EasyChair Preprint

№ 872

Detecting Toxic Content Online and the Effect of Training Data on Classification Performance

Zhixue Zhao, Ziqi Zhang and Frank Hopfgartner

April 1, 2019

Detecting Toxic Content Online and the Effect of Training Data on Classification Performance

Zhixue Zhao, Ziqi Zhang and Frank Hopfgartner
Information School, University of Sheffield
{zzhao33, ziqi.zhang, f.hopfgartner}@sheffield.ac.uk

Abstract. The spread of toxic content online has attracted a wealth of research into methods of automatic detection and classification in recent years. However, two limitations still exist: 1) the lack of support for multi-label classification; and 2) the lack of understanding of the impact of the typical unbalanced datasets on such tasks. In this work, we build three state of the art methods for the task of multi-label classification of toxic content online, and compare the effect of training data size on their performance. The three methods of choice are based on Support Vector Machine (SVM), Convolutional Neural Networks (CNN) and Long-Short-Term Memory Networks (LSTM), respectively. We conduct learning curve analysis and show that CNN is the most robust method as it outperforms the other two regardless of the sizes of the dataset, even on very small amounts of data. This challenges the conventional belief that Neural Networks require significant amounts of data to train accurate models. We also empirically derive indicative thresholds of training data size to help determine a reliable estimate of classifier performance, or maximise potential classifier performance in such tasks.

Keywords: toxic content, hate speech, machine learning, CNN, LSTM, SVM, deep learning, natural language processing, information retrieval

1 Introduction

The rise of various social media platforms encourages the ever-increasing user-generated content (UGC), which broadly refers to any form of content created by users and made available publicly online. While such platforms are changing the ways we access and share information, at the same time, they are also increasingly exploited for the creation and propagation of 'toxic content'. This refers to the UGC that is abusive, disrespectful or otherwise likely to make someone leave a discussion. It can include many types of content such as hate speech, cyber bullying and abusive language [1]. Previous research showed that at least 41% of Americans have been subjected to abusive comments online [2], while 80% of young people in the European Economic Area could have been targets of hate speech [3].

Building effective countermeasures for toxic content online requires as the first step, identifying and tracking such content efficiently. While social media companies have been investing hundreds of millions of euros each year to moderate and remove such content manually [4], the industry is now looking at methods that partially automate these processes in order to cope with the Web scale [5] .

An enabling task for this process is to distinguish toxic from non-toxic content, or different types of toxic content. We call this task 'toxic content detection', or 'classification', and it broadly covers an extensive range of recent research on detecting hate speech [4, 6], abusive and offensive language [7], and cyberbully [8]. Despite an increasing number of related research in recent years, we identify three limitations. First, previous work only studied a particular type of toxic content, such as hate speech [4, 6], or cyberbully [8]. Toxic content in general, can include these but also other types, and there could be interesting correlations among these types. Second, the majority of studies have assumed the 'one-against-others' rule, i.e., the different types of toxic content are mutually exclusive. Practically this is hardly the real case, as hate speech could be at the same time abusive, or threatening. Third, a consistent finding from previous work is that typical datasets of toxic content are extremely unbalanced, such that there are 'minority' types suffering from insufficient data to train accurate classifiers [6]. However, it remains unclear how the size of training data effects classification accuracy, or whether there is a threshold of 'sufficient' training data for such tasks.

We address these limitations by developing three state of the art methods applied to multi-label classification of the largest collection of toxic content dataset, and empirically study the effect of training data size on the classification accuracy, or in other words, their 'learning curve'. We show that a convolutional neural network (CNN) based method consistently performs best, and also 'learns faster' as it was able to achieve better results with reduced training data. This challenges our conventional perspective that deep neural network based methods require significant amount of data to perform well. The remainder of this paper is structured as follows. Section 2 reviews related work. Section 3 describes our methodology. Section 4 presents our experiments and discusses findings, followed by conclusion in Section 5.

