the general detection of offensive tweets. The categories OFFENSE and OTHER need to be assigned.

(1) @SusanBrenning In Sachen Verrat war die Kirche schon immer groß. OFFENSE

(2) @Doodoofist Das mach dir was zu essen Kamerad OTHER

In the second, fine-grained task, offensive tweets have to be further categorized into PROFANITY, INSULT and ABUSE where profanity depicts the least and abuse the most offensive class.

(3) Wie viel Oblaten muss ich denn jetzt essen bis ich ein Steak von Jesus zusammen hab?^^. PROFANITY

(4) Sagt mal, kommt .es nur mir so vor, oder ist das Staasfunk Fernsehprogramm wirklich so scheiße? INSULT

(5) @YigidoYosi58 @Mesut__A @ntvde @ntv Bald seid ihr alle hier ”Entsorgt”! ABUSE

The final task is binary and is directed at the distinction between EXPLICIT and IMPLICIT offense.

(6) @sozialromantik Eine Schande für Deutschland ist diese BOLSCHEWISMUS Regierung! EXPLICIT

(7) Was tut Ihr, wenn Ihr merkt, dass jemand grün wählt? IMPLICIT

2 Classification Approach

We used different system designs to solve the specified tasks. For subtask 1 and 2, we choose a SVM system that was created with the help of scikit-learn (Pedregosa et al., 2011). For subtask 3 we used a LSTM neural net as well as a CNN implemented with Keras (Chollet, 2015).

2.1 Data

To develop classification systems that can achieve satisfying results for the described tasks, a sufficient amount of training data is essential. The organisers provided two sets of training data, one for subtasks 1 and 2 and another one for subtask 3. The set for subtask 3 does not include additional tweets but categorizes OFFENSE examples taken from the first data set further into IMPLICIT and EXPLICIT. The data distribution of all training sets is presented in table 1.

Class	Tweets	%
Subtask 1		
Offense	4117	33.32
Other	8359	66.68
Total	12536	100
Subtask 2		
Profanity	271	2.16
Insult	1601	12.77
Abuse	2305	18.39
Other	8359	66.68
Total	12536	100
Subtask 3		
Implicit	259	13.23
Explicit	1699	86.77
Total	1958	100

Table 1: Training data distribution

To evaluate and optimize our systems during the training phase, we took a random sample of 20% of the provided data to form a development set for each task. The distribution of these samples can be found in table 2.

Class	Tweets	%
Subtask 1		
Offense	840	33.49
Other	1668	66.51
Total	2508	100
Subtask 2		
Profanity	41	1.63
Insult	321	12.80
Abuse	478	19.06
Other	1668	66.51
Total	2508	100
Subtask 3		
Implicit	47	15.61
Explicit	254	84.39
Total	301	100

Table 2: Development set distribution

2.2 Feature Description

For the classification with the SVM, a number of features were used during training. We combined these features into groups and assigned transformer weights to them.

Group	Feature
Sentiment	Sentiment Score
Character content	Character n-grams
Tweet content	Number of words Number of mentions Number of capital words Number of hashtags Number of emojis Number of exclamation and question marks Number of URLs
Pre-process	Removal of stop words Lemmatization

Table 3: Features used in subtask 1 and 2

The sentiment scores were extracted with the help of the Python module textblob-de [1]. We used character n-grams that are weighted by their TF-IDF. Lemmatization was implemented with the Spacy lemmatizer [2]. The described features were used for subtask 1 as well as subtask 2.

For the LSTM neural net and the CNN that were utilized in subtask 3, we merely pre-processed the data. No specific features were fed into the net. During pre-processing, we converted the text to lowercase and removed all punctuation and stop words. A German stop word list was acquired from the Python module stop-words [3]. Furthermore, we removed the line break token "lbr" and stemmed the text with the GermanStemmer by NLTK [4]. To be able to use the Tweets as input for the neural net, we created sequences out of the examples given, with a maximum length of 100. Shorter instances were padded.

3 Preliminary Results

We present our preliminary results. These results were obtained by testing our systems on self-compiled development sets that comprise 20% of the training data respectively. In addition, we report on 10-fold cross validation results.

[1] https://textblob.readthedocs.io/en/dev/
[2] https://spacy.io/api/lemmatizer
[3] https://pypi.org/project/stop-words/
[4] https://www.nltk.org/_modules/nltk/stem/snowball.html

3.1 Subtask 1 and Subtask 2

We evaluated our systems with the described development sets. During the optimization phase we firstly experimented with different character n-gram ranges. The decision to start the evaluation with a search for the best performing character n-grams is based, among others, on findings of last year's GermEval Shared Task on Identification of Offensive Language that show that character n-grams are rewarding features (Ruppenhofer et al., 2018). Detailed results for subtask 1 that were obtained during the development process can be found in tables 4 and 5.

