https://doi.org/10.31891/csit-2024-3-9
UDC [004.021:004.91]+004.415.53

Yakiv YUSYN, Nataliia RYBACHOK
National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute"

DICTIONARY-BASED DETERMINISTIC METHOD OF GENERATION OF TEXT CORPORA

This paper examines the problem of solving software engineering tasks in developing information systems for natural language processing. Generating corpora of text data is highlighted as a specific task of this problem. An analysis of the basic CorDeGen method was carried out, which is one of the corpus generation methods specially developed for this specific problem. This study shows that this method has a limited scope due to the use of "artificial" terms to fill the texts.

The paper proposes a new modified DBCorDeGen method that solves this shortcoming thanks to the use of an additional dictionary of terms that is supplied to the input of the method. The DBCorDeGen method preserves most of the characteristic features of the basic method, which are important for its use in solving software engineering tasks: determinism, speed of operation (including the possibility of combining with parallel modification), the possibility of a priori description of the structure and properties of the generated corpus. The only disadvantage compared to the basic method is the increase in the number of input parameters, however, compared to other methods of generating corpora presented in the literature, it is relatively small, and due to it, the scope of application of corpora generated by this method significantly increases.

As an experimental test of the proposed modified DBCorDeGen method, the task of sentiment analysis of the texts of the generated corpus is considered. The study shows that when using the basic CorDeGen method, it is impossible to obtain sentiment analysis results different from neutral polarity for all texts. When using the proposed method, it is possible to obtain different results using different dictionaries. Thus, it is confirmed that the proposed DBCorDeGen method has a larger scope than the basic method.

Keywords: natural language processing; text corpora; corpora generation; software quality assurance; sentiment analysis.

Яків ЮСИН, Наталія РИБАЧОК
Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського»

ДЕТЕРМІНОВАНИЙ МЕТОД ГЕНЕРУВАННЯ ТЕКСТОВИХ КОРПУСІВ НА ОСНОВІ СЛОВНИКА

У даній роботі розглядається проблематика вирішення задач інженерії програмного забезпечення при розробленні інформаційних систем оброблення природної мови. В якості конкретної задачі даної проблематики виділено задачу генерування корпусів текстових даних. Проведено аналіз базового методу CorDeGen – одного із методів генерування корпусів, спеціально розроблених для визначеної проблематики. У даному дослідженні показано, що цей метод має обмежену сферу застосування через використання «штучних» термів для наповнення текстів.

У роботі запропоновано новий модифікований метод DBCorDeGen, що вирішує даний недолік завдяки використанню додаткового словника термів, що подається на вхід методу. Метод DBCorDeGen зберігає більшість характерних ознак базового методу, що є важливими для його використання при вирішенні задач інженерії програмного забезпечення: детермінованість, швидкодію (включаючи можливість поєднання з паралельною модифікацією), можливість апріорного опису структури та властивостей генерованого корпусу. Єдиним погіршенням відносно базового методу є збільшення кількості вхідних параметрів, проте, у порівнянні з іншими методами генерування корпусів, що представленні у літературі, воно є відносно малим, а за його рахунок значно збільшується сфера застосування корпусів, що генеровані цим методом.

В якості експериментальної перевірки запропонованого модифікованого методу DBCorDeGen розглянуто задачу аналізу тональності текстів генерованого корпусу. У роботі показано, що при використанні базового методу CorDeGen неможливо отримати результати аналізу тональності, відмінні від нейтральної тональності для всіх текстів, а при використанні запропонованого методу можливо отримувати різні результати, використовуючи різні словники. Таким чином підтверджено те, що запропонований метод DBCorDeGen має більшу сферу застосування, ніж базовий метод.

Ключові слова: оброблення природної мови; корпуси текстів; генерування корпусів; забезпечення якості програмного забезпечення; аналіз тональності тексту.

Introduction

Information systems designed to address various natural language processing tasks have become ubiquitous in modern life, serving everyday needs and numerous professional fields. The unique features of natural language processing present challenges in solving different software engineering tasks when developing such systems. One major challenge is that these systems are usually designed to handle large text corpora and this complicates the development process. When, for example, it is necessary to test the developed information system (at the unit or integration level) or measure its performance, a large number of different (by structure, properties, or size) corpora of text data may be necessary. Only on-demand corpus generation can provide this because manual preparation of corpora is not efficient in terms of time and human effort.