2 Related Work

2.1 Text classification

The detection of toxic content is a type of text classification problem, which is a widely used technique in natural language processing that aims to classify the given texts or documents (called 'instances') into categories (classes, labels, or types) [9]. The problem is solved by training a 'supervised' machine learning classifier using human-labelled training data (already classified texts). A typical workflow involves data pre-processing, feature extraction, and algorithm or model training. Pre-processing reduces the language complexity by, e.g., stopword removal and stemming. Feature extraction represents instances in a high dimensional vector space for algorithmic consumption. Then an algorithm is applied to the training data using

the extracted features, to train a model able to classify similar, unseen instances represented in the data.

2.2 Toxic content classification

Types of toxic content. While recent years have seen significant increase in research on toxic content classification, the majority of these focus on only a single type (and its subtypes) of content. These include a large number of work on hate speech classification [4, 6, 10-14], some work on abusive and offensive language [5, 15-18, 28], cyberbully [7, 8, 19, 20], or discrimination detection [21]. To our knowledge, Kumar et al. [29] organised the first effort examining different types of toxic content including trolling, aggression, and cyber bullying. However, these studies adopt the 'one-against-others' classification principle. As an example, a hateful message can be 'sexism' or 'racism', but not both [11].

Types of features. Although the specific types of toxic content can vary in the tasks, they often share similar features. Schmidt and Wiegand [14] summarised several commonly used types of features. The most frequently used, *simple surface features,* include words, word or character n-grams, content length, capitalisation, and in the context of social media platforms, URLs and hashtags. *Word generalisation* includes examples such as word clusters [22], or word embeddings [23]. *Linguistic features* can be, e.g., Part of Speech and dependency relations. Features such as *sentiment*, *dictionaries* of indicative keywords and *user information* (e.g., gender, frequent vocabulary) were also used.

Types of algorithms. Zhang et al. [6] summarised machine learning algorithms used in the literature into two types: classic and deep learning based algorithms. The first requires manually designing and encoding features of instances - such as those described above - into feature vectors, which are then directly used by the algorithms for classification. Examples include Support Vector Machines (SVM) [6, 15, 16, 18, 22], Logistic Regression [18, 23], Naive Bayes [12, 15, 18, 21], and Random Forest [16], with SVM being the most frequently used. The second employs deep artificial neural networks (DNN) to learn abstract feature representations from input data through its multiple stacked layers. The input is often simply the raw text data, but can also take various forms of feature encoding as those used in the classic methods. The most popular network structures use either Convolutional Neural Networks (CNN) [4, 6, 10, 13] or Long-Short Term Memory (LSTM) [6, 10] that is a type of Recurrent Neural Networks. A conventional viewpoint is that DNN based methods require significant amount of data to train [24], as they naturally involve significantly more parameters to learn compared to classic algorithms. However, in the context of toxic content detection, there are no comparative studies on the learning curves of DNN or classic algorithms. The question of 'how much data are enough' remains unanswered.

3 Methodology

To address the limitations described before, we propose a comparative study of existing methods on multi-label toxic content classification, with an analysis of their

learning curves. In the following we firstly describe our methods for multi-label classification, followed by an explanation of how we conduct learning curve analysis using these methods.

3.1 Classification methods

We choose three most often used methods for comparison: SVM, CNN, and LSTM.

SVM. This represents the most popular 'classic' algorithm for toxic content classification. The principle of SVM is to learn parameters to draw a 'separating hyperplane' that maximises the distance between the different classes in the training data. For each text to classify, we extract both unigram and bigrams as its features as they were shown to be the most effective features for text classification [25]. These are then weighted by the Term Frequency Inverse Document Frequency algorithm often used in Information Retrieval.

SVM essentially classifies data based on the 'one-against-others' principle. To adapt it for multi-label classification, we apply the concept of 'label powerset', which transforms a multi-label problem into a single-label, multi-class problem by considering every label combination set as a single label [26]. As an example, instances assigned both the class labels 'sexism' and 'racism' are treated as a separate class from those assigned 'sexism' or 'racism' alone. Although this increases the potential number of classes in a dataset and reduces the amount of training data per class, it is the easiest, natural solution that allows taking into account the correlation between class labels [27].

