UDC 004.855.5(045)
DOI:10.18372/1990-5548.76.17663

[1]V. M. Sineglazov,
[2]I. M. Savenko

COMPARATIVE ANALYSIS OF TEXT VECTORIZATION METHODS

[1]Aviation Computer-Integrated Complexes Department, Faculty of Air Navigation Electronics and Telecommunications, National Aviation University, Kyiv, Ukraine
[2]Department of Artificial Intelligence, Institute for Applied System Analysis, National Technical University of Ukraine "Sikorsky Kyiv Polytechnic Institute," Kyiv, Ukraine
E-mails: [1]svm@nau.edu.ua ORCID 0000-0002-3297-9060, [2]savenko.ilya@lll.kpi.ua

***Abstract*—*The paper considers methods of vectorization of textual properties of natural language in the context of the task of intellectual text analysis. The most common methods of statistical analysis of feature extraction and methods that taking into account the context are analyzed. The work describes the above types of text embeddings and their most common variations and implementations. Their comparative analysis was performed, which showed the relationship between the type of task of intellectual text analysis and the method showing the best metrics. The topology of the neural network, which is the basis for solving the problem and obtaining metrics, is described, and implemented. The comparative analysis was carried out using the relative time analysis of the theory of algorithms and classification metrics: accuracy, f1-score, precision, recall. The classification metrics are taken from the results of building a neural network model using the described framing methods. As a result, in the task of analyzing the tonality of the text, the statistical method of framing based on n-grams of character sequences turned out to be the best.*

Index Terms—Intellectual text analysis; natural language processing; text embeddings; opinion mining; machine learning; Word2Vec; TF-IDF; statistical embeddings; context-based embeddings.

I. INTRODUCTION

The field of neural networks nowadays makes it possible to build and solve many of today's pressing problems. This scientific field has become a leader in the last decade and shapes the development in all related areas of analytics. This mechanism allows building the architecture of neural networks that are universal "accumulators of experience" using already existing methods.

More than most of the sphere of human activity is based on writing. And since the development of a person is interconnected with society, writing and communication play a key role in the development of a person. Connecting this area with the one previously described, questions arise about the possibility of constructing tasks for the intellectual analysis of natural language texts – the language apparatus commonly used by humans.

The essence of the methods of intellectual analysis of texts consists in the extraction of properties that help in building a model of their analysis (construction of classification problems, regression, generative problems, etc.). As an example, you can take an ordinary forward propagation neural network with a linear function. To build intellectual analysis tasks on its basis, there is a need for a certain representation of the natural text.

In previous works [1] already existing approaches to solving the problem of text vectorization were described and considered. Tomas Mykolov proposed and implemented methods considering the context in the corpus of documents [2], [3]. Subsequently, several modifications appeared that improved the performance of the vectorized representation of documents, such as GloVe [4], [5].

When considering this research in the subject area of opinion mining analysis, it is possible to cite already existing works [6], [7], where a combination of statistical methods of framing and methods of considering the context was carried out, in particular tf-idf and word2vec.

This paper will present statistical and contextual methods of natural language representation, for the further construction of solutions to the problems of intellectual analysis of texts. It is worth investigating their positive and negative sides, as well as analyzing the contexts of tasks where they can be used in the most effective way.

II. PROBLEM STATEMENT

We present a mathematical model of machine learning for natural language processing. Let we have the problem of machine learning of natural

language, the model of which is described by the function $f(x)$:

$$f(x) = x \cdot W + b,$$

where $x \in R^n$ is a document representation vector built based on the extracted characteristics of the document, in a previously specified format. We have d – a document from which the characteristics will be extracted. Let us use the function to extract characteristics:

$$x = \text{vec}(d).$$

It is necessary to find a way to present textual characteristics – choose a method of forming the vector x for its further use in the model:

$$J = \sum_i \text{Loss}\left(y^{(i)}, \hat{y}^{(i)}\right) \to \min.$$

It is worth conducting research and comparative analysis and extracting the connection between the structural characteristic of the text together with the specificity of the task of intellectual analysis of natural language and the method that will build a vectorized representation in the best possible way.