As will be shown below, the field of corpus generation methods tailored specifically for software engineering tasks is currently underexplored despite the growing necessity for these methods. Thus, creating new methods and enhancing existing ones for generating text corpora to support software engineering tasks is an ongoing and pressing challenge.

State of the art

Even though today's literature presents a large number of studies devoted to the construction and generation of text data corpora, the vast majority of approaches and methods presented in these works cannot be applied (partially or completely) to solve software engineering problems.

For example, the methods of constructing corpora described in [1–4] require a large amount of natural text data based on which the corpus is generated. This requirement significantly limits the possibility of their use in developing information systems because the initial data must be stored somewhere additionally.

The methods presented in [5, 6] also require natural text data, but they can also work with data of a small volume, which simplifies their storage. However, these methods have a low speed, so their use will significantly slow down the process of solving software engineering problems.

Also, all these methods (presented in [1–6]) have the following in common:

- They are deterministic, therefore, to obtain different generated corpora, it is necessary to store a set of different initial natural data.
- Determining the structure and properties of the generated corpus can be difficult when using them.

Among the methods presented in the literature, specially designed for their use in solving software engineering problems, it is worth highlighting the CorDeGen method [7].

The basic CorDeGen method consists of the following steps [7]:

1. Input the parameter N_{terms} – the number of unique terms that should be contained in the texts of the generated corpus.
2. Calculate the parameter N_{docs} – the number of texts in the corpus – using the function $\mathrm{f}(x)$ represented by the formula (1):

$$\mathrm{f}(x): \left\lfloor \sqrt[4]{N_{terms}} \right\rfloor. \tag{1}$$

3. For each term index i from 0 to N_{terms} (exclusive):

a. Receive the string representation of the term corresponding to the index i by converting the index to the hexadecimal number system: $term_i = \mathrm{itoa}(i,16)$.

b. Calculation of the vector $\overline{tf_i}$, containing the number of occurrences of the term in texts, using the calculation of the function $\mathrm{g}(x)$ by the formula (2):

$$\mathrm{g}(x): tf_{ij} = \begin{cases} 0,\ j \notin (c_i - r \ldots c_i + r) \\ \dfrac{1}{2r+2} N_i,\ j \in (c_i - r \ldots c_i + r),\ j \neq c_i \\ \dfrac{2}{2r+2} N_i,\ j = c_i \end{cases},\ \text{where}$$
$$c_i = i \bmod N_{docs},$$
$$r = \left\lfloor \frac{N_{docs}}{5} \right\rfloor + 1,$$
$$N_i = N_{docs}(i \bmod N_{docs} + 1). \tag{2}$$

c. Record to each text the string representation of the term based on the calculated number of occurrences from the vector $\overline{tf_i}$.

In practice, the algorithm implementing the basic CorDeGen method will have asymptotic computational complexity equal to $O(N^{1.5})$.

The basic CorDeGen method is also deterministic like the methods presented in papers [1–6], but it accepts a minimum number of input parameters (only N_{terms}, which determines the size of the corpus), so this does not pose a problem when applying this method in solving software engineering problems.

The literature also presents modifications of the basic CorDeGen method, which are focused on its improvement and/or correction of shortcomings, namely: improvement of performance at large N_{terms} due to parallelization [8], avoidance of removing part of the terms from the generated texts during preprocessing [9].

However, despite the work already done to improve the basic CorDeGen method, it and its modifications presented in the literature still have certain shortcomings, which lead to the limitation of the applicability of these methods. First of all, such a shortcoming is the use of "artificial" terms to fill the texts, namely the hexadecimal representation of their indices. In this case, only certain terms can match natural language terms that contain only letters a-f. This greatly limits the possibilities of using generated corpora because natural language processing methods that use term semantics analysis or use natural language term dictionaries will not return practically meaningful results for such corpora generated by the basic CorDeGen method. An example of such methods of natural language processing can be methods of sentiment analysis, the vast majority of which are based on machine learning methods or dictionaries. As of the time of this study, no corpora generated by the basic CorDeGen method or its known modifications can be used with such methods, which is the goal of this study to correct.

Purpose of the paper

The paper aims to eliminate the identified shortcoming of the basic CorDeGen method related to the lack of semantics of the generated terms. This issue can be addressed by developing a modification of the method that will generate semantically filled terms.