CNN. The CNN model consists of a sequence of an embedding layer taking an input text, followed by a convolutional layer, a max pooling layer and a dense layer to output the final classification result. The embedding layer assigns weight to each word in the input text as a fixed dimension, real-valued vector. Each dimension indicates the relative weight of the word for a 'latent' concept. These weights and latent concepts are learned from a large text corpus, either from the training dataset itself as part of the training process, or on an external, domain-independent corpus such as the 840 billion words GloVe corpus[1]. Here we choose not to use pre-trained embeddings to exclude potential influences from external resources. The embedding layer also assumes an input text to contain a uniform number of $\boldsymbol{l}$ words, and learns to project each word into a $\boldsymbol{d}$ dimension vector space. Longer texts will be truncated while shorter ones are padded with arbitrary placeholders.

Then a convolutional layer scans a fixed sliding window of $\boldsymbol{cw}$ consecutive words with a stride of $\boldsymbol{cs}$ from the output of the embedding layer and transforms each scanned sequences into an abstract feature. These output features are concatenated, and further 'pooled' by a max pooling layer, which scans the previous feature representation by a fixed sliding window of $\boldsymbol{pw}$ and a stride of $\boldsymbol{ps}$ to generate new features. They are finally flattened into a vector as the final feature representation to be classified by the last dense layer. This uses the softmax function to calculate a probability distribution over the set of possible classes seen in the training dataset.

[1] https://nlp.stanford.edu/projects/glove/, last retrieved in September 2018

Intuitively, the CNN examines adjacent sequences of words in a text and attempts to extract abstract features from such sequences.

LSTM. The LSTM model starts with the same embedding layer as that in the CNN, followed by an LSTM layer that extracts further abstract features from the embedding layer, and finally passes these features to the same dense layer for classification. LSTM is designed to capture long distance dependencies between its input features. Intuitively, it 'remembers' words it has seen before the current word in a sentence and their order, and hypothesizes that this 'history' and order would 'explain' the meaning of the current word.

For both CNN and LSTM, we use the categorical cross entropy loss function and Adam optimiser to compile and train the model, using a batch size of 64 with 5 epochs. The first is empirically found to be more effective on classification tasks than other loss functions including classification error and mean squared error [31], and the second is designed to improve the classic stochastic gradient descent (SGD) optimiser and in theory combines the advantages of two other common extensions of SGD (AdaGrad and RMSProp) [30]. Both methods produce a probability distribution over candidate classes seen in the dataset for an input text. We assert that classes with a probability greater than 0.5 are the final labels for the text.

For all models, we conduct a 5-fold cross validation experiment and calculate the the classic Precision, Recall and F1 scores for each class as well as their micro-average for the dataset. Compared to methods used in the literature, all of our methods are relatively simple. The reason is to focus on the algorithmic difference, but excluding influence from feature engineering and complex network structures.

3.2 Learning curve analysis

To investigate the effect of training data on a method's performance, given a target dataset, we create random subsets of the dataset in 5% increments. In other words, we obtain 20 datasets, starting with a minimum size of 5% of the original dataset, up to 100%. We then conduct 5-fold validation of a method using each of these datasets to evaluate its performance. This allows us to gauge the change in the method's performance as we increase training data.

4 Experiment and Discussion

4.1 Dataset

We use a dataset published by Conversation AI[2]. This contains comments collected from Wikipedia Talk Page, which is a function of Wikipedia allowing users to discuss improvements to the articles. The comments are anonymised and annotated with toxicity reasons and toxicity levels. This creates six class labels: 'toxic', 'severe

[2] https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge. Last retrieved in September 2018

toxic', 'obscene', 'threat', 'insult' and 'identity hate'. 'Obscene', 'threat', 'insult' and 'identity hate' are four sub-labels for 'toxic' and 'severe toxic' comments (therefore they can co-occur on a comment). The 'toxic' comments that are not 'obscene', 'threat', 'insult' or 'identity hate' are assigned to either 'toxic' or 'severe toxic'. The 'clean' comments are excluded from this dataset. This results in a dataset of a total of 16,225 toxic comments, with a label distribution shown in Figure 1. The dataset is then pre-processed by 1) lowercase conversion; 2) filtering non-alphabetic characters; and 3) removing extra white spaces resulted in this process.

