UDC 004.421
DOI: 10.15587/1729-4061.2022.263421

DEVELOPMENT OF A THEMATIC AND NEURAL NETWORK MODEL FOR DATA LEARNING

Akerke Akanova
Corresponding author
PhD, Senior Lecturer*
E-mail: akerkegansaj@mail.ru
Nazira Ospanova
PhD, Associate Professor, Head of Department
Department of Information Technology
Toraighyrov University
Lomova ave., 64, Pavlodar, Republic of Kazakhstan, 140008
Saltanat Sharipova
Master of Science in Informatics*
Gulalem Mauina
Master of Engineering and Technology
Department of Information Systems**
Zhanat Abdugulova
Candidate of Economic Sciences, Associate Professor
Department of Systems Analysis and Management
L. N. Gumilyov Eurasian National University
Satpayev str., 2, Nur-Sultan, Republic of Kazakhstan, 010008
*Department of Computer Engineering and Software**
**S. Seifullin Kazakh Agro Technical University
Zhenis ave., 62, Nur-Sultan, Republic of Kazakhstan, 010000

Research in the field of semantic text analysis begins with the study of the structure of natural language. The Kazakh language is unique in that it belongs to agglutinative languages and requires careful study. The object of this study is the text in the Kazakh language. Existing approaches to the study of the semantic analysis of text in the Kazakh language do not consider text analysis using the methods of thematic modeling and learning of neural networks. The purpose of this study is to determine the quality of a topic model based on the LDA (Latent Dirichlet Allocation) method with Gibbs sampling, through neural network learning. The LDA model can determine the semantic probability of the keywords of a single document and give them a rating score. To build a neural network, one of the widely used LSTM architectures was used, which has proven itself well in working with NLP (Natural Language Processing). As a result of learning, it is possible to see to what extent the text was trained and how the semantic analysis of the text in the Kazakh language went. The system, developed on the basis of the LDA model and neural network learning, combines the detected keywords into separate topics. In general, the experimental results showed that the use of deep neural networks gives the expected results of the quality of the LDA model in the processing of the Kazakh language. The developed model of the neural network contributes to the assessment of the accuracy of the semantics of the used text in the Kazakh language. The results obtained can be applied in systems for processing text data, for example, when checking the compliance of the topic and content of the proposed texts (abstracts, term papers, theses, and other works)

Keywords: multilayer neural network, LDA model, deep learning, backpropagation

Received date 22.06.2022
Accepted date 12.08.2022
Published date 31.08.2022

***How to Cite**: Akanova, A., Ospanova, N., Sharipova, S., Mauina, G., Abdugulova, Z. (2022). Development of a thematic and neural network model for data training. Eastern-European Journal of Enterprise Technologies, 4 (2 (118)), 40–50. doi: https://doi.org/10.15587/1729-4061.2022.263421*

1. Introduction

High technologies occupy a large place in solving the issue of automating text processing (hereinafter referred to as ATP), in other words, processing big data, which speeds up the process of solving problems of high complexity. To solve problems with the processing of textual information, various technologies and methods of artificial intelligence, and in particular machine learning (ML), are widely used. AI (artificial intelligence) is used in various areas, for example, in the field of automating the processes of pattern recognition [1], separating data into certain classes [2], adaptive control [3], clustering [4], forecasting [5] and others. One of the tasks of scientists in this field is the classification of the submitted text on certain topics. Long before the use of machine learning in automated text processing, developers had difficulty in presenting a translation from one language to another of large texts, sentences, the semantics of the translation expected better. The problem was also in presenting accurate information to queries made in Internet browsers. But with the advent of machine learning, it became possible to make the task easier for both the user and the developer. Scientists began to study in more depth the application of machine learning in automated text processing. The main task of scientists was to bring the machine's understanding of the meaning of the text closer to human understanding. Most of the textual information is transmitted electronically, especially in educational institutions and organizations leading a large document flow. In connection with the pandemic, semantic text analysis has become the most pressing issue in the field of automated text processing. However, to process text documents, users have to read huge amounts of text data, which takes a lot of time. However, to automate this process, models and tools are required that will perform high-quality semantic analysis of the text in the Kazakh language. This development will allow creating intelligent systems for the semantic analysis of text not only in the Kazakh language, but also in all agglutinative languages. This will facilitate the work when checking text documents in large volumes.

Thus, the research conducted in the field of automated text processing in the Kazakh language using the LDA model is relevant.

2 Literature review and problem statement

Over the past decade, algorithms and models based on neural networks have been widely used in the field of automated text processing. The work [6] considers the processing and classification of feedback from students during distance learning through the use of various RNN architectures, such as simple RNN, LSTM and GRU. As a result of the study, the author found that Macro-F1 is of the greatest importance when using the LSTM architecture. The usefulness of this study lies in the implementation and evaluation of the performance of several RNN architectures in text classification, but only for the Indonesian language. In addition, in [7], the microblogging texts of the provinces and cities of the national network were analyzed as the main data, including the relevant comments, which helped to understand the events in the electric power industry and people's attitudes towards these events. In the course of the study, a professional dictionary of electricity was extracted. The result of the experiment showed that the LSTM-RNN classification is more accurate. The indicator was 83.1 %, which is significantly better than the traditional LSTM and RNN text classification models (78.4 % and 73.1 %). This was the reason for choosing this architecture as one of the neural network layers in this study. In this study, the object of study covers the text only on the electric power industry, which limits the scope of its results. However, the use of LSTM and its performance once again prove its superiority over other architectures in text processing.

Much research has focused on applying the LDA model to Web services clustering using word2vec using the K-mean++ algorithm. The result of their use demonstrates a significant improvement in the accuracy of clustering compared to other methods [8]. However, the application of the K-mean++ algorithm for the Kazakh language did not show a good result. The Latent Dirichlet Allocation (LDA) method was used in particular for the analysis of international standards, where the top 10 keywords were extracted using the LDA method. Also, this method made it possible to apply it in working with cloud computing to solve problems with data transfer, which is a great success in this area [9]. The Latent Dirichlet Distribution (LDA) model was used in the one-pass clustering method to extract hidden information about microblogging topics. This method has shown that the probability of skipping false information (not relevant to blog topics) when extracting data is reduced, and the costs of normalized topic detection are also reduced [10]. The Dirichlet Latent Allocation (LDA) model was applied to identify groups of skills required by employers on job sites. The advantage of the method is that this method can be used for announcements in various sectors of the industry, but this method is only used for analyzing text on the web [11].