For the comparative analysis, two aspects of the algorithm analysis were considered: its implementation complexity and the aspect of integration in the subject area.

As a calculation of complexity, the standard theory of algorithms will be used and the total (average) number of operations and the relative execution time that will occur during its operation will be calculated. The theory of numerical methods for calculating operations of more complex functions will also be considered.

The second aspect considers the integration of the algorithm in the generalized context of the subject area where it will be used. In our case, this is the problem of opinion mining in text from social networks was chosen (in the case of consideration of this article, posts from the Telegram social network will be selected). The context of the posts consists in the description of news events taking place in the period of 2022–2023 in Ukraine. The following metrics are used to get conclusions.

To check the effectiveness of the models, a confusion matrix was built, and the classification quality analysis metrics were used: *precision, recall, f_1-score*.

Accuracy: Accuracy defines the percentage ratio of the total number of correct instances found to the total number of instances found. In this paper, the ratio of the number of correctly recognized views to the total number of set views considered to determine the accuracy.

$$\text{Acc} = \frac{TP + TN}{TP + FP + TN + FN}.$$

Before describing other performance parameters, the terms "true positives", "false positives", "true negatives" and "false negatives" need to be defined according to the work performed:

True Positive (TP) is a correct recognition of a user's view, i.e. the view that the user actually posed turns out to be the same view.

False Positive (FP) is a false prediction of a view, i.e. one that was not prevalent over the post but was recognized by the model as correct.

True Negative (TN) defines the case of correctly recognizing the falseness of the view, that is, a post for which the reaction (view) is different.

False Negative (FN) is a false recognition of reaction that is classified into a different category.

Precision is a performance measure that is calculated based on true and false prediction results, resulting in correct recognition of views prevailing over the post. The higher the accuracy, the higher the level of gaze recognition. On the other hand, lower accuracy values reflect the presence of more false positives.

$$\text{Precision} = \frac{TP}{TP + FP}.$$

Recall is the result of true and false negatives, which determines the correct recognition of a view for which the other primary views prevail. A high recall value reflects a high level of recognition of the prevailing view for which the other is correct, while a low recall value indicates the presence of more false negatives, that is, views that are prevalent but are incorrectly recognized.

$$\text{Recall} = \frac{TP}{TP + FN}.$$

F-score: If both false positives and false negatives are equally serious, a combination of precision and recall measures called *F*-score is used and is calculated as:

$$F_{\text{score}} = \frac{2 \cdot \text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}.$$

The main goal of the research is to find such a method for which characteristics, for which the accuracy, precision, recall scores were as high as possible.

III. METHODS OVERVIEW

A. One-hot vectorization

The easiest way to convert text to vector representation. With this coding method, each text element is represented as a vector consisting only of the numbers 0 and 1. There is a 0 in each position of the vector, except for the position corresponding to the text representation element. Passing the sliding window along the text, we will get a set of vectors whose elements are 0 and 1. An example of such a representation is shown in Fig. 1.

The world caught me and did not catch

The world	100000
caught	010000
me	001000
and	000100
did not	000010
catch me	000001

Fig. 1. One-hot vector text representation

This method makes it possible to uniquely identify a text element by its vector representation and vice versa. The disadvantage of using this method is its processing and presentation in a computer. Considering that the dictionary of the English language can be composed of up to a million words, when constructing a matrix of cross-meeting of pairs of words, we will get a huge array of data that is difficult to process and store.

B. Bag-of-Words

Describing the problem of data representation as a feature of direct representation of textual elements, it is also worth noting the structure of non-fixedness of the representation for the further construction of the model. The vectorization model in the form of a bag of words solves this problem.

Forming the input vector for the document, we aggregate element-by-element representations and average them. The input vector **x** in our speech classification example contains the normalized number of bigrams in document *D*. This vector can be decomposed into the average of **D** vectors, each of which corresponds to a specific position in document *i*:

$$x = \frac{1}{|D|}\sum_{i=1}^{|D|} x^{D_{[i]}}.$$

In this case, D_i is the bigram in the document at position *i*, and $x^{D_{[i]}}$ is a one-hot vector in which all positions are zero except the position corresponding to the bigram. The value in this position is 1.

