Identification of Semantic Patterns in Full-text Documents Using Neural Network Methods

O. Zolotarev[1], Y. Solomentsev[2], A.Khakimova[3], M. Charnine[4]
ol-zolot@yandex.ru, solomencev-yaroslav@mail.ru, aida_khatif@mail.ru, 1@keywen.com
[1]Russian New University, Moscow, Russia
[2]Moscow Institute of Physics and Technology, Moscow, Russia
[3]Research Center for Physical and Technical Informatics, Nizhny Novgorod, Russia
[4]Institute of Informatics Problems FRS CSC of the Russian Academy of Sciences, Moscow, Russia

Abstract *Processing and text mining are becoming increasingly possible thanks to the development of computer technology, as well as the development of artificial intelligence (machine learning). This article describes approaches to the analysis of texts in natural language using methods of morphological, syntactic and semantic analysis. Morphological and syntactic analysis of the text is carried out using the Pullenti system, which allows not only to normalize words, but also to distinguish named entities, their characteristics, and relationships between them. As a result, a semantic network of related named entities is built, such as people, positions, geographical names, business associations, documents, education, dates, etc. The word2vec technology is used to identify semantic patterns in the text based on the joint occurrence of terms. The possibility of joint use of the described technologies is being considered.*

Keywords: *intelligent text analysis, natural language, neural networks*

Abbreviations

Pullenti = SDK extract named entities from unstructured texts (Puller of Entities).
Word2vec = a technology (set of models, method) for the analysis of the semantics of natural languages.

1. Introduction

This article is devoted to the development of new approaches to the analysis of natural language texts based on the mechanism of neural networks. The article also discusses issues of machine learning, the goal of which is to analyze big data, identify patterns and build data processing algorithms based on the patterns found. Initially, the text is marked up, sentences, tokens are highlighted, and morphological analysis of parts of speech takes place. Text processing is carried out using the program Pullenti [5]. With the help of neural networks are hidden patterns in the text. Words for analysis are represented as normalized vectors. For analysis, the word2vec method is used.

2. Features of the Pullenti program

Pullenti is a program for processing unstructured natural language texts. Program functions: breaking down text into words, performing morphological analysis, determining of all possible parts of speech of words (regardless of context), normalizing words, bringing words to the desired case / gender / number, highlighting named entities, multiplication of functions with numeric, nominal and verbal groups, brackets, quotes and other useful features [6].

In Pullenti, such objects as persons, organizations, dates, geographic objects, sums of money, etc. are distinguished. There are specialized analyzers that cover a certain subject area. For example, identifying the structure of a regulatory act and a contract with its details, analyzing the title pages, literary characters, incidents, etc [7]. Here is an incomplete list of named entities that allocate a program: dates, date ranges, phone numbers, websites, sums of money, bank details, keywords and phrases, definitions, measured values and their ranges, countries, regions, seas, lakes, planets, addresses, streets, organizations, persons, passport data, electronic addresses, business facts, links, promotions, product attributes, weapons, relations etc. Selected entities can be represented as a connected graph, see fig. 1.

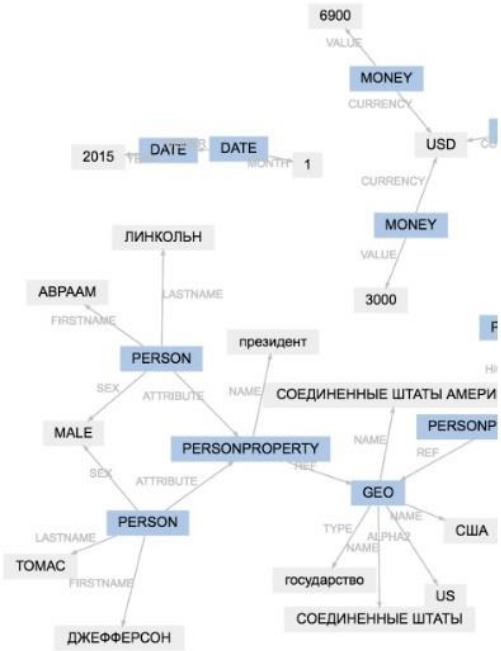

Fig. 1. Graph of selected named entities.