LDA has also been used in conjunction with node2vec and GraphGAN to detect matching between user and movie characteristics. This was done in order to subsequently offer the user a movie suitable for viewing. This combination of methods provides text and graphics analysis [12]. LDA has shown its power by serving to extract related keywords from a user's profile. Then automatically classify keywords into several categories according to its types. However, this study is currently narrow-profile and can only be used in considering expert profiles [13]. To improve the quality of sensitive information topic recognition, an extended vocabulary of such sensitive word results can be included in the LDA model. It is this approach that was used in [14], however, in this study, a word-form dictionary was developed in advance and then applied in LDA. An analysis of the literature suggests that the LDA model provides ample opportunities for studying various aspects of automatic text processing. Each study was used for a specific task and for a specific language. The advantage of these studies is that a large number of combined methods with LDA and experimentally proven good results. Separate parts can be used in the study for the processing of the Kazakh language.

Thus, the existing studies show that it is expedient to conduct a study on the semantic analysis of the text. For it, it is possible to use the LDA topic model using a neural network, which is still an open question in the field of word processing.

In modern science, there are a sufficient number of models and methods for automating text processing, but most of them are for texts in English, Chinese, Indian and other world languages. For the semantic analysis of the text in the Kazakh language, no similar models were found.

3. The aim and objectives of the study

The aim of the study is to determine the quality of the LDA model through the learning of a neural network when processing text in the Kazakh language. This will improve the quality of the semantic analysis of texts in the Kazakh language.

To achieve the aim, the following objectives were set:

– create a matrix of evaluations of the distribution of words by topics and topics by documents using LDA;

– develop a neural network model for learning, which determines the quality of the LDA model in text processing in the Kazakh language.

4. Materials and methods of research

The object of the study is the text in the Kazakh language. It should be expected that the use of a topic model and a neural network for learning text in the Kazakh language will show a good result in extracting and distributing keywords by topic. Assumptions in the study are processes such as extracting keywords from text and learning a neural network. In the neural network in the SpatialDropout1D and LSTM layers, the Dropout method was used to reduce input signals.

Analysis of research in the field of automated text processing in the Kazakh language (agglutinative language) showed that the use of a deep neural network will contribute to obtaining good results in the semantic analysis of text. One of the previous stages was the creation of a text corpus and a dictionary-word forms for processing texts in the Kazakh language. To create a corpus from the text, words that do not carry a semantic load were cleared. These are stop words, words from other languages and other noises. The process of creating a dictionary-word

forms includes the stemmatization of the text, which occupies a special place in the further work of the thematic model. For the stemmatization of texts in the Kazakh language, Porter's algorithm was used [15]. This process simplified the text to word forms by removing endings and suffixes.

Topic modeling is often used in the analysis of news, archival documents and the identification of hidden topics. For example, there is a text about neural networks, in which 90 % of the text includes the word "recognition" in combination with the word "graphics", and the abbreviation "NLP" occurs in 10 % of the text. From the percentage, let's intuitively understand that it is about image recognition using neural networks, but not natural language processing, although it was also mentioned.

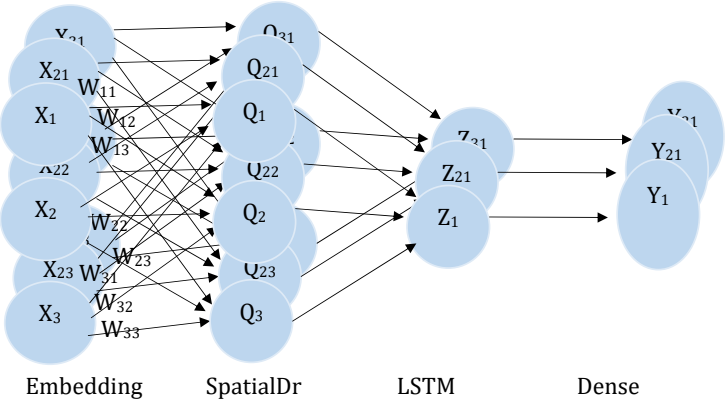

Fig. 1. Multilayer neural network for learning text in the Kazakh language

One of the well-known topical models is LDA – Latent Dirichlet Allocation (latent Dirichlet distribution). The LDA model was presented in 2001 at the Conference on Achievement in Neural Information Processing Systems [16] as a graph model for discovering hidden topics. The LDA model is a generative model of the probabilistic LSA model and creates a semantic model of the vector space. The LDA model significantly outperforms other topic models by smoothing the frequency estimates of conditional probabilities. The semantic connection of words in LDA, which determines the topic, is formed by highlighting the frequent use of words in combination with other words. Rarely used words are accepted as noise and are ignored when reducing the dimension of the vectors. To obtain estimates of the mathematical expectation of distributions of topics over texts and words over topics, a classical method was chosen – Gibbs sampling. The process of calculating the estimate of the received data using Gibbs sampling includes fixing all variables at each step and choosing the remaining variable according to the probability distribution of this variable [17]. The evaluation of the received data determines the importance (weight) of keywords in topics and topics in documents, where keywords with the highest weight reflect the topic of the document.

When constructing artificial neural networks, a number of assumptions and simplifications are made.

In the built neural network, the architecture of recurrent networks LSTM, which is widely used in NLP, was chosen as the main architecture. The presented neural network, according to the type of structure of neurons, belongs to heterogeneous networks, since it has neurons with different activation functions.

These layers are organized using the Sequential model, which is implemented in the Keras framework. The sequential neural network model is a complex architecture that combines Embedding (matrix initializer layer), SpatialDropout1D (full connection layer before recurrent layer), LSTM (recurrent layer) and Dense (fully connected layer). Where Embedding is the input layer, SpatialDropout1D and LSTM are the hidden layers, Dense is the output layer (Fig. 1).

The Embedding layer (embedding) converts the data obtained from thematic modeling (matrices for assessing the distribution of words by topics and topics by documents) into vectors. The vectors are in an embedding matrix. The nesting matrix associates each object index and its translation into a vector.

The SpatialDropout1D layer is used to split the data into packets and determine their independence. Package independence makes it easier to remove packages that act as noise (in particular, these are sentences that do not carry key information).

The LSTM (longshort-termmemmory) layer is a recurrent neural network and is commonly referred to as long short-term memory. Conventional RNNs may have the problem of associating the current position with long-stored data, i.e. distance may be a hindrance and the issue of long-term dependency may not be resolved. LSTM can solve the problem of data persistence and feedback, as well as word-level text generation [18]. Usually LSTM is compared with the memory of a person who once remembered the information, and then can return to the memory to apply it to new information. There is no problem with long-term dependency in LSTM. LSTM is implemented in such a way that it has 4 hidden layers that are interconnected with each other, especially through the state of the memory cell (cellstate), which is a key component in LSTM. The LSTM uses a sigmoid function to calculate error backpropagation over time, which is used to minimize the learning error between the actual response and the prediction.

The Dense layer implements an exit operation with little dimensionality.

