CNN-Based Offensive Language Detection

Jian Xi[⋆] and **Michael Spranger**

University of Applied Sciences Mittweida
Forensic Science Investigation Lab (FoSIL)
Mittweida, Germany
{xi, spranger}@hs-mittweida.de

Dirk Labudde

Fraunhofer SIT
Cyber Security
Darmstadt, Germany
labudde@hs-mittweida.de

Abstract

Sentiment analysis of short social media texts is a challenging task due to limited contextual information and noise in texts. We present a deep convolutional model that utilizes unsupervised pre-trained word embeddings to detect offensive texts. Unfortunately, the model cannot outperform the baseline model in task-1 of the Germeval Task 2018 in terms of the F_1-measure.

1 Introduction

Sentiment Analysis (SA) is a subtask in Text Classification (TC) that focuses on the contextual mining of texts that are related to some specific objects. SA has great potential for several different applications. For instance, for a recommender system it is critical to know the interests of the customers. Furthermore, SA is also useful to find out the public opinion concerning highly sensitive political topics, as was the case in the study by Ross et al. (2016), in which Twitter texts were used to detect hate speech in the European refugee crisis. Usually, SA includes methods from different disciplines such as natural language processing (NLP) and machine learning (ML) (Pang et al., 2002).

The detection of offensive language in the Germeval Task 2018 is a typical task in SA. The submitted models should be able to categorize tweets into offensive or neutral for task-1 and into more fine-grained categories, namely neutral, profanity, insult and abuse, in task-2. Both, basic features and deep learning features, were used and combined with a classical ML model and a deep model in order to find out how the best result for the task can be achieved.

The paper is organized as follows: in Section II the architecture for the task is presented. Section III details the experimental setup and results. Finally, Section IV gives a short conclusion and discusses future work.

2 Model Description

The deep learning model shows remarkable performance in SA tasks as was shown by Nogueira dos Santos and Gatti (2014) as well as in NLP sequential text generation (Sutskever et al., 2011). The former study used a Convolution Neural Network (CNN) that uses convolution filters to extract local features in order to classify texts. In the latter study, a Recurrent Neural Network (RNN) captures the dependencies of data in a time-sequential way. In our case, we used a CNN model due to its performance in NLP tasks.

2.1 Architecture

Our model is a variation of the CNN by Kim (2014) as depicted in Figure 1. For the model, two channels were used for static and non-static representations of inputs with word embeddings (Mikolov et al., 2013). After maximizing the *feature map* with a max pooling operator as was presented by Kim (2014) a dense layer was added and its output entered into a second convolution layer

$$\mathbf{c}_s = f(\mathbf{w} \cdot \max\{\mathbf{c}\} + \mathbf{b}), \tag{1}$$

where $\mathbf{c}$ is the *feature map*, $\mathbf{w}$ and $\mathbf{b}$ the weights connected to the dense layer. It was found that, without this structure, the results are even worse. The output of the second convolution layer is concatenated and used as the input for the last dense layers. The final predicted sentiment label is output by a softmax layer.

2.2 Network Training

In our task let $T = t_1, ..., t_m$ be a set of texts to be categorized, and $c = c_1, ..., c_n$ a set of sentiment classes, then the task of categorizing can be described as a surjective mapping $f : T \to C$, where $f(t) = c \in C$ yields the correct class for $t \in T$. Given a text, the model calculates a score for each sentiment class $c \in C$. The network is hence trained

Proceedings of GermEval 2018, 14th Conference on Natural Language Processing (KONVENS 2018)
Vienna, Austria – September 21, 2018