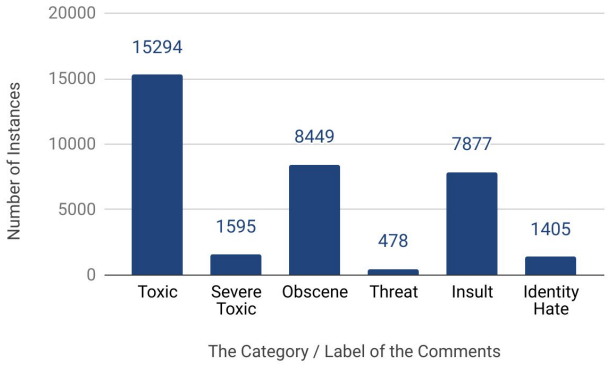

Fig. 1. Distribution of different class labels in the dataset

4.2 Implementation and parameters

For the SVM based method, we used the Python Scikit-Learn[3] library implementation. All default hyperparameters of the algorithm are used. For the CNN and LSTM methods, we used the Keras[4] library with TensorFlow[5] for implementation. For CNN, we set cw=4, cs=1, pw=2, ps=2. For both CNN and LSTM, we set d=128, and l=120, i.e., we assume a uniform length of 120 words for every comment, which is long enough to encode the entirety of most comments as most of them have fewer than 50 words, as shown in Figure 2. For the longer comments that are minority in the dataset, this implies that their contents are not fully represented as words beyond the 120 limit are truncated. However, this is a reasonable trade-off for practical reasons, as very long texts can significantly increase computation. All the parameters are rather arbitrary, as our goal is not to maximise the performance of any method. All experiments were performed using GPUs on Google Colaboratory[6].

[3] http://scikit-learn.org/stable/. Last retrieved in September 2018
[4] https://keras.io/. Last retrieved in September 2018
[5] https://www.tensorflow.org/. Last retrieved in September 2018
[6] https://colab.research.google.com/. Last retrieved in September 2018.

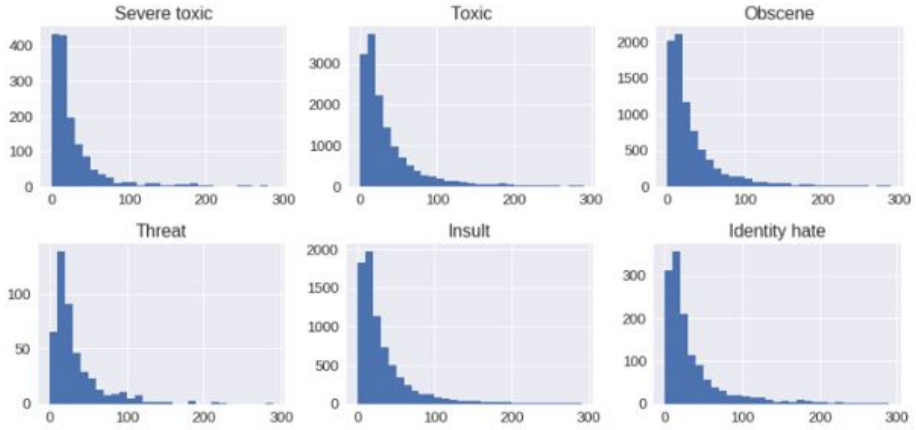

Fig. 2. Distribution of length (as number of words) of toxic comments

4.3 Correlation analysis

Considering a Pearson correlation of at least 0.5 to be 'strong', Figure 3 shows three pairs of strongly correlated classes: 'insult' and 'toxic', 'obscene' and 'toxic', and 'insult' and 'obscene'. The last pair has the strongest correlation of 0.74, implying that 'insult' comments also tend to be 'obscene' at the same time, and vice versa. Such correlations could be useful clues to classifiers.