During the learning of the neural network, the best set of weights is searched for to maximize the accuracy of the prediction. In particular, the learning process includes tuning the input data (neural network parameters) by modeling the environment. The model includes regulators (weights or weight coefficients) of sensitivity to different types of input data. If the weight is too high, then even the smallest input value can generate a large predictive output value. If the weight is too small, then a large input value at the output will give a small output value. During the learning process, the neural network receives an input vector and tries to predict in the form of a probability distribution of the words in the topic and the topics in the document.

To train data in the Kazakh language, a deep learning method was applied, which is based on data processing in several layers. Deep learning of the network includes three stages: feeding the learning data to the inputs of the network, backpropagation of the error, and adjusting the weights. The computational process goes from the input layer to the output layer and as a result an error is dis-

played. After receiving the error, the reverse computational process is activated - using the error backpropagation method. The entire computational process moves in the opposite direction to the input, and after reaching the input, the process again goes to the output, adjusting the weights. The stage of error backpropagation and adjustment of the weights is called the learning process.

The main idea of the error backpropagation method is to obtain an error estimate for neurons in hidden layers. Errors that are allowed by neurons of the output layer arise when calculating unknown errors of neurons of hidden layers. The error of the first one affects the error of the second one more strongly if there are large values of the synaptic connection between the neurons of the hidden layer and the output layer. Therefore, the estimate of the error of the elements of the hidden layers can be obtained as a weighted sum of the errors of subsequent layers. In summary, the error propagation algorithm, using the hyperbolic tangent in the hidden layer, rebuilds the weights in the direction of the error minimum. To assess the success of classification in LDA, in this task, the accuracy (1) accuracy calculation method and its visualization through the error matrix (confusion matrix – Fig. 2) were applied:

$$accuracy = \frac{P}{N}, \quad (1)$$

where P is the number of topics for which the classifier made the correct decision, N is the size of the learning sample.

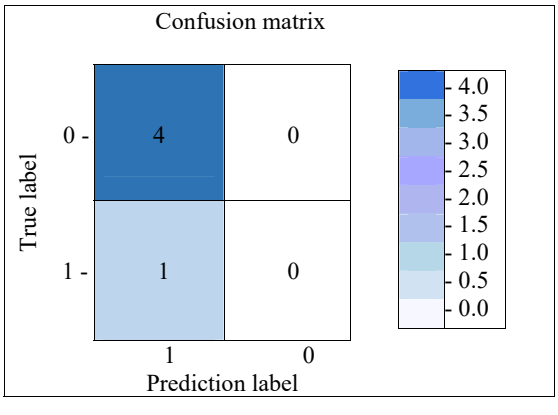

Fig. 2. Error matrix after learning

Thus, the accuracy of learning a neural network within a class is the proportion of words that really belong to a given topic relative to the entire document, and test data processing shows the completeness of learning. The completeness of learning is the proportion of topics found by the classifier in the test sample. These values can be calculated on the basis of a table that is compiled for each class separately (Fig. 2). For example, after learning data in the error matrix in the TR (true positive decision) section, there were 4 topics in which the keywords are close to expectations and provide more information about the topic. In the FN (false negative) section, there were 6 topics that were predicted as negative topics, but in fact, these 6 topics have keywords that are closest to the topic. This process is called learning error.

5. Results of semantic text analysis using the Latent Dirichlet Allocation model

5. 1. Creation of a matrix of estimates for the distribution of words by topics and topics by documents using the Latent Dirichlet Allocation model

The task of constructing a topic model is to determine the hidden topics T, by obtaining a set of one-dimensional conditional distributions φ (w, t) and θ (t, d) based on the text corpus and words used.

Thus, the LDA model assumes that document vectors θ_d and topic vectors φ_w are formed by Dirichlet distributions with parameters $\alpha \in R^{|T|}$ and $\beta \in R^{|W|}$ ((2), (3)):

$$\theta_d = (\theta_{td}) \in R^{|T|}, \quad (2)$$

$$\varphi_w = (\varphi_{wt}) \in R^{|W|}, \quad (3)$$

where, $\varphi_{w,t}$ is the distribution of words by topics, and $\theta_{t,d}$ is the distribution of topics by documents [16].

The initial stage of building a thematic model is the creation of a text corpus (a collection of documents) and a dictionary – word forms. Gensim technology uses the Dictionary function to assign a unique identifier to each token word. The Phraser.bigrams function detects frequently occurring words or bigrams in sentences and returns the frequency of bigrams in a document. Otherwise, a two-dimensional frequency matrix of indexed words or phrases (bigrams) is created [14]. To carry out the distribution of topics by texts (documents) and words by topics with Dirichlet parameters, the following calculations (4), (5) are used:

$$Dir(\theta_d;\alpha) = \frac{\Gamma(\alpha_0)}{\prod_t \Gamma(\alpha_t)} \prod_t \theta_{td}^{\alpha_t - 1},$$

where

$$\alpha_t > 0, \quad \alpha_0 = \sum_t \alpha_t, \ \theta_{td} > 0, \quad \sum_t \theta_{td} = 1, \quad (4)$$

$$Dir(\varphi_t;\beta) = \frac{\Gamma(\beta_0)}{\prod_\omega \Gamma(\beta_w)} \prod_\omega \varphi_{wt}^{\beta_w - 1},$$

where

$$\beta_w > 0, \quad \beta_0 = \sum_w \beta_w, \ \varphi_{wt} > 0, \quad \sum_w \varphi_{wt} = 1. \quad (5)$$

In this formula, α and β are hyperparameters.

The result of applying (4), (5) to the body of the text (collection of documents) forms a matrix of the frequency of the use of words relative to the body of the text (collection of documents). The doc2bow function is applied to the received frequency matrix, which starts creating patterns from bigrams or unigrams with high frequency into topics (topics are obtained) in accordance with Fig. 3. Fig. 3 shows how one document can include three topics: the topic "finance", "jurisprudence" and "religion", which is determined by finding frequently occurring words in one or another combination.

Thus, in LDA, each document is considered as a set of different topics, which can be determined by the frequency of use of keywords. The resulting topics contribute to the formation of a new vocabulary of keywords.

