Social Media Toxic-Text Analysis Using Deep Learning Techniques

Tripti Agrawal, Shweta Sankhwar(✉), Tanya Chaudhary, and Aaditri Saraswat

Maitreyi College, University of Delhi, New Delhi 110021, India
{tagrawal,ssankhwar}@maitreyi.du.ac.in

Abstract. The widespread use of social media in contemporary society has created a vast platform for individuals to express their opinions openly. However, certain anti-social groups misuse this freedom to propagate toxic behavior, including verbal sexual harassment, threats, insults, and obscenities, among others. These behaviors hinder the free exchange of opinions and have led even major social media platforms to limit or disable user comments to counter toxicity. Consequently, the automatic detection and identification of such behavior through machine learning models have become increasingly critical. In this context, this paper examines various machine learning techniques for the classification of toxicity in online comments, utilizing Kaggle's toxic comment classification dataset. Furthermore, the study assesses the performance of both shallow learning algorithms and deep learning methods, using various evaluation metrics to comprehensively evaluate their effectiveness.

Keywords: Social media · Toxic Text analysis · Machine Learning · Deep-learning · Shallow Learning

1 Introduction

Social media toxicity refers to the presence of harmful, abusive, or inappropriate content within the context of social media platforms. This can include various forms of negative behavior such as hate speech, cyberbullying, harassment, and offensive language. This term encompasses any content that has the potential to create a toxic or hostile environment, leading to negative impacts on users' well-being, mental health, and overall online experience. Dealing with social media toxicity involves the implementation of content moderation strategies, user protection mechanisms, and the use of technologies such as natural language processing and machine learning for the identification and mitigation of harmful content, primarily through classification techniques.

In essence, toxic social media text classification involves utilizing natural language processing and machine learning techniques to automatically detect, categorize, and manage toxic or harmful content across social media platforms.

O. Castillo et al. (Eds.): PerSOM 2023, LNICST 517, pp. 293–303, 2024.
https://doi.org/10.1007/978-3-031-66044-3_21

The primary goal is to develop models that can effectively identify and label different forms of toxic text, such as hate speech, offensive language, cyberbullying, and other types of inappropriate or harmful communication. By employing these classification techniques, social media platforms can enhance their content moderation strategies, create a safer online environment, and promote more respectful and constructive interactions among users. This area of research has received significant attention due to the growing concern about online harassment and the need for effective content moderation strategies in various online communities and social media platforms. Thus, the suggested study intends to analyze toxic social media content using machine learning methods to enhance the effectiveness of toxic comment classification. In the paper, experiments have been implemented on Kaggle's Toxic Classification dataset[1] using different machine learning algorithms. In addition to this, a result comparison between shallow learning algorithms and deep learning algorithms for the purpose of toxic comment classification has been done.

The outline of the rest of the paper is as follows - Sect. 2 provides an overview of the existing literature. Section 3 presents the methodology followed in this study. Section 4 presents the experiments and their outcomes. Finally, Sect. 5 concludes the paper with a summary of the findings, implications of the study, and an outline of future work.

2 Related Work

In recent years, scholars have increasingly focused on exploring methodologies for identifying toxicity levels through sentiment analysis, particularly within social media data. Various machine learning techniques have been employed to address this challenge, including the analysis of sentiment and the detection of hate speech in text. Notably, authors in [16] utilized machine learning models to detect toxicity levels, particularly targeting personal attacks. They achieved promising results, with an accuracy of over 90% in identifying toxic comments.

Furthermore, a number of researchers [3–9,11,12] have made significant contributions to various aspects of the field. These include the development of innovative deep-learning methods for identifying toxic comments, understanding sentiment in YouTube video comments, and detecting harmful language, cyberbullying, and sexual harassment. Some have also proposed novel hybrid models combining LSTM and CNNs for cyberbullying detection.

In [13], authors have demonstrated the potential of transfer learning with a text-to-text transformer model for developing effective toxic comment classification models. Additionally, the study by authors in [15] identified the challenges associated with toxic comment classification, including the complexities of context-dependent, non-normative, and subtle forms of toxicity and the potential biases against certain identity groups. The implications of these findings underline the need for the development of more robust and less biased classification models for toxic comments.

[1] https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge/data.

These cumulative efforts highlight the ongoing developments and challenges in the field of toxic comment classification, emphasizing the significance of continuous research and advancement to address the complexities and nuances involved in this domain.

