Applıcatıon of Paragraphs Vectors Model for Semantıc Text Analysıs

Irina Gruzdo [1[0000-0002-4399-2367]], Iryna Kyrychenko [1[0000-0002-7686-6439]],
Glib Tereshchenko [1[0000-0001-8731-2135]], Olga Cherednichenko [2[0000-0002-9391-5220]]

[1]Kharkiv National University of Radioelectronics, Kharkiv, Ukraine
{irina.gruzdo,iryna.kyrychenko, hlib.tereshchenko}@nure.ua
[2]National Technical University "KhPI", Kharkiv, Ukraine
olha.cherednichenko@gmail.com

Abstract. The paper examined a model of paragraph vectors, as well as its methods of distributed memory and distributed bag of words. The peculiarity of this model lies in the definition of the objective functions of individual sentences and their representation in the form of some local vectors, on the basis of which a global vector is constructed, which determines the semantic component of the text as a whole. Various aspects of the application of distributed memory and distributed bag of words methods were considered, as well as the sets of algorithms of the underlying distributed memory and distributed bag of words methods, which allow obtaining distributed vectors of text parts to solve the problem of determining similar articles, where the search will be carried out key words, annotations, and articles of various sizes. It was experimentally established that Doc2Vec and its Bag-of-Words method, the most complete, allows you to determine borrowing and analogues depending on the structural elements of the text, in accordance with the review and the task. Also Bag-of-Words allows the user to make an exact picture of the lexical meaning of a word and its semantic relations in language and texts.

Keywords: Text Meaning Definition, Semantic Analysis, Latent-Semantic Analysis, Experiment, Textual Information, Model, Semantic Analysis Library, Text Analysis, Text Fragment.

1 Introduction

At the present stage of development of information technologies, both worldwide and in Ukraine, the tasks related to the processing of textual information for solving a number of tasks such as plagiarism detection, text recognition, highlighting the structural blocks of text, analysis and issuance of recommendations, etc. [1, 2, 3]. Among all these tasks, one of the essential problems, which has been solved for more than 60 years and is the “cornerstone”, is the problem of semantic analysis of the text [1, 4, 5]. In [9–15], approaches to checking semantic correctness are shown. During the analysis of the primary sources of the first works devoted to semantic analysis, a tendency was observed to divide the work into those that consider solving abstract theoretical problems, and those that are aimed at facilitating work with a computer and software implementation of solutions. It should be noted that all approaches and models described in them are aimed at solving a specific problem and therefore can only be applied to a narrow circle of subject areas. The mathematical apparatus is also suitable only for formalization of some linguistic mechanisms [14, 15]. Determining the meaning of texts will, to a considerable degree, allow solving a number of word processing tasks with a greater degree of correctness, since it allows improving the analysis procedure.

In [4], a review and analysis of existing solutions for determining the meaning of text documents was carried out, the most used models and methods of semantic text processing were considered, and the classical text processing process for semantic analysis was also described. It should also be noted that when applying the models considered in [4] in practice, in most cases there is a partial loss of the meaningful meaning of the text. This fact is not always justified from the point of view of the problem being solved, although it allows to perform some procedures of semantic analysis. Among the considered models, the model of paragraph vectors allows to solve the problem of word processing with greater accuracy when plagiarism is detected, as well as text recognition.

Therefore, by virtue of the above, this article will consider practical aspects related to determining the meaning of textual information and using the model of paragraph vectors. The work of the model of the vectors of paragraphs and how it behaves on various text blocks of information to solve the problem of finding similarity of documents will be considered in more detail. Two methods will be considered: distributed memory and distributed bag of words. Also in the work will be tested the work of the model of the vectors of paragraphs on the task of processing different length of text fragments for the models of Wikipedia and APNews. Namely, for the task of determining the search for similar articles, where the search will be conducted by keywords, by annotation, and by articles of different size.

The purpose of this work is to test the applicability of the model of paragraph vectors for semantic text analysis as a subtask aimed at solving text processing problems when plagiarism is detected, as well as recognizing text and justifying the choice of the considered method for practical use. As a result of solving this problem, a reasonable conclusion will be made about the applicability of the model of paragraph vectors for semantic text analysis as a subtask aimed at solving text processing problems when plagiarism is detected and text is recognized.