In this way, we get a vector representation of a document of a fixed dictionary size, which will be used later. This allows you to reduce the load on computing power and be widely used. Among the disadvantages, this presentation does not take into account the order of text elements.

C. TF-IDF

The problem of using BOW is the uneven distribution of text elements in the representation vectors, that is, rare elements or common words will not be distinguished. To solve this problem, the TF-IDF vectorization method was implemented, consisting of two parts: Term-Frequency (TF) and Inverse Document Frequency (IDF). The calculation of such values is as follows:

$$TF \sim IDF_{td} = TF_{td} \cdot IDF_{tc}.$$

Term Frequency – calculation of the frequency of occurrence of a text element in the document. For a specified text element, this metric can be defined as the ratio of the number of times the element appears in the document to the total number of words in the document.

$$TF_{td} = \frac{N(t)}{N_d},$$

where $N(t)$ is the quantity of *t* occurrences in document *d*. N_d is the overall quantity of words on document *d*.

Inverse Document Frequency (IDF) is a metric that determines the importance of a word in a corpus of multiple documents. It is calculated as a logarithmic ratio of the number of all documents to the number of documents with the specified word. Formalizing, we have a formula definition:

$$IDF_{tc} = \frac{N_c}{N_c(t)},$$

where N_c is the number of documents in corpus *c*, $N_c(t)$ is the number of documents that contains the word *t* in corpus *c*.

We highlight the following features of this vectorization method: each column represents a separate unique text element, each cell contains a weight value indicating how important the text element is for the corpus or document, occurrences in the entire corpus and not the document are considered. The metric provides an opportunity to distribute the values of the text elements of the

model more evenly, increasing the weight values of rare elements and weakening more frequent elements.

D. Word2Vec

The widely popular Word2Vec algorithm was developed by Tomasz Mikolov [2] and his colleagues in a series of papers [2], [3]. Similar to Collober and Weston's algorithm, Word2Vec also starts with a natural language processing neural network model and modifies it to produce results with faster learning processes. Word2Vec is not a single algorithm, it is a software package that implements two different context representations (CBOW and Skip-Gram) and two different improvements (Negative-Sampling and Hierarchical Softmax).

Putting Colbert and Weston's algorithm in line, the learning process based on the negative sampling of NS in the Word2Vec algorithm includes in the learning process the discrimination of word elements that fit the context and those that do not. Word2Vec implements a probabilistic approach. Let us have D context pairs and $\bar{D}$ mismatched pairs. The essence of the method is to make an approximate estimate of the probability on the estimation function $s(w,c)$:

$$P(D=1|w,c)=\frac{1}{1+e^{-s(w,c)}}.$$

The essence of the algorithm is to maximize the logarithm of the likelihood of DD data:

$$L(\Theta,D,\bar{D})=\sum_{(w,c)\in D}\log P(D=1|w,c)+\sum_{(w,c)\in\bar{D}}\log P(D=0,w,c).$$

Words in the context create a corpus D. A negative selection approach is used to form $\bar{D}$. In this way, a machine learning model is built that predicts words according to the context in which they are used (CBOW) and vice versa (Continuous Skip-Grams).

E. CBOW

The CBOW architecture was proposed by a group of researchers from Google in 2013 together with the Skip-gram model. The logic of the CBOW architecture is very simple: predict a word depending on the context where the word is. An example of the neural network architecture in the simplest version of the CBOW implementation using only one word (context) before the target word is shown in Fig. 2.

At the input of the neural network, the context is given, at the output, the model determines a quantitative value that predicts how well the given word fits the context. After forming the dictionary, it is necessary to choose the input word and its context. We will consider the context to be a sliding window that can pass through the document, thus choosing a group of text elements – words – at each step.