The following section provides a detailed methodology workflow followed in this study for the toxic classification task.

3 Methodology Workflow

This section presents an outline of the methodological workflow designed for building classifiers capable of categorizing toxic comments extracted from the dataset. The workflow includes essential steps, including (i) data collection, (ii) data pre-processing, (iii) feature extraction, and (iv) an explanation of both shallow and deep learning models used for the toxic classification task.

Detailed elaboration of the methodology workflow follows subsequently.

3.1 Dataset

The research utilizes data from the Jigsaw/Conversation AI dataset available through the Kaggle Toxic Comment Classification Challenge. This dataset contains a significant volume of comments sourced from Wikipedia, which have been manually annotated by human raters into six different types of toxicity. These toxicity classes include Toxic, Severe toxic, Obscene, Threat, Insult, and Identity threat.

Regarding toxic comment classification, the aforementioned dataset is comprised of two primary files, one intended for training machine learning models and the other for testing these models. Additionally, the training dataset contains 159,571 observations, while the test dataset comprises 153,164 observations.

Further details on data pre-processing follows next.

3.2 Data Pre-processing

Data pre-processing involves a series of steps to convert raw data into comprehensible format suitable for data analysis. In this research work, following basic data pre-processing steps are implemented on the dataset.

- **Lowercase Text Conversion** On social media, online users exhibit diverse writing patterns. This often leads to an irregular use of letter cases, with some data presented in lowercase, some in uppercase, or a combination of both. Consequently, this inconsistency can pose challenges during data pre-processing, particularly in tasks involving dictionary look-ups, as mismatches in letter cases can lead to failures. To avoid this, the entire text in the data is converted to lowercase [1].
  After this step, the next pre-processing step is the expansion of acronyms and slangs.

- **Expansion of Acronyms and Slangs** The presence of acronyms and slangs in social media text can make it challenging to precisely evaluate the toxicity of a comment. Expanding these acronyms and slang expressions, such as converting "f*ck" to "fuck" or "mf" to "motherfucker," helps in capturing the intended meaning and maintaining the overall context of the text. This process leads to more accurate classification of toxic content. Thus, for expansion of acronyms and slangs, a dictionary compiled by authors in [1,2] has been used.
  After this step, the next step is the removal of punctuation marks.
- **Removal of Punctuation Marks**
  Punctuation marks generally do not convey significant meaning in text analysis. However, retaining them can disrupt the tokenization process and introduce unwanted noise to the text data. Therefore, their removal simplifies the tokenization process and cleans the data, facilitating smoother and more precise text analysis.
  To achieve this, regular expressions are utilized to remove the punctuation marks. The subsequent step involves the elimination of stopwords.
- **Removal of Stopwords** Stopwords such as "the," "is," "and," and "but," are frequently found in the social media text, yet they usually do not hold significant meaning for analysis. The elimination of these stopwords reduces the dimensionality of the data, leading to more effective and efficient text analysis and processing. During this step, the NLTK(Natural Language Toolkit) stopwords list is used for removing stopwords.
  The next data pre-processing step is the removal of HTML entities, URLs and newline characters.
- **Removal of HTML entities, URLs and Newline characters** The elimination of HTML entities, URLs, and newline characters is essential to ensure the cleanliness of the text data and remove any irrelevant elements that might disrupt subsequent text analyses.This process is crucial as it refines the text, making it more suitable for subsequent procedures such as tokenization and feature extraction. Thus, these elements have been removed with the help of regular expressions.
  The final step in data pre-processing involves the tokenization and lemmatization of text.
- **Tokenization and Lemmatization** Tokenization helps in extracting important features from the text, which are then used as inputs for machine learning models for different analysis tasks. Lemmatization, on the other hand, helps in normalizing words with similar meanings and reduces the overall dimensionality of the data, which can lead to improved text analysis.
  In this study, text tokenization for shallow machine learning models has been executed using the 'build_tokenizer()' function from the scikit-learn[2] library. For deep learning models, the 'Tokenizer' class from the Keras[3] library has

[2] https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html.
[3] https://rb.gy/aexz8.

been employed for text tokenization. Furthermore, for text lemmatization, the "WordNetLemmatizer" from the NLTK[4] library has been utilized.

After data pre-processing, the next step is extraction of features from the processed data.