2 Checked model

The model of the vectors of paragraphs is very well described in [6, 7, 8]. This model, while defining the meaning of text documents, adds a memory vector to the standard language model aimed at defining the subject of the document. This model allows you to work with documents of different lengths, such as fragments of texts, sentences, paragraphs and documents. That is why this model was chosen for analysis, since it is this model that allows to determine borrowing and analogs depending on the structural elements of the text, and also satisfies the search task: by keywords, by annotation, and by articles with different volumes.

The model of paragraph vectors in the classical form allows us to solve the problem of predicting a word given in other words in the context of the analyzed text. In context, each word is displayed as a unique vector represented by a column in the matrix. The column is indexed by the position of the word in the dictionary. The concatenation or sum of the vectors is subsequently used as signs to predict the next word in the sentence. In general, the model of paragraph vectors can be represented schematically (see Fig. 1).

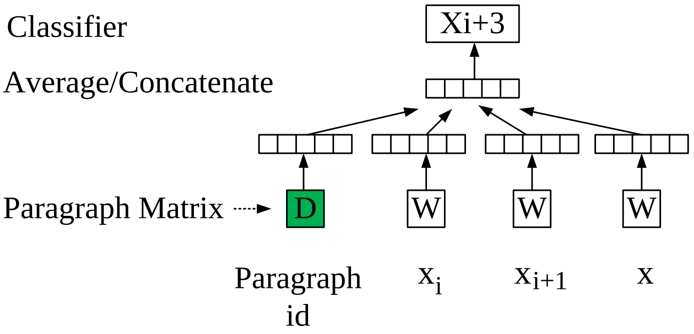

Fig. 1. Paragraph vector associative memory model for input sentence [2]

The figure shows that the paragraph vector is concatenated or averaged using local contextual word vectors to predict the next word. The prediction task changes the word vector and paragraph vector.

It should be noted that the classical model does not fully allow implementing software calculations and testing in practice, as well as drawing reasonable conclusions about the applicability to the task of finding similarity of documents. Also in the classical form in the model of the vectors of paragraphs with a larger amount of data, an increase in the size of the body is observed, therefore, the dimension of the vectors, which will lead to high computational complexity. Therefore, it is necessary to consider this model in more detail and describe the work in the form of an algorithm. To do this, you need to understand what methods underlie it.

Considering the above, it should be noted that the paragraph vector model consists of two methods: distributed memory (DM, distributed memory) and distributed bag of words (DBOW, distributed bag of words). The DM method predicts a word from known preceding words and a paragraph vector. DBOW predicts random groups of words in a paragraph only on the basis of the paragraph vector. These methods are fully implemented in Python and are presented by Word2Vec and Doc2Vec algorithms, in the gensim library.

Consider their work in more detail.

Word2Vec is a set of algorithms for calculating vector representations of words, assuming that words used in similar contexts are semantically close. First, a dictionary is created, and then a vector representation of the words is calculated. A vector representation is based on contextual proximity, the essence of which is that words found in the text next to the same words will have close coordinates of the vectors - words.

Algorithm Word2Vec work as follows [9]:

-the body is calculated and calculated the occurrence of each word in the body;

-the array of words is sorted by frequency and deleted rare words;

-build a Huffman tree (for dictionary coding - it greatly reduces computational and time complexity algorithm);

-from the case is read so-called. Submission which is the basic element of the body - sentence, paragraph, article, after Subsampling is carried out most frequent words) from analysis;

-pass on the proposal, calculating maximum distance between current and the predicted word in the sentence;

-further applied neural network direct distribution with activation function hierarchical softmax and / or negative sampling.

Word2Vec is based on two Continuous Bag-of-Words (CBOW) and Skip-gram algorithms. CBOW and Skip-gram are neural network architectures that describe exactly how a neural network "learns" from data and "remembers" the representations of words. Their working principles are different, CBOW performs the prediction of a word for a given context, and skip-gram allows you to predict a context for a given word.

The operation of the Continuous Bag-of-Words (CBOW) algorithm is described in great detail in the article by David Meyer [6].

The basis of CBOW is the log likelihood calculation:

$$\mathcal{L} = \sum \mathrm{w} \in \mathrm{D} \log \mathrm{p}(\mathrm{w}|\mathrm{c},\ \theta)\ , \tag{1}$$

where θ - model parameters; w is the current word; c is the context of the current word. Schematically, the work of CBOW can be represented in the form (see Fig. 2).

When considering, it is necessary to take into account that the same matrix is used for learning CBOW, which receives several input vectors representing different context words. Training is performed using a simple neural network, using the passage through the entire collection. In- put: one-hot representation of the word (vector of length |T|). Output: distribution in words of the collection (vector of length |T|). The probability p(w|c, θ) is modeled by a softmax-function.