Pullenti ChatBot technology is designed to develop the intellectual part of chat bots. The technology is based on the SDK Pullenti (www.pullenti.ru), which contains various linguistic processing procedures, including morphological analysis. In addition, the technology offers a number of specific handlers of typical situations arising in the process of dialogue. For example, the selection of a phone number from a sequence in which the numbers are given in words, which takes place at the output of voice recognition systems ASR (automatic speech recognition), assessment of emotional state, typical situation (agreement, refusal, greetings ...), etc. The technology is aimed at developing the part of the chat bot that mimics its "brain", that is, responsible for analyzing text fragments from the user (if it's a voice, then after recognizing it), understanding, extracting data from the text and generating text answer.

Here is an example of using Pullenti through Python (Jupyter Notebook). The following program selects name groups from arbitrary text: «American President Donald Trump wrote on Twitter on Thursday that it was time for the US to recognize the Golan Heights as Israel in the interests of the security of Israel and the region as a whole. A number of countries in the Middle East and Europe have already expressed regret in connection with this decision, and the Russian Foreign Ministry called it irresponsible and leading to the destabilization of the region».

The result of the program: «['AMERICAN PRESIDENT', 'PRESIDENT', 'TRUMP', 'THURSDAY', 'TWITTER', 'PORA', 'GOLANA HEIGHT', 'HEIGHT', 'INTEREST', 'SECURITY', 'REGION', 'WHOLE', 'SERIES', 'STRANA', 'NEAR EAST', 'EAST', 'EUROPE', 'Uzh', 'COMPLAINT', 'CONNECTION', 'DECISION', 'DECISION', 'MFA', 'MASTER','DESTABILIZATION', 'REGION']».

Pullenti does not include context definition functions, therefore the meaning of a word must be performed by other means, not by the program Pullenti. One of these tools is a program from Google – word2vec.

3. The principle of the technology word2vec on the algorithm Skip-Gram

Word2vec – is a set of models for the analysis of the semantics of natural languages, which is a technology that is based on distributional semantics and vector representation of words [1]. The word2vec model provides two global operation algorithms: CBOW and Skip-Gram [8,9]. CBOW determines the most appropriate word for a given set of words (by context). Skip-Gram, on the contrary, determines the most appropriate set of words to a given word. This article will consider the algorithm Skip-Gram.

Before the appearance of neural networks, to analyze the proximity of words, a table of frequency of each word was compiled. That is, they made a matrix where words were horizontally and vertically laid out, and the frequency of the word in the specified line with the word specified in the column was indicated in the cells.

This paper discusses the skip-gram algorithm for predicting a neighboring word. Further, this approach covers several words [2].

At the input of the neural network, pairs of words are fed, the window size is selected, and then the moving window slides through the text over all the pairs of words in this window. If you select a window equal to one, the window will contain one word to the left of the target word and one word to the right of the target word. If the size of the window is equal to two, then to the left and right of the target word there will be two words each. Below is an example for a window equal to two (articles removed):

Old *abandoned house stands* on the edge of the forest
Training phrases: (*old, abandoned*), (*old, house*)

Old ***abandoned*** *house stands* on the edge of the forest
Training phrases: (*abandoned, old*), (*abandoned, house*), (*abandoned, stands*)

Old abandoned ***house*** *stands* on the edge of the forest
Training phrases: (*house, old*), (*house, abandoned*), (*house, stands*), (*house, on*)

Old abandoned house ***stands*** on the edge of the forest
Training phrases: (*stands, abandoned*), (*stands, old*), (*stands, on*), (*stands,* edge)

The neural network will learn statistics on the frequency of occurrence of each pair of words. Every word needs to be converted to digital form. One of the common ways is to present it as a column vector (one-hot encoding), for example, like this:

$$\vec{x} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ \dots \\ 0 \end{bmatrix}$$

Here our word, which we represent as a vector, takes second place in the dictionary. Transformations using a neural network can be represented as follows (fig. 2).

Here ***x*** is the input word (or several words) by which we want to predict, ***y*** is the word (or several words).

h (hidden layer of the neural network) is a vector obtained by multiplying the word vector ***x*** by the matrix of weight coefficients ***w***:

$$\vec{h} = w^T \cdot \vec{x}$$

w – is a matrix containing weights, it has the dimension: (dictionary length) * (number of attributes). The number of signs is set once before the launch of the neural network, it is selected to obtain the best result. Example: Google used 300 tags to train a neural network on a variety of Google News data. The weighting coefficients at the initial moment of time take random

values, then they are adjusted in accordance with the method of back error propagation.

