UDC 004.622 **doi: 10.20998/2522-9052.2020.3.14**

Oleksandr Orlovskyi, Sergey Ostapov

Yuriy Fedkovych Chernivtsi National University, Chernivtsi, Ukraine

ANALYSIS OF THE TEXT PREPROCESSING METHODS INFLUENCE ON THE DESTRUCTIVE MESSAGES CLASSIFIER

Abstract. Problem. Social networks are increasingly becoming an environment for threats, insults, profanity and other destructive manifestations of human communication. Today, a huge number of people are involved in online platforms, and the amount of content created and reactions to it is constantly breaking records. Therefore, there is a need to automate the detection and counteraction of antisocial influences. One of the important areas of such activities is the detection of toxic comments that contain threats, insults, profanity, contempt for others and more. To perform this task, researchers usually build a classifier based on neural networks. And for their training they use a collected or publicly available set of data. The article investigates how different methods of pre-processing of input data affect the final accuracy of the classifier. Previous studies in this direction have confirmed the presence of an impact on the result, but did not allow to draw definitive conclusions about the effectiveness. **Goal.** Research of preliminary processing of text data methods influence on the destructive messages classifier. **Results.** It has been shown that the effect of a particular method can be quite dependent on the content in the data set. In addition, it is noted that sometimes the impact may be insignificant, and in some cases may even lead to a worsening of the result. It is also justified the need to pre-check the data set for the percentage of elements that fall under the impact of a particular method. **Originality.** The methods of data processing are evaluated on the basis of English and Russian data sets. **Practical significance.** The obtained results allow to make better decisions about the usage of certain pre-processing methods to improve the accuracy of the destructive messages classifier.

Keywords: data preprocessing; destructive text data detection; neural networks; data mining; data set; data processing; classifier.

Introduction

Nowadays, especially in quarantine conditions, people increasingly prefer to communicate in online environments. Based on data from the Statista portal, we can say that in July 2020, three of the most popular social networks broke the bar of 2 billion active users, while three more hit 1 billion. Also in the same place we can mention other top growth rate online platforms like TikTok [1]. Accordingly, the amount of content and reactions to it is also growing rapidly. Among this large-scale volume of information, users encounter both cognitive, entertaining, developmental content, and various negative manifestations of human communication. These include cyberbullying, threats, obscene language and any messages created to offend the feelings and dignity of the interlocutors.

According to the need for timely response by the communication platforms to the above destructive manifestations, the task of automating the detection of such negative factors for a healthy atmosphere on the platform is extremely important. In addition, having detected such a manifestation, it is necessary to quickly neutralize it. Due to the size of the platforms and according to the number of interaction points between users, we note that even an army of moderators will not be able to cope with the flow of information as quickly as automated solutions can. Therefore, the study of factors influencing the accuracy and speed of their work is an urgent task.

Previous research analysis. A variety of approaches are used to automatically detect destructive messages, from simple statistical analysis of textual data to complex neural network-based approaches with complex architectures, reinforced with large amounts of data. The study [2] presents a manual selection of features of destructive messages and the use of context to improve accuracy. It is also noted that taking into account features that are based on uppercase and lowercase letters, emoji does not lead to a significant increase in the accuracy of the final result.

Research [3] demonstrates a wide range of destructive messages categories, to identify each of which you can choose an individual approach. This paper also emphasizes the importance of understanding the nature of the origin of each of the categories. Moreover, authors show some possibilities to experiment with the architecture of neural networks for accuracy improving.

From [4] we can learn the basic methods of statistical analysis applied to text for solving problems in our chosen topic. In addition, the paper pays great attention to the influence of syntactic connections in the sentence on the quality of the final classification result.

From the study [5] we can better understand the role of context in solving the domain problems. At [6] authors analyze a wide range of classifiers in the context of our problem, and at [7] it is demonstrated how to reverse different classification methods errors in favor.

In the context of our study, the paper [8] requires the most attention. Authors studied 35 preprocessing methods applied on English-based dataset and showed that the usage of most of them is a rather ambiguous solution. Based on the above study, we have the opportunity to investigate the effectiveness of using some of the described methods on the Russian- and English- language based data sets.

The aim of this paper is to study the impact of some text pre-processing methods on improving the accuracy of destructive message detectors.

To solve the selected problem, we train a classifier based on an artificial neural network and compare the

results of toxic comments identification after applying each of the methods with the initial result of identification. We will also consider the option of combining these methods with each other.

Data samples

The experiments were performed using two data sets. The first – "Toxic Comment Classification Challenge" [9] has the following characteristics, presented in Table 1.