CBOW learning difficulties:

$$\mathrm{Q1} = \mathrm{O}\ (\mathrm{N} \times \mathrm{D} + \mathrm{D} \times \log2\ |\mathrm{V}|), \tag{2}$$

The output layer remains the same, and learning is performed in the manner described above.

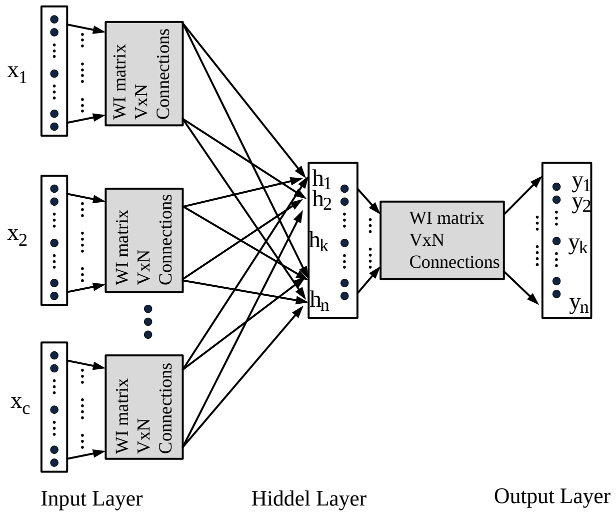
x1
x2
xc
WI matrix VxN Connections
WI matrix VxN Connections
WI matrix VxN Connections
h1
h2
hk
hn
WI matrix VxN Connections
y1
y2
yk
yn
Input Layer
Hiddel Layer
Output Layer

Fig. 2. Continuous Bag-of-Words [7]

Skip-gram is based on calculating a maximizing objective function.

$$L = \sum t,k \sum j \in Contextk(t) \log P(wj |wt). \qquad (3)$$

The Skip-gram neural network is a two-layer network. The second layer implements hierarchical softmax. The cardinal difference from the CBOW model is that the word wt is predicted as many times as there are words in its context, and each time it is predicted based on only one of the words in the context.

The work of Skip-gram can be represented graphically (see Fig. 3).

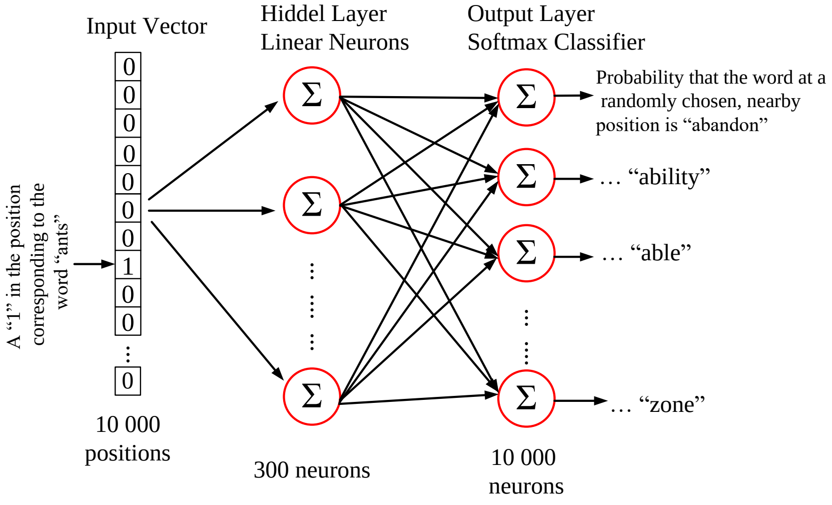

Fig. 3. Neural network for Skip-Gram [8]

The complexity of learning Skip-gram is calculated as:

$$Q2 = O\ (N \times D + N \times D \times \log2\ |V|). \qquad (4)$$

According to the authors, training in the Skip-gram model is more expensive, however, according to [12], this model usually gives the best results.

Given the above, we can conclude that CBOW works faster and Skip-gram works better, especially for relatively rare words.

Since words in a Word2Vec package may appear anywhere in the entire package, each associated vector will receive several adjustments early, middle and late in the process as the model improves - even with a single pass.