Dictionary-based CorDeGen (DBCorDeGen)

The extreme limitation of the input data of the method only to the required size of the corpus causes the above-described drawback of the basic CorDeGen method. In this case, the only way to obtain a string representation of the next term is to perform certain manipulations with its index – the basic method uses the hexadecimal representation for this. Thus, the identified shortcoming of the basic CorDeGen method can only be solved by extending the input data that the method accepts.

The new modified DBCorDeGen method proposed in this study uses this idea and expands the set of input parameters with an additional dictionary of terms $dict$, which will be used when determining the string representation of the next term by its index. This dictionary can be task-specific, taking different forms and structures depending on the natural language processing task with which the generated corpus will be used. The only mandatory requirement that the proposed method imposes on this dictionary is that its size must be at least as large as the given value of N_{terms}. Otherwise, it may lead to an error when generating the corpus or to the use of the same term several times for different indexes, depending on the specific implementation of the method.

Thus, the proposed DBCorDeGen method consists of the following steps:

1. Input the following parameters:

a. N_{terms} – the number of unique terms that should be contained in the texts of the generated corpus.

b. $dict$ – the terms dictionary, $\text{size}(dict) \geq N_{terms}$.

2. Calculate the parameter N_{docs} – the number of texts in the corpus – using the function $\text{f}(x)$ represented by the formula (3):

$$\text{f}(x) : \left\lfloor \sqrt[4]{N_{terms}} \right\rfloor. \tag{3}$$

3. For each term index i from 0 to N_{terms} (exclusive):

a. Receive the string representation of the term corresponding to the index i from the dictionary $dict$: $term_i = \text{dict}(i)$.

b. Calculation of the vector $\overline{tf_i}$, containing the number of occurrences of the term in texts, using the calculation of the function $\text{g}(x)$ by the formula (4):

$$\text{g}(x) : tf_{ij} = \begin{cases} 0,\ j \notin (c_i - r \ldots c_i + r) \\ \dfrac{1}{2r+2} N_i,\ j \in (c_i - r \ldots c_i + r),\ j \neq c_i \\ \dfrac{2}{2r+2} N_i,\ j = c_i \end{cases}, \text{where}$$

$$c_i = i \bmod N_{docs},$$

$$r = \left\lfloor \frac{N_{docs}}{5} \right\rfloor + 1,$$

$$N_i = N_{docs}(i \bmod N_{docs} + 1). \tag{4}$$

c. Record to each text the string representation of the term based on the calculated number of occurrences from the vector $\overline{tf_i}$.

The algorithm implementing the proposed DBCorDeGen method will have the same asymptotic computational complexity as for the basic CorDeGen method because it is possible to get terms from the dictionary with constant time.

The modified DBCorDeGen method, in addition to increasing the scope of application of generated corpora, preserves most of the other advantages of the basic CorDeGen method in solving software engineering problems for information systems for natural language processing: determinism, relatively low computational complexity, the possibility of a priori description of the structure and properties of the generated corpus. In addition, the proposed DBCorDeGen method can also be combined with a parallel modification of the basic method [8], thus obtaining even better performance in practice.

The main disadvantage of the DBCorDeGen method, in comparison with the basic method, is the increase in the number of input parameters, due to which the main advantage of this method is achieved – the increase in the scope of application. However, if we compare the possible volume of the dictionary with the required volume of input data of other methods [1–6] presented in the literature review, it is much smaller. Also, the proposed method may retain other shortcomings of the basic CorDeGen method, which have not yet been discovered and/or not presented in the literature. As for the drawback described in [9], the DBCorDeGen method can also effectively avoid it by excluding stop words from the used dictionaries.

Experiment

Experiment hypothesis

As a practical example of how the proposed DBCorDeGen method can be used in fields inaccessible to the basic CorDeGen method, let's consider the above-mentioned example of sentiment analysis of the texts of the generated corpus.

As a final assessment of the polarity of the entire text, we will take the polarity that occurs most often among the sentences that make it up.

In the case of the basic CorDeGen method, it is possible to hypothesize that since most of the terms generated by this method do not occur in natural languages, all generated texts will have a neutral polarity. Individual constituent sentences may have a non-neutral polarity (negative or positive) due to some random fluctuations in polarity scoring methods, but such sentences will not be sufficient to change the final score based on the most frequent polarity.