3.3 Feature Extraction

Machine learning models do not possess the innate ability to directly understand human language. As a result, there is a need to convert language into a mathematical representation, typically a vector with specific dimensions, to serve as input for these models. This conversion can be accomplished using various methods. For instance, statistical techniques such as TF-IDF, bag-of-words etc. can be utilized, or alternatively, pre-trained models like word2vec or glove can be employed to convert textual data into vectors.

In the present study, the TF-IDF statistical measure has been applied with shallow learning models using TfidfVectorizer of the scikit-learn library, while pre-trained embeddings from keras[5] library have been employed with deep learning algorithms for feature extraction. The resulting vector outputs are subsequently employed as input for the machine learning models to generate the final output.

The following section presents comprehensive details regarding different machine learning models that will be implemented for the classification of text in terms of toxicity.

3.4 Machine Learning Models

For the classification of toxic comments, two types of machine learning models have been utilized - shallow learning and deep learning models. Shallow learning models include conventional machine learning algorithms whereas, deep learning models include deep learning algorithms. The difference between these two models lies in the feature extraction process as described in Subsect. 3.3.

Both shallow and deep learning models are further classified as supervised and unsupervised learning algorithms. In supervised learning, data is labelled, and the algorithms predict the output from the input data. Whereas, in unsupervised learning, data is unlabelled, and algorithms learn to identify inherent structure from the input data. The supervised and unsupervised algorithms of both the models are shown in Fig. 1.

Initially, the study outlines the specifics of shallow learning algorithms that have been applied to the toxic classification task, followed by a subsequent section detailing the deep learning algorithms.

[4] https://www.nltk.org/_modules/nltk/stem/wordnet.html.
[5] https://keras.io/examples/nlp/pretrained_word_embeddings/.

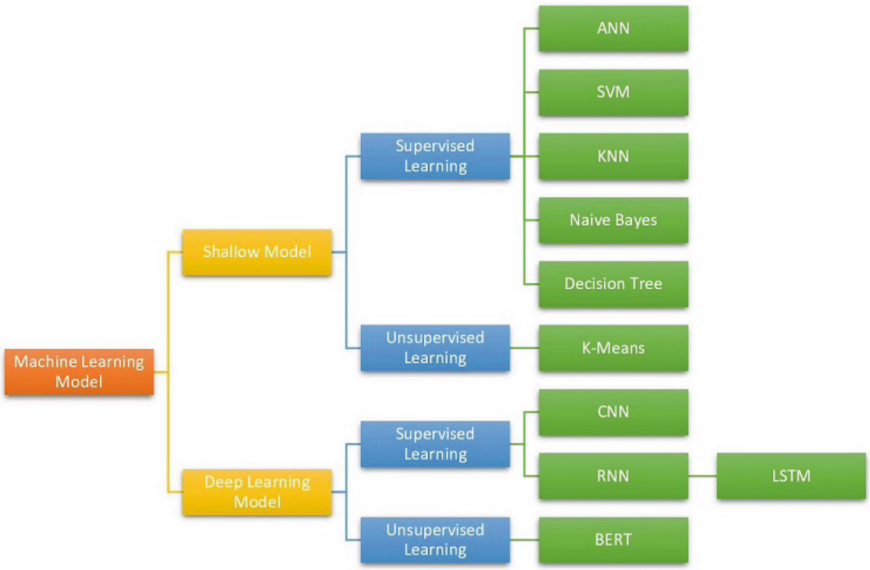

Fig. 1. Classification of Machine Learning Models.

Shallow Learning Models. Shallow learning models are typically used for various tasks, including classification, regression, and clustering. Unlike deep learning models, shallow learning models have a limited number of layers, and they are more interpretable and easier to train on smaller datasets.

In this particular research, the focus is on three of the most widely used shallow model algorithms applied to the toxic classification dataset, namely Logistic Regression (LR), Naive Bayes (NB), and Decision Trees (DT) [10,14,18]. The subsequent sections provide a detailed examination of these algorithms.

Logistic Regression. Logistic Regression is a commonly used classification algorithm that is well-suited for binary classification tasks, where the goal is to predict one of two possible outcomes. It works by modeling the probability of the binary outcome using a logistic function. The algorithm's output predicts the binary 0 or 1 by determining the degree of probability generated by the logistic regression.

Naive Bayes. Naive Bayes is a generative probability model commonly used for classification tasks. This model operates under the assumption that the features used for classification are conditionally independent, which simplifies its calculations. It is renowned for its speed and simplicity, making it a popular choice, particularly in text classification tasks. In essence, for each of the k possible classes (Ck), this classifier assigns probabilities (p(Ck — x1, ..., xn)), where (x1, ..., xn) denotes the input vector. This approach allows it to estimate the likelihood of a given class given the observed features efficiently.

Decision Tree. Regression and classification problems can be effectively addressed using decision trees, which are a type of tree-based technique. To generate the output, a decision tree is constructed, branching from a root node with homogeneously distributed data to increasingly heterogeneous leaf nodes. The rules for branching are selected by the tree based on the optimal informational gain from each split. This splitting process is recursively repeated for each child node until the algorithm determines that there is no further advantage to be gained from additional splits.

After shallow learning models, details of Deep Learning algorithms used for toxic classification follows next.

Deep Learning Algorithms. In the context of machine learning, artificial neural networks serve as a specialized branch underlying all deep learning algorithms. These networks are designed to simulate the learning process of the human brain, processing input data and generating output. The three essential functions performed by a neural network include receiving input through the input layer, performing computations via the hidden layer, and producing the final output from the output layer.

The input layer receives raw input and transmits data (features) to the hidden layer, without conducting any computations. The hidden layer applies the activation function to the features received from the input layer, performing various calculations and providing the output to the next layer. The final layer, the output layer, generates the ultimate output based on the information processed by the hidden layer.

Neurons, nodes, or perceptrons constitute the components of these layers. Each neuron is characterized by its weight, bias, and activation function, which collectively determine the significance of the input in predicting the final value. Various mathematical procedures, including the sigmoid, hyperbolic tangent (tanh), rectified linear unit (ReLU), and softmax, are utilized within the activation function to assess the input's significance.

The training process of a neural network involves two key steps:

- **Forward Propagation** Input data is passed through the network in the forward direction, with each hidden layer processing the data according to the activation function, leading to the final prediction as the output.
- **Backward Propagation** After generating the final output, a comparison is made with the actual value. Parameter values are updated based on the deviation or proximity of the output to the actual value. The forward propagation process is repeated with the modified parameter values to generate new outputs.

In this research, three widely used deep learning algorithms are explored for toxic classification. The details of these algorithms are discussed subsequently. The first algorithm is the Convolutional Neural Network (CNN) followed by Long Short-Term Memory (LSTM)(a type of Recurrent Neural Networks (RNNs)), and finally, the Bidirectional Encoder Representations from Transformers (BERT).

Convolutional Neural Networks (CNNs). Convolutional neural networks are multilayer neural network architectures specifically designed for classification tasks. They typically comprise three key layers: the convolutional layer, responsible for the majority of computations, the pooling layer, which reduces input data dimensionality by minimizing parameter count, and the fully connected layer, which integrates information from the preceding layers. The convolutional layer serves as the backbone of a CNN, involving input data, a filter, and a feature map. The pooling layer is instrumental in reducing the parameters within the input data.

Recurrent Neural Networks (RNNs). Recurrent Neural Networks are specialized neural networks tailored for sequential problems, wherein each node possesses a memory component for handling sequences of inputs. This unique capability allows RNNs to compress prior inputs into a low-dimensional space and effectively manage issues related to position invariance, making them particularly efficient in processing text data where order and structure are crucial.

Unlike conventional feed-forward neural networks, RNNs are equipped with a memory component that enables the processing of input sequences [17].

Long Short-Term Memory (LSTM). Long Short-Term Memory is a type of recurrent neural network (RNN) architecture that is specifically designed to handle the long-term dependencies in sequential data.

LSTMs are equipped with memory cells that can maintain information over a long period, allowing them to remember information for a significant duration. The architecture of LSTM networks includes various gates, such as the input gate, forget gate, and output gate, which regulate the flow of information. These gates enable LSTMs to selectively remember or forget certain information.

Bidirectional Encoder Representations from Transformers (BERT). Unlike traditional models that read text input sequentially, BERT is designed to understand the context of words in a sentence by considering the surrounding words on both sides simultaneously. This bidirectional approach enables BERT to capture intricate linguistic patterns and context, leading to more accurate and context-aware language understanding. Its ability to generate high-quality contextualized word embeddings has made it a valuable asset in the field of natural language processing (NLP).

After machine learning models, the details of experiments and theirs outcomes follows next.

4 Experiments and Results

The methodology outlined in the preceding section was applied to the dataset specified in Sect. 3.1 for the purpose of toxic classification experiments. This involved a series of steps, including data collection, data pre-processing, feature extraction, selection of a machine learning model, training the model using the training dataset, testing the model with separate test data, and assessing the classifier's performance using various metrics which provide insights into the models' ability to correctly identify different types of toxic content in the dataset.

In the context of toxic classification as a multi-label classification problem, an extensive evaluation of shallow machine learning algorithms namely, Multinomial Naive Bayes, Logistic Regression and Decision Trees, has been carried out for six different types of toxicity classes present in the dataset. This evaluation employed metrics such as F1 score and recall. These metrics were chosen because the accuracy metric alone may not adequately capture the complexities of multi-label classification tasks, particularly in cases involving imbalanced datasets.

The performance of the shallow learning algorithms in toxic classification for each toxic class is illustrated in Table 1. The experiment implementation has been done using scikit-learn[6] python library.

Table 1. Performance Evaluation of Shallow Learning Models for Toxic Classification.

Index	Classifier Model	Label	Recall	F1 Score
0	MultinomialNB	toxic	0.482998670582251	0.63656203029990
1	MultinomialNB	severe_toxic	0.021937893081761	0.04224362130181
2	MultinomialNB	obscene	0.469167110687344	0.62214786031211
3	MultinomialNB	threat	0.0	0.0
4	MultinomialNB	insult	0.36701974986939	0.51139399757082
5	MultinomialNB	identity_hate	0.00783181357649	0.01534644366480
6	LogisticRegression	toxic	0.6105000491585	0.73133891997268
7	LogisticRegression	severe_toxic	0.25643081761006	0.35153007835561
8	LogisticRegression	obscene	0.63700201912561	0.74736262671441
9	LogisticRegression	threat	0.12331560283688	0.20663228803617
10	LogisticRegression	insult	0.52354568850418	0.63817671120194
11	LogisticRegression	identity_hate	0.20074974670719	0.31037916632762
12	DecisionTreeClassifier	toxic	0.28004419993417	0.43018988752237
13	DecisionTreeClassifier	severe_toxic	0.05012971698113	0.09034301625967
14	DecisionTreeClassifier	obscene	0.46443450461314	0.61318146764799
15	DecisionTreeClassifier	threat	0.07508865248227	0.12884417058563
16	DecisionTreeClassifier	insult	0.37819032630499	0.49853390670456
17	DecisionTreeClassifier	identity_hate	0.12246200607903	0.20903444007670

Based on the findings presented in Table 1, it can be concluded that the Logistic Regression model demonstrates superior performance in toxic classification, exhibiting the highest F1 scores for three classes, specifically, toxic, obscene, and insult. The Multinomial Naive Bayes model follows, while the Decision Tree classifier model performs the least effectively. Further enhancements in the toxic classification performance can be achieved through the adoption of state-of-the-art deep learning algorithms. Therefore, this paper proceeds to evaluate the

[6] https://scikit-learn.org/stable/supervised_learning.html#supervised-learning.

performance of three prominent deep learning algorithms, namely CNN, LSTM, and BERT, using the recall and F1 score metrics.

The performance of these algorithms for toxic classification is shown in Table 2. Deep learning models implementation uses Keras library.

Table 2. Performance Evaluation of Deep Learning Models for Toxic Classification.

S. No.	Classifier Model	Recall	F1 Score
1	Convolutional Neural Networks(CNN)	0.86133	0.84528
2	Long Short-Term Memory(LSTM)	0.87212	0.85988
3	Bidirectional Encoder Representations from Transformers (BERT)	0.86824	0.83631

Based on the analysis presented in Table 2, it is evident that LSTM emerges as the best performer, exhibiting the highest F1 score among the evaluated models. Moreover, a result comparison of Table 1 and Table 2 reveals that deep learning models consistently outperform shallow learning models in terms of performance.

The subsequent section provides the conclusive remarks for this study.

5 Conclusion and Future Work

The prevalent use of offensive language within the vast volume of user-generated content on social media platforms is a significant concern. To foster a positive online environment, the implementation of an automated system for identifying and categorizing toxic comments using machine learning techniques is imperative. Research findings suggest that deep learning models tend to outperform shallow machine learning models in this context.

Furthermore, it has been emphasized that selecting an appropriate model should be based on the specific requirements of the task rather than solely relying on the most popular choice. Notably, in the context of toxic classification, the LSTM model has demonstrated better performance compared to the latest BERT model. Moreover, it has been observed that the performance of deep learning models can be further improved through the use of ensemble methods.

In future work, the research will involve conducting detailed experiments on deep learning models with the integration of ensembling techniques. Additionally, there will be a focus on creating a toxic dictionary based on social media language, along with emojis, which will be integrated into the data pre-processing steps to enhance the effectiveness of classifiers in toxic classification.

References

1. Agrawal, T., Singhal, A.: An effective knowledge-based pre-processing system with emojis and emoticons handling on Twitter and Google+. Int. J. Innovative Technol. Exploring Eng. (IJITEE) **8**(11), 349–361 (2019)

2. Agrawal, T., Singhal, A.: An efficient knowledge-based text pre-processing approach for Twitter and Google+. In: Singh, M., Gupta, P.K., Tyagi, V., Flusser, J., Ören, T., Kashyap, R. (eds.) ICACDS 2019 Part II. CCIS, vol. 1046, pp. 379–389. Springer, Singapore (2019). https://doi.org/10.1007/978-981-13-9942-8_36
3. Alhabash, S., McAlister, A.R., Lou, C., Hagerstrom, A.: From clicks to behaviors: the mediating effect of intentions to like, share, and comment on the relationship between message evaluations and offline behavioral intentions. J. Interact. Advert. **15**(2), 82–96 (2015)
4. 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, pp. 759–760 (2017)
5. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching word vectors with subword information. Trans. Assoc. Comput. Linguistics **5**, 135–146 (2017)
6. Burnap, P., Williams, M.L.: Us and them: identifying cyber hate on twitter across multiple protected characteristics. EPJ Data Sci. **5**, 1–15 (2016)
7. Cunha, A.A.L., Costa, M.C., Pacheco, M.A.C.: Sentiment analysis of YouTube video comments using deep neural networks. In: Rutkowski, L., Scherer, R., Korytkowski, M., Pedrycz, W., Tadeusiewicz, R., Zurada, J.M. (eds.) ICAISC 2019 Part I. LNCS (LNAI), vol. 11508, pp. 561–570. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-20912-4_51
8. Farag, N., El-Seoud, S.A., McKee, G., Hassan, G.: Bullying hurts: a survey on non-supervised techniques for cyber-bullying detection. In: Proceedings of the 8th International Conference on Software and Information Engineering, pp. 85–90 (2019)
9. Gada, M., Damania, K., Sankhe, S.: Cyberbullying detection using LSTM-CNN architecture and its applications. In: 2021 International Conference on Computer Communication and Informatics (ICCCI), pp. 1–6. IEEE (2021)
10. Gautam, G., Yadav, D.: Sentiment analysis of twitter data using machine learning approaches and semantic analysis. In: 2014 Seventh International Conference on Contemporary Computing (IC3), pp. 437–442. IEEE (2014)
11. Karlekar, S., Bansal, M.: Safecity: understanding diverse forms of sexual harassment personal stories. arXiv preprint arXiv:1809.04739 (2018)
12. Kurita, K., Belova, A., Anastasopoulos, A.: Towards robust toxic content classification. arXiv preprint arXiv:1912.06872 (2019)
13. Raffel, C., et al.: Exploring the limits of transfer learning with a unified text-to-text transformer. J. Mach. Learn. Res. **21**(1), 5485–5551 (2020)
14. Soumya George, K., Joseph, S.: Text classification by augmenting bag of words (BOW) representation with co-occurrence feature. IOSR J. Comput. Eng. **16**(1), 34–38 (2014)
15. Van Aken, B., Risch, J., Krestel, R., Löser, A.: Challenges for toxic comment classification: an in-depth error analysis. arXiv preprint arXiv:1809.07572 (2018)
16. Wulczyn, E., Thain, N., Dixon, L.: Ex machina: personal attacks seen at scale. In: Proceedings of the 26th International Conference on World Wide Web, pp. 1391–1399 (2017)
17. Yin, W., Kann, K., Yu, M., Schütze, H.: Comparative study of CNN and RNN for natural language processing. arXiv preprint arXiv:1702.01923 (2017)
18. Zhao, C., Zhang, H., Zhang, X., Liu, M., Hu, Z., Fan, B.: Application of support vector machine (SVM) for prediction toxic activity of different data sets. Toxicology **217**(2–3), 105–119 (2006)