Doc2Vec (the original name of the Paragraph Vector), as well as Word2Vec, is a set of algorithms that allow to obtain distributed vectors for parts of texts [12]. Texts can be of variable length: from a sentence to a large document. Doc2Vec allows you to work simultaneously with the words and labels on the document, this fact is very useful in solving the problem of finding similarity of documents as a subtask of finding plagiarism. In Doc2Vec, vector representations of documents are trained to predict words in a document, more precisely, a document vector is taken and combined with several word vectors from it, i.e. tries to predict the next word according to the context. Word and document vectors are trained using the stochastic gradient descent method and the back- propagation method. Document vectors are unique, and the vectors of the same words in different documents are the same.

There are two architectures for building vector representations of documents: Distributed Memory (DM, D2V-DM) and Distributed Bag-of-Words (DBOW, D2V-DBOW).

In Distributed Memory, each document is represented by a unique vector as a column in the matrix, and each term is represented by a unique vector as a column in the matrix. A vector document and word vectors in it are combined or averaged to predict the next word from the context. In fig. 4 shows a graphical interpretation of this architecture.

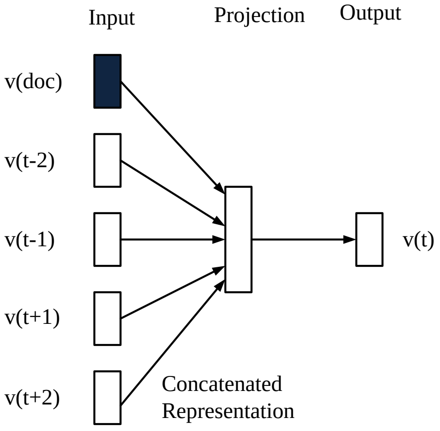

Fig. 4. Distributed Memory (DM, D2V-DM)

In Distributed Memory, you can think of tokens as separate words. They act as a memory that remembers what is missing in the current context or subject of the document. For the same reason, the model is called Distributed Memory.

Distributed Bag-of-Words is simpler than DM, it ignores word order, and as a result, the learning phase goes faster. In DBOW, ignoring words from an input context is used, but it is necessary to predict randomly selected words for an output document. At each iteration of the stochastic gradient descent, a text window is viewed, then a random word is viewed in the text window and a classification task is formed based on the document vector.

Doc2vec is a logical development of the Word2vec model that implements the popular bag-of-words method. The difference is that when creating a vector of sentences, the word order is taken into account.

In Fig. 5 shows a graphical interpretation of this architecture [8].

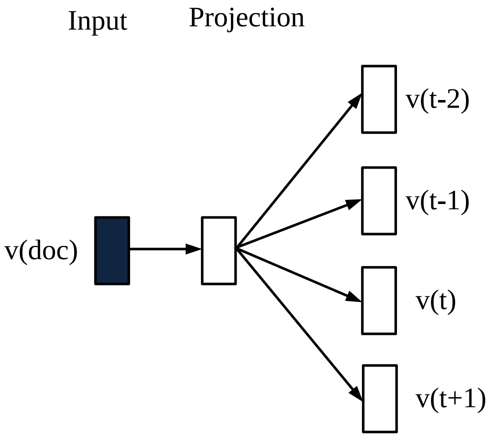

Fig. 5. Doc2Vec - DBOW (distributed bag of words)

In Doc2Vec, it is possible to place both vectors and vectors words in “the same space”, which makes them more interpretable by their proximity to words.

It should also be remembered that the learning rate can sometimes decrease during the iteration according to Labeled Sentence during training, and sometimes this can lead to non-optimal results, therefore it is necessary to double-check the results by repeating the data several times.

Because Doc2Vec often uses unique identifier tags for each document, more iterations may be more important for each vector document to come in for learning several times as the training progresses, as the model gradually improves.

Doc2Vec, according to the authors, is faster than Word2Vec and consumes less memory, since there is no need to save word vectors. Although Word2Vec is good at the vector word representation, it was not intended to generate a single word from several words found in a sentence, paragraph, or document. Doc2Vec also allows you to find a cosine similarity between two documents.

Word2vec shows high results in relation to very short texts, such as messages on the social network Twitter [8, 10]. Doc2Vec works more efficiently with longer messages.

In view of the above, for the experiment on solving the problem of finding similarity of documents as a subtask of finding plagiarism, Doc2Vec was chosen.

3 Experiment

Independent software assessments at the design stage in most cases are performed by people who do not always take into account the relationship between software quality and the development team and resources that are on the project, which in turn leads to erroneous results and is impractical. Word2Vec and Doc2Vec are implemented in Python in the gensim library. For the experiment, Doc2Vec and the Bag-of-Words method were chosen, as it allows to determine borrowing and analogues depending on the structural elements of the text, and also satisfies the search task: by keywords, by annotation, and by articles with different volume. Also, the study of the speed of execution and analysis of the texts of the considered models.