In the case of the proposed DBCorDeGen method, polarity estimates of the texts of the generated corpus will depend on the method of construction of the used dictionary. As part of this experiment, it is proposed to use a dictionary constructed as follows: if the index of a term in the dictionary (taken with the remainder from division by N_{docs}) is less than $N_{docs}/2$, then the term must have a negative polarity, otherwise the term must have a positive polarity. For the texts of the corpus generated using such a dictionary, it is possible to hypothesize that the first $N_{docs}/2$ texts will have a final evaluation in the form of negative polarity, and the second $N_{docs}/2$ texts will have a final evaluation in the form of positive polarity. This is due to the peculiarities of the distribution of terms across texts, namely, the existence of a "central" text for each term.

Experiment results

To conduct the experiment, the software implementation of the basic CorDeGen method, all its modifications known at the time of writing the paper, and the proposed DBCorDeGen modification were used. The software implementation was created using the 8th version of the .NET platform [10] in the C# 12 programming language [11]. The developed software implementation is tested using unit tests written with the xUnit library [12, 13]. Also, the repository of the software implementation contains benchmarks of all implemented methods written with the BenchmarkDotNet library [14, 15].

The .NET port of the Stanford CoreNLP library, developed by the authors of the paper, was used to perform the sentiment analysis of the text [16, 17]. This library supports five polarities for sentences: very negative, negative, neutral, positive, and very positive.

Fig. 1 shows the results of the sentiment analysis of the corpus of texts generated using the basic CorDeGen method with the parameter N_{terms} equal to 256 (correspondingly, $N_{docs} = 4$).

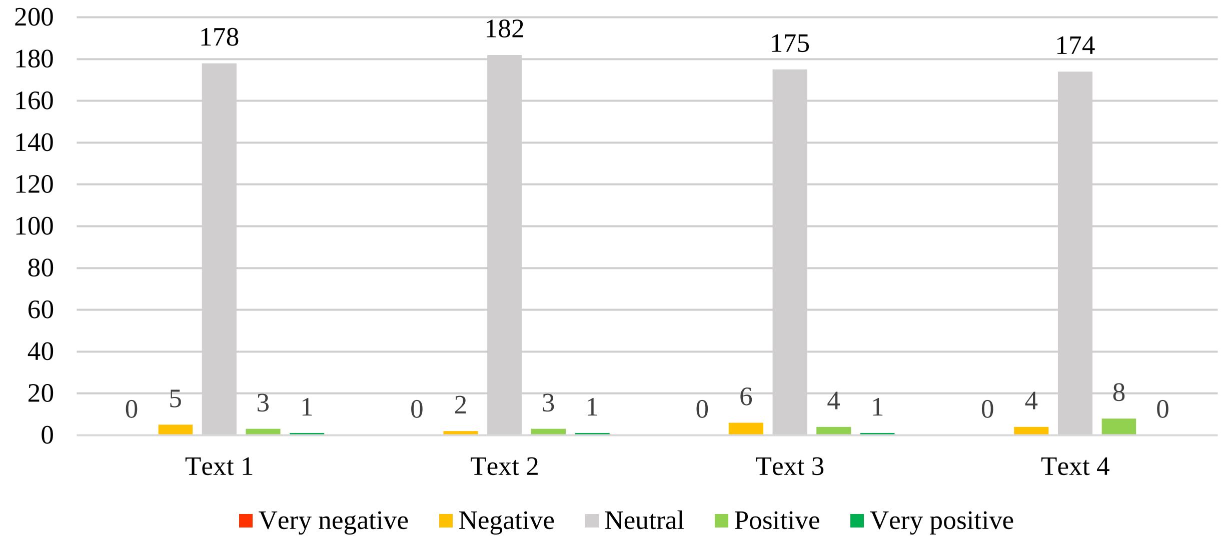

Fig. 1. The result of the sentiment analysis of the corpus of texts generated by the basic CorDeGen method

As can be seen from Fig. 1, the first hypothesis of the proposed experiment turned out to be true – for each text generated by the basic method, the final assessment of polarity is neutral because sentences with this polarity significantly prevail in all generated texts. Thus, the basic CorDeGen method cannot be applied together with the tasks that solve the problems of software engineering for the methods of sentiment analysis because the obtained results will not have practical meaning.

For the software implementation of the proposed modified DBCorDeGen method, a dictionary constructed according to the principle described in the framework of the second hypothesis was used. Given that the Stanford CoreNLP library recognizes all text as written in English by default, the term “awful” is used for terms that should have a negative polarity, and the term “amazing” for terms that should have a positive polarity. Fig. 2 shows the results of sentiment analysis of texts generated by the proposed DBCorDeGen method with such a dictionary for the same parameter N_{terms}.

