The HUIU Contribution for the GermEval Shared Task 2

Melanie Andresen, Melitta Gillmann, Jowita Grala, Sarah Jablotschkin, Lea Röseler, Eleonore Schmitt†, Lena Schnee, Katharina Straka, Michael Vauth, Sandra Kübler‡, Heike Zinsmeister

Universität Hamburg, †Universität Bamberg, ‡Indiana University

{melanie.andresen, melitta.gillmann, sarah.jablotschkin}@uni-hamburg.de
{Jowita.Grala, Lea.Roeseler}@studium.uni-hamburg.de
{Lena.Schnee, Katharina.Straka}@studium.uni-hamburg.de
{michael.vauth, heike.zinsmeister}@uni-hamburg.de
eleonore.schmitt@uni-bamberg.de, skuebler@indiana.edu

Abstract

In this paper, we present the HUIU system (a collaboration of University of Hamburg and Indiana University) for the GermEval 2019 shared task 2, subtask 1 – the coarse-grained classification of tweets into the classes OFFENSE or OTHER. Our system uses linear SVMs with character n-grams (5 \( \leq n \leq 10 \)), POS n-grams (3 \( \leq n \leq 9 \)) and the tweet’s length in number of tokens as features. We obtain a macro-averaged F-score of 65.32 on the test data.

1 Introduction

In this paper, we report on the HUIU team’s submission to the GermEval Task 2, 2019 - Shared Task on the Identification of Offensive Language. Three subtasks were offered. Subtask I was a binary classification task and required discriminating offensive from non-offensive tweets. Subtask II consisted of a more fine-grained classification: Each of the offensive tweets had to be marked with one of the following labels: PROFANITY, INSULT, ABUSE. Subtask III required labeling the offensive tweets as explicitly or implicitly offensive. We participated in Subtask I, i.e., the detection of offensive language in binary classified twitter data.

Our contribution is the result of a class project conducted at the University of Hamburg. The authors participated in a 6-day compact class that provided an introduction to machine learning for linguistics and digital humanities, under the supervision of Kübler and Zinsmeister. Most of the participants had basic knowledge in programming, but no experience with machine learning. The class was structured to provide a practical introduction to machine learning. Therefore, the Shared Task offered a good opportunity to familiarize the participants with every step in the process of translating a problem into a machine learning problem, deciding on a machine learning algorithm, specifying feature sets, extracting features, and training the machine learning algorithm. In addition to this task, the group also participated in GermEval 2019 Task I on hierarchical classification of blurbs [reference to be added in the final version].

Using the python library scikit-learn (Pedregosa et al., 2011), we tested different models and features for the binary classification task of identifying offensive tweets. A bag-of-words approach which employs a linear SVM classifier using character n-grams combined with additional features yielded the best results.

The rest of the paper is structured as follows: We will briefly present the best systems of last year’s GermEval Shared Task as well as this year’s SemEval Shared Task in the section 2. In section 3, we will describe experimental setup, i.e., the data of our Shared Task, how we preprocessed the tweets, which features we extracted, which classifier algorithm, implementation, and evaluation we used for our experiments. The best scores that we achieved during development are presented together with the final test scores in section 4. For our overall ranking in the Shared Task we have to refer to the summary published at the Workshop in Erlangen 2019. At the time of submitting this paper we did not have this information. In section 5, we will conclude our paper with a short summary and outlook on additional features and methods that we have not taken into account.

2 Related Work

The GermEval 2019 task on the Identification of Offensive Language is the second edition of the original task from 2018 (Ruppenhofer et al., 2019). This year’s task is based on a different data set than last year’s. But parallel to 2018, Subtask I
requires a binary classification system that discriminates between offensive and non-offensive tweets. In 2018 the team that reached the highest results (Montani and Schüller, 2019) extracted as features character n-grams, stemmed token n-grams, the TF/IDF scores of both feature classes, and word embedding vectors. The TF/IDF scores of the token n-grams proved to be their most important features, i.e., removing those from the model caused a large drop of the F1-score. Their classification system was an ensemble of supervised learning methods (Logistic Regression and Random Forests) implemented in scikit-learn (Pedregosa et al., 2011). Montani and Schüller (2019) reported worse results using deep learning models (LSTM, CNN, Convolution+GRU).

The GermEval shared tasks are the German equivalent of the SemEval-2019 Task 6 (OffensEval) Subtask A (Zampieri et al., 2019), which uses a data set consisting of English tweets. The data set is also remarkably larger than the one used in the GermEval 2018 task: It comprises more than 14,000 tweets (Zampieri et al., 2019), as compared to the approximately 4,000 tweets in the current task. In contrast to the results reported by Montani and Schüller (2019), the best performing team at OffensEval Subtask A used a deep learning model – BERT (Devlin et al., 2018), based on bidirectional training of the attention model Transformer (Liu et al., 2019).