Table 1 – **"Toxic Comment Classification Challenge" data set characteristics**

Parameter	Value
Language	English
Size (MB)	~ 52 MB
Number of records	~ 128.000
Number of classification categories	6 (toxic, severe toxic, obscene, threat, insult, identity hate)

The second is "Russian Language Toxic Comments. Small dataset with labeled comments from 2ch.hk and pikabu.ru" [10] has the following characteristics presented in Table 2.

Table 2 – **"Russian Language Toxic Comments. Small dataset with labeled comments from 2ch.hk and pikabu.ru" data set characteristics**

Parameter	Value
Language	Russian
Size (MB)	4.45 MB
Number of records	~ 11.500
Number of classification categories	1 (toxic)

The following data distribution principle, shown in Fig. 1, was applied to each of the datasets in the experiment.

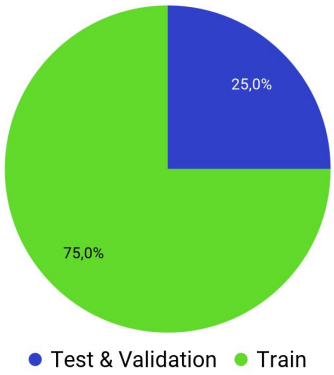

Fig. 1. Distribution of data in datasets

Classifier architecture

In our experiments, we rely on the neural network architecture presented in [11] and in Fig. 2.

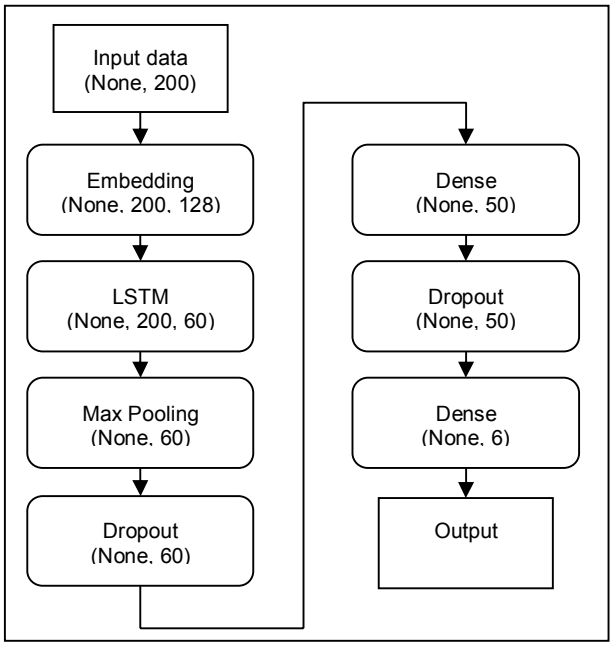

Fig. 2. Classifier architecture

As can be seen from the figure, our classifier contains seven layers. The Embedding layer is responsible for representing words in the coordinate space. LSTM preserves the connection between words in a sentence. Max Pooling helps to reduce the dimensionality of data for further processing. Subsequent block of duplicate Dropout-Dense layer combinations prevents network overfitting and helps with further dimensional reduction and, consequently, helps to speed up classifier training.

Consider the architecture in a more detailed view. The input data in the presented classifier, based on the neural network, is a list of English and Russian sentences (depending on the data set) of different lengths. As for training of a network we cannot use words in their usual state, we carry out over a data set the following manipulations:

1. Tokenization. For example: we transform every object that looks like "Good evening, gentlemen! - Good!" on an array of unique words ["good", "evening", "gentlemen"] without punctuation.
2. Indexing. We form from the received array of all unique words in a data set the dictionary which looks as follows: {1: "good", 2: "evening", 3: "gentlemen", ...}.
3. Representation of indices. For each of the objects in the data set, we form an array in which the words involved in the object are represented as indexes from our dictionary. For example: [1, 2, 3, 1].

The next problem we face is the problem of different sized objects among our data. And the training process of our network requires elements of the same dimension. This is solved by using the padding technique: set the maximum number of words, such as 200, and fill the remaining spaces in each object with

zeros. If the object contains more words than the selected maximum value – each subsequent word after the maximum is being cut off. Although in the data sets we have chosen, the size of most sentences does not exceed the value of 50 words, nevertheless, the dimension with a value of 200 is chosen to capture atypical cases, if any.

The next step is the Embedding layer, which projects words into a specific vector space using the well-known WordToVec algorithm [12]. This allows us to significantly reduce the size of the model. As a result of this layer work we receive the list of coordinates of words in vector space. In this case, we can use the distance between these coordinates to identify relevance and context.

The resulting tensor with coordinates is fed to the LSTM layer. We used recursive LSTM to preserve the coherence of words in the object.