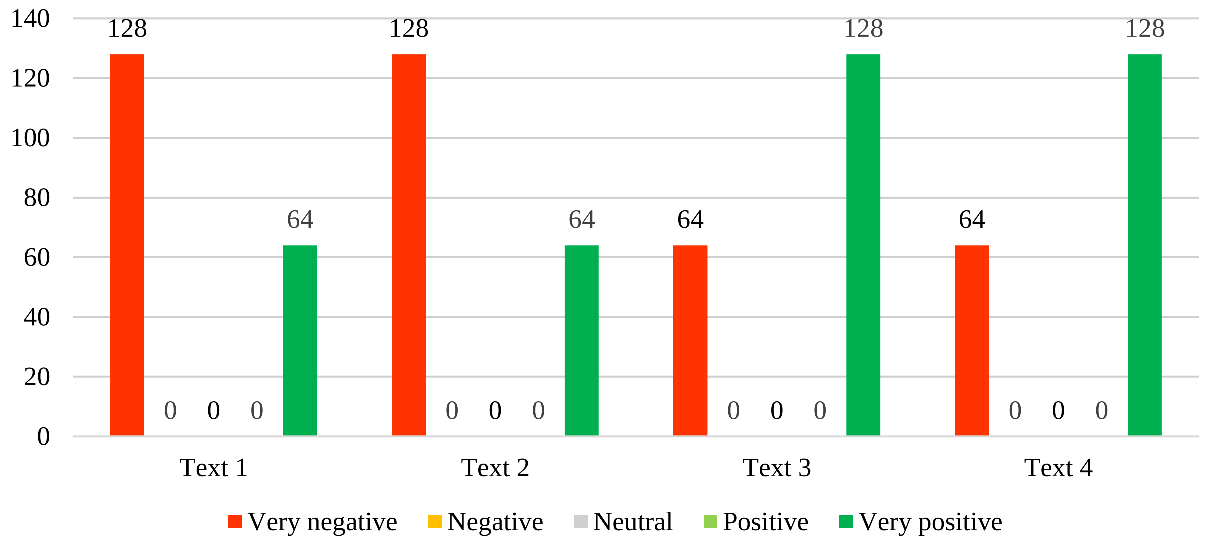

Fig. 2. The result of the sentiment analysis of the corpus of texts generated by the proposed DBCorDeGen method

As can be seen from Fig. 2, the second hypothesis of the proposed experiment also turned out to be true – the first two generated texts have a “very negative” rating as the final assessment of polarity, and the second two generated texts have a “very positive” rating. Thus, using the DBCorDeGen method, it is possible to obtain different sentiment analysis results for different corpora generated using dictionaries constructed according to different principles. This is qualitatively different from the always identical result that can be obtained using the basic CorDeGen method.

Conclusions

This study shows the need for further development of new methods of generating corpora of text data, including modification of existing methods to correct their existing shortcomings and expand the limits of applicability. All this concerns the application of these methods in solving the problems of developing software for natural language processing.

The performed analysis of the basic CorDeGen method, which is presented in the literature as specially developed for use during solving software engineering problems, revealed the presence of shortcomings and limitations in its applicability, despite its existing modifications that are quite widely presented in the literature. This study presents a new modification of the CorDeGen method, which extends the limits of applicability to natural language processing methods based on semantic analysis or dictionaries of natural language terms.

The given example of the practical use of the proposed modified method confirmed the elimination of the identified shortcoming and the expansion of the limits of applicability of the CorDeGen method of text data corpus generation. The conducted experiment showed the possibility of building dictionaries of terms for the DBCorDeGen method, which have different properties, as a result of which corpora with different predicted properties will be generated. Such practical results can be used, for example, when solving the problem of software quality assurance, namely testing. The proposed DBCorDeGen method can be applied together with a property-based testing methodology, where the predicted properties of corpora generated by different dictionaries will be used to validate the developed natural language processing software.

Further research can be focused in different directions:

- Important for academia: research on the properties of corpora generated using different dictionaries in the context of different natural language processing tasks; combining the studied properties with different methods of solving software engineering problems, such as quality assurance or benchmarking.
- Important for industry: software implementation of the proposed method in various programming languages.