As discussed in the introduction, our goal of participating in the Shared Task was to acquire a basic understanding of machine learning in the setting of a compact introductory course. Therefore, we chose a common and easy to adapt SVM algorithm with a number of features, as described in the following section, and did not take into consideration prior more elaborated attempts at approaching this specific machine learning problem.

### 3 Experimental Setup

#### 3.1 Data Set

We used the training and test data sets provided by the Shared Task, which consisted of 3,995 manually annotated tweets. Each tweet was labeled either as OFFENSE or as OTHER. On the second annotation level, each of the tweets of the category OFFENSE was marked with one of the following more fine-grained labels: ABUSE, PROFANITY, INSULT. Subtask I, in which we participated, did only take the binary coarse-grained labels into account.

<table>
<thead>
<tr>
<th></th>
<th>OFFENSE</th>
<th>OTHER</th>
<th>sum</th>
</tr>
</thead>
<tbody>
<tr>
<td>train</td>
<td>1,141</td>
<td>2,455</td>
<td>3,596</td>
</tr>
<tr>
<td>dev</td>
<td>146</td>
<td>253</td>
<td>399</td>
</tr>
<tr>
<td>sum</td>
<td>1,287</td>
<td>2,708</td>
<td>3,995</td>
</tr>
</tbody>
</table>

Table 1: Distribution of OFFENSE and OTHER in our training and development splits.

Examples (1) to (3) show example tweets from the annotated data set provided by the Shared Task. The annotation guidelines can be found online in the repository of the Shared Task 2018.

1. @DrDavidBerger Die wirklichen Rassisten sitzen in der GroKo und bei den Grünen (@DrDavidBerger The real racists are sitting in the GroKo and in the Green Party)

   OFFENSE ABUSE

2. Sein Charakter war ihm wichtiger anstatt als billige Nute für Korrupte Regierungen zu arbeiten. Er hat das Leben begriffen (‘His character was more important to him instead of working as cheap whore for corrupt governments. He understands life’)

   OFFENSE PROFANITY

3. @de_sputnik Eine Weltherrschaft führt zum Krieg bis zum bitteren Ende (@de_sputnik World domination leads to war to the bitter end)

   OTHER OTHER

In order to optimize our system, we split the provided training set into 90% for our actual train(ing) set and 10% for our dev(elopment) set by taking every tenth instance for development. Table 1 shows the distribution of tweets and coarse-grained labels in the train and dev set respectively. For the final submission, we trained the system on the complete training set.

#### 3.2 Extracted Features

For preprocessing, we tokenized and part-of-speech (POS) tagged the data. We used the python implementation of twokenizer, a tokenizer especially designed for twitter data (Owoputi et al., 2013). For POS tagging, we employed TnT (Brants, 2013).
1998), trained on the Tübingen Treebank of Written Language (Tüba-D/Z) (Telljohann et al., 2006), version 10, assigning STTS labels (Schiller et al., 1999).

As basic features, we extracted token and POS n-grams for bag-of-words approaches of various classifiers, see Section 3.3 for the description of the classifiers.

We found that for our final system the bag-of-words approach was most effective when using character n-grams of the tokens (crossing word boundaries) combined with POS n-grams. For the tokens, the best results were achieved with a range of 5-10 characters; for the POS tags, a range of 3-9 words led to the best results.

In order to identify additional features that help to improve our model, we extracted further textual features per tweet. First, we counted the number of tokens per tweet as well as the number of @’s and #’s. Our hypothesis was that the emotional language of offensive tweets differed from other tweets in length and in the number of addressing terms and hashtags. In addition, we tested the length of elements starting with # because we assumed that in emotional tweets writers tend to use hashtags consisting of longer phrases or even full sentences. Therefore, we also determined the mean length of hashtags. Since we assumed that offensive tweets might be characterized by a specific use of punctuation marks and their combinations, we counted the occurrences of the following standalone punctuation marks . , ! ? as well as sequences of more than one of the same or different of these punctuation marks. We added an n-gram analysis of the (unicodes of) emojis extracted from the data, which we tested on character as well as on word basis.

Figure 1 illustrates the distribution of three of these features in our train set. The boxplots show that the difference are small at best. Most of our expectations have not been met.

Among all these additional features, only the number of words per tweet improved our original results based on character and POS n-grams. We will present detailed results in Section 4.

3.3 Methodology

We used the machine learning library scikit-learn (v0.20.1) (Pedregosa et al., 2011) for Python (v3.7.1) and selected the Support Vector Classifier as our model. We achieved best results on our development data with a linear model and setting the constructor option probability to True along with the remaining default settings. Additional experiments with the Random Forest Classifier, including grid search for parameter tuning, did not yield better results.

3.4 Evaluation

For evaluation, we used the scorer provided by the shared task. It reports accuracy as percent correct, precision and recall for each subset (OFFENSE, OTHER), and the macro-averaged F1-score which is the harmonic mean of the results of the two subsets. Averaging this way makes sure that in unbalanced settings, in which one subset is much larger than the other, the results on the larger subset do not obliterate the results on the smaller one. In case of the training and development data, the subset

---

Figure 1: Boxplots of example feature distributions in the train set (n=3,956).
Table 2: Results on the development set and on the final test set (Average= macro-averaged score, Perc.= percent correct, corr.= number of correct tweets; total= number of all tweets P=precision, R=recall, F= F1-score).