After that, before transferring the original data to the next layer, we need to reformat the 3D tensor to 2D. To achieve this goal, we use the Global Max Pooling layer, which helps reduce the dimension of the tensor as follows: we review each sample of data and take the maximum values. This process helps to create a new set of data already reduced in size, which we will use.

After receiving the 2D tensor, we pass it to the Dropout layer, which randomly eliminates a certain percentage of nodes, the number of which we set, replacing them with a value of zero. This is necessary so that nodes at the next level are forced to process the representation of missing data. In this way we achieve the whole network has a better level of generalization – we avoid the overfitting effect, according to which the network can show good results on a familiar data set and far from the desired results on an unfamiliar set. In our experiments, we set the value of the drop rate as 10%. Of course, this value can be tuned empirically.

After the first Dropout layer, we transfer the data to the Dense layer and the output goes through the RELU activation function. After re-applying the Dropout-Dense layers, the result goes through the sigmoidal activation function, where classification is performed for each of the labels and we get a value between 0 and 1.

It should be noted that the model uses Adam optimizer to work with the loss function (loss). We selected "binary crossentropy", as we have just a binary classification between the values: this object belongs to a set to one of the categories or not.

Of course, the chosen architecture can be customized. However, research on this topic is beyond the scope of this article.

Research results

Initially, the classifier was trained in the "basic mode" – using the pre-processing methods described in the previous section, which are critical to its operation. Next, we conducted an iterative experiment consisting of the following steps:

1. In addition to the "basic mode" methods, we use another method of data processing or a combination of several methods.
2. We train the classifier on the selected data set without changing the parameters described in the previous section.
3. Test and validate the accuracy of the classification.

For the English-based data set [9], the accuracy in the basic mode reached 98.28% of successful classification cases. And the results of the application of additional used pre-processing algorithms and their combinations are shown in Table 3.

Table 3 – **Classifier accuracy results after applying text preprocessing methods on a data set [9]**

Method	% covering	Accuracy
to_lower (1)	~ 96.1	0.9825
remove_whitespaces (2)	~ 59.2	0.9829
remove_ip (3)	~ 6.3	0.9827
remove_username (4)	~ 0.3	0.9824
methods 1-4 in combination	~ 97.2	0.9821

For the Russian-based data set [10], the accuracy in the basic mode reached 86.12%, and the results of additional pre-processing methods and their combinations application are shown in Table 4.

Table 4 – **Classifier accuracy results after applying text preprocessing methods on a data set [10]**

Method	% covering	Accuracy
to_lower (1)	~ 91.4	0.8699
remove_whitespaces (2)	~ 93.3	0.8612
remove_ip (3)	~ 0.01	0.8729
remove_username (4)	~ 0.01	0.8638
methods 1-4 in combination	~ 98.9	0.8552

Consider more details about functionality of the methods used:

1) to_lower – lowercase all words;

2) remove_whitespaces – removal of extra spaces and service characters of carriage transfer;

3) remove_ip – delete all elements that fall under the next predicate described by a regular expression:

\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3},

with which you can find IP addresses;

4) remove_username – delete all elements described by the regular expression:

\[\[*\],

with which you can find nicknames.

The "% coverage" column in Tables 3 and 4 shows the percentage of the data set processed by this method. As you can see, given the content of a particular data set, it is extremely important to check this indicator. After all, in one case, the method affects almost the entire dataset, such as method 1 in table 3, and in another case – the method affects less than one hundredth of a percent of the data in the set, such as methods 3 and 4 in table 4.

According to the results described in tables 3 and 4, we note that often the increase of accuracy is relatively small. And sometimes there is even a deterioration of the result compared to the baseline result, as in the case of using a combination of methods, as described in table 4.

It is also obvious that the impact of the methods used on the English-language data set is much smaller than that for the Russian-language set. Indeed, in the first case, the deviation of the classification accuracy is maximum when using all four methods, and is only 0.07%.

As for the Russian-language set, again the maximum effect was shown by a combination of all methods, with differences of 0.4%, which is 5.7 times more than in English.

A possible explanation for this fact, in our opinion, is the relatively small size of the Cyrillic dataset, which led to less accurate classification, as well as a greater impact of pre-processing.

Another reason may be the presence of only one category of classification (in English there are six such categories). This can also significantly affect the accuracy of the classification.

However, in both cases, the use of this type of pre-processing leads to a slight decrease in the accuracy of the classifier. And although this reduction is quite small, but on huge amounts of data, for the processing of which are mostly used such classifiers, the effect can lead to significant negative consequences.