In comparison, two models were used:

- Wikipedia is a model that has been trained on articles in the English version of Wikipedia.
- APNews - a model that was trained on Associated PressNews articles.

For the problem of determining the search for similar articles, the analysis will be conducted depending on the words in the text. Analysis by keywords - the size of textual information is from 1-5 words to 75 words, by annotation - 75-150 words to 400 words, and by articles of various sizes - 400-500 words to 2000 (small) and 2000-3000 words (average).

Below is a summary table that shows the average processing time for different lengths of text fragments for both models (Table 1).

Table 1. The average processing time of different along the length of the text fragments of the considered models.

	Wikipedia	AP News
Text 1-5 words	2.5 seconds	1.9 seconds
Text 75-150 words	75 seconds	51 seconds
Text 400-500 words	100 seconds	93 seconds
Text 2000-3000 words	5.7 seconds	4.6 seconds

Looking at the results, you can see that the processing of information is accelerated on a larger number of words. To verify the results obtained, the data were re-checked several times, and the results were the same. Therefore, it is safe to say that information processing is accelerating with an increase in the number of words. It is not clear why such a decrease in the time spent occurs with an increase in the number of words, since it was not possible to find a mathematical description of the Doc2Vec family of algorithms.

The following assumption can be made: the semantic core itself occupies 2.2 GB, then there is a Python server that processes client requests — it also takes a certain place, and the browser itself, which takes about 1.5 GB of memory. Testing was conducted on a laptop with 4 GB of RAM. When a project is launched for testing, it either fits completely into RAM (which does not correspond to the observed reality in the task manager), or uses the memory completely and then begins to use a paging file that is on a slow HDD and is no longer part of the RAM.

Algorithmic speed of driving text through the semantic core is linear, that is, O (n), where n is the number of words in the text. Therefore, we can conclude that, provided there is a sufficient amount of RAM, the processing speed of processing the client's request will be approximately the same over relatively large intervals of the length of the input text, that is, the comparison table could have approximately the same numbers for each text size. However, given the lack of RAM, errors are added to the runtime that affect the results. Those. for a more complete picture you need to rent an external server, for additional verification of the results obtained.

For the time being, it can be concluded that using the method of paragraph vectors for semantic analysis of text as a subtask aimed at solving text processing problems when plagiarism is detected, as well as text recognition, is appropriate. With this in mind, it can be assumed that the additional time that will be allocated to semantic word processing will not greatly influence the overall text processing process.

By virtue of the above, we can conclude that when analyzing various types of work that differ in volume, the time spent on semantic word processing will be approximately the same. This, in turn, allows, without significant time expenditures in determining the meaning of text documents, to add additional steps related to semantics, and thereby increase the accuracy of the results obtained.

4 Conclusions

In the course of this work:

-Considered a model of paragraph vectors, as well as methods of distributed memory and distributed bag-of-words. The peculiarity of the approach under consideration is the definition of the objective functions of individual sentences and their representation in the form of some local vectors, on the basis of which a global vector is constructed, defining the semantic component of the text as a whole;

-Various aspects of the application of distributed memory and distributed bag-of-words;

-Studied a set of algorithms that allow to obtain distributed vectors for parts of texts for solving the problem of determining similar articles, where the search will be carried out by keywords, by annotation and by articles different in size;

-An experiment was conducted for the Bag-of-Words method Doc2Vec, since it was he who most fully, in accordance with the overview and the task, allows to determine borrowing and analogs depending on the structural elements of the text;

-It has been confirmed in practice that the Bag-of-Words Doc2Vec method allows the user to get an accurate picture of the lexical meaning of a word and its semantic relationships in language and texts.

As a conclusion throughout the paper, we can say that semantic analysis has a high practical application for determining the meaning of text documents.

In a number of publications devoted to the model of paragraph vectors, which deals with the methods of distributed memory and distributed bag-of-words, it is noted that these methods work in the same way, but they allow solving a different spectrum of problems. But after conducting a study, it was found that Word2vec shows high results in relation to very short texts, and Doc2Vec works more efficiently with longer messages. Consequently, the algorithm of their work should be different.