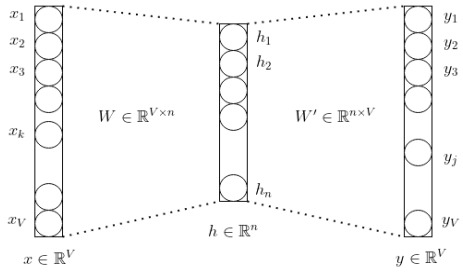

Fig. 2. CBOW for one word in context

The number of context words will depend on the number of sentences and the size of the sliding window (Fig. 3).

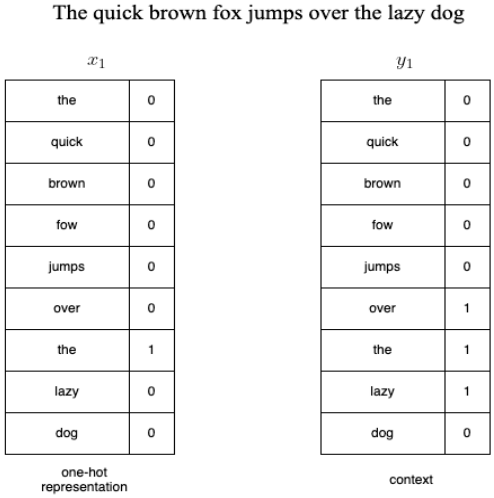

Fig. 3. Context training definitions

Subsequently, a neural network is built and the process of its training is carried out, where the already defined context is the vector of inputs to the network, and the word from the context is the value on which the above model is trained – y_{true}.

The result of multiplying the sum of one-hot representations of the context of words by the matrix of frames gives a vector representation of the word, which is subsequently used in the output layer, the result of which is obtained by the product of this vector by the neuron's weights. The activation function of the hidden layer is linear, and at the output layer it is a generalized logistic activation function:

$$f(u)_i=\frac{e^{u_i}}{\sum_{k=1}^{K}e^{u_k}}.$$

The values of the hidden layer will be calculated as follows:

$$h = W^T x.$$

After passing the hidden layer, the values of the original layer will be calculated in the same way, but with a different weight matrix:

$$y = W'^T h.$$

It is worth paying attention that the vectorization of the dictionary in the initial layer in the above example is a BOW. But the functions of any statistical vectorization of the text, such as TF-IDF, the calculation mechanism of which was described above, are also admissible.

At the beginning of the process, the weights are initialized using a standard distribution. The selection of the loss function for the evaluation of the output layer of the neural network is based on obtaining a result in the range from 0 to 1. The logistic activation function can be interpreted as the conditional probability of the appearance of a word in each context. The loss function will look like this:

$$L = -\ln P(w_t \mid w_c) = -\ln\left[f(u_j) \right] = -\ln\left(\frac{e^{u_j}}{\sum_i e^{u_i}} \right).$$

To solve the given problem, it is necessary to minimize the loss function and maximize the probability of the word appearing in its context.

F. Skip-Gram

This mechanism has the same structure as CBOW, but with one difference, that one word is fed to the input of the neural network, and the output must be an estimate of the context around this word.

The inner layer remains the same as in the case of CBOW. The activation function of the inner layer is linear.

The output layer of the constructed neural network has the size of the input layer – the size of the dictionary. It has an activation function (softmax). Each source element is determined by the quantitative characteristic of the corresponding word belonging to the context being evaluated. The training process is identical to CBOW.

IV. Comparative Analysis

As an assessment of the methods themselves in the context of the subject area, a solution to the problem of text classification was taken. The problem to be solved was the opinion mining problem of news text posts. During the study, the structure of a neural network was built, which was subsequently trained on labeled data. The general topology of the neural network is as follows (Fig. 4).