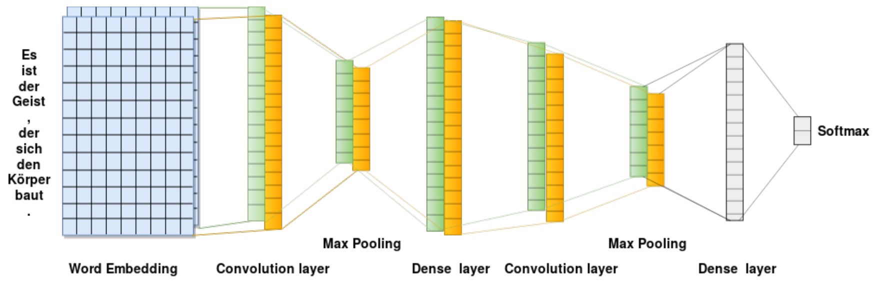

Figure 1: The architecture of the model with two input channels.

by minimizing the negative likelihood for the training set T defined in Equation 2.

$$\log L(c|t,\Theta) = \sum_{i=1}^{m} p(c|t_i,\Theta) - \log \sum_{j=1}^{n} e^{s_\Theta(t_i)_{c_j}} \quad (2)$$

For each input text t_i, the sentiment score $s_\Theta(t_i)_c$ for the sentiment label c is calculated by the network with the parameter Θ. The probability of a sentiment class c_k given the input t_i is the proportion of the sentiment class c over all sentiment classes $c_j \in C, j = 1,...,n$ and is calculated as shown in Equation 3.

$$p(c_k|t_i,\Theta) = \frac{e^{s_\Theta(t_i)_{c_k}}}{\sum_{j=1}^{n} e^{s_\Theta(t_i)_{c_j}}} \quad (3)$$

To predict a sentiment class it has to be determined which Θ maximizes the probability for a certain class as is shown in Equation 4.

$$\tilde{c} = \arg\max_{\Theta}\, p(c|t_i,\Theta) \quad (4)$$

In order to solve this optimization task ADADELTA, as proposed by Zeiler (2012), was applied.

2.3 Regularization

In order to regularize the parameters the L_2 norm was used in the convolution layers and a batch normalization (Ioffe and Szegedy, 2015) in the dense layers. The training does not stop until the validation accuracy does not improve any further within 25 epochs.

3 Experimental Setup and Results

The tasks are implemented with NLTK (Loper and Bird, 2002), Keras (Chollet, 2017), Scikit-learn (Pedregosa et al., 2011) and TreeTagger (Schmid, 1995). For task-1 four machine learning approaches were used: Naïve Bayes, SVM, a Multi-layer Perceptron (MLP) and our deep model. The basic models give a base-line performance for task-1. Afterwards, the deep model was build to upgrade the results for both tasks. All models are evaluated with respects of precision, recall and F_1-measure. Before the setup is explained in more detail, the features used are briefly introduced.

3.1 Feature Selection

In text classification tasks the selection of features is a critical step. On the one hand, well selected features are necessary to achieve highly accurate results. On the other hand, they help to reduce the feature space and as a consequence to minimize the time complexity (Yang and Pedersen, 1997).

Basic Features: Before the selection of features, all stop-words, repeated words and the punctuation were removed. Wang and Castanon (2015) showed that emoticons help in sentiment analysis tasks, however, this was not taken into account in our classification. The following three representations of text documents incorporating different features were compared:

- bag of words (BoW),
- TF-IDF of the BoW,
- Word $n-$grams (bi- and trigrams)

We also tried to select the top most common k $n-$grams to serve as a dictionary. However, due to an almost uniform distribution of $n-$grams in the corpus, this approach gives less informative feature representations.

Deep Learning Features: In order to use the similar contextual semantic of words, we used unsupervised pre-trained word embeddings (Mikolov et al., 2013) from the following resources:

- German twitter data between 2013 and 2017, with 100 dimensions and window size 5 provided by Ruppenhofer (2018),

- German Wikipedia and news articles, with 300 dimensions and window size 5 from Müller (2015)

3.2 Setup

Features: Table 1 shows the abbreviations for the features considered in both classification tasks.