Feature	**Macro-F1 Development**
Char 1-4 grams	67.62
Char 1-5 grams	**69.58**
Char 3-6 grams	**69.65**
Char 3-7 grams	69.38

Table 4: Character n-gram evaluation

Due to the minimal difference between the performance of character 1-5 grams and character 3-6 grams, we decided to continue the optimization of our system with both ranges. Character 1-5 grams outperformed character 3-6 grams regarding the classification of OFFENSE slightly (F1-score of 55.60 vs. 55.52) while character 3-6 grams exhibited better results for the OTHER class (F1-score of 83.79 vs. 83.55).

Feature Combination	**Macro-F1 Development**
1-5 grams + tweet content	68.85
1-5 grams + sentiment	69.27
1-5 grams + pre-process	74.40
1-5 grams + sentiment + pre-process	74.40
1-5 grams + sentiment + tweet content	68.94
1-5 grams + pre-process + tweet content	**74.66**
1-5 grams + sentiment + pre-process + tweet content	**74.65**
3-6 grams + tweet content	69.05
3-6 grams + sentiment	70.27
3-6 grams + pre-process	74.34
3-6 grams + sentiment + pre-process	**74.44**
3-6 grams + sentiment + tweet content	69.93
3-6 grams + pre-process + tweet content	73.91
3-6 grams + sentiment + pre-process + tweet content	74.31

Table 5: Feature evaluation

As can be seen in table 5, we achieved the best macro-F1 score, 74.66, when using character 1-5 grams in combination with pre-process and tweet content. It can be observed that even though the feature group tweet content does not improve the results in combination with character 1-5 grams alone, it *does* contribute to a higher macro-F1 score when used together with other features. We achieved a nearly equally high macro-F1 score of 74.65 with the combination of 1-5 grams and all other features. In the case of character 3-6 grams, a combination of all features except tweet content yields the best results, 74.44. We continue our training and evaluation process with the combination of character 1-5 grams, sentiment scores, tweet content and pre-process as well as the combination of character 1-5 grams, tweet content and pre-process.

As all feature groups were combined in a feature union, we were able to assign transformer weights to the different groups. The best performing combination was the following:

- Character content: 0.8
- Sentiment: 0.6
- Tweet content: 1.0
- Pre-process: 0.8

We obtained a final, highest macro-F1 score of 75.21. This score was obtained by using character 1-5 grams, pre-process and tweet content. The additional inclusion of sentiment scores yields a slightly lower macro-F1 score of 75.02.

	Precision	Recall	F1	Support
OTHER	80.68	90.89	85.48	1668
OFFENSE	75.83	56.79	64.94	840
macro avg	78.26	73.84	**75.21**	2508

Table 6: SVM results subtask 1

10-fold cross validation of the highest scoring system results in a mean macro-F1 score of 73.96.

For the second, fine grained subtask we implemented the same optimization process as for subtask 1. We found the best feature combination for the SVM to be character 3-6 grams and pre-process.

We achieved a macro-F1 score of 55.42 on our development set. Other combinations that yielded relatively high macro-F1 scores were character 3-6 grams, pre-process, tweet content and sentiment, (55.05), and character 3-6 grams, pre-process and sentiment (54.97).

	Precision	Recall	F1	Support
OTHER	78.15	92.63	84.77	1668
PROFANITY	60.71	41.46	49.28	41
INSULT	53.80	30.84	39.21	321
ABUSE	60.50	40.38	48.43	478
macro avg	63.29	51.33	**55.42**	2508

Table 7: SVM results subtask 2

We evaluated the best performing system with 10-fold cross validation and obtained a mean macro-F1 score of 46.18. The discrepancy to the results achieved when using a fixed development set can and should be attributed to variations in the data.

3.2 Subtask 3

For subtask 3, we trained and tested a LSTM neural net as well as a CNN. Even though we achieved the best results for subtask 1 and 2 with the SVM model, we obtained distinctly better results for subtask 3 when training and evaluating the corresponding data on a neural net (macro-F1 score of 0.46 with the SVM vs. 0.64 with a neural net). This was due to difficulties of predicting IMPLICIT tweets with the SVM. We obtained a very low F1 score of 02.82 for this category which impacted the final macro-F1 score negatively.

The input for the neural nets was pre-processed as described in subsection 2.2. With the LSTM neural net, we achieved a macro-F1 score of 64.20, the CNN produced a score of 64.06.

	Precision	Recall	F1	Support
EXPLICIT	89.63	85.04	87.27	254
IMPLICIT	36.67	46.81	41.12	47
macro avg	63.15	65.92	**64.20**	301

Table 8: LSTM neural net results

	Precision	Recall	F1	Support
EXPLICIT	88.89	88.19	88.54	254
IMPLICIT	38.78	40.43	39.58	47
macro avg	63.83	64.31	**64.06**	301

Table 9: CNN results

3.3 Submitted Results

The following files were submitted:

1. fkie_coarse_1.txt — SVM, character 1-5 grams, pre-process, tweet content
2. fkie_coarse_2.txt — SVM, character 1-5 grams, pre-process, tweet content, sentiment
3. fkie_fine_1.txt — SVM, character 3-6 grams, pre-process
4. fkie_fine_2.txt — SVM, character 3-6 grams, pre-process, sentiment
5. fkie_fine_3.txt — SVM, character 3-6 grams, pre-process, tweet content, sentiment
6. fkie_implicit_1.txt — LSTM
7. fkie_implicit_2.txt — LSTM
8. fkie_implicit_3.txt — CNN

For subtask 1 (coarse) we submitted one run (fkie_coarse_1.txt) which uses character 1-5 grams, pre-process and tweet content as features for the SVM and another run (fkie_coarse_2.txt) which in addition uses sentiment scores.