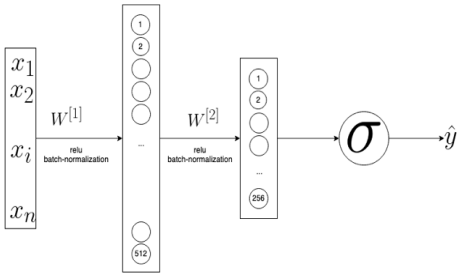

Fig. 4. Feed-forward NN

A neural network with two hidden layers will be used as the analysis. At the entrance to the network, already vectorized data is provided in a statistical or context-sensitive way. After the learning process, the test sample is evaluated and the results are obtained. The following methods were analyzed: BOW, TF-IDF for n-grams of groups of words from 1 to 4, TF-IDF for groups of characters from size 3 to 5, word2vec with a frame of size 1000, and TF-IDF with word combinations from 2 to 5 words.

TABLE I. ACCURACY SCORE FOR EACH TEXT EMBEDDING METHOD

Method	**Test accuracy**	**Train accuracy**
BOW	0.7577	0.9666
TF-IDF *n*-gram words (1.4)	0.7408	0.9767
TF-IDF *n*-gram chars (3–5)	**0.7551**	**0.9839**
Word2Vec 100	0.7557	0.7494
TF-IDF *n*-gram words (2.5)	0.7530	0.7537

The best score turned out to be tf-idf for *n*-gram characters. 98% accuracy was achieved on the training samples, 75% on the test samples. One point to focus is that a similar result was achieved using BOW embeddings – 96% on the training set and 75% in the test set.

V. Conclusions

Using Word2Vec as a framework to complement TF-IDF in natural language processing (NLP) tasks offers several advantages.

- Extracting semantic meaning: Word2Vec excels at capturing the semantic meaning of words by learning distributed word embeddings. It can capture contextual relationships and similarities between words. On the other hand, TF-IDF

represents the importance of words based on their frequency in a document collection. TF-IDF does not explicitly capture semantic relationships.

- Handling non-vocabulary words: Word2Vec can handle out-of-vocabulary (OOV) words by providing vector representations based on their context. In contrast, TF-IDF struggles with OOV words since they are not present in the training data. Word2Vec embeddings can still provide meaningful representations for OOV words, which can be advantageous in some scenarios.
- Generalization improvements: Word2Vec embeddings capture contextual relationships, which can enhance the generalization of models. This can be particularly useful in tasks such as text classification, where understanding the underlying semantics is important. TF-IDF, on the other hand, relies solely on word importance, which may not capture nuanced semantic information as effectively.
- Reducing high-dimensionality: TF-IDF vectors are high-dimensional, especially when working with large dictionaries. On the other hand, Word2Vec frames are usually smaller.

It is important to note that the performance comparison between TF-IDF and Word2Vec depends on the specific task and dataset. There might be scenarios where TF-IDF performs better, especially in tasks where word importance or interpretability are crucial. In other cases, Word2Vec might outperform TF-IDF by capturing semantic relationships and improving generalization. Evaluating their performance on your specific task and dataset is essential for determining which approach is more suitable.

In this work, a comparative analysis of existing text vectorization methods was carried out. The key emphasis was given to the group of Word2Vec algorithms, as the best illustration of the implementation of algorithms for the extraction of text features in context, which are sufficiently optimized for applied use in the field of natural language analysis. Statistical methods of text vectorization such as TF-IDF, BOW, one-hot coding were also considered. Subsequently, a quantitative analysis of the algorithms was carried out, TF-IDF for word groups in the task of text tonality analysis turned out to be the best solution in the problem of opinion mining on a small news posts.

REFERENCES

[1] Christopher Manning, Prabhakar Raghavan, and Hinrich Schütze, "Introduction to Information Retrieval," *Cambridge University Press*, 2008. https://doi.org/10.1017/CBO9780511809071.

[2] Tomáš Mikolov, *Statistical language models based on neural networks*, Ph.D. thesis, Brno University of Technology, 2012.

[3] Tomáš Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean, *Efficient estimation of word representations in vector space*. arXiv:1301.3781 [cs], January 2013.

[4] Jeffrey Pennington, Richard Socher, and Christopher Manning, "GloVe: global vectors for word representation," *In Proc. of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543, Doha, Qatar. Association for Computational Linguistics, October 2014. https://doi.org/10.3115/v1/D14-1162.