4.4 Multi-label classification: overview of results

Table 1 shows the evaluation results obtained by the three methods on the *entire* dataset. In terms of F1, CNN outperforms both SVM and LSTM, sometimes quite significantly (e.g., on 'threat'). However, SVM is arguably the best performer in terms Precision, while LSTM only has very marginal advantage in Recall. Comparing different classes, there is a strong pattern that all models perform much better on 'large' classes, i.e., 'toxic', 'obscene' and 'insult' which have more than 7,000 instances, where all three methods have obtained F1 scores higher than 80% except SVM on 'insult'. In contrast, their performance on the other three 'smaller' classes containing less than 2,000 instances is significantly worse, with F1 scores between 47% and 73%.

The conclusions we can draw seem to be that, on the whole, CNN is the most effective method on this task. All methods suffer significantly on 'small' classes when the amount of training data is small, or in other words, they can benefit from more data. It is worth to note that on the smallest class 'threat', CNN performs particularly well, even beating its own performance on the much larger class 'severe toxic'.

Thus the questions still remain that to what extent can a method benefit from more training data, and does CNN consistently perform better than the other methods on small data. We answer these questions in the next section.

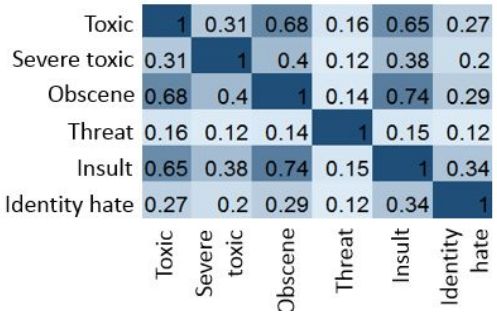

Fig. 3. Correlation analysis between different classes

Table 1. Results obtained on the entire dataset. P - Precision, R - Recall. For each measure and each class, the highest figure is highlighted in **bold**. Numbers in brackets indicate the number of instances for that class.

Class	SVM			LSTM			CNN		
	P	R	F1	P	R	F1	P	R	F1
Toxic (15,294)	**94.1**	70.0	80.3	87.9	86.0	**86.9**	90.5	86.5	**88.4**
Severe toxic (1,595)	70.1	47.8	56.8	67.7	54.2	**60.1**	**74.8**	57.2	**62.8**
Obscene (8,449)	**92.5**	77.8	84.5	89.9	86.6	**88.2**	90.5	87.3	**88.8**
Threat (478)	**85.2**	48.0	**61.3**	82.2	34.3	47.1	83.6	60.3	**69.7**
Insult (7,877)	**86.0**	72.8	63.9	85.4	79.0	82.0	85.8	**83.2**	**84.4**
Identity hate (1,405)	**83.0**	52.0	63.9	76.9	49.3	60.1	80.6	**66.4**	**72.8**
Average	**90.3**	70.5	79.2	86.7	81.0	83.8	88.3	**83.5**	**85.8**

4.5 **Multi-label classification: learning curve**

To address the questions we raised before, here we analyse the learning curves of the three methods. Figure 4 compares the learning curves (micro-average F1) of the three methods on a 5% increment of the entire dataset. To show a complete picture, below 5%, we record the performance at every 1% increment.

The figure shows that, clearly, CNN 'learns' faster and better than both SVM and LSTM, as it consistently outperforms the two methods starting from just 2% of the entire dataset. LSTM on the other hand, only overtakes SVM beyond the 25% mark, suggesting that it learns much slower and arguably, the most sensitive method to data size. For all three methods, it appears that the benefits from the increase of data size are the most significant before 25% where the growth of performance is nearly exponential. This slows down dramatically and beyond 80%, there is tendency for performance to plateau, which did not happen on our dataset, suggesting that all three models may still benefit from additional data. But arguably, the trade-off between

gains in performance and effort spent on obtaining additional training data may become significantly unbalanced thereafter.