The first submission for subtask 2 (fkie_fine_1.txt) was obtained by using a SVM with character 3-6 and pre-process. For the second submission (fkie_fine_2.txt), we again used character 3-6 grams, pre-process and added sentiment scores. The third submitted run (fkie_fine_3.txt) uses all available features.

For subtask 3 (implicit), three runs were submitted. Two of these (fkie_implicit_1.txt, fkie_implicit_2.txt) include results obtained with the LSTM neural net. The other one (fkie_implicit_3.txt) presents the CNN results.

4 Discussion

In general, the binary classification systems yield better macro-F1 scores than the multi-class system. This was to be expected. The best performing system is the one that focuses on the simple distinction between offensive and not offensive tweets. This is also intuitive: categorizing offensive tweets into profanity, insult, abuse or explicit, implicit, requires more precise feature engineering.

A fine-grained classification is, in this case, additionally difficult as the annotation of some subcategories is at times not coherent, e.g.:

> (8) @KingGeorgVI @EngelGert Ich kann es nicht mehr sagen. Bild und Artikel sind verschwunden und ich habe es nicht gespeichert. Das Bild zum Tweet ist ebenfalls weg. Sorry. OFFENSE ABUSE

> (9) @JuttaMBrandt @jouwatch Ich muss da nicht überlegen. OFFENSE INSULT

The examples above are annotated as OFFENSE even though they do not appear to be insulting or

abusive. Especially example (8), which is categorized as abuse, the *most* offensive class, does not include offensive content but rather conveys a factual and even apologetic tone.

We assume that the categorization of such instances is based on background knowledge that is not accessible to us and subsequently not accessible to the classification systems. Still, it is advisable to train the systems on the provided data set and therefore inevitably on examples that break ranks, to make them applicable to the test set.

Regarding subtask 3, it is instinctive that the detection of EXPLICIT instances can be achieved more easily than that of IMPLICIT ones. The term implicit as such can be defined as "capable of being understood from something else though unexpressed" (Merriam-Webster, 2011) which already hints at the problem that something that is not overtly expressed might be difficult to identify. The impact of this can be observed clearly in the results depicted in tables 8 and 9. In addition, the distribution of EXPLICIT and IMPLICIT tweets in the training data is skewed (86.77% EXPLICIT, 13.23% IMPLICIT). This complicates the eventual detection of IMPLICIT tweets in the test data.

5 Conclusion

We presented our submission to GermEval Task 2, 2019 - Shared Task on the Identification of Offensive Language. We described the generation and implementation of a SVM, a CNN and a LSTM neural net as well as feature engineering and preprocessing strategies that were used.

For future work in this area, some issues should be considered and, if possible, improved. The data set that was provided for the training of the systems should be more balanced with regard to the individual categories. Especially for subtask 3, the small number of IMPLICIT examples was problematic. In addition, it would be helpful if the data was annotated in a more consistent manner. A data set that is fully coherent will quite likely improve the performance of the classification systems in the end.

Acknowledgements

We would like to thank our colleague Albert Pritzkau for his invaluable help and insights during the process of solving the task of offensive language detection.

References

Francois Chollet. 2015. Keras. `https://keras.io`. [Online; accessed 24-July-2019].

Thomas Davidson, Debasmita Bhattacharya, and Ingmar Weber. 2019. Racial bias in hate speech and abusive language detection datasets. *CoRR*, abs/1905.12516.

Ali Hakimi Parizi, Milton King, and Paul Cook. 2019. UNBNLP at SemEval-2019 task 5 and 6: Using language models to detect hate speech and offensive language. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 514–518, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Merriam-Webster. 2011. implicit. `https://www.merriam-webster.com/dictionary/implicit`. [Online; accessed 14-August-2019].

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Josef Ruppenhofer, Melanie Siegel, and Michael Wiegand. 2018. Overview. In *Proceedings of the GermEval 2018 Workshop, 14th Conference on Natural Language Processing (KONVENS 2018)*, pages 1–10.

Leandro Silva, Mainack Mondal, Denzil Correa, Fabricio Benevenuto, and Ingmar Weber. 2016. Analyzing the targets of hate in online social media. In *Proceedings of the Tenth International AAAI Conference on Web and Social Media (ICWSM 2016)*, pages 687–690, Palo Alto, California, USA. AAAI Press.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019. SemEval-2019 task 6: Identifying and categorizing offensive language in social media (OffensEval). In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 75–86, Minneapolis, Minnesota, USA. Association for Computational Linguistics.