Fig. 3. Keyword frequency

Thus, the LDA model can determine the semantic probability of the keywords of one document and give them a rating factor [19]. The visualization of such an assessment is as follows:

(*0, '0.002*"адамның"+0.002*"тұқым"+0.002*"сүреті́ндіктен"+0.002*"соңғы"+0.002*"сперматоғ"++0.002*"стресс"+0.002*"сызбанұсқа"+0.002*"сызбанұсқасы"+0.002*"сыртқы"+0.002*"соның"'*)

(*2, '0.002*"тұқым"+0.002*"адамның"+0.002*"әр"++0.002*"әдіс"+0.002*"цитогенетикалық"+0.002*"қуалаитын"+0.002*"бір"+0.002*"адам"+0.002*"сызбанұсқа"+0.002*"сау"'*)

(*3, '0.023*"тұқым"+0.020*"адамның"+0.002*"бір"++0.011*"қуалаитын"+0.017*"түрлі"+0.010*"әдіс"++0.008*"кейбір"+0.008*"әр"+0.007*"мысалы"*)

where 0, 1, 2, 3 denote topics, and next to them are words with weights that reflect the importance of the keyword for this topic. From the above example, it is obvious that it is talking about genetics, because words with high weights predominate in the 3rd topic, these are «тұқым» – 0.023, «адам» – 0.020, «қуалаитын» – 0.011 and so on. The given figures are the weight of the topic in the document, which is reflected in Fig. 4 «Мәтін номері» is the number of the document, which corresponds to the number of the topic «Басым тақырып», which prevails in this document and shows the coefficient of its content in the document «Тақырыптың пайыздық үлесі». And the line «кілттік сөздер» reflects the key words that define this topic.

After obtaining the distribution matrices and estimating, the consistency of topics is calculated, which is determined by the coherent model. Topic coherence is calculated by the Coherence model function, which allows to visually see what is the optimal number of topics from the resulting model that has a high coherence score to obtain well-trained data. During the execution of the Coherence model function, the probability is initially calculated once, after which the coherence is estimated for each model (Fig. 5).

The main parameters for building an LDA model for the text in the Kazakh language in the developed thematic model are:

1) corpus_c – a set of keywords on topics, which is obtained after applying the bigram and doc2bow functions to the dictionary, contributing to the formation of topics;

2) id2word – a dictionary of words that represents the identifier of phrases (words) and the frequency of occurrence of phrases (words);

3) num_topics=int (self.ui.spinBox.value()) – number of predicted topics;

4) random_state=100, used in case of repetition of the learning process;

5) update_every=1 – determines how often the model parameters should be updated;

6) passes=100 – the total number of iterations of the sample through the entire corpus;

7) alpha='auto' – Dirichlet hyperparameter for topic distribution (default=1, learns asymmetric previously known knowledge;

8) per_word_topics=True - setting True retrieves the most likely topics with the given word. During learning, each word is assigned to a topic, less likely ones are ignored;

9) nzw_: array, shape=[n_topics, n_features] – matrix of counts of word ratings by topics , which is recorded in the final iteration;

10) ndz_:array, shape=[n_samples, n_topics] – matrix of counts of topic ratings by documents, which is recorded in the final iteration;

11) doc_topic_: array, shape=[n_samples, n_features] – a matrix of evaluations of the distribution of topics among documents, denoted in formulas as "θ, Θ";

12) nz_:array, shape=[n_topics], a matrix for counting topic ratings, writing in the final iteration.

Thus, the result of thematic modeling is a matrix of evaluations of the distribution of words by topics and topics by documents. This matrix feeds the input data to the first layer of the neural network – Embedding. In the Embedding layer, the learning sample (data matrix) is written to the X_train.shape variable and is described in the parameters of this layer.

Мәтін нөмері	Басым тақырып	Тақырыптын пайыздық үлесі	Кілттік сөздер
0	9	0.8382999897003174	электр, болат, кел, қасиеттер, қалпы, тотықтан, со, пештер, қоспалар,
1	3	0.9704999923706055	элементтер, қорытпа, темір, көмірте, қосыл, болат, көмірт, жасал, деген
2	23	0.8766999840736389	болат, өңдіру, әдістер, вакку, жас, бұрын, бөліне, атмосфера, до, кел
3	13	0.9855999946594238	болат, конвертер, кірпіш, ағылшы, бастапқы, бессем, қолдан, жылдар, асты
4	18	0.9811000281622162	мар, пеш, кеңістігі, оты, металургтар, мазут, табыла, балқыт, жағыла, регенератив
5	1	0.9496999979019165	балқыту, электр, жата, мына, пештерді, салыстырға, агрегаттары, атмосфера
6	3	0.9386000037193298	элементтер, қорытпа, темір, көмірте, қосыл, болат, көмірт, жасал, деген
7	9	0.9861000180244446	электр, болат, кел, қасиеттер, қалпы, тотықтан, со, пештер, қоспалар,
8	15	0.9855999946594238	шөміштер, пештеріб болат, сиымдылы, балқжерыт, биікті, аралығы, жер, балқы

Fig. 4. Distribution of topics by texts and keywords by topics

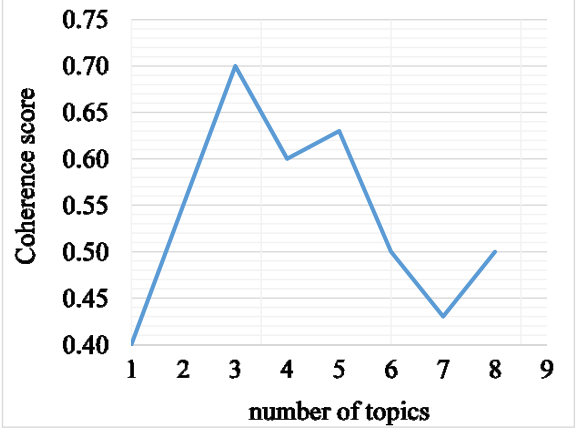

Fig. 5. Coherent model

5. 2. Development of a neural network model for learning, which determines the quality of the LDA model in text processing in the Kazakh language

For learning, a neural network was created, consisting of four consecutive layers: Embedding, SpatialDropout1D, LSTM, Dense (Fig. 6)

Layer (type)	Output Shape	Param #
embedding (Embedding)	(None, 130, 128)	1024000
spatial_dropout1d (SpatialDr	(None, 130, 128)	0
lstm (LSTM)	(None, 64)	49408
dense (Dense)	(None, 25)	1625

Fig. 6. Neural network model for learning a text model in the Kazakh language

In this study, the Embedding layer received 1024000 neurons, which after SpatialDropout1d was sparse to 49408 neurons and transferred to the LSTM layer, then sparse is also performed in the LSTM and 1625 neurons are transferred to the Dense layer.

At the input to the neural network, there are the keywords and their weights obtained through the LDA model. Keywords have numeric identifiers that are used for calculations in learning.

The formation of vectors (Fig. 6) can be traced in the Embedding layer. The dimension of the input layer is set relative to the dimension of the input data sequence. In this case, there is a text corpus with a maximum number of words of 8000, and it is also assumed that the dimension of the data sample will have a maximum sequence of 130. Therefore, if the dimension is not specified during learning, then Embedding will build a matrix of 130x128 and if the number of samples is less than the specified sequence , then the remaining places are filled with zeros (Fig. 7).