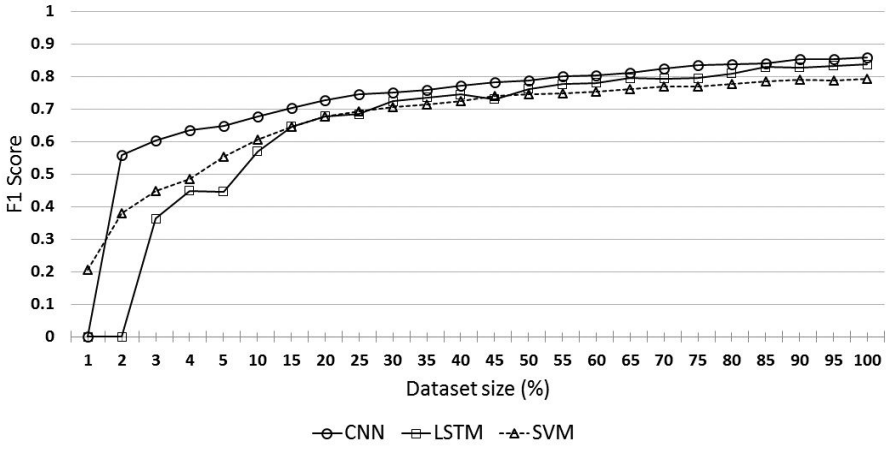

Fig. 4. Learning curves of the three different methods

Using the best performing CNN as example, Figure 5 shows its detailed learning curve on each class in the dataset. We can see that the learning curves for 'toxic', 'insult' and 'obscene' classes follow the general pattern described above. This is because they are the majority classes in the dataset. For all three classes, it follows that beyond the 80% mark, the performance tends to plateau. Among them, 'obscene' is the smaller class, and this threshold corresponds to around 7,000 instances. This could be considered a general indication of the minimum number of instances in such tasks to maximise the potential performance of a classifier.

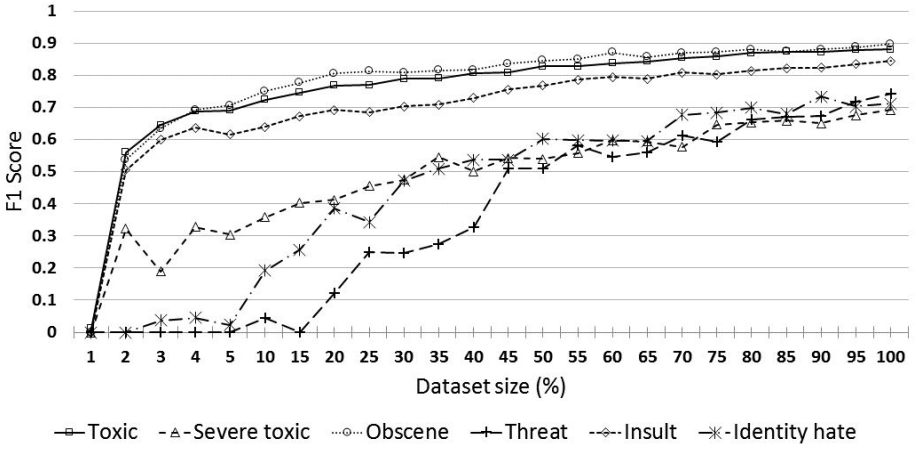

Fig. 5. Learning curves of the CNN method on a per-class basis

The story for the three minority classes, i.e., 'severe toxic', 'threat', and 'identity hate', is quite different. For all these classes, the performance of the CNN classifier continues to rise as we increase the amount of data. While the speed of increase slows down gradually, there is no tendency of them to plateau, suggesting that the classifier can still benefit from further training data on these classes. In other words, the amount of training data for these classes (between 478 and 1,595) could be insufficient.