Conclusions

Data preprocessing is an extremely important step in preparing data for neural network training. At the same time, it is necessary to carefully choose the processing methods, pre-analyzing the data set and estimating what percentage of data from this set is affected by the method.

Based on the results of previous researchers, this paper investigates the impact of some methods of pre-processing of text data and their combination on the accuracy of the destructive messages classifier.

It has been shown that some methods have almost no effect on the accuracy of the work in the described experiment, such as removing extra spaces or IP addresses. However, caution should be exercised with the use of combinations of different preprocessing methods, which may have a small but significant effect on the accuracy of the classification on large data sets.

Similar conclusions were reached by the author of [8], who identified 15 appropriate methods of preprocessing, some of which and their combinations we studied.

Subsequent research should be aimed at finding or self-generating a relatively large Cyrillic data set (at least commensurate with the English), it is better if it is a Ukrainian-based dataset. In the case of self-formation it is necessary to design a crawler capable of collecting text data from certain resources. Further study of this data set and comparison with the results in English will allow us to draw deeper conclusions about the feasibility of different methods of pre-processing of data for automatic detection of toxic messages on the Internet.

It is also necessary to consider the use of additional methods and their combination with the selected ones and to take into account data sets with a significant occurrence of instances that fall under the action of a particular method.

References

1. (2020), *Social Network Ranking*, available at: https://www.statista.com/statistics/272014/global-social-networksranked-by-number-of-users/.
2. Dadvar. M., Trieschnigg. D., Ordelman. R. and de Jong, F. (2013), "Improving Cyberbullying Detection with User Context", Serdyukov P. et al. (eds), *Advances in Information Retrieval. ECIR 2013*, Lecture Notes in Computer Science, vol 7814. Springer, Berlin, Heidelberg.
3. Salminen, J., Almerekhi, H., Milenkovic, M., Jung, S., An, J., Kwak, H., & Jansen, B.J. (2017), "Anatomy of Online Hate: Developing a Taxonomy and Machine Learning Models for Identifying and Classifying Hate", *Online News Media. ICWSM*.
4. Shtovba, S. D., Shtovba, O. V., Yakhymovych, O. V. and Petrychko, M. V. (2019), "Vplyv syntaksychnykh zviazkiv u rechenniakh na yakist identyfikatsii toksychnykh komentariv v sotsialnii merezhi", *Informatsiini tekhnolohii ta kompiuterna tekhnika*, VNTU, Vinnytsia, No. 4, DOI: https://doi.org/10.31649/2307-5376-2019-4-35-42.
5. Pavlopoulos, J., Sorensen, J., Dixon, L., Thain, N., & Androutsopoulos, I. (2020), "Toxicity Detection: Does Context Really Matter?", *arXiv preprint,* arXiv: 2006.00998.
6. Noever, D. (2018), "Machine learning suites for online toxicity detection", *arXiv preprint,* arXiv:1810.01869.
7. van Aken, B., Risch, J., Krestel, R., & Löser, A. (2018), "Challenges for toxic comment classification: An in-depth error analysis", *arXiv preprint,* arXiv:1809.07572.
8. Mohammad, Fahim (2018), "Is preprocessing of text really worth your time for toxic comment classification?", *Proceedings on the International Conference on Artificial Intelligence (ICAI)*, The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), pp. 447-453.
9. (2020), *Toxic Comment Classification Challenge*, available at: https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge/data.
10. (2020), *Russian Language Toxic Comments. Small dataset with labeled comments from 2ch.hk and pikabu.ru*, available at: https://www.kaggle.com/blackmoon/russian-language-toxic-comments.
11. (2020), *Tackling Toxic Using Keras,* available at: https://www.kaggle.com/sbongo/for-beginners-tackling-toxic-using-keras.

12. (2020), *An Intuitive Understanding of Word Embeddings: From Count Vectors to Word2Vec* , available at: https://www.analyticsvidhya.com/blog/2017/06/word-embeddings-count-word2veec/.

Received (надійшла) 12.06.2020
Accepted for publication (прийнята до друку) 19.08.2020

ABOUT THE AUTHORS / ВІДОМОСТІ ПРО АВТОРІВ

Орловський Олександр Валер'янович – аспірант кафедри програмного забезпечення комп'ютерних систем, Чернівецький національний університет імені Юрія Федьковича, Чернівці, Україна;
Oleksandr Orlovskyi – graduate student Department of Software Engineering, Yuriy Fedkovych Chernivtsi National University, Chernivtsi, Ukraine;
e-mail: orlovskyi.alex@gmail.com; ORCID ID: https://orcid.org/0000-0003-4782-566X.