[5] Jeffrey Pennington, Richard Socher, and Christopher Manning, "GloVe: global vectors for word representation," *In Proc. of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543, Doha, Qatar. Association for Computational Linguistics, October 2014. https://doi.org/10.3115/v1/D14-1162.

[6] T. T. Vu, V. A. Nguyen, & T. B. Le, "Combining Word2Vec and TF-IDF with Supervised Learning for Short Text Classification," *In 2020 3rd International Conference on Computational Intelligence (ICCI)*, 2020, pp. 241–245, IEEE.

[7] M. Lin, S. Liao, & Y. Huang, "Hybrid word2vec and TF-IDF approach for sentiment classification," *Journal of Information Science*, 45(6), 797–806, 2019.

Received March 14, 2023

Sineglazov Victor. ORCID 0000-0002-3297-9060. Doctor of Engineering Science. Professor. Head of the Department of Aviation Computer-Integrated Complexes.
Faculty of Air Navigation Electronics and Telecommunications, National Aviation University, Kyiv, Ukraine.
Education: Kyiv Polytechnic Institute, Kyiv, Ukraine, (1973).
Research area: Air Navigation, Air Traffic Control, Identification of Complex Systems, Wind/Solar power plant, artificial intelligence.
Publications: more than 700 papers.
E-mail: svm@nau.edu.ua

Savenko Illia. MSc in Computer Science.
Artificial Intelligence Department, Institute for Applied System Analysis, National Technical University of Ukraine "Ihor Sikorsky Kyiv Polytechnic Institute", Kyiv, Ukraine, (2023).

Research Interests: artificial neural networks, artificial intelligence, programming.
Publications: 2.
E-mail: savenko.ilya@lll.kpi.ua

В. М. Синєглазов, І. М. Савенко. Порівняльний аналіз методів векторизації тексту
В роботі розглянуто способи векторизації текстових властивостей природної мови в контексті задачі інтелектуального аналізу тексту. Проаналізовано найпоширеніші способи статистичного аналізу вилучення ознак та методи з урахуванням контексту. В роботі проведено опис вищезазначених типів обрамлення тексту та їх найпоширеніші реалізації. Виконано їх порівняльний аналіз, який показав зв'язок між типом задачі інтелектуального аналізу тексту та методом, що показує найкращі метрики. Описано та реалізовано топологію нейронної мережі, яка стоїть в основі вирішення задачі та отримання метрик. Порівняльний аналіз проведено за допомогою відносного аналізу часу теорії алгоритмів та метрик класифікації: accuracy, f1-score, precision, recall. Метрики класифікації узято з результатів побудови моделі нейронної мережі з використанням описаних методів обрамлення. В результаті в задачі аналізу тональності тексту найкращим виявився статистичний метод обрамлення на основі n-грамів символьних послідовностей.
Ключові слова: інтелектуальний аналіз тексту; обробка тексту природної мови; вставлення тексту; аналіз думок; машинне навчання; Word2Vec; TF-IDF; статистичні вкладення; контекстні вбудовування.

Синєглазов Віктор Михайлович. ORCID 0000-0002-3297-9060.
Доктор технічних наук. Професор. Завідувач кафедри авіаційних комп'ютерно-інтегрованих комплексів. Факультет аеронавігації, електроніки і телекомунікацій, Національний авіаційний університет, Київ, Україна.
Освіта: Київський політехнічний інститут, Київ, Україна, (1973).
Напрям наукової діяльності: аеронавігація, управління повітряним рухом, ідентифікація складних систем, вітроенергетичні установки, штучний інтелект.
Кількість публікацій: більше 700 наукових робіт.
E-mail: svm@nau.edu.ua

Савенко Ілля Михайлович. Магістр комп'ютерних наук.
Кафедра штучного інтелекту, Інститут прикладного системного аналізу, Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна.
Освіта: Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна, (2023).
Напрям наукової діяльності: штучний інтелект, машинне навчання, штучні нейронні мережі, програмування.
Кількість публікацій: 2.
E-mail: savenko.ilya@lll.kpi.ua