Finally, we can notice the 'zig-zag' pattern in the performance figures for all classes except 'toxic' and 'obscene' below the 5% mark, suggesting that below this threshold, the amount of training data could be far too inadequate such that the performance estimates of the classifiers may be unreliable. Based on the largest class of the four (i.e., 'insult'), this corresponds to about 400 instances, which can be considered a general indication of the minimum amount of training data in order to obtain reliable estimation of a classifier performance in such tasks.

5 Conclusion

This work looked at the task of multi-label toxic content classification, which is more general than the extensively studied problems of hate speech, cyberbully, and abusive content detection, and has not been examined before. Using the largest dataset known to date, we implemented and compared three frequently used methods in similar tasks, including SVM, CNN and LSTM. We showed that among the three, CNN performs the best overall in terms of F1, while SVM is arguably the winner in terms of Precision. However, in terms of learning curves, contrary to the conventional belief that deep neural network based methods require substantial data to perform well, our experiments have shown that on this very specific task, CNN appears to be the most robust among the three, as it 'learns' faster and consistently outperforms SVM and LSTM on varying sizes of training data. We also empirically derived indicative 'thresholds' of the minimum amount of training data in order to obtain reliable estimate of a classifier performance, and that to maximise its potential performance.

Our future work will focus on two directions. First, we will apply our study to other similar tasks and datasets to understand the generality of our findings. Second, we will investigate methods that 'compensate' the lack of training data, such as transfer learning and multi-task learning.

6 References

1. Waseem, Z., Davidson, T., Warmsley, D., Weber, I.: Understanding Abuse: A Typology of Abusive Language Detection Subtasks. In 1st Workshop on Abusive Language Online. Vancouver (2017)
2. Duggan, M.: Online Harassment. Pew Research Centre. Last retrieved in September, 2018, from: http://www.pewinternet.org/2014/10/22/online-harassment/ (2018)
3. EEA News: Countering hate speech online, Last retrieved in September, 2018, from: http://eeagrants.org/News/2012/ (2018)

4. Gambäck, B., Sikdar, U.: Using convolutional neural networks to classify hate-speech. In Proceedings of the First Workshop on Abusive Language Online, pages 85–90. Association for Computational Linguistics (2017)
5. Chu, T., Jue, K., Wang, M.: Comment Abuse Classification with Deep Learning. Stanford University (2017)
6. Zhang, Z., Robinson, D., Tepper, J.: Detecting hate speech on twitter using a convolution-gru based deep neural network. In Proceedings of the 15th Extended Semantic Web Conference, ESWC18, pages 745–760. Springer (2018)
7. Nobata, C., Tetreault, J., Thomas, A., Mehdad, Y., Chang, Y.: Abusive language detection in online user content. In Proceedings of the 25th International Conference on World Wide Web, pages 145–153 (2016)
8. Dadvar, M., Trieschnigg, D., Ordelman, R., de Jong, F.: Improving cyberbullying detection with user context. In Proceedings of the 35th European Conference on Advances in Information Retrieval, ECIR'13, pages 693–696, Berlin, Heidelberg, Springer-Verlag (2013)
9. Uysal, A.: An improved global feature selection scheme for text classification. Expert Systems With Applications, 43, 82–92 (2015)
10. Badjatiya, P., Gupta, S., Gupta, M., Varma, V.: Deep learning for hate speech detection in tweets. In Proceedings of the 26th International Conference on World Wide Web Companion, pages 759–760 (2017)
11. Burnap, P., Williams, M.: Us and them: Identifying cyber hate on twitter across multiple protected characteristics. EPJ Data Science, 5(11):1–15 (2016)
12. Kwok, I., Wang, Y.: Locate the hate: Detecting tweets against blacks. In Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence, pages 1621–1622. AAAI Press (2013)
13. Park, J., Fung, P.: One-step and two-step classification for abusive language detection on twitter. In ALW1: 1st Workshop on Abusive Language Online, Vancouver, Canada, Association for Computational Linguistics (2017)
14. Schmidt, A., Wiegand, M.: A survey on hate speech detection using natural language processing. In International Workshop on Natural Language Processing for Social Media, pages 1–10. Association for Computational Linguistics (2017)
15. Chen, Y., Zhou, Y., Zhu, S., Xu, H.: Detecting offensive language in social media to protect adolescent online safety. In Proceedings of the 2012 ASE/IEEE International Conference on Social Computing and 2012 ASE/IEEE International Conference on Privacy, Security, Risk and Trust, pages 71–80, Washington, DC, USA, IEEE Computer Society (2012)
16. Xiang, G., Fan, B., Wang, L., Hong, J., Rose, C.: Detecting offensive tweets via topical feature discovery over a large scale twitter corpus. In Conference on Information and Knowledge Management, pages 1980–1984. ACM, (2012)
17. Davidson, T., Warmsley, D., Macy, M., Weber, I.: Automated hate speech detection and the problem of offensive language. In Proceedings of the 11th Conference on Web and Social Media. AAAI, (2017)
18. Mehdad, Y., Tetreault, J.: Do characters abuse more than words? In Proceedings of the SIGDIAL 2016 Conference, pages 299–303, Los Angeles, USA, Association for Computational Linguistics (2016)
19. Dinakar, K., Jones, B., Havasi, C., Lieberman, H., Picard, R.: Common sense reasoning for detection, prevention, and mitigation of cyberbullying. ACM Trans. Interact. Intell. Syst., 2(3):18:1–18:30 (2012).
20. Zhong, H., Li, H., Squicciarini, A., Rajtmajer, S., Griffin, C., Miller, D., Caragea, C.: Content-driven detection of cyberbullying on the instagram social network. In Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, pages 3952–3958. AAAI Press (2016)