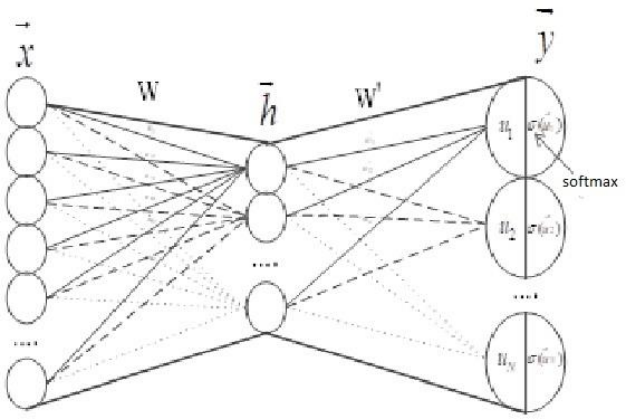

Fig. 2. Schematic diagram of the neural network.

After the hidden layer ***h***, taking into account another matrix of weight coefficients ***w '***, the vector ***u*** is formed:

$$\vec{u} = w'^T \cdot \vec{h} = w'^T \cdot w^T \cdot \vec{x}$$

The dimension of the vector $\vec{U}$ coincides with the dimension of the vector $\vec{x}$.

To normalize the output vector $\vec{y}$ in the range [0; 1], we use the softmax function (it is used as the activation function, see $\sigma(\vec{u}_i)$ figure 2):

$$y_i = \sigma(u_i) = \frac{e^{u_i}}{\sum_{k=1}^{N} e^{u_k}}$$

where N is the number of signs.

As a result, we obtain that $\boldsymbol{y_i}$ is the probability of observing (predicting) the i-th word (or phrase) in the dictionary with the incoming word (context) ***x***.

The purpose of the neural network, shown in Figure 1, is to determine the weights ***w*** and ***w'.*** The criterion for convergence of calculations is the maximization of the probability ***y*** for all possible output words (phrases). As a result of mathematical transformations (taking the logarithm of the probability ***y***, then calculating the derivative of the logarithm of the probability ***y*** using the variable ***w')*** we get an equation for which it is impossible to find the optimum. Therefore, it is necessary to use numerical methods. One of the best numerical methods is the gradient descent method. The result is that you need to solve a recursive task:

$$w'^{new} \leftarrow w'^{old} - G(w \cdot (1 - y))$$

Here ***G*** is a gradient descent function.

Thus, if the probability for the output word being searched is maximal, then the expression in parentheses is close to zero, and $w'^{new} \leftarrow w'^{old}$. Otherwise, when the probability of output word is very small, then from ***w'*** subtracted proportion of values ***w***, so the matrix ***w'*** approaches to the matrix ***w***.

Similarly, ***w*** can be brought closer to ***w':***

$$w^{new} \leftarrow w^{old} - G(w' \cdot (1 - y))$$

The described above method is not applied in practice, since the calculation of the *softmax* function is expensive in duration. Therefore, the authors of *word2vec* proposed an amendment to the algorithm in the form of technology (inclusion function), called "Negative Sampling" (negative sample). This inclusion function is not covered in this article [3,4].

4. Word2vec example on multiple articles

In the example below, we are processing several articles and collections of articles related to virtual reality and modeling.

The processing objects are the following documents: materials of conferences on programming, computer science, collections of articles, presentations, dissertations. The processing program is written in python.

Before processing was only 473 337 words.

As a result of processing module Pullenti formed 320 564 words.

After the processing the above documents for 20 cycles with the word2vec module there were highlighted 8 words closest to the word 'virtual'. Here they are:

Virtual	scholar	0.3870989680290222
	reality	0.2791272401809692
	system	0.2551341354846954
	google	0.2481471002101898
	pubmed	0.2445603758096695
	research	0.1964012682437896
	analysis	0.1937002688646316
	environment	0.0998007102039157

As a result, there were highlighted a set of closest words in the vicinity of term 'virtual'. For each extracted word there were highlighted their neighborhood of the most similar words.

One can construct for each significant term its neighborhood portrait, characteristic of a given subject area.

Many significant domain terms form a characteristic portrait of the domain using a neighborhood approach. The research results can be used as an original method for semantic comparison and classification of documents.