Остапов Сергій Едуардович – доктор фізико-математичних наук, професор кафедри програмного забезпечення комп'ютерних систем, Чернівецький національний університет імені Юрія Федьковича, Чернівці, Україна;
Sergey Ostapov – Doctor of physical and mathematical sciences, Professor Department of Software Engineering, Yuriy Fedkovych Chernivtsi National University, Chernivtsi, Ukraine;
e-mail: sergey.ostapov@gmail.com; ORCID ID: https://orcid.org/0000-0002-4139-4152.

Аналіз впливу методів попередньої обробки тексту на роботу класифікатора деструктивних повідомлень

О. В. Орловський, С. Е. Остапов

Анотація. Проблема. Соціальні мережі все частіше стають середовищем для погроз, образ, ненормативної лексики та інших деструктивних проявів людського спілкування. В онлайнових платформах сьогодні задіяна величезна кількість людей, а об'єм створеного контенту та реакцій на нього постійно б'є рекордні показники. Тому виникає потреба в автоматизації діяльності із детектування та протидії антисоціальним впливам. Одним із важливих напрямків такої діяльності є виявлення токсичних коментарів, що містять погрози, образи, ненормативну лексику, зневагу до оточуючих тощо. Для виконання такої задачі зазвичай будують класифікатор, заснований на нейронних мережах. А для їх навчання використовують зібраний власно або публічно доступний набір даних. В статті досліджується, як різні методи попередньої обробки вхідних даних впливають на кінцеву точність роботи класифікатора. Попередні дослідження в цьому напрямку підтвердили присутність впливу на результат, але не дозволили зробити остаточних висновків про ефективність. **Мета.** Дослідження впливу методів попередньої обробки текстових даних на результат роботи класифікатора деструктивних повідомлень. **Результати.** Продемонстровано, що вплив конкретного методу може досить сильно залежати від контенту в наборі даних. Крім цього, відзначено, що інколи вплив може бути незначним, а в деяких випадках може призводити навіть до погіршення результату. Також обґрунтовано необхідність попередньої перевірки набору даних на відсоток елементів, що підпадають під дію конкретного методу. **Оригінальність.** Проведено оцінку методів попередньої обробки даних на прикладі англомовного та російськомовного наборів даних. **Практична значущість.** Отримані результати дозволяють якісніше приймати рішення про використання тих чи інших методів попередньої обробки для підвищення точності прогнозів класифікатора деструктивних повідомлень.

Ключові слова: препроцесінг даних; виявлення деструктивних текстових даних; нейронні мережі; інтелектуальний аналіз даних; набір даних; обробка даних; класифікатор.

Анализ влияния методов предварительной обработки текста на работу классификатора деструктивних сообщений

О. В. Орловский, С. Э. Остапов

Аннотация. Проблема. Социальные сети все чаще становятся средой для угроз, оскорблений, ненормативной лексики и других деструктивных проявлений человеческого общения. В онлайновых платформах сегодня задействовано огромное количество людей, а объем созданного контента и реакций на него постоянно бьет рекордные показатели. Поэтому возникает потребность в автоматизации деятельности по детектированию и противодействию антисоциальным действиям. Одним из важных направлений такой деятельности является выявление токсичных комментариев, содержащих угрозы, оскорбления, ненормативную лексику, пренебрежение к окружающим и тому подобное. Для выполнения такой задачи обычно строят классификатор, основанный на нейронных сетях. А для их обучения используют собранный собственно или публично доступный набор данных. В статье исследуется как различные методы предварительной обработки входных данных влияют на конечную точность работы классификатора. Предыдущие исследования в этом направлении подтвердили присутствие влияния на результат, но не позволили сделать окончательные выводы об эффективности. **Цель**. Исследование влияния методов предварительной обработки текстовых данных на результат работы классификатора деструктивных сообщений. **Результаты**. Продемонстрировано, что влияние конкретного метода может достаточно сильно зависеть от контента в наборе данных. Кроме этого, отмечено, что иногда влияние может быть незначительным, а в некоторых случаях может приводить даже к ухудшению результата. Также обоснована необходимость предварительной проверки набора данных на процент элементов, подпадающих под действие конкретного метода. **Оригинальность**. Проведена оценка методов предварительной обработки данных на примере англоязычного и русскоязычного наборов данных. **Практическая значимость**. Полученные результаты позволяют качественно принимать решения об использовании тех или иных методов предварительной обработки для повышения точности прогнозов классификатора деструктивных сообщений.

Ключевые слова: препроцессинг данных; выявление деструктивных текстовых данных; нейронные сети; интеллектуальный анализ данных; набор данных; обработка данных; классификатор.