21. Yuan, S., Wu, X., Xiang, Y.: A two phase deep learning model for identifying discrimination from tweets. In Proceedings of 19th International Conference on Extending Database Technology, pages 696–697 (2016)
22. Warner, W., Hirschberg, J.: Detecting hate speech on the world wide web. In Proceedings of the Second Workshop on Language in Social Media, LSM '12, pages 19–26. Association for Computational Linguistics (2012)
23. Djuric, N., Zhou, J., Morris, M., Grbovic, M., Radosavljevic, V., Bhamidipati, N.: Hate speech detection with comment embeddings. In Proceedings of the 24th International Conference on World Wide Web, pages 29– 30. ACM (2015)
24. Ahn, J., Park, J., Park, D., Paek, J., Ko, J.: Convolutional neural network-based classification system design with compressed wireless sensor network images. PLoS ONE 13(5): e0196251 (2018)
25. Johnson, R., Zhang, T.: Effective Use of Word Order for Text Categorization with Convolutional Neural Networks. In HLT-NAACL (2015)
26. Madjarov, G., Kocev, D., Gjorgjevikj, D., & Džeroski, S.: An extensive experimental comparison of methods for multi-label learning. Pattern Recognition, 45(9), 3084–3104 (2012)
27. Padmanabhan, D., Bhat, S., Shevade, S., Narahari, Y.: Multi-Label Classification from Multiple Noisy Sources Using Topic Models. Information, 8(2), 52 (2017)
28. Wiegand, M., Siegel, M., Ruppenhofer, J.: Overview of the GermEval 2018 Shared Task on the Identification of Offensive Language. Proceedings of GermEval 2018, 14th Conference on Natural Language Processing (2018)
29. Kumar, R., Ojha, A., Malmasi, S., Zampieri, M.: Benchmarking Aggression Identification in Social Media. First Workshop on Trolling, Aggression and Cyberbullying (2018)
30. Kingma, D., Adam, J.: A method for stochastic optimization. In Proceedings of the 3rd International Conference for Learning Representations, 2015.
31. McCaffrey, J.: Why you should use cross-entropy error instead of classification error or mean squared error for neural network classifier training, Last accessed: Jan 2018, https://jamesmccaffrey.wordpress.com.