In general, the considered methods Word2Vec and Doc2Vec for the semantic analysis act in the same type, the results of their work are quite similar, but they allow to solve different problems associated with semantic analysis. It should also be noted that in the literature Doc2Vec is very poorly described and it is impossible to fully judge its mathematical apparatus, although Python has the gensim library and allows using the necessary set of algorithms.

It was also found that when analyzing various types of work that differ in volume, using Doc2Vec and the Bag-of-Words method, the time spent on semantic word processing will be approximately the same. This, in turn, allows, without significant time expenditures in determining the meaning of text documents, to add additional steps related to semantics, and thereby increase the accuracy of the results obtained.

The obtained results will allow to continue the work on solving the problem of analyzing texts for the presence of text borrowings and borrowing ideas, as well as determining the authorship of the text, taking into account its paraphrasing.

References

1. Sitikhu, P., Pahi, K., Thapa, P., Shakya, S.: A Comparison of Semantic Similarity Methods for Maximum Human Interpretability. arXiv preprint arXiv:1910.09129 (2019).
2. Panigrahi, A., Simhadri, H. V., Bhattacharyya, C.: Word2Sense: Sparse Interpretable Word Embeddings. In:Proceedings of the57th Annual Meeting of the Association for Computational Linguistics, pp. 5692-5705 (2019).
3. Kanishcheva, O., Cherednichenko, O., Sharonova, N.: Image Tag Core Generation. In: 1st International Workshop on Digital Content & Smart Multimedia (DCSMart 2019) Ukraine, CEUR Workshop Proceedings, Volume 1, pp. 35-44. Lviv (2019). http://ceur-ws.org/Vol-2533/preface.pdf
4. Gruzdo, I.: Overview and Analysis of Existing Decisions of Gothic, International Scientific and Practical Conference of Infocommunications. Science and Technology PIC S & T, pp.645-653. Kharkiv, Ukraine (2018).
5. Vysotska, V., Lytvyn, V., Kovalchuk, V., Kubinska, S., Dilai, M., Rusyn, B., Pohreliuk, L., Chyrun, L., Chyrun, S., Brodyak, O.: Method of Similar Textual Content Selection Based on Thematic Information Retrieval. In: Proceedings of the International Conference on Computer Sciences and Information Technologies, CSIT, 1-6. (2019)

6. Dai, A., Olah, Ch., Le Q.: Document Embedding with Paragraph Vectors (2015) https://arxiv.org/abs/1507.07998
7. Beutel, A., Covington, P., Jain, S., Xu, C., Li, J.: Latent Cross: Making Use of Context in Recurrent Recommender Systems, Ed H. Chi. WSDM'18, Marina Del Rey, CA, USA (2018).
8. Le,Q., Mikolov, T.: Distributed Representations and Documents. In: 31st International Conference on Machine Learning, JMLR: W & CP, vol. 32 (2), pp. 1188-1196. Beijing, China (2014).
9. Smooth, A.V., Ideas, M.M.: Bakhtin on utterance and dialogue and their significance for the formal semantics of natural language. In the book: Interactive systems: Reports and theses of reports and messages of the third school-seminar.-Tbilisi: Metsniereba, vol. I, p. 33-43 (1981).
10. Golovina, E.A., Kolmychek, K.N., Terzian, V.N.: The principles of verifying the semantic correctness of natural language utterances. In the book. : Problems of Bionics.-Kharkov: KSU, no. 32. pp. 64-72 (1984).
11. Golovina, E.A., Terzian, V.Ya.: Express analysis of natural language utterances. In the book: Interactive systems: Materials of the fifth school-seminar.-Tbilisi: Metsniereba, pp. 385-388 (1983).
12. Apresyan, D.Yu.: Toward a formal model of semantics: rules for the interaction of meanings. In the book: Representation of knowledge and modeling of understanding processes.-Novosibirsk: Computing Center of the Academy of Sciences of the USSR, pp. 47-78 (1980).
13. Skorokhodko, D.F.: Semantic networks and automatic text processing.-Kiev: Naukova Dumka (1983).
14. Terzian, V. Ya.: Theoretical and experimental study of the problem of semantic analysis of natural language utterances: dis. cand. tech. Sciences: 05.13.01 "Technical cybernetics and information theory" Terziyan Vagan Yakovlevich; Kharkiv. Institute of Radio Electronics. - Kharkov (1984).
15. Harris, L.R.: Using a Data Base as a Semantic Component to Aid in the Parsing of Natural Language Data Base Querries. In: Journal of Cybernetics, v. 10, No. 1-3, pp. 77-96 (1980).