<table>
<thead>
<tr>
<th></th>
<th>Accuracy</th>
<th>OFFENSE</th>
<th>OTHER</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Perc.</td>
<td>corr.</td>
<td>total</td>
<td>P</td>
</tr>
<tr>
<td>dev</td>
<td>76.94</td>
<td>307</td>
<td>399</td>
<td>84.62</td>
</tr>
<tr>
<td>test</td>
<td>72.58</td>
<td>2,200</td>
<td>3,031</td>
<td>64.76</td>
</tr>
</tbody>
</table>

Table 3: Results of the ablation study and a model with all features (on the development set, n=399, w/tw= number of words per tweet, p/tw= number of periods per tweet, Average= macro-averaged).

<table>
<thead>
<tr>
<th></th>
<th>Accuracy</th>
<th>OFFENSE</th>
<th>OTHER</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>char POS w/tw p/tw</td>
<td>75.94</td>
<td>303</td>
<td>82.05</td>
<td>43.84</td>
</tr>
<tr>
<td>char POS w/tw</td>
<td>76.94</td>
<td>307</td>
<td>84.62</td>
<td>45.21</td>
</tr>
<tr>
<td>char POS</td>
<td>76.44</td>
<td>305</td>
<td>84.21</td>
<td>43.84</td>
</tr>
<tr>
<td>char</td>
<td>76.19</td>
<td>304</td>
<td>84.00</td>
<td>43.15</td>
</tr>
<tr>
<td>all features</td>
<td>74.94</td>
<td>299</td>
<td>80.26</td>
<td>41.78</td>
</tr>
</tbody>
</table>

OTHER was much larger than the subset OFFENSE, cf. Table 1.

We optimized our system for the macro-averaged F1-score on our development set, since this score was the official ranking function in the shared task.

4 Results

4.1 Official Shared Task Results

Our best system on the development set achieved a macro-averaged F1-score of 65.32 on the shared task’s test data, see Table 2.

Overall, the system did not generalize well to the final test data. We observe a loss of about 6 points in macro-averaged F1-score from 71.45 on the development data to 65.32 on the test data. The main decrease is due to a loss of about 20 points in precision on the OFFENSE class, followed by about 14 points in recall. The effect on the OTHER class is much smaller with only about 3 points loss in the F1-score from 83.97 to 82.02.

4.2 Ablation Study

We tested the best bag-of-words setting (character n-grams of size 5-10 and POS n-grams of size 3-9) with different additional feature combinations. Table 3 shows a selection of the results. The full version includes the features Words per tweet, @’s per tweet, #’s per tweet, max_lengths, multi word lengths, commas per tweet, periods per tweet, exclamation marks per tweet, question marks per tweet, punctuation sequences longer than one per tweet.

Using only character n-grams provides a solid baseline. Adding POS n-grams improves precision in both classes marginally. The best combination used only the number of words per tweet as an additional feature. It outperformed models with other feature combinations only slightly. The difference was mostly due to improved recall of offensive tweets (of about 1.4 points in comparison to the model based on characters and POS n-grams only).

Interestingly, adding the number of periods to the feature set decreased results across both classes. The full version performed systematically worse than systems with less features. We assume that other features are either not as characteristic for offense as we thought or simply were to sparse in the data to affect the training in a positive way.

5 Conclusion and Future Work

By participating in the GermEval Shared Task in the setting of a compact introductory course we learned how to conduct the basic steps that are necessary when approaching a machine learning problem: From choosing a model to setting the parameters to extracting features from the data set and implementing them in the algorithm. For our final classifier, we used an SVM algorithm and optimized the system using several features of which character n-grams, POS n-grams, and the length of the tweets proved to be most effective.

We obtained an F-score of 65.32 on the test data. In future experiments, the score could possibly be improved by selecting more elaborate features: Montani and Schüller (2019), for example, ob-
tained a high F-score in the GermEval 2018 Shared Task on the Identification of Offensive Language making use of the TF/IDF scores of token n-grams. They calculated the TF/IDF for each n-gram within each class (i.e. OFFENSE and OTHER) and created a feature that contained only those TF/IDF scores with a document frequency within a certain range (determined by a grid search). Thereby, they reduced the token n-gram counts to only those n-grams that are important for one of the classes.

Another promising source for potential feature extraction could be the emojis in the tweets. They are an important characteristic of twitter data and reveal valuable information about the author’s intentions or emotional state. A semantic annotation (i.e. positive, negative) of each emoji type, perhaps with the help of the emoji descriptions in the unicode table, would precede the feature creation. This annotation would have to be done manually because emojis can be represented not only by simple but also by complex unicode containing variation selectors.

References