Name: Table, dTaype:Int64						
[[383	1	88	…	1	425	1]
[113	1	31	…	1	144	1]
[2	1	21	…	1	333	1]
[0	0	0	…	1	245	1]
[0	0	0	…	0	334	1]
[0	0	0	…	0	19	1]]
130						

Fig. 7. Sequence of neurons

After receiving the vectors, the SpatialDropout1D layer performs a calculation with the received data, that is, each input data element is multiplied by its weight coefficient (weights), the resulting products (partial predictions) are summed and the weighted sum of the inputs (scalar product) is displayed, for example, as in (6), (7):

$$(X_1 * W_{11}) + (X_2 * W_{21}) + (X_3 * W_{31}) = Q_1, \quad (6)$$

$$(X_1 * W_{12}) + (X_2 * W_{22}) + (X_3 * W_{32}) = Q_2. \quad (7)$$

The matrix of input vectors is multiplied by the weight vectors, the product of which is summed with the bias value b (8). The sigmoid activation function (9) is applied to the obtained value. The dot product of the input is then added to the offset value b (default is 1).

$$Q\_in_i = \sum_i^n x_i * w_{ij} + b, \quad (8)$$

$$Q_j = f(Q\_in_i), \quad (9)$$

where Q_int is the argument of the activation function, the result of the function is sent to the next layer.

The SpatialDropout1D layer is used to supply vectors in batches (batch), instead of individual elements, and thus prevents memorization when learning entire sentences.

The SpatialDropout1D layer is dimensionally reduced using the Dropout method, which is a regularization method to prevent the network from overfitting. If at the learning stage the neuron is not deleted with probability q to emulate

the behavior of neural networks, then at the testing stage the activation function $f(Q)$ is multiplied by the coefficient q. Hence, Dropout for the i-th neuron at the testing stage looks like this:

$$Q_i = q_i f\left(\sum_{k=1}^{d} w_k x_k + b\right). \quad (10)$$

The data that SpatialDropout1D gave out is received by the third LSTM layer, which performs recursive calculations.

The LSTM architecture will have one input, one output and one hidden layer, which is usually called a memory cell, this layer has forget states [20]. Also has two more hidden layers Dropout1.

The process in the LSTM layer when processing text data will look like this:

1) the input state $i(t)$ controls which of the new value will be stored in the memory cell and is calculated as follows:

$$i_t = f\left(x_t W_{xi} + h_{t-1} W_{hi} + c_{t-1} W_{ci} + b\right), \quad (11)$$

where i – the input state;

2) the forgetting state $l(t)$ controls what goes from the previous cell value to the current cell value and is calculated as:

$$l_t = f\left(x_t W_{xf} + h_{t-1} W_{ht} + c_{t-1} W_{cf} + b_f\right), \quad (12)$$

where l – the state of forgetting;

3) a candidate for a new value of the memory cell $c(t)$, the scale of which depends on the update of each state value, is calculated as follows:

$$c_t = l_t c_{t-1} + i_t \tanh\left(x_t W_{xc} + h_{t-1} W_{hc} + b\right), \quad (13)$$

where c – a memory cell;

4) the output state $h(t)$ is the hyperbolic tangent (14) multiplied by the output state [21].

$$h(t) = \tanh\left(c_t o_t\right), \quad (14)$$

$$o_t = f\left(x_t W_{xo} + h_{t-1} W_{ho} + c_t W_{co} + b_i\right), \quad (15)$$

here o – the output state, x – the input cell activation vectors and h – the hidden vector, W_{hi} – the matrix of hidden inputs, W_{xo} – the matrix of inputs and outputs, and b_i – the base vector.

The hyperbolic tangent in LSTM is taken by default and is calculated from (16):

$$\tanh(x) = \frac{2}{1+e^{-2x}} - 1. \quad (16)$$

The LSTM layer consists of two hidden layers: dropout=0.7, recurrent_dropout=0.7 in which the dimensionality of the network decreases with a probability of 0.7, the result is sent to Dense.

Dense is the last output layer, in which the usual function for calculating the weighted sum of the output layer is carried out and the Softmax activation function is implemented. Softmax, gives the probability distribution over all labels. The activation function S_i for the i-th neuron is calculated by the formula:

$$S(y)_i = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}}. \quad (17)$$

Incoming signals in the Dense layer are calculated in the same way as in the other layers (18), (19), y_in_k – the total value transmitted to the input of the output element y_k:

$$y\_in_i = \sum_i^n x_i * w_{ij} + b, \quad (18)$$

to which the activation function is applied:

$$y_i = f\left(y\_in_i\right). \quad (19)$$

The next step is learning the neural network.

So, there is a 4-layer neural network that is ready for learning. The learning process is an error calculation by backpropagation. After calculating the error in the last Dense layer, the calculation goes back to the input. After reaching the input layer, the process begins to adjust the scales in the direction of the output (Fig. 8).

Each layer uses an activation function to calculate the cross-entropy error: LSTM – hyperbolic tangent, Dense – Softmax.

Neural network learning begins with a backpropagation process, where each output neuron in turn computes the result of its activation function, which is nothing more than the output y of that neuron for the corresponding input. In the last Dense layer, the output neuron y is displayed and the error (20) is calculated:

$$\sigma_k = \left(z_k - y_k\right) * f'\left(y\_in_k\right), \quad (20)$$

where y –the output signal, σ – the error and z – the desired signal.

Here, the value by which the weight of the connection (21) will change is calculated

$$\Delta w_{jk} = \upsilon * \sigma_k * x_j, \quad (21)$$

the bias correction is calculated, where υ is the learning rate (22):

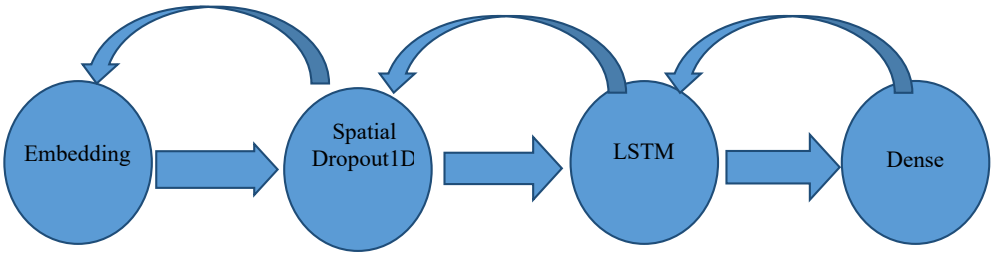

Fig. 8. Backpropagation Process

$$\Delta w_{ik} = \upsilon * \sigma_k, \quad (22)$$

and cross entropy (32).

The crossentropy loss between labels and predictions is calculated, and how often the predictions match the labels is calculated.