References

1. Tanaka K., Chu C., Kajiwara T., "Corpus Construction for Historical Newspapers: A Case Study on Public Meeting Corpus Construction Using OCR Error Correction," SN Computer Science Vol. 3, (2022). https://doi.org/10.1007/s42979-022-01393-6.
2. Qumar S. M. U., Azim M., Quadri S. M. K., "Addressing the data gap: building a parallel corpus for Kashmiri language," International Journal of Information Technology (2024). https://doi.org/10.1007/s41870-024-01979-8.
3. Meden K., Erjavec T., Pančur A., "Slovenian parliamentary corpus siParl," Language Resources and Evaluation, (2024). https://doi.org/10.1007/s10579-024-09746-8.
4. Aichaoui S. B., Hiri N., Dahou A. H., "Automatic Building of a Large Arabic Spelling Error Corpus," SN Computer Science Vol. 4, (2023). https://doi.org/10.1007/s42979-022-01499-x.
5. Boujelbane R., Ellouze Khemekhem M. Belguith L., "Mapping Rules for Building a Tunisian Dialect Lexicon and Generating Corpora," in Proceedings of the Sixth International Joint Conference on Natural Language Processing, (2013). https://aclanthology.org/I13-1048/.
6. Nazura J., Muralidhara B. L., "Automating Corpora Generation with Semantic Cleaning and Tagging of Tweets for Multi-dimensional Social Media Analytics," International Journal of Computer Applications Vol. 127, pp. 11–16, (2015). https://doi.org/10.5120/ijca2015906548.
7. Yusyn Y., Zabolotnia T., "Text data corpora generation on the basis of the deterministic method", KPI Science News, no. 3, pp. 38–45, (2021). http://scinews.kpi.ua/article/view/240780. [in Ukrainian].
8. Yusyn Y., Zabolotnia T., "Accelerating the process of text data corpora generation by the deterministic method," Eastern-European Journal of Enterprise Technologies Vol. 1, no. 2, pp. 26-34, (2024). https://doi.org/10.15587/1729-4061.2024.298670.
9. Yusyn Y., Rybachok N., "Improvement of the deterministic method of the text data corpora generation," Herald of Khmelnytskyi National University. Technical sciences Vol. 333, p. 437–445, (2024). https://doi.org/10.31891/2307-5732-2024-333-2-69.
10. Microsoft, "What's new in .NET 8," (2023). [Online]. Available: https://learn.microsoft.com/en-us/dotnet/core/whats-new/dotnet-8/overview.
11. Microsoft, "What's new in C# 12," (2023). [Online]. Available: https://learn.microsoft.com/en-us/dotnet/csharp/whats-new/csharp-12.
12. .NET Foundation and contributors, "Home > xUnit.net," (2019). [Online]. Available: https://xunit.net/.
13. .NET Foundation and contributors, "xunit/xunit at 2.6.2," (2023). [Online]. Available: https://github.com/xunit/xunit/tree/2.6.2.
14. .NET Foundation and contributors, "Overview | BenchmarkDotNet," (2018). [Online]. Available: https://benchmarkdotnet.org/articles/overview.html.
15. .NET Foundation and contributors, "dotnet/BenchmarkDotNet at v0.13.10," (2023). [Online]. Available: https://github.com/dotnet/BenchmarkDotNet/tree/v0.13.10.
16. Manning C. D., Surdeanu M., Bauer J., Finkel J., Bethard S. J., McClosky D., "The Stanford CoreNLP Natural Language Processing Toolkit," in Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations, (2014).
17. Socher R., Perelygin A., Wu J. Y., Chuang J., Manning C. D., Ng A. Y., Potts C., "Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank," in Proceedings of Conference on Empirical Methods in Natural Language Processing, (2013).

Yakiv Yusyn **Яків Юсин**	PhD, Assistant of Computer Systems Software Department, National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", Kyiv, Ukraine, e-mail: yusyn@pzks.fpm.kpi.ua. orcid.org/0000-0001-6971-3808	доктор філософії, асистент кафедри програмного забезпечення комп'ютерних систем, Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна.

Nataliia Rybachok **Наталія Рибачок**	PhD, Associate Professor of Computer Systems Software Department, National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", Kyiv, Ukraine, e-mail: rybachok@pzks.fpm.kpi.ua. orcid.org/0000-0002-8133-1148	кандидат технічних наук, доцент кафедри програмного забезпечення комп’ютерних систем, Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна.