Toxic Messages Classification in Social Media

Mikhail Dolgushin and Yuliya Bidulya(✉)

University of Tyumen, Tyumen, Russia

Abstract. The aim of our work is to develop a software that could detect toxicity in the Russian segment of social media. In this paper, we investigated the problem of toxic detection in messages in Russian language. We implemented a set of features using selected vector models, trained some classifiers on the dataset about fourteen thousand annotated messages and compare results. Experiments were conducted with a calculation of accuracy, precision, and recall values. F1 measure reached the value 0.91, accuracy value is 0.87.

Keywords: Toxic detection · Feature extraction · Social media analysis

1 Introduction

The analysis of messages in social networks is one of the urgent tasks in the modern Internet community. It constitutes the most important stage in the work of a group moderator in a social network. Typically, the moderation process can be considered as a classification task for messages according to a few criteria. One of the criteria is the presence of toxicity in the messages: a rude, disrespectful comment that can cause someone to leave the discussion [1]. There are media and social media monitoring applications that offer social media comment moderation services [2]. As a rule, checking comments on social networks is done manually by moderators. Currently, many social networks are implementing simple filters based on a list of forbidden words compiled by a moderator. These filters allow you to delete messages with obscenities or some special forbidden words. However, such filters do not consider the semantics of the message, which often leads to either the deletion of completely harmless messages, or the omission of comments with more subtle insults.

Determination of toxicity in a text using machine learning methods has been discussed for a long time in publications. However, in the Russian-speaking segment of social networks, this problem has not yet been sufficiently developed. One of the main reasons for this is the lack of labeled training data. In this work, we present the study of methods for detecting toxic expressions in messages from the social network VKontakte (https://vk.com/) using the example of a Russian-language dataset provided on the website of the well-known platform Kaggle [3]. We set a goal to choose the most effective algorithm for the automated classification by the presence or absence of toxic statements in messages. The final aim of our work is to develop a software that could detect toxicity in the Russian segment of social media.

A. Rocha and E. Isaeva (Eds.): Perm Forum 2021, LNNS 342, pp. 60–65, 2022.
https://doi.org/10.1007/978-3-030-89477-1_7

2 Related Works

Finding toxicity in messages can be viewed as a text classification task. This task includes determining to which class the text of message belongs. In this work, each message can be toxic or non-toxic.

Many papers are devoted to similar tasks: recognizing aggressiveness or hate speech in the text [4], searching for insults or offensive vocabulary, sentiment analysis with a negative connotation, etc. Review article [5] shows the main problems of this task: general terms, jargon, memes, vocabulary and cultural canons of social groups can vary significantly; attackers use special methods to bypass automatic moderation on social networks; the presence of messages with spelling errors and typos complicates the technical processing of texts.

A common approach is to solve the binary classification problem for individual messages. In this case, the following text preprocessing script is used: tokenization, removal of stop words, POS markup, extraction of text features, which leads to the presentation of a bag of words. In addition, the representation of texts in word embeddings models is also widely used in classification problems [6]. The most popular model is Word2vec, presenting words from the vocabulary mapped to vectors of real numbers. [7].

Among the classification methods, both traditional approaches (naive Bayesian, random forest, etc.) [8] and methods of deep learning and neural networks are popular [9].

Neural networks show the most stable and highest performance. In turn, traditional methods provide more opportunities for implementation as web applications, since they require less extensive computing resources.

The authors [10] propose to use the next features for rumor detection in tweets: words expressing an opinion, vulgar words, emoticons, abbreviations, verbs describing speech acts, average complexity of a word in a tweet, negative tone, speech act category. In this work, we use a similar approach to improve the results of traditional classifiers. Besides, we used the word embedding models as texts representation for classifier training.

3 Data and Preprocessing

We used Kaggle Russian Language Toxic Comments Dataset [3]. This collection of annotated comments from 2ch and Pikabu was published on Kaggle in 2019 and consists of 14,412 unique comments, where 4,826 texts were labeled as toxic, and 9,586 as non-toxic. The average length of comments is 175 characters; the minimum length is 21, and the maximum is 7,403.

Message preprocessing includes tokenization, stop-word and punctuation removal, lemmatization provided by NLTK [11] and Pymorphy2 [12] libraries.

4 Features

4.1 Vector Models

To extract message signs, we vectorized message texts using libraries. Implemented set of features we used as the input data for the machine learning classification. At the output, we get labels for messages, depending on presence of toxic in the message text.

1) Term Frequency (TF) vector model is a matrix of token usage frequencies in documents [13].