The error σ is the difference between the output predicted signal y and the desired signal *z*, i.e.

$$\sigma = y - z. \quad (23)$$

The hidden LSTM layer takes incoming errors from Dense and computes them as follows (24):

$$\sigma\_in_j = \sum_{k=1}^{n} \sigma_k * w_{jk}. \quad (24)$$

Then the error value (26) is calculated, that is, the product of the error is multiplied by the derivative of the activation function (hyperbolic tangent) (25)

$$f(x) = \tanh(x), \quad (25)$$

$$\sigma_k = \sigma\_in_j * f'(x), \quad (26)$$

where the function activation derivative in LSTM is calculated by the formula:

$$f'(x) = 1 - f(x^2), \quad (27)$$

here the value by which the weight of connection (28) will change is calculated:

$$\Delta v_{ij} = \upsilon * \sigma_i * x_i, \quad (28)$$

then offset adjustments (29) are calculated:

$$\Delta v_{ij} = \upsilon * \sigma_i. \quad (29)$$

Each output neuron changes the weights of its connections with the bias element and output neurons

$$w_{jk}(\text{new}) = w_{jk}(\text{old}) + \Delta b_{ij}. \quad (30)$$

Dropout in the learning phase

$$Q_i = X_i f\left(\sum_{k=1}^{d} w_k x_k + b\right). \quad (31)$$

During the forward pass, all synaptic weights of the network are fixed, and during the backward pass, all synaptic weights are adjusted according to the error correction rule (Fig. 5). So, there is data at the output: *y_pred*, error σ and the desired signal *z*.

Before learning the neural network, it is required to tune the constructed model using the compile() method, which has the following arguments: optimizer – 'adam', error function – 'categorical_crossentropy' and metric list – 'acc'.

The Adam optimizer is an algorithm for iteratively updating network weights based on learning data. The method calculates individual adaptive learning rates for various parameters [22].

The categorical cross-entropy error function calculates the error between the actual and received data, which helps to minimize the error. The cross-entropy error function should not depend on the activation values. Categorical cross-etropy is used in data classification and is calculated as follows

$$H(p,y) = \\ = -\sum_x p(x) \log y(x) t * \log(\sigma) + (1-t) * \log(1-\sigma). \quad (32)$$

During learning, forward movement along the network corresponds to the activation of neurons, and during backward passage, the node weights and neuron displacement weights are corrected, then the weights are updated in such a way that the error for the input vector is minimized. This process occurs during iteration (epoch), after each iteration, the weights are redistributed between neurons using the gradient descent method for the same minimization of the loss function. Learning stops after passing a given number of iterations [22].

In this neural network, the metrics=['acc'] – accuracy metric was used during learning, since it is a basic metric and measures the number of correctly issued keywords relative to the total number of all words. The measurements of the ratio between them are calculated in a well-known way (33):

$$\text{accuracy} = \frac{TP + TN}{TP + TN + FP + FN}, \quad (33)$$

where

$$TP(\text{True positiv}) = \sum_{i=0}^{n} \left[\mu(x_i) = +1\right]\left[y_i = +1\right], \quad (34)$$

$$TN(\textit{True negativ}) = \sum_{i=0}^{n} \left[\mu(x_i) = -1\right]\left[y_i = -1\right], \quad (35)$$

$$FP(\textit{False positiv}) = \sum_{i=0}^{n} \left[\mu(x_i) = +1\right]\left[y_i = -1\right], \quad (36)$$

$$FN(\textit{ False negativ}) = \sum_{i=0}^{n} \left[\mu(x_i) = -1\right]\left[y_i = +1\right]. \quad (37)$$

Here TP shows the sum of the number of classes $\mu(x_i)$, in which the pair "topic and keywords" has high accuracy and is true both in prediction and in the results of the learning algorithm (34). Section TN shows the sum of the number of classes $\mu(x_i)$ in which the pair "topic and keywords" has false precision both in prediction and in the results of y_i learning algorithm (35). The FP section shows the sum of the number of classes $\mu(x_i)$, where the pair "topic and keywords" has true predictive accuracy and false according to the results of y_i learning algorithm (36). Section FN shows the sum of the number of classes $\mu(x_i)$, where the pair "topic and keywords" has a false prediction accuracy and a true accuracy according to the results of the y_i learning algorithm (37). In the FN and FP section, the system shows the errors that were received during the learning.

Hence, the multilayer neural network model for working with NLP has 4 layers: an input layer, two hidden ones and an output layer. One of the hidden layers of the LSTM contains two more hidden layers, which are used as a regularizer to reduce dimensionality, reduce the number of neurons. In LSTM, the activation function is the hyperbolic tangent, and in the Dense layer, the activation function is Softmax. In Python, the neural network architecture is represented as follows:

```
model.add(Embedding(n_most_common_words,emb_dim,input_length=X_train.shape))
model.add(SpatialDropout1D (0.7))
model.add(LSTM(64, dropout=0.7, recurrent_dropout=0.7))
model.add(Dense(num_classes, activation='softmax'))
```

Before learning, all having vectors are divided into trainees and testees. The tested subset of vectors and their labels are stored in the variables *x_test* (for vectors), *y_test* (for labels of all data in *x_test*). The learning subset of vectors is stored in *x_train* (for learning vectors), *y_train* (for labels of all data in *x_train*). Therefore, let's obtain a matrix of the form Tables 1, 2.

Table 1

Matrix with data in *x_test*

0	1	2	3	..	...	...	126	127	128	129
0	0	0	0	...	...	...	0	0	101	1
0	0	0	0	...	...	...	186	1	187	1
0	0	0	0	...	...	...	255	1	256	1
0	0	0	0	...	...	...	2	1	25	1
0	0	0	0	...	...	...	5	1	197	1

Table 2

Label matrix in *y_test*

0	1	2	3	..	...	...	26	27	28	29
0	0	0	0	...	...	...	0	0	0	0
0	0	0	0	...	...	...	0	0	0	0
0	0	0	0	...	...	...	0	0	0	0
0	1	0	0	...	...	...	0	0	0	0
0	0	0	0	...	...	...	0	0	0	0

In this study, in learning, it is necessary to determine the coefficient of dependence (which is unknown) between pairs of neurons – this is a pair of "topics and keywords". These dependencies are obtained using the LDA model and the components of the learning sample. To solve such problems, a method is used that, on the basis of learning samples, determines the relationship between the "topic and keywords" and evaluates the qualitative dependence of the pair. To evaluate the dependence, the model.compile(), model.fit(), model.evaluate, model.save methods are used. These methods allow to compile a neural network model, write it to the database, calculate the accuracy of the learning quality and save the resulting model. The parameters in model.compile() have a key effect on the quality of the resulting work.

