Replicating Waseem Hovy's examination of abusive language

Rebecca Myers
Department of Computer Science

Holly Lopez Long
Department of Information Science

Abstract

The following discussion is a replication of Wasseem and Hovy's Waseem and Hovy (2016). In particular, this examination uses the data

1 Introduction

The examination presented here replicates a portion of Waseem and Hovy (2016)'s investigation of abusive language. In their examination, the authors create a corpus of abusive tweets. To increase the number tweets represented in the corpus the authors searched for topics that attracted racist and sexist language, including political topics, sports, and religion. The results of their examination demonstrated that character n-grams and gender performed best when trying to classify abusive tweets using a logistic regression classifier. Our examination uses the same corpus to train a logistic regression classifier and, to the best of our ability, replicates Waseem and Hovy (2016)'s examination. Our method for abusive language classification and findings are summarized in the following sections.

2 Method

2.1 Data Preprocessing

Waseem and Hovy (2016)'s corpus consisted of total of 15,715 tweets that were annotated as either 'abusive' or 'non-abusive. The corpus was divided into train and test sets. To preserve stylistic choices that might be informative in detecting abusive language, limited pre-processing was performed on the tweets. Although URLs were removed during preprocessing, emojis, @tags, tags, and stylistic capitalization were not removed or altered. Abusive post were more frequent in both the train and test set. The number of tweets and the distribution of annotations are described in Table 1 (below).

Set	Tweets	Abusive	Non-Abusive
Train	14,143	9,683	4,460
Test	1,572	1,076	496

Table 1: Distribution of annotation in train and test sets.

2.2 Classifier, Parameters, Fine Tuning

We used Scikit-learn to train, test, and perform evaluations of abusive language classification (Buitinck et al., 2013) on the corpus described above. The Tfidf Vectorizer library (from sklearn.feature_extraction.text) was used in the process of extracting character n-gram features. The frequency inverse document frequency was used, and the parameter 'analyzer' was set to 'char', which allowed us to analyze all characters including spaces. Since we wanted to examine character n-grams across words we chose this parameter value rather than only analyzing characters that were parts of words.

The cleaned data which was input as a list of strings, created an instance of the vectorizer and was assigned to a variable, 'scaler. To replicate Waseem and Hovy (2016)'s examination we chose to train on character n-grams as the research from the paper showed that this produced superior results when compared with word n-grams. Character n-grams for values ranging from 1 to 3 were created. An additional parameter choice was made by the researchers to use the strip_accents parameter with default setting of None. We chose to remove accents by setting this to 'ascii'.

The smooth_idf parameter was considered as we wanted to make sure division by zero was mitigated. The default value adds 1 to document frequency, as if every document in the corpus is seen one additional time. This prevents division by zero. No additional smoothing parameters was set because

we did not want to risk potentially losing information. Train and test were fit to the scaler, and using the transform built-in function of the package, both sets were transformed into sparse matrices (14,143 x 23,289 for train vectors, and 1,572 x 23,289 for test vectors).

We used Scikit-learn's logistic regression classifier (from sklearn.linear.model import LogisticRegression) to construct the model used. The default for penalization is set to L2 normalization. No other model parameters were utilized initially, and accuracy results came in at 84%. Various solvers available were examined. Ultimately, the LR model, and saga performed best with a L1 penalty. L2 limits the weights, but the L1 drives many of the weights to zero, and so some characters are ignored. The parameter, C, took on values of, 0.001, 0.01, 0.1, 1.0, and 10.0 for purposes of tuning the model. This value determines how aggressive the penalization is on the L1 normalization (how much the weights are zeroed out). Best performance is when C=10.0. This is the inverse of regularization. Smaller values are more aggressive. The results of the task are described in the following section.

3 Findings

The evaluation of the classifier used in the present investigation demonstrated a marked performance improvement when comparing with Waseem and Hovy (2016)'s examination. In terms of the classifiers performance on the target measure (e.g., instances of abuse), the classifiers precision was 0.89 when C was set to 10.0, which was well above the overall precision of (Waseem and Hovy, 2016)'s investigation. It is difficult to attribute this improvement to particular differences in the process which was used in this examination. However, it is interesting to note that the evaluation results during the fine-tuning process showed more comparable results when C was set to 0.10. Table 2 shows the macro average of our evaluation as well as, the evaluation results provided by Waseem and Hovy (2016). The features listed in the table are character n-grams for Waseem and Hovy (2016)'s examination does not depict other classifiers that used word n-grams or a combination of character n-grams and other features like gender. Even still, the classifier used in this investigation still outperformed Waseem and Hovy (2016)'s best evaluation results.

Moreover, when we extracted the 10 most offensive and 10 most predictive character n-grams many of the same character n-grams are among Waseem and Hovy (2016)'s most indicative character n-grams for sexism including, 'fem', 'sex', and 'itc'. This can be observed in Table 3, which depicts the top 10 most offensive and predictive character n-grams.

	Precision	Recall	F-1 Score
Original[1]	0.74	0.73	0.78
C = 10.0	0.85	0.85	0.85
C = 1.0	0.84	0.84	0.84
C = 0.1	0.77	0.77	0.75
C = 0.01	0.47	0.68	0.56
C = 0.001	0.47	0.68	0.56

Table 2: Evaluation Results

Most Offensive	Most Predictive
'sex'	'sex'
'fem'	'fem'
'kat'	'kat'
'@sy'	'@sy'
'cun'	'mee'
' .@'	'cun'
'itc'	' .@'
'cu'	'itc'
'l..'	'cu'
'hus'	'l..'

Table 3: 10 Most Offensive Predictive Character N-grams

References

Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, et al. 2013. Api design for machine learning software: experiences from the scikit-learn project. *arXiv preprint arXiv:1309.0238*.

Zeerak Waseem and Dirk Hovy. 2016. Hateful symbols or hateful people? predictive features for hate speech detection on twitter. In *Proceedings of the NAACL student research workshop*, pages 88–93.