Abbrv.	Feature
RAW	only raw texts
RAW*	with replacement of mention and hash tag
STM	BoW after stemming
LEM	BoW after lemmatizing
TFI	TF-IDF of BoW
STF	TF-IDF of BoW after stemming
LTF	TF-IDF of BoW after lemmatizing
BIG	word bigrams after stemming
TRG	word trigrams after stemming
MIG	mixture of BIG and TRG

Table 1: Features considered in the classification tasks.

In order to evaluate the fitting of the models for our data, a 10-fold cross validation was used. In each cross step, models with different features were evaluated regarding precision, recall and f-measure. After the best accuracy was achieved the most appropriate features and model was selected. The results will be given in 3.3.

Models: For the three basic models the default parameter settings from NLTK were used. In order to select the best version for the deep model, the following model variations were tested:

- Random: the word embeddings are initialized randomly and learned during training,
- Static: the word embeddings are initialized with previously pre-trained word embeddings and not changed during training,
- Non-static: one channel is set as static and the other as non-static. The static channel gives a basic word representation in the semantic space, while the other channel is adjusted during the learning process, so it can give a plausible representation of words in the given context.

3.3 Results

The results for the 10-fold cross-validation of three basic machine learning models for task-1 with different features are given in Table 2. As can be seen, unigram features lead to less information in the classification, while trigrams give the best precision results. Since the sequential and contextual information between words are encoded in trigrams, it enables a model to classify offensive texts better. Of all three basic models, the Naïve Bayes using BoW and stemmed texts performs best in terms of the F_1 measure.

Model	Feature	P	R	F_1
Naïve Bayes	RAW	0.542	**0.789**	0.623
	RAW*	0.536	0.756	0.627
	STM	0.556	0.784	**0.651**
	LEM	0.558	0.779	0.650
	BIG	0.570	0.225	0.323
	TRG	0.775	0.018	0.036
	MIG	0.565	0.222	0.319
MLP	RAW	0.654	0.473	0.549
	RAW*	0.651	0.439	0.524
	STM	0.661	0.493	0.565
	LEM	0.669	0.495	0.569
	TFI	0.629	0.511	0.564
	STF	0.626	0.509	0.561
	LTF	0.638	0.490	0.554
	BIG	0.748	0.069	0.126
	TRG	0.875	0.012	0.025
	MIG	0.836	0.033	0.064
SVM	TFI	0.663	0.513	0.579
	STF	0.677	0.524	0.591
	LTF	0.680	0.523	0.591
	BIG	0.777	0.056	0.104
	TRG	**0.917**	0.007	0.013
	MIG	0.857	0.025	0.048

Table 2: Evaluation results of the basic models for task-1.

Additionally, Table 3 shows stems of words that often occur in offensive twitter texts. They were selected by their informativeness which is based on the prior probability that features occur for each label. These may be useful in a later approach in order to set up a knowledge base.

Table 4 shows the best results for our deep model for task-1, achieved using word embeddings pre-trained on Twitter data, as suggested by Rezaeinia et al. (2017). The model performs best with a static

Stem	Informativeness
murksel	21.68
scheiss	19.09
pack	17.95
idiot	17.34
wand	14.20
deutschfeind	12.09
entsorgt	10.07
gehirn	8.31
hitl	7.18
altmai	6.65

Table 3: The 10 most informative features detected by the Naïve Bayes model.

Class	P	R	F_1
OTHER	0.778	0.918	0.840
OFFENSIVE	0.754	0.470	0.572

Table 4: Evaluation results of the CNN model for task-1.

initialization. However, the Naïve Bayes model performs better in this task. One possible explanation for the poor performance of our model is the lack in sufficient training data. For example Kim's (2014) training data set was on average of double the size. Another possible explanation is that the quality of the pre-trained word embeddings is not sufficient. As we have seen the word embeddings include a lot of noise. Subsequently, three runs of the static deep model using Twitter word embeddings were submitted as:

- `FoSIL_coarse_1.txt`,
- `FoSIL_coarse_2.txt`, and
- `FoSIL_coarse_3.txt`.