   For the TF implementation matrix, the Count Vectorizer acquired in the Scikit-learn library [14] is used with parameter: size 300.
2) Word2Vec [15] is a tool for calculating vector representations of words that implements two main architectures: Continuous Bag of Words and Skip-gram. The input is a text corpus, and the output is a set of word vectors.
3) We applied Word2vec model of Gensim with the following model parameters: size 150, window 10. The final value for each word is obtained by averaging all the numbers in its vector.
4) Doc2Vec [16] is a tool like Word2Vec, but the input is a whole text document. We applied Doc2vec model of Gensim with the following model parameters: size 10. The final value for each word is obtained by averaging all the numbers in its vector.
5) FastText [17] is a library containing pre-trained ready-made word representations and a classifier, that is, a machine learning algorithm that breaks words into classes.We used FastText model of Gensim with the following model parameters: size 150, window 5.

Scaling from 0 to 100 results was applied using the MaxMinScaler class from the Scikit-learn library for all described above models except Count Vectorizer.

4.2 Content Features

To improve the quality of the classification, we have expanded the properties matrix with additional features determined from the message text: the presence in the text of persona's names, names of locations, organizations, positive, negative, neutral emotional colors, the presence of vulgar speech. The definition of features was carried out using the Dostoevsky [18], SpaCy [19] libraries and the obscene vocabulary.

Further, the results of Pearson's correlation were obtained for the selected features with the sign of the presence of toxicity from the marked dataset. The calculated values are shown in Table 1.

As shown, the largest modulus coefficient is shown by features: negative, neutral coloring and the presence of vulgar words.

Table 1. Pearson's correlation coefficients.

Feature	Coefficient
Person	–0.013
Location	–0.063
Organization	–0.017
Positive	–0.047
Negative	0.361
Neutral	–0.267
Vulgar	0.413

5 Classification Methods

In this section the classification methods that we applied in our work are described [14].

1) Support vector classification (SVC) [20] is a set of similar supervised learning algorithms used for classification and regression analysis problems. The training time has a quadratic dependence with the number of samples; therefore, this algorithm is not used on samples larger than several tens of thousands of values.
2) The Multinomial Naive Bayesian Classifier (MNB) [20] is a modification of the Naive Bayesian Classifier where function vectors represent the frequencies with which certain events were generated by a polynomial distribution. This model is commonly used to classify documents.
3) A support vector machine modification using a naive Bayesian algorithm (NBSVM) is a support vector machine implementation in which vectors are constructed based on the coefficients of the logarithm of the naive Bayesian algorithm as characteristic values. The NBSVM implementation is taken from the Kaggle [21].
4) The Random Forest Classification (RFC) [20] is a classification model using a machine learning algorithm that uses an ensemble of decision trees (decision trees).

6 Experiments and Results

In the software implementation of the classifiers Multinomial Naive Bayesian, Support vector classification and Random Forest the classes presented in the SciKitLearn library were used [14].

In all methods we used cross-validation with stratification to prevent overfitting and to decrease the influence of class imbalance.

Table 2 demonstrates a Precision, Recall and F1 metrics values calculated as a result of experiments with Russian Language Toxic Comments Dataset described above in Sect. 3.

Table 2. The results of message classification.

Vector model	Classifier	Class "Toxic"			Class "Un-toxic"			Accuracy
		Precision	Recall	F1	Precision	Recall	F1	
Term frequency	SVC	.819	.438	.571	.780	.954	.858	.787
	MNB	.878	.247	.385	.732	.984	.839	.745
	NBSVM	.825	.433	.568	.779	.956	.859	.787
	RFC	**.902**	.265	.410	.737	**.986**	.844	.753
Doc2Vec	SVC	.881	.745	.745	.881	.902	.892	.851
	MNB	.797	.698	.744	.864	.915	.889	.845
	NBSVM	.716	.715	.715	.864	.864	.864	.816
	RFC	.784	.744	.763	.881	.902	.891	.851
Word2Vec	SVC	.842	.707	.769	.842	.842	.902	.862
	MNB	.834	.668	.742	.855	.936	.894	.849
	NBSVM	.778	.779	.778	.894	.894	.894	.857
	RFC	.846	.709	.771	.871	.938	.903	.864
Fast text	SVC	.837	.742	**.786**	.883	.931	**.906**	**.870**
	MNB	.760	**.792**	.776	**.899**	.880	.889	.852
	NBSVM	.838	.728	.779	.878	.933	.904	.867
	RFC	.861	.662	.748	.854	.949	.899	.856

From the results of Table 2, it follows that the FastText vectorization and SVC classification have the highest accuracy, although in some special cases FastText vectorization with the MNB classification does not lag this classification. The best of the investigated classification algorithms on a toxic comment dataset to classify the alignment of SVC classifier and FastText vectorization with an accuracy of 87%.