The main arguments of the model.compile() method are:

1. The optimizer used to update the weights in the program looks like this: optimizer='Adagrad', optimizer='SGD' and so on. To optimize the learning process, the SGD optimizer was chosen. The argument in this optimizer is learning_rate and the initial learning rate must be float>=0.

2. The metric by which the quality of the model is determined, that is, the proportion of correctly guessed answers, the program uses the following metrics: metrics=['binary_accuracy'], metrics=['accuracy'] and others.

3. Error function, the program uses such types as: loss='binary_crossentropy', loss='crossentropy' and others.

To select the optimal variant of the parameters, experiments were carried out, the results of which are given in Table 3.

The main idea behind deep learning is to use this estimate to adjust the weight values to reduce the loss. Based on the above experiments, the optimal parameters were chosen:

model.compile ('binary_crossentropy', optimizer='Adagrad', metrics=['cosine_proximity']).

From Table 3 shows that the error is 0.2443, and the accuracy=0.8333.

Thus, the result of the experiment to determine the quality measure of the learning algorithm showed that the optimal combination of the compilation method parameters are the following parameters: 'binary_crossentropy', optimizer='Adagrad', metrics='cosine_proximity'. This result was also facilitated by the neural network parameters n_most_common_words=8000, emb_dim=128, batch_size=256, epochs=400. In general, the main role in determining the error loss accuracy was played by the neural network learning model: Embedding(), SpatialDropout1D (0.7)), LSTM(64, dropout=0.7, recurrent_dropout=0.7)), Dense()).

Table 3

Parameters of the model.compile() method

Experiment	Learning time (min)	n_most_common_words	number of topics	epochs	Loss_type	optimizer	metrics	loss	accuracy	val_loss	val_acc
1	2	3	4	5	6	7	8	9	10	11	12
E1	~40	36 912	50	400	mean_squared_error	SGD	binary_accuracy	0.0033	0.9837	0.3125	0.0033
E 2	~20	19865	50	200	mean_absolute_error	Adam	accuracy	2.1209	0.9975	0.3002	2.1209
E 3	13.1	7 933	50	200	mean_absolute_percentage_error	RMSprop	cosine_proximity	0.0033	0.9767	0.2675	0.0033
E 4	28.98	7 933	70	500	mean_squared_logarithmic_error	SGD	categorical_accuracy	0.0910	0.4483	0.2855	0.0910
E 5	57.4	7 933	80	500	squared_hinge	Adagrad	sparse_categorical_accuracy	5.6881	1.0000	0.3471	5.6881
E 6	62	7 933	50	600	hinge	Adadelta	top_k_categorical_accuracy	0.0014	0.9883	0.2192	0.0014
E 7	70	5 932	70	600	categorical_hinge	Adamax	binary_accuracy	0.0059	0.9802	0.1750	0.0059
E 8	71	5 932	70	600	logcosh	Adam	categorical _accuracy	0.0170	0.8560	0.1901	0.0170

Continuation of Table 3

1	2	3	4	5	6	7	8	9	10	11	12
E 9	47.1	2400	30	400	huber_loss	Adagrad	cosine_proximity	0.0015	0.9833	0.1465	0.0015
E 10	6.93	2400	50	600	categorical_crossentropy	Adagrad	binary_accuracy	5.3517	1.0000	0.1540	5.3517
E 11	23.9	2400	25	500	poisson	SGD	sparse_top_k_categorical_accuracy	9.9523	1.0000	0.2010	9.9523
E 12	28	2400	30	800	categorical_crossentropy	Nadam	categorical_accuracy	0.0017	1.0000	0.1864	0.0017
E 13	9.5	1685	10	600	binary_crossentropy	Adam	cosine_proximity	3.4168	1.0000	0.2022	3.4168
E 14	1,8	1685	20	400	binary_crossentropy	Adagrad	cosine_proximity'	0.2443	0.8333	0.1342	0.2443

6. Discussion of the results of the study using the LDA model for the semantic analysis of the text in the Kazakh language

The advantage of using the LDA model in comparison with the known methods in this study is provided by the proposals for using the Dictionary function, the Phraser. bigrams function to build the frequency matrix of the bigram, and the doc2bow function (Fig. 3). The LDA model determines the semantic probability of the keywords of one document and gives them an evaluation coefficient; relative to the obtained evaluation, the words are distributed by topic (Fig. 4). It has been found that the best way to identify topic consistency is to use a coherent model. Topic coherence is calculated by the Coherence model function and shows what is the optimal number of topics from the resulting model that has a high coherence score to obtain well-trained data (Fig. 4).

The neural network was built from a sequence of Embedding SpatialDropout1d, LSTM, Dense layers (Fig. 6). The selection of optimal parameters for the efficiency of the constructed neural network was carried out by selecting the optimizer parameters, metrics and error functions that contribute to data learning (Table 3). The result of this study is considered the best in contrast to [23], where the neural network includes Embedding, LSTM, LSTM, Dense layers and the accuracy result shows 0.71. The result of the experiment in the course of this study found that the use of the LDA model and the proposed sequence of neural network layers approximates the learning accuracy to 0.833. It should be noted that the constructed neural network using the LDA model gives keywords distributed on topics with an accuracy of 83 %.

The learning of the neural network is limited to the Kazakh language, because the word-form dictionary includes only the word forms of the Kazakh language.

The disadvantage of this study is that the use of the LDA model in learning a neural network only for the Kazakh language, in the future it is possible to expand the base for other unconsidered world languages. For artificial intelligence, there is a question of 100 % understanding of the text by a computer, but existing research requires additional development, because each language has its own characteristics.

7. Conclusions

1. During the study, the text in the Kazakh language was processed using the LDA model. To solve the problem of developing a rating matrix, an LDA model with Gibbs simulation was used. This model performs smoothing of frequency estimates of conditional probabilities. The semantic relationship of words in LDA that defines the topic is formed by highlighting the frequent use of certain words in combination with other words, rarely used words are taken as noise and ignored by reducing the dimension of the vectors obtained after the Dirichlet distribution. One of the factors affecting the quality of the distribution of words by topics and topics by documents is the previously prepared word form dictionary and text corpus, as well as the process of applying the doc2bow method in the program (corpus_c=[id2word.doc2bow(text) for text in texts]). The result of thematic modeling was a matrix of estimates of the distribution of words by topics and topics by documents with a dimension of 130×128. As a result of the computational processes performed, the distribution of keywords by topics and topics by documents are obtained.