4 Conclusions and Future Work

In this paper we used basic ML methods and a deep CNN model in order to classify texts into different categories regarding offensive language. The results show that the Naïve Bayes model performs better in task-1 in comparison to our proposed CNN model. The reasons might be the small amount of training data as well as the poor quality of the provided word embeddings. Tai et al. (2015) showed that sequential models perform best in sentiment analysis tasks, which is why these models should be further tested. However, also further features should be considered. For instance, in order to distinguish texts including profanity from those, that include abuse and insults, it would be useful to take Part-of-Speech (POS) into account as Rezaeinia et al. (2017) suggest to use POS and word embeddings to improve classification accuracy. As emoticons occur in both, neutral texts and offensive texts, it should be analyzed how they might influence the classification results. Furthermore, Nogueira dos Santos and Gatti (2014) used word-level embeddings as well character-level embeddings to catch morphological information in order to classify short texts.

References

Francois Chollet. 2017. *Deep learning with python*. Manning Publications Co.

Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, pages 448–456. JMLR.org.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751. Association for Computational Linguistics.

Edward Loper and Steven Bird. 2002. Nltk: The natural language toolkit. In *Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics - Volume 1*, ETMTNLP '02, pages 63–70, Stroudsburg, PA, USA. Association for Computational Linguistics.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, pages 3111–3119, USA. Curran Associates Inc.

Andreas Mueller. 2015. German Wikipedia Embeddings. URL: `https://devmount.github.io/GermanWordEmbeddings/` [accessed: 2018-08-09].

Cicero Nogueira dos Santos and Maira Gatti. 2014. Deep convolutional neural networks for sentiment analysis of short texts. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 69–78. Dublin City University and Association for Computational Linguistics.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up? sentiment classification using machine learning techniques. In *Proceedings of EMNLP*, pages 79–86.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2011. Scikit-learn: Machine learning in python. *J. Mach. Learn. Res.*, 12:2825–2830, November.

Seyed Mahdi Rezaeinia, Ali Ghodsi, and Rouhollah Rahmani. 2017. Improving the accuracy of pre-trained word embeddings for sentiment analysis. *CoRR*, abs/1711.08609.

Björn Ross, Michael Rist, Guillermo Carbonell, Ben Cabrera, Nils Kurowsky, and Michael Wojatzki. 2016. Measuring the Reliability of Hate Speech Annotations: The Case of the European Refugee Crisis. In Michael Beißwenger, Michael Wojatzki, and Torsten Zesch, editors, *Proceedings of NLP4CMC III: 3rd Workshop on Natural Language Processing for Computer-Mediated Communication*, pages 6–9.

Josef Ruppenhofer. 2018. German Twitter Embeddings. URL: `http://www.cl.uni-heidelberg.de/english/research/downloads/resource_pages/GermanTwitterEmbeddings/GermanTwitterEmbeddings_data.shtml` [accessed: 2018-08-09].

Helmut Schmid. 1995. Improvements in part-of-speech tagging with an application to german. In *In Proceedings of the ACL SIGDAT-Workshop*, pages 47–50.

Ilya Sutskever, James Martens, and Geoffrey Hinton. 2011. Generating text with recurrent neural networks. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, pages 1017–1024, USA. Omnipress.

Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. *CoRR*, abs/1503.00075.

H. Wang and J. A. Castanon. 2015. Sentiment expression via emoticons on social media. In *2015 IEEE International Conference on Big Data (Big Data)*, pages 2404–2408, Oct.

Yiming Yang and Jan O. Pedersen. 1997. A comparative study on feature selection in text categorization. In *Proceedings of the Fourteenth International Conference on Machine Learning*, ICML '97, pages 412–420, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Matthew D. Zeiler. 2012. Adadelta: An adaptive learning rate method. *CoRR*, abs/1212.5701.