At the next stage, we chose the Word2Vec representation and the SVC classifier to expand the features with the content features described above in Sect. 4.2. The features with the highest correlation were added to the set of features of the training sample for the Word2Vec model, the classifier was trained, and the results were compared with previous experiments. It turned out that the expansion of the feature set led to a slight increase in accuracy. We find it interesting to conduct a similar experiment for all studied classifiers and vector representations.

7 Conclusion

After analyzing the results of the experiments, we chose an approach to implement a Web application for the automatic determination of toxicity in VKontakte messages. API was developed by the Flask framework and uses the trained Word2Vec + NBSVM set, as well as to determine additional message signs: the presence of proper names, emotional color, the presence of curses. In the future, we plan to use own dataset for training models and evaluating results.

References

1. Georgakopoulos, S.V., Tasoulis, S.K., Vrahatis, A.G., Plagianakos, V.P.: Convolutional neural networks for toxic comment classification (2018). arXiv preprint arXiv:1802.09957
2. Medialogiya - monitoring and analysis of media and social networks (rus.). https://www.mlg.ru
3. Russian Language Toxic Comments. https://www.kaggle.com/blackmoon/russian-language-toxic-comments
4. Ventirozos, F.K., Varlamis, I., Tsatsaronis, G.: Detecting aggressive behavior in discussion threads using text mining. In: Gelbukh, A. (ed.) CICLing 2017. LNCS, vol. 10762, pp. 420–431. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-77116-8_31
5. Levonevskiy, D., Malov, D., Vatamaniuk, I.: Estimating aggressiveness of russian texts by means of machine learning. In: Salah, A.A., Karpov, A., Potapova, R. (eds.) SPECOM 2019. LNCS (LNAI), vol. 11658, pp. 270–279. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26061-3_28
6. Camacho-Collados, J., Pilehvar, M.T.: From word to sense embeddings: a survey on vector representations of meaning (2018). arXiv:1805.04032. Bibcode:2018arXiv180504032C
7. Pietro, M.D.: Text Classification with NLP: TF-IDF vs Word2Vec vs BERT. https://towardsdatascience.com/text-classification-with-nlp-tf-idf-vs-word2vec-vs-bert-41ff868d1794
8. Plaza-del Arco, F.M, Molina-Gonzalez, D., Martın-Valdivia, T., Urena-Lopez, A.: SINAI at SemEval-2019 Task 6: incorporating lexicon knowledge into SVM learning to identify and categorize offensive language in social media. In: The 13th International Workshop on Semantic Evaluation (SemEval) (2019)
9. Pavlopoulos, J., Thain, N., Dixon, L., Androutsopoulos, I.: ConvAI at SemEval-2019 task 6: offensive language identification and categorization with perspective and BERT. In: SemEval, Minneapolis, USA (2019)
10. Chernyaev, A., Spryiskov, A., Ivashko, A., Bidulya, Y.: A rumor detection in Russian tweets. In: Karpov, A., Potapova, R. (eds.) SPECOM 2020. LNCS (LNAI), vol. 12335, pp. 108–118. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-60276-5_11
11. NLTK documentation. https://www.nltk.org
12. Morphological analyzer pymorphy2. https://pymorphy2.readthedocs.io
13. Document-term matrix. https://en.wikipedia.org/wiki/Document-term_matrix
14. Pedregosa, F., et al.: Scikit-learn: machine learning in python. J. Mach. Learn. Res. **12**, 2825–2830 (2011)
15. Rehurek, R., Sojka, P.: Software framework for topic modelling with large corpora. In: LREC 2010 Workshop on New Challenges for NLP Frameworks, pp. 45–50, Valletta, Malta, May. ELRA (2010). http://is.muni.cz/publication/884893/en
16. Gensim: Doc2vec. https://radimrehurek.com/gensim/models/doc2vec.html
17. Mestre, M.: FastText: stepping through the code. https://medium.com/@mariamestre/fasttext-stepping-through-the-code-259996d6ebc4
18. Dostoevsky: Sentiment Analysis Library for Russian Language. https://pypi.org/project/dostoevsky
19. SpaCy: Industrial-Strength Natural Language Processing. https://spacy.io
20. Wang, S., Manning, C.D.: Baselines and Bigrams: Simple, Good Sentiment and Topic Classification, Department of Computer Science, Stanford University, Stanford, CA 94305. https://nlp.stanford.edu/pubs/sidaw12_simple_sentiment.pdf
21. Wang, Z.: NBSVM. https://www.kaggle.com/ziliwang/nbsvm