There are a lot of methods of text classification such as Word Mover's Distance, Smooth Inverse Frequency, Pre-trained encoders and so on. but these methods are not based on building a deep multi-level neighborhood of many significant terms [10].

The approach presented in this paper to construct a multi-level neighborhood portrait of a document based on the selection of significant terms using the word2vec algorithm is original.

In this work, we use only certain functions of Pullenti that have common functions for highlighting some entities. Pullenti can use different libraries for different situations. The quality of building model depends on which class the text belongs to. Classification of texts using neural networks will allow us to choose special methods of text processing and improve the quality of the resulting model.

In Pullenti for complex mining tasks, a higher level presentation of data may be required.

Pullenti denotes named entities based on the construction of a chain of adjacent words. The use of neural networks and, in particular, the genism library for additional analysis of the text, allows us to define significant verbose terms that are in the sentence quite far from each other. In this case, it will be possible to form semantic named entities and carry out their identification throughout the text based on the analysis of the word environment.

Conclusion

An example of the work of the program Pullenti has been analyzed, a drawback has been revealed - the lack of definition of the context of words.

An example of the work of the word2vec technology has been analyzed, and the problem of training on a small amount of data has been revealed.

During the training of the word2vec model, satisfactory results were obtained with the number of cycles equal to 20.

The use of methods based on neural networks for the analysis of texts will allow us to switch from text parsing to partially semantic modeling.

The approach outlined in this document can be used to analyze texts, compare and classify documents.

Acknowledgments

This work is supported by Russian Foundation for Basic Research, grants 18-07-01111, 18-07-00909, 19-07-00857 and 16-29-09527.

We are grateful to the Russian Foundation for Basic Research for financial support of our projects.

References

[1] Word2Vec: how to work with vector representations of words [Electronic resource]. // https://neurohive.io/ru/osnovy-data-science/word2vec-vektornye-predstavlenija-slov-dlja-mashinnogo-obuchenija/ (appeal date 08/04/2019).

[2] Word2Vec Tutorial - The Skip-Gram Model [Electronic resource]. // http://mccormickml.com/2016/04/19/word2vec-tutorial-the-skip-gram-model/ (appeal date 08/04/2019).

[3] Ali Ghodsi, Lec 13: Word2Vec Skip-Gram [Electronic resource]. // https://www.youtube.com/watch?v=GMCwS7tS5ZM/ (appeal date 08/04/2019).

[4] models.word2vec - Word2vec embeddings [Electronic resource]. // https://radimrehurek.com/gensim/models/word2vec.html#gensim.models.word2vec.Word2Vec/ (appeal date 08/04/2019).

[5] Zolotarev OV, Sharnin MM, Klimenko SV, Kuznetsov KI System PullEnti - extracting information from natural language texts and automated building of information systems // Proceedings of the International Conference. Situation centers and class 4i information and analytical systems for monitoring and security tasks. SCVRT2015-16, Pushchino, TsarGrad, November 21-24, 2015-2016, Pushchino, pp. 28-35.

[6] Deep Contextualized Word Representations / Matthew Peters, Mark Neumann, Mohit Iyyer et al. // Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. — Association for Computational Linguistics, 2018. — Pp. 2227–2237..

[7] Zolotarev OV, MM Sharnin, S.V. Klimenko, A.G. Matskevich. Research of methods of automatic formation of associative-hierarchical portrait of the subject area // Bulletin of the Russian New University. Series "Complex systems: models, analysis and management." - 2018. № 1. - p. 91 96.

[8] Distributed Representations of Words and Phrases and their Compositionality. / Tomas Mikolov, Ilya Sutskever, Kai Chen et al. // NIPS / Ed. by Christopher J. C. Burges, L´eon Bottou, Zoubin Ghahramani, Kilian Q. Weinberger. — 2013. — Pp. 3111–3119.

[9] Enriching Word Vectors with Subword Information / Piotr Bojanowski, Edouard Grave, Armand Joulin, Tomas Mikolov // Transactions of the Association for Computational Linguistics. — 2017. — Vol. 5. — Pp. 135–146.

[10] Enriching Word Vectors with Subword Information / Piotr Bojanowski, Edouard Grave, Armand Joulin, Tomas Mikolov // Transactions of the Association for Computational Linguistics. — 2017. — Vol. 5. — Pp. 135–146.