2. The result of LDA modeling, presented as a data matrix, was transferred to a neural network of 4 layers. The neural network was trained using the deep learning method and the error propagation method. Each inner layer had a Dropout method to sparse neurons with a probability of 0.7 to avoid overfitting the neural network. The essence of the developed neural network is that it shows the degree of data learning. That is, the result of sufficiently high-quality learning is a more accurate distribution of keywords by topic and topic by document. The result of high-quality learning is obtained by selecting the optimizer parameters, metrics and error functions. In this study, when fitting parameters, error function='binary_crossentropy', optimizer='Adagrad', metric='cosine_proximity', the following results were obtained: error=0.2443 and learning accuracy=0.8333. Approximation of the error to zero, and the accuracy of learning to one indicates the correctness of learning.

Conflict of interest

The authors declare that they have no conflict of interest in relation to this research, whether financial, personal, authorship or otherwise, that could affect the research and its results presented in this paper.

References

1. Garcia-Arroyo, J. L., Garcia-Zapirain, B. (2017). Segmentation of skin lesions based on fuzzy classification of pixels and histogram thresholding. arXiv. doi: https://doi.org/10.48550/arXiv.1703.03888
2. De Falco, I., De Pietro, G., Della Cioppa, A., Sannino, G., Scafuri, U., Tarantino, E. (2019). Evolution-based configuration optimization of a Deep Neural Network for the classification of Obstructive Sleep Apnea episodes. Future Generation Computer Systems, 98, 377–391. doi: https://doi.org/10.1016/j.future.2019.01.049
3. Jafari, M., Xu, H. (2018). Intelligent Control for Unmanned Aerial Systems with System Uncertainties and Disturbances Using Artificial Neural Network. Drones, 2 (3), 30. doi: https://doi.org/10.3390/drones2030030
4. Feng, B., Xu, J., Lin, Y., Li, P. (2020). A Period-Specific Combined Traffic Flow Prediction Based on Travel Speed Clustering. IEEE Access, 8, 85880–85889. doi: https://doi.org/10.1109/access.2020.2992657
5. Mehedi, I. M., Bassi, H., Rawa, M. J., Ajour, M., Abusorrah, A., Vellingiri, M. T. et. al. (2021). Intelligent Machine Learning With Evolutionary Algorithm Based Short Term Load Forecasting in Power Systems. IEEE Access, 9, 100113–100124. doi: https://doi.org/10.1109/access.2021.3096918
6. Christianto, Christian, J., Rusli, A. (2020). Evaluating RNN Architectures for Handling Imbalanced Dataset in Multi-Class Text Classification in Bahasa Indonesia. International Journal of Advanced Trends in Computer Science and Engineering, 9 (5), 8418–8423. doi: https://doi.org/10.30534/ijatcse/2020/217952020
7. Shen, M., Lei, J., Du, F., Bi, Z. (2020). Power Micro-Blog Text Classification Based on Domain Dictionary and LSTM-RNN. Testbeds and Research Infrastructures for the Development of Networks and Communications, 36–45. doi: https://doi.org/10.1007/978-3-030-43215-7_3
8. Shi, M., Liu, J., Zhou, D., Tang, M., Cao, B. (2017). WE-LDA: A Word Embeddings Augmented LDA Model for Web Services Clustering. 2017 IEEE International Conference on Web Services (ICWS). doi: https://doi.org/10.1109/icws.2017.9
9. Hwang, M.-H., Ha, S., In, M., Lee, K. (2018). A Method of Trend Analysis using Latent Dirichlet Allocation. International Journal of Control and Automation, 11 (5), 173–182. doi: https://doi.org/10.14257/ijca.2018.11.5.15
10. Huang, B., Yang, Y., Mahmood, A., Wang, H. (2012). Microblog Topic Detection Based on LDA Model and Single-Pass Clustering. Lecture Notes in Computer Science, 166–171. doi: https://doi.org/10.1007/978-3-642-32115-3_19
11. Gao, L., Eldin, N. (2014). Employers' Expectations: A Probabilistic Text Mining Model. Procedia Engineering, 85, 175–182. doi: https://doi.org/10.1016/j.proeng.2014.10.542
12. Tang, Z., Zhang, X., Niu, J. (2020). LDA Model and Network Embedding-Based Collaborative Filtering Recommendation. 2019 6th International Conference on Dependable Systems and Their Applications (DSA). doi: https://doi.org/10.1109/dsa.2019.00043
13. Xu, Y., Zuo, X. (2016). A LDA model based text-mining method to recommend reviewer for proposal of research project selection. 2016 13th International Conference on Service Systems and Service Management (ICSSSM). doi: https://doi.org/10.1109/icsssm.2016.7538568
14. Xu, G., Wu, X., Yao, H., Li, F., Yu, Z. (2019). Research on Topic Recognition of Network Sensitive Information Based on SW-LDA Model. IEEE Access, 7, 21527–21538. doi: https://doi.org/10.1109/access.2019.2897475
15. Akanova, A., Ospanova, N., Kukharenko, Y., Abildinova, G. (2019). Development of the algorithm of keyword search in the Kazakh language text corpus. Eastern-European Journal of Enterprise Technologies, 5 (2 (101)), 26–32. doi: https://doi.org/10.15587/1729-4061.2019.179036
16. Blei, D. M., Ng, A. Y., Jordan, M. I. (2003). Latent dirichlet allocation. Journal of machine Learning researchm 3, 993–1022. Available at: https://jmlr.org/papers/volume3/blei03a/blei03a.pdf
17. lda: Topic modeling with latent Dirichlet Allocation. Available at: https://lda.readthedocs.io/en/latest/
18. Buddana, H. V. K. S., Kaushik, S. S., Manogna, P., P.S., S. K. (2021). Word Level LSTM and Recurrent Neural Network for Automatic Text Generation. 2021 International Conference on Computer Communication and Informatics (ICCCI). doi: https://doi.org/10.1109/iccci50826.2021.9402488
19. Zhang, D., Luo, T., Wang, D. (2016). Learning from LDA Using Deep Neural Networks. Lecture Notes in Computer Science, 657–664. doi: https://doi.org/10.1007/978-3-319-50496-4_59
20. Understanding LSTM Networks. Available at: http://colah.github.io/posts/2015-08-Understanding-LSTMs/
21. Graves, A. (2014). Generating Sequences With Recurrent Neural Networks. arxiv.org. doi: https://doi.org/10.48550/arXiv.1308.0850
22. Kingma, D., Ba, J. (2015). Adam: A Method for Stochastic Optimization. Published as a conference paper at the 3rd International Conference for Learning Representations. San Diego. doi: https://doi.org/10.48550/arXiv.1412.6980
23. Smirnova, O. S., Shishkov, V. V. (2016). The choice of the topology of neural networks and their use for the classification of small texts. International Journal of Open Information Technologies, 4 (8), 50–54. Available at: https://cyberleninka.ru/article/n/vybor-topologii-neyronnyh-setey-i-ih-primenenie-dlya-klassifikatsii